An Incremental Dual nu-Support Vector Regression Algorithm

Hang Yu1, Jie Lu2[0000-0003-0690-4732] and Guangquan Zhang3

[1,2,3] Decision System and e-Service Intelligence Laboratory, Centre for Artificial Intelligence,
Faculty of Engineering and Information Technology,
University of Technology Sydney Australia
Hang.Yu@student.uts.edu.au, Jie.Lu@uts.edu.au,
Guangquan.Zhang@uts.edu.au

Abstract. Support vector regression (SVR) has been a hot research topic for several years as it is an effective regression learning algorithm. Early studies on SVR mostly focus on solving large-scale problems. Nowadays, an increasing number of researchers are focusing on incremental SVR algorithms. However, these incremental SVR algorithms cannot handle uncertain data, which are very common in real life because the data in the training example must be precise. Therefore, to handle the incremental regression problem with uncertain data, an incremental dual nu-support vector regression algorithm (dual-*v*-SVR) is proposed. In the algorithm, a dual-*v*-SVR formulation is designed to handle the uncertain data at first, then we design two special adjustments to enable the dual-*v*-SVR model to learn incrementally: incremental adjustment and decremental adjustment. Finally, the experiment results demonstrate that the incremental dual-*v*-SVR algorithm is an efficient incremental algorithm which is not only capable of solving the incremental regression problem with uncertain data, it is also faster than batch or other incremental SVR algorithms.

Keywords: Support Vector Regression, Regression Learning Algorithm, Incremental Regression Problem, Uncertain Data.

1 First Section

1.1 A Subsection Sample

Support vector regression (SVR) has been a hot research topic for several years because it is an effective regression learning algorithm [1-4]. It aims to minimize a combination of the empirical risk and a regularization term [5]. Early studies on SVR mostly focus on solving large-scale problems [6-8]. Nowadays, an increasing number of researchers are focusing on incremental SVR algorithms [9-13]. Junshui et al. introduced ε-SVR and developed an accurate online support vector regression (AOSVR) [9]. Omitaomu et al. propose AOSVR with varying parameters that uses varying SVR parameters rather than fixed SVR parameters [11]. Later, Gu et al. proposed an exact incremental ν-SVR algorithm (INSVR) [13].

These incremental SVR algorithms require precise data for the training examples. However, the data in many practical applications is not precise yet represented by an uncertain data. For example, the height of a man is between 180 cm and 185 cm. Therefore, some researchers have proposed many improved SVR algorithms [14-16] which explicitly handle uncertain data and perform better than traditional SVRs. Hao et.al incorporate the concept of fuzzy set theory into the SVM regression model [15]. Peng proposed an interval twin support vector regression algorithm for interval input-output data [16]. Several SVR algorithms treat uncertain data as random noise [17-19]. By replacing the constraints in the standard ε-SVR with probability constraints, chance-constrained, robust regression formulations can be obtained. For example, an robust SVR algorithm which is robust to bounded noise was proposed in [19]. However, the quadratic programming problems (QPPs) of these algorithms is too complex to translate these algorithms into incremental algorithms directly.

Hence, to handle the incremental regression problem with uncertain data, the incremental dual nu-support vector regression (dual-*v*-SVR) algorithm is proposed. In the algorithm, a dual-*v*-SVR formulation be designed to handle the uncertain data at first, then we design two special adjustments to enable the dual-*v*-SVR model to learn incrementally: incremental adjustment and decremental adjustment. Finally, the experiment results demonstrate that the incremental dual-*v*-SVR algorithm is an efficient incremental algorithm which is not only capable of solving the incremental regression problem with uncertain data, it is also faster than batch or other incremental SVR algorithms.

The rest of this paper is organized as follows. In Section 2, we describe the formulation, KKT conditions and two adjustments of the incremental dual-*v*-SVR algorithm. The experimental setup, results and discussions are presented in Section 3. Section 4 provides the concluding remarks.

2 An Incremental Dual-v-SVR

As previously mentioned, the QPPs of many SVR algorithms are too complex to translate into online algorithms directly. Hence we propose a dual-v-SVR algorithm estimates the upper bound functions $f_1(x)=\langle w_1 \cdot x\rangle + b_1$ and lower bound functions $f_2(x)=\langle w_2 \cdot x\rangle + b_2$ at same time, and the final regression function is constructed as follows: $f(x)=\frac{1}{2}\left[f_1(x)+f_2(x)\right]$.

2.1 The formulation

For cases with data uncertainties, we suppose the independent variables are perturbed by noise: $x_i = x_i + \delta_i$, such that $\|\delta_i\| \le \tau$, where δ_i represents a bounded perturbation with $\tau > 0$ and x_i constructs a nominal vector $X=(x_1, x_2, \ldots, x_N)$. The dependent variable Y is also perturbed by noise: $Y = y + \sigma = [u, l]$, such that $\|\sigma\| \le \hat{\tau}$, where σ represents a bounded perturbation with $\hat{\tau} > 0$. Thus, we can get:

$$\left\langle w \cdot \Phi\left(x_i + \delta_i\right)\right\rangle = \left\langle w \cdot \Phi\left(x_i\right)\right\rangle + \left\langle w \cdot \Phi\left(\delta_i\right)\right\rangle \tag{1}$$

By the Cauchy-Schwarz inequality, we have:

$$\left|\left\langle w \cdot \Phi(\delta_i)\right\rangle\right| \le \|w\| \cdot \|\delta_i\| \le \tau \|w\| \tag{2}$$

Hence a formulation of dual-*v*-SVR is:

$$\min_{w_1, b_1, \xi_{1i}} \frac{N}{2}\|w_1\|^2 + C_1\left(v_1 b_1 N + \sum_{i=1}^{N} \xi_{1i}\right)$$

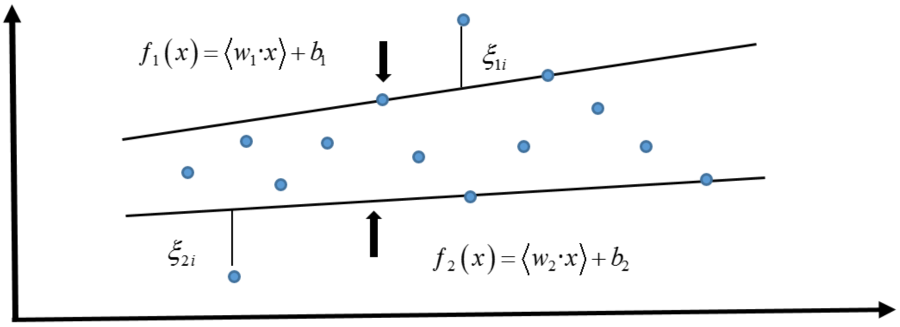
$$\text{s.t. } \left\langle w_1 \cdot \Phi\left(x_i\right)\right\rangle + \tau\|w_1\| + b_1 \ge u - \xi_{1i},\ \xi_{1i} \ge 0,\ i = 1, \ldots, N \tag{3}$$

and

$$\min_{w_2, b_2, \xi_{2i}} \frac{N}{2}\|w_2\|^2 + C_2\left(v_2 b_2 N + \sum_{i=1}^{N} \xi_{2i}\right)$$

$$\text{s.t. } \left\langle w_2 \cdot \Phi\left(x_i\right)\right\rangle + \tau\|w_2\| + b_2 \ge l + \xi_{2i},\ \xi_{2i} \ge 0,\ i = 1, \ldots, N \tag{4}$$

where Φ is a nonlinear transform: $R^N \rightarrow F$ to map the data points into a higher dimensional feature space F, $\|w_{1,2}\|^2$ is the regularization term, C_1, $C_2 \ge 0$ are the regularization parameters and ξ_{1i}, ξ_{2i} are the slack variables. Parameter $v_1 \in (0, 1)$ controls the tradeoff between the minimization of $b_{1,2}$ and the minimization of errors.



$f_1(x) = \langle w_1 \cdot x\rangle + b_1$
ξ_{1i}
ξ_{2i}
$f_2(x) = \langle w_2 \cdot x\rangle + b_2$

Fig. 1. Incremental dual-*v*-SVR

2.2 KKT Conditions

Let $Q_{i,j}=\frac{1}{N}k\langle x_i,x_j\rangle=\frac{1}{N}\langle(\Phi(x_i)+\tau)\cdot(\Phi(x_j)+\tau)\rangle$, the dual problem of (3) can be written as:

$$\min_{\alpha}\frac{1}{2}\sum_{i,j=1}^{N}\alpha_{1i}\alpha_{1j}Q_{ij}-\sum_{i=1}^{N}y_{1i}\alpha_{1i}$$

$$\text{s.t. } \sum_{i=1}^{N}\alpha_{1i}=C_1v_1N\,,\ 0\le\alpha_{1i}\le C_1,\ i=1,\dots,N \tag{5}$$

From equation (5), we can see the box constraints of α_{1i} are independent of the size of the training sample set.

Then, we introduce the extended training set *S*, which is defined as $S=S^{-}\cup S^{+}$, where $S^{-}=\{(x_{1i},y_{1i},z_{1i}=-1)\}_{i=1}^{N}$, $S^{+}=\{(x_{1i},y_{1i},z_{1i}=+1)\}_{i=1}^{N}$ and z_i is the label of the training sample (x_{1i},y_{1i}). Thus, the minimization problem (5) can be further re-written as:

$$\min_{\alpha}\frac{1}{2}\sum_{i,j=1}^{2N}\alpha_{1i}\alpha_{1j}Q_{ij}$$

$$\text{s.t. } \sum_{i=1}^{2N}z_{1i}\alpha_{1i}=0\,,\ \sum_{i=1}^{2N}\alpha_{1i}=2C_1v_1N\,,\ 0\le\alpha_{1i}\le C_1,\ i=1,\dots,2N \tag{6}$$

The solution of the minimization problem (6) can also be obtained by minimizing the following convex quadratic objective function under constraints:

$$\min_{0\le\alpha_{1i}\le C_1}W=\frac{1}{2}\sum_{i,j=1}^{2N}\alpha_{1i}\alpha_{1j}Q_{ij}+\mu\left(\sum_{i=1}^{2N}z_{1i}\alpha_{1i}\right)+\varepsilon\left(\sum_{i=1}^{2N}\alpha_{1i}-2C_1v_1N\right) \tag{7}$$

Then by the KKT theorem, the first-order derivative of *W* leads to the following KKT conditions:

$$\frac{\partial W}{\partial\mu}=\sum_{i=1}^{2N}z_{1i}\alpha_{1i}=0 \tag{8}$$

$$\frac{\partial W}{\partial\varepsilon}=\sum_{i=1}^{2N}\alpha_{1i}=2C_1v_1N \tag{9}$$

$$\forall i\in S: g_{1i}=\frac{\partial W}{\partial\alpha_{1i}}=\sum_{j=1}^{2N}\left(Q_{ij}\alpha_{1i}+z_{1i}\mu+\varepsilon\right)$$

$$\begin{cases}\ge 0 & for\ \alpha_{1i}=0\\ =0 & for\ 0<\alpha_{1i}<C_1\\ \le 0 & for\ \alpha_{1i}=C_1\end{cases} \tag{10}$$

According to the value of the function g_{1i}, the extended training set S is partitioned into three independent sets (see Fig.2):

$$\begin{array}{ll} \text{Support Set} & S_S = \{ i \mid g_{1i}(x_{1i}) = 0, 0 < \alpha_{1i} < C_1 \} \\ \text{Error Set} & S_E = \{ i \mid g_{1i}(x_{1i}) \le 0, \alpha_{1i} = C_1 \} \\ \text{Remaining Set} & S_R = \{ i \mid g_{1i}(x_{1i}) \ge 0, \alpha_{1i} = 0 \} \end{array} \tag{11}$$

The function $f_2(x)$ can use the same procedure for the analyses and the KKT condition is same as equation (9). Therefore, the two functions build an insensitive zone. Furthermore, only errors outside the insensitive zone contribute to the cost function, and only those points (i.e. SVs [2]) determine the final regression model. Hence, the obtained regression model of dual-v-SVR is sparse.

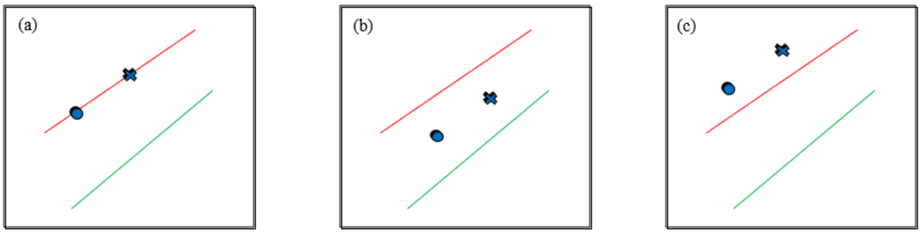


Fig. 2. The partitioning of the training samples S into three independent sets by KKT conditions. (a) S_S. (b) S_E. (c) S_R.

2.3 Incremental & Decremental Adjustment

From section 2.2, we know when a new sample arrives, the weights of the new sample are set to 0 initially, and then it needs to be assigned into a set to satisfy the KKT conditions. Hence, if the assignment violates the KKT conditions, the weights of the new sample will be adjusted. Furthermore, due to a conflict between equations (8) and (9), the adjustment of dual-v-SVR involves two steps: incremental and decremental.

Incremental Adjustment.

In the incremental adjustment step, we need to ensure all the samples satisfy the KKT conditions, but the restriction $\sum_{i=1}^{N} \alpha_{1i} = 2C_1 vN$ does not need to hold for all the weights, so we have the following linear system:

$$\Delta g_{1i} = \sum_{j \in S_S} \Delta\alpha_{1j} Q_{ij} + z_{1i}\Delta\mu + \Delta\varepsilon - \Delta\alpha_c Q_{ic} = 0 \tag{12}$$

$$\sum_{j \in S_S} z_{1j}\Delta\alpha_{1j} + z_{1c}\Delta\alpha_{1c} = 0 \tag{13}$$

where $\Delta\alpha_{1j}$, z_{1j}, $\Delta\varepsilon$ and Δg_{1i} denote the corresponding variations. Then we define $e_{S_S} = [1, \ldots, 1]^T$ as the $|S_s|$ dimensional column vector with all ones, let $z_{S_S} = \left[z_1, \ldots z_{|S_S|} \right]^T$, and let $Q_{S_S S_S}$ denote the matrix, the above liner system can be further rewritten as:

$$\underbrace{\begin{bmatrix} 0 & z_{S_S}^T \\ z_{S_S} & Q_{S_S S_S} \end{bmatrix}}_{Q_{\setminus\varepsilon\varepsilon}} \cdot \underbrace{\begin{bmatrix} \Delta\mu \\ \Delta\alpha_{S_S} \end{bmatrix}}_{\Delta h_{\setminus\varepsilon}} = -\begin{bmatrix} z_{1c} & 0 \\ Q_{S_S c} & e_{S_S} \end{bmatrix} \cdot \begin{bmatrix} \Delta\alpha_{1c} \\ \Delta\varepsilon \end{bmatrix} \tag{14}$$

where $Q_{\setminus\varepsilon\varepsilon}$ is the abbreviation of the matrix above the under brace, $\Delta h_{\setminus\varepsilon}$ is also the abbreviation which is defined in the same way.

Let $R = Q_{\setminus\varepsilon\varepsilon}^{-1}$ and $\Delta\varepsilon = 0$, then the linear relationship between $\Delta h_{\setminus\varepsilon}$ and $\Delta\alpha_{1c}$ can be easily solved as follows:

$$\Delta h_{\setminus\varepsilon} = \begin{bmatrix} \Delta\mu \\ \Delta\alpha_{S_S} \end{bmatrix} = -R \cdot \begin{bmatrix} z_{1c} \\ Q_{S_S c} \end{bmatrix} \equiv \underbrace{\begin{bmatrix} \beta_\mu^c \\ \beta_{S_S}^c \end{bmatrix}}_{\beta_{\setminus\varepsilon}^c} \Delta\alpha_{1c} \tag{15}$$

where $\beta_{\setminus\varepsilon}^c$ stands for the dimension corresponding to μ in the vector $\beta_{\setminus\varepsilon}^c$. $\beta_{S_S}^c$ is the vector with the same meaning of $\beta_{\setminus\varepsilon}^c$. Accordingly, let $\beta_{\setminus\varepsilon}^c = 0$, then the relationship between $\Delta h_{\setminus\varepsilon}$ and $\Delta\alpha_c$ can also be defined as:

$$\Delta h_\varepsilon = \Delta\varepsilon = \beta_\varepsilon^c \Delta\alpha_c \tag{16}$$

By substituting (15) and (16) into (12), we can get the linear relationship between Δg_{1i} and $\Delta\alpha_c$ as follows:

$$\Delta g_i = \left(\sum_{j \in S_S} \beta_j^c Q_{ij} + z_{1i} \beta_\mu^c + \beta_\varepsilon^c + Q_{ic} \right) \Delta\alpha_{1c} \equiv \gamma_{1i}^c \Delta\alpha_{1c}, \forall i \in S \tag{17}$$

Obviously $\forall i \in S_S$, so we have $\gamma_{1i}^c = 0$. Thus, for each incremental adjustment, we can compute the maximal increment of $\Delta\alpha_{1c}$ (here denoted as $\Delta\alpha_{1c}{}^{\max}$), update *α*, *g*, *S_S*, S*_E*, S*_R* and the inverse matrix R, similar to the approaches in [20].

Decremental Adjustment.

In the decremental adjustment step, we gradually adjust $\sum_{i\in S}\alpha_{1i}$ to restore the equality $\sum_{i\in S}\alpha_{1i}=2Cv(N+1)$, so that the KKT conditions are satisfied by all the samples. For each adjustment of $\sum_{i\in S}\alpha_{1i}$, in order to ensure all the samples satisfy the KKT conditions, the weights of the samples in S_S, the Lagrange multipliers μ and ε should also be adjusted accordingly, and these have the following linear system:

$$\Delta g_{1i}=\sum_{j\in S_S}\Delta\alpha_{1j}Q_{ij}+z_{1i}\Delta\mu+\Delta\varepsilon=0\,,\ \forall i\in S_S \tag{18}$$

$$\sum_{j\in S_S}z_{1i}\Delta\alpha_{1j}=0 \tag{19}$$

$$\sum_{j\in S_S}\Delta\alpha_{1j}+\eta\Delta\varepsilon+\Delta\omega=0 \tag{20}$$

where $\Delta\omega$ is the introduced variable of adjusting $\sum_{i\in S}\alpha_{1i}$, η is any negative number, and $\eta\Delta\varepsilon$ is incorporated in (20) as an extra term. Using this extra term can prevent the recurrence of conflicts between equations (8) and (9). The above linear system can also be further rewritten as:

$$\underbrace{\begin{bmatrix}0 & 0 & z_{S_S}^T\\ 0 & \eta & e_{S_S}^T\\ z_{S_S} & e_{S_S} & Q_{S_SS_S}\end{bmatrix}}_{Q}\underbrace{\begin{bmatrix}\Delta\mu\\ \Delta\varepsilon\\ \Delta\alpha_{S_S}\end{bmatrix}}_{\Delta h}=-\begin{bmatrix}0\\1\\0\end{bmatrix}\Delta\omega \tag{21}$$

Let $R=Q^{-1}$, then the linear relationship between Δh and $\Delta\omega$ can be obtained as follows:

$$\Delta h=\begin{bmatrix}\Delta\mu\\ \Delta\varepsilon\\ \Delta\alpha_{S_S}\end{bmatrix}=-R\begin{bmatrix}0\\1\\0\end{bmatrix}\Delta\omega\equiv\underbrace{\begin{bmatrix}\beta_\mu\\ \beta_\varepsilon\\ \beta_{S_S}\end{bmatrix}}_{\beta}\Delta\omega \tag{22}$$

From equation (20), we have $\sum_{i\in S}\Delta\alpha_{1i}=-\left(1+\eta\beta_\varepsilon\right)\Delta\omega$, which implies that the control of the adjustment of $\sum_{i\in S}\Delta\alpha_{1i}$ is achieved by $\Delta\omega$.

Finally, substituting equation (20) into equation (16), we can also get the linear relationship between Δg_{1i} and $\Delta\omega$ as follows:

$$\Delta g_i = \left(\sum_{j \in S_S} \beta_j Q_{ij} + z_{1i} \beta_\mu + \beta_\varepsilon \right) \Delta\omega \equiv \gamma_{1i}^c \Delta\omega \,, \forall i \in S \tag{23}$$

Obviously, $\forall i \in S_S$, and we also have $\gamma_{1i} = 0$. However, the decremental adjustments cannot be used directly to obtain the optimal solution to the minimization problem (6). Hence, to solve this problem, we need to compute the adjustment quantity $\Delta\omega^*$ for each decremental adjustment such that a certain sample migrates among the sets S_S, S_R and S_E. If $\sum_{i \in S} \alpha_{1i} > 2Cv(N+1)$, we will compute the maximal adjustment quantity $\Delta\omega^{\max}$ and let $\Delta\omega^* = \Delta\omega^{\max}$; otherwise, we will compute the minimal adjustment quantity $\Delta\omega^{\min}$ and let $\Delta\omega^* = \Delta\omega^{\min}$.

3 Experiment Result

3.1 Experiment setup

In this section, we validate the performance of the proposed incremental dual-v-SVR algorithm (IDVSVR) through several experiments which compare our algorithm with standard TSVR [22], AOSVR [9], and INSVR [12]. All regression models are implemented in MATLAB 2016a version on Windows 7 running on a PC with system configuration Intel Core i5 processor (2.40 GHz) with 8-GB RAM. We also use *cadata*, and *Friedman* data sets. Cadata [12] is a real data set, *Friedman* is an artificial data set [21]. The details of the three data sets are shown in Table 1.

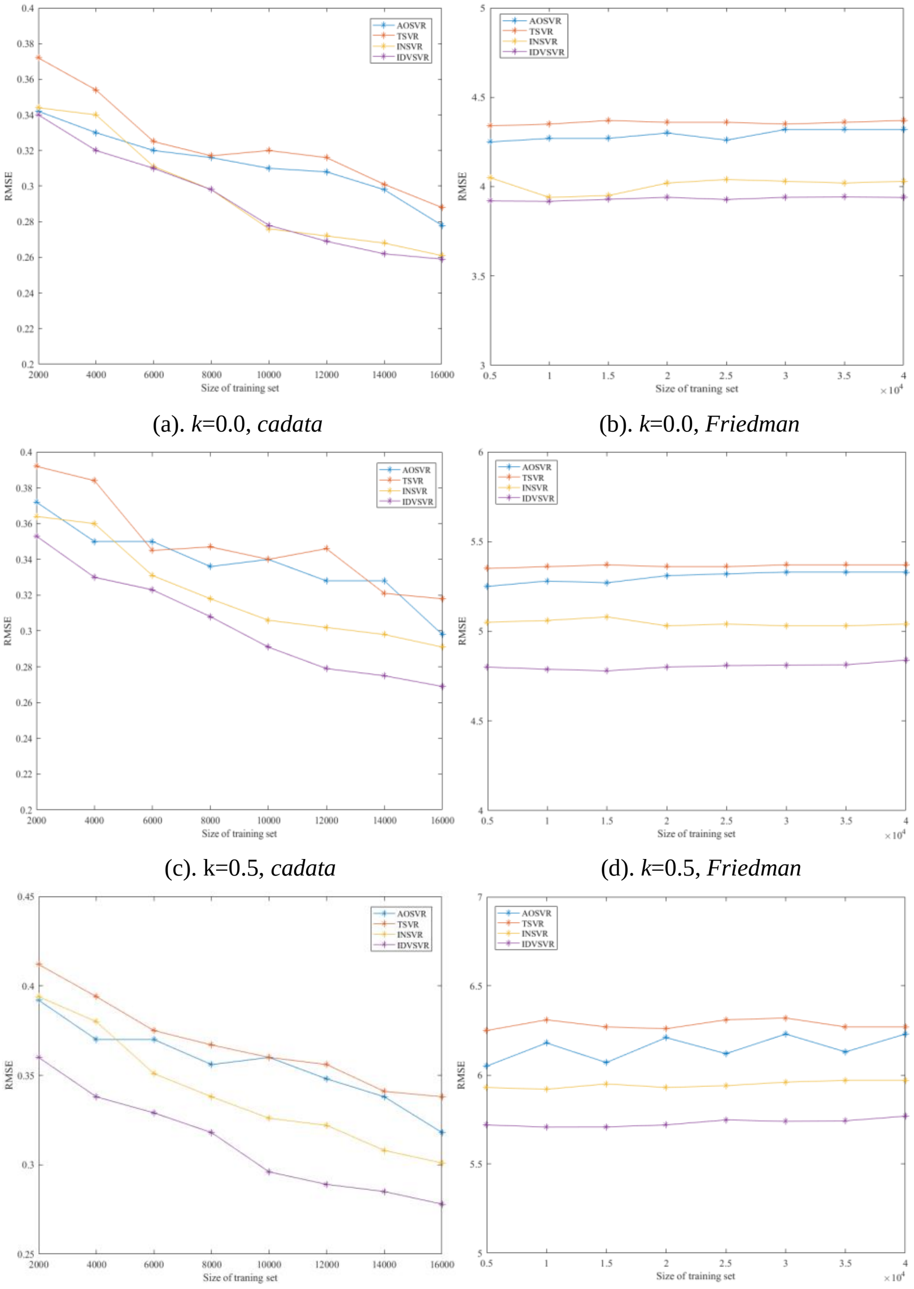
Table 1. Data sets used in the experiments

Data set	#training set	# attributes
cadata	20000	8
Friedman	40000	10

For simplicity, the Gaussian radial basis function kernel is adopted for all examples. We set the model parameters $C_1 = C_2 = C$ and $v_1 = v_2 = v$. The values of parameter C, q, ε, v are, respectively, selected from the sets $\{10^i \mid i = 0, 1, \ldots, 6\}$, $\{2^i \mid i = -9, -8, \ldots, 2\}$, $\{0.01, 0.02, \ldots, 0.4\}$, and $\{0.01, 0.02, \ldots, 0.6\}$. As for uncertainty, as the aforementioned data sets are not noisy, we artificial introduce a noisy e_i into predictor variable X and dependent variable Y, and e_i is drawn from a uniform distribution on U(-*k*, *k*). Here, U(-*k*, *k*) represents the uniformly random variable in [-*k*, *k*].

3.2 Performance Evaluation

In the first experiment, we compare their trends in relation to regression risk on noisy data when the data size N is increased. We use RMSE [22] to represent the accuracy where a smaller RMSE represents a lower risk. Fig.3 shows the comparison result for different data sets, different data size N and different k.



(a). *k*=0.0, *cadata*

(b). *k*=0.0, *Friedman*

(c). k=0.5, *cadata*

(d). *k*=0.5, *Friedman*

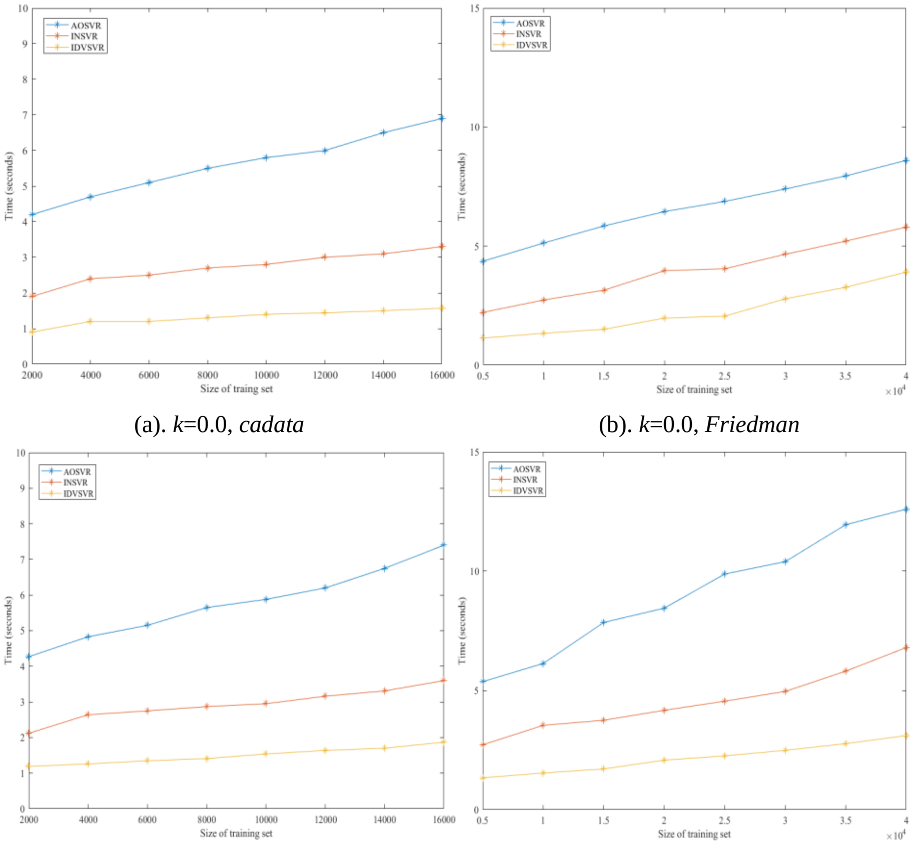
(e). k=1.0, *cadata*

(f). *k*=1.0, *Friedman*

Fig. 3. Comparison result of RMSE

From Fig.3, we can see that when the data size N is increased, no matter how much k is, the RMSE of any regression is gradually decreasing. When k = 0, IDVSVR and INSVR have the better performance. These results identify that both IDVSVR and INSVR have the advantage of using parameter v to control the bounds on the fraction of SVs and errors. However, when k increases, or in other words, when the data is perturbed by noise, the IDVSVR still has the better performance, as the performance of INSVR is very poor. Furthermore, Fig.3 also shows that the accuracy of IDVSVR remains relatively stable even when k=1. Hence, a major advantage of the proposed IDVSVR over the other algorithms is its effectiveness in handling uncertain data.

In the second experiment, we compare the training speed on noisy data when the data size N is increased. In the previous experiment, we know TSVR has the worst generalization ability. Hence, we only test the training time of AOSVR, INSVR, and IDVSVR. Fig.4 shows the comparison results in terms of time for the different data sets, different data size N and different k.

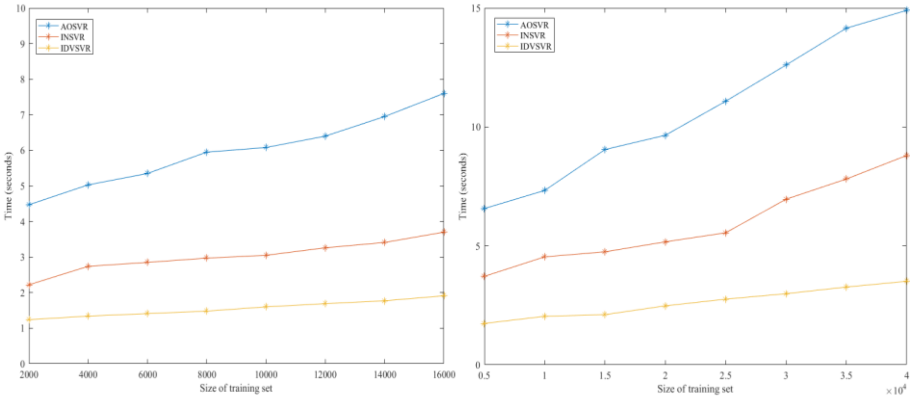


(a). k=0.0, *cadata*

(b). k=0.0, *Friedman*

(c). k=0.5, *cadata*

(d). k=0.5, *Friedman*



(e). *k*=1.0, *cadata*

(f). *k*=1.0, *Friedman*

Fig. 4. Comparison result of training speed.

Fig.4 demonstrates that the learning speed of the IDVSVR algorithm is generally much faster than the other online SVR algorithms when data size *N* increases. One reason for this is that in IDVSVR, the two nonparallel functions are estimated by solving two SVR-type QPPs of smaller size at the same time, so the learning speed of IDVSVR is faster. The second reason is using an extra term to prevent the recurrence of conflicts between the equation (8) and (9) and reduce the number of adjustment. Fig.4 furthermore suggests that the prediction speed of IDVSVR model remained relatively stable in different *k*. But the training speed of other online SVR algorithms becomes slower with the *k* increased.

4 Concluding remarks

As there is no effective SVR algorithm which can handle incremental regression problem with uncertain data, we design an incremental dual-*v*-SVR algorithm in this paper. Our proposed incremental dual-*v*-SVR has good robustness against uncertainty and can handle the incremental regression problem efficiently. Furthermore, there are a total of five advantages of our proposed incremental algorithm: (1) the learning speed of incremental dual-v-SVR algorithm is fast; (2) the sparsity of incremental dual-*v*-SVR algorithm is improved; (3) the incremental dual-*v*-SVR algorithm has good generalization performance; (4) the incremental dual-*v*-SVR algorithm also can use parameter *v* to control the bounds on the fractions of SVs and errors; and (5) in incremental dual-*v*-SVR algorithm, the box constraints are independent of the size of the training sample set. The experimental results also prove our conclusion.

References

1. Scholkopf, Bernhard Smola, A.J.: New Support Vector Algorithms. Neural Comput. Learn. II. 1245, 1207–1245 (1998).

2. Chang, C.-C., Lin, C.-J.: Libsvm. ACM Trans. Intell. Syst. Technol. 2, 1–27 (2011).
3. Gu, B., Sheng, V.S.: A Robust Regularization Path Algorithm for ν-Support Vector Classification. IEEE Trans. Neural Networks Learn. Syst. 28, 1241–1248 (2017).
4. Wang, G., Zhang, G., Choi, K., Lu, J.: Deep Additive Least Squares Support Vector Machines for Classification With Model Transfer. IEEE Trans. Syst. Man, Cybern. Syst. 1–14 (2017).
5. Smola, A.J., B.~Schölkopf: A Tutorial on Support Vector Regression. Stat. Comput. 199–222 (2001).
6. Shevade, S.K., Keerthi, S.S., Bhattacharyya, C., Murthy, K.R.K.: Improvements to the SMO algorithm for SVM regression. IEEE Trans. Neural Networks. 11, 1188–1193 (2000).
7. Takahashi, N., Guo, J., Nishi, T.: Global convergence of SMO algorithm for support vector regression. IEEE Trans. Neural Netw. 19, 971–982 (2008).
8. Collobert, R., Williamson, R.C.: SVMTorch: Support Vector Machines for Large-Scale Regression Problems. J. Mach. Learn. Res. 1, 143–160 (2001).
9. Ma, J., Theiler, J., Perkins, S.: Accurate on-line support vector regression. Neural Comput. 15, 2683–2703 (2003).
10. Omitaomu, O.A., Jeong, M.K., Badiru, A.B., Hines, J.W.: Online support vector regression approach for the monitoring of motor shaft misalignment and feedwater flow rate. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 37, 962–970 (2007).
11. Omitaomu, O.A., Jeong, M.K., Badiru, A.B.: Online support vector regression with varying parameters for time-dependent data. IEEE Trans. Syst. Man, Cybern. Part ASys-tems Humans. 41, 191–197 (2011).
12. Gu, B., Sheng, V.S., Tay, K.Y., Romano, W., Li, S.: Incremental Support Vector Learning for Ordinal Regression. IEEE Trans. Neural Networks Learn. Syst. 26, 1403–1416 (2015).
13. Gu, B., Sheng, V.S., Wang, Z., Ho, D., Osman, S., Li, S.: Incremental learning for ν -Support Vector Regression. Neural Networks. 67, 140–150 (2015).
14. Hong, D.H.H.D.H., Hwang, C.H.C.: Interval regression analysis using quadratic loss support vector machine. IEEE Trans. Fuzzy Syst. 13, 229–237 (2005).
15. Yang, X., Zhang, G., Lu, J., Ma, J.: A kernel fuzzy c-Means clustering-based fuzzy support vector machine algorithm for classification problems with outliers or noises. IEEE Trans. Fuzzy Syst. 19, 105–115 (2011).
16. Peng, X., Chen, D., Kong, L., Xu, D.: Interval twin support vector regression algorithm for interval input-output data. Int. J. Mach. Learn. Cybern. 6, 719–732 (2015).
17. Chen, G., Zhang, X., Wang, Z.J., Li, F.: Robust support vector data description for outlier detection with noise or uncertain data. Knowledge-Based Syst. 90, 129–137 (2015).
18. Yang, X., Tan, L., He, L.: A robust least squares support vector machine for regression and classification with noise. Neurocomputing. 140, 41–52 (2014).
19. Huang, G., Member, S., Song, S., Wu, C., You, K.: Robust Support Vector Regression for Uncertain Input and Output Data. IEEE Trans. NEURAL NETWORKS Learn. Syst. 23, 1690–1700 (2012).
20. Gu, B., Wang, J., Yu, Y., Zheng, G., Huang, Y., Xu, T.: Accurate on-line ν-support vector learning. Neural Networks. 27, 51–59 (2012).
21. Meyer, D., Leisch, F., Hornik, K.: The support vector machine under test. Neurocomputing. 55, 169–186 (2003).
22. Peng, X.: TSVR: An efficient Twin Support Vector Machine for regression. Neural Networks. 23, 365–372 (2010).