2011 International Conference on Computer Science and Information Technology (ICCSIT 2011)
IPCSIT vol. 51 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V51.54

A Novel Incremental Learning Regularized Least Squares Support Vector Fuzzy Regression

Yuan jian[+] and Chen yongqi

College of Science and Technology, Ningbo University, Ningbo, China

Abstract. For estimating imprecise system, a incremental learning regularized least squares support vector fuzzy regression model is proposed. This model is applying the fuzzy sets principle and Incremental Learning method. As against the solution of a complicated quadratic programming problem in previous support vector fuzzy regression model, the proposed model reduces memory and calculates time because of utilizing the historical training results and equality constraints. Numerical examples are given to demonstrate the effectiveness of the proposed model.

Keywords: Fuzzy regression; regularized least squares; incremental learning

1. Introduction

For these years, Support vector machines(SVM) is developed for function estimation by Vapnik [1-3]. SVM has emerged as an alternative approach to neural networks. This is due to the fact that the LS-SVM is established based on the structural risk minimization principle rather than the empirical error commonly implemented in the neural networks. But tradition SVM can not estimate interval data. For estimation interval data, Hong et al.[17] presented multivariate fuzzy linear and nonlinear support vector regression model based the idea of Tanaka. The fuzzy support vector regression model was not only used to estimate crisp input-fuzzy output system but also fuzzy input-fuzzy output system. Chih-Chia Yao et al.[18] presented Fuzzy regression model based on asymmetric support vector machines by incorporating the concept of fuzzy set theory. However, weight vector and the bias term of all these support vector fuzzy regression model are solved by complicated quadratic programming problem. In this paper, a new incremental learning regularized least squares support vector fuzzy regression machines is proposed. The proposed fuzzy regression model is applying the fuzzy sets principle in weight vector and bias term. For this reason, this model can estimate fuzzy input-fuzzy output system. In addition, Training of the least squares support vector fuzzy regression can be implemented in a way of incremental learning avoiding computing large-scale matrix inverse but maintaining the precision when training and testing data[6]. Because this method fully utilizes the historical training results and reduces memory and time.

This paper is organized as follows. Section II introduces regularized least squares support vector machines. Section III proposes the incremental learning algorithm for regularized least support vector fuzzy regression machines

Experiments are presented in Section IV. Finally, Section V puts forward the concluding remarks.

2. Regularized Least Squares Support Vector Fuzzy Regression

In order to get the regularized least squares support vector fuzzy regression model for fuzzy input-fuzzy output system, there are some preliminaries: 1) Given fuzzy

[+] Corresponding author.
E-mail address: 605277691@qq.com, lingfen7781@163.com

samples $X_i, Y_i, i = 1 \cdots n$. $X_i = (x_i, e_{xi}) \in T(R^l)$, $Y_i = (y_i, e_i) \in T(R)$. They are symmetric triangular fuzzy numbers. 2) In the proposed fuzzy regression model, weight vector $W = (w)$ is a crisp number. Fuzzy bias term $B = (b, d)$ are symmetric triangular fuzzy number. W is denoted in the vector form which is $W = [w_1 \quad w_2 \quad \cdots \quad w_n]^T$. b and d are scalar parameters. The fuzzy function regression model can be presented as: $y(x) = W^T \phi(x) + B = (w\phi(x) + b, w\phi(|e_{xi}|) + d)$. The degree of the fitting between the fuzzy regression model and fuzzy training samples can be obtained as $h_i = 1 - \dfrac{|y_i - (w\phi(x_i) + b)|}{(w\phi(|e_{xi}|) + d) - e_i} = H - \xi_i$. The meaning of this equation is equal to the work by Tanaka[15].

$$\min J(w, \xi) = \frac{1}{2}(w^T w + b^2 + d^2) + \frac{\gamma}{2}\sum_{i=1}^{n}(\xi_{1i}^2 + \xi_{2i}^2) \text{ Subject to:}$$

$$\begin{cases} w\phi(x_i) + b + (1-H)(c\phi(|x_i|) + d) + \xi_{1i} \\ \quad = y_i + (1-H)e_i \\ -(w\phi(x_i) + b) + (1-H)(w\phi(|e_{xi}|) + d) + \xi_{2i} \\ = -y_i + (1-H)e_i, \quad i = 1, 2, \cdots, n, \end{cases} \tag{1}$$

This optimization problem including constraints can be solved by using the Lagrange function as follows:

$$\begin{aligned} L(w, b, \xi, \alpha) &= \frac{1}{2}(w^T w + b^2 + d^2) \\ &+ \frac{\gamma}{2}\sum_{i=1}^{n}(\xi_{1i}^2 + \xi_{2i}^2) - \sum_{i=1}^{n}\alpha_{1i}[(w\phi(x_i) + b) \\ &+ (1-H)(w\phi(|e_{xi}|) + d) - y_i - (1-H)e_i) + \xi_{1i}] \\ &- \sum_{i=1}^{n}\alpha_{2i}[-(w\phi(x_i) + b) + (1-H)(w\phi(|e_{xi}|) + d) \\ &+ y_i - (1-H)e_i) + \xi_{1i}] \end{aligned} \tag{2}$$

where α_{1i}, α_{2i} are Lagrange multiplier sets. Through computing the partial derivatives of $L(w, \alpha_{1i}, \alpha_{2i}, b, d, \xi_{1i}, \xi_{2i})$, one can derive:

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N}(\alpha_{1i} - \alpha_{2i})\phi(x_i) + (\alpha_{1i} + \alpha_{2i})(1-H)\phi(|e_{xi}|)$$

$$\begin{aligned} &\frac{\partial L}{\partial \alpha_{2i}} = 0 \Rightarrow -(w\phi(x_i) + b) + (1-H)(w\phi(|e_{xi}|) + d) \\ &+ y_i - (1-H)e_i + \xi_{2i} = 0 \end{aligned} \qquad \frac{\partial L}{\partial b} = 0 \Rightarrow b = \sum_{i=1}^{n}(\alpha_{1i} + \alpha_{2i})$$

$$\frac{\partial L}{\partial d} = 0 \Rightarrow d = \sum_{i=1}^{n}(\alpha_{1i} + \alpha_{2i})(1-H) \tag{3}$$

$$\frac{\partial L}{\partial \xi_{1i}} = 0 \Rightarrow \alpha_{1i} = \gamma\xi_{1i}$$

$$\frac{\partial L}{\partial \xi_{2i}} = 0 \Rightarrow \alpha_{2i} = \gamma\xi_{2i}$$

Then, from (3), the parameters $\alpha_{1i}, \alpha_{2i}, b$ are the solution of the next matrix equation:

$$
\begin{bmatrix}
-1 & 1 & \cdots & 1 \\
1 & \begin{matrix}\phi(x_1)^T\phi(x_1)+(1-H)[\phi(x_1)^T\phi(|e_{x1}|) \\ +\phi(|e_{x1}|)^T\phi(x_1)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{x1}|)]+(1-H)^2+1/\gamma\end{matrix} & \cdots & \begin{matrix}\phi(x_1)^T\phi(x_n)+(1-H)[\phi(x_1)^T\phi(|e_{xn}|) \\ +\phi(|e_{x1}|)^T\phi(x_n)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix} \\
\vdots & \vdots & \ddots & \vdots \\
1 & \begin{matrix}\phi(x_n)^T\phi(x_1)+(1-H)[\phi(|x_n|)^T\phi(|e_{x1}|) \\ +\phi(|e_{xn}|)^T\phi(x_1)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & \cdots & \begin{matrix}\phi(x_n)^T\phi(x_n)+(1-H)[\phi(|x_n|)^T\phi(|e_{xn}|) \\ +\phi(|e_{xn}|)^T\phi(x_n)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{xn}|)]+(1-H)^2+1/\gamma\end{matrix} \\
-1 & \begin{matrix}-\phi(x_1)^T\phi(x_1)+(1-H)[-\phi(x_1)^T\phi(|e_{x1}|) \\ +\phi(|e_{x1}|)^T\phi(x_1)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & \cdots & \begin{matrix}-\phi(x_1)^T\phi(x_n)+(1-H)[-\phi(x_1)^T\phi(|e_{xn}|) \\ +\phi(|e_{x1}|)^T\phi(x_n)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix} \\
\vdots & \vdots & \cdots & \vdots \\
-1 & \begin{matrix}-\phi(x_n)^T\phi(x_1)+(1-H)[-\phi(|x_n|)^T\phi(|e_{x1}|) \\ +\phi(|e_{xn}|)^T\phi(x_1)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & & \begin{matrix}-\phi(x_n)^T\phi(x_n)+(1-H)[\phi(|x_n|)^T\phi(|e_{xn}|) \\ +\phi(|e_{xn}|)^T\phi(x_n)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix}
\end{bmatrix}
$$

$$
\begin{bmatrix}
1 & \cdots & 1 \\
\begin{matrix}-\phi(x_1)^T\phi(x_1)+(1-H)[\phi(x_1)^T\phi(|e_{x1}|) \\ -\phi(|e_{x1}|)^T\phi(x_1)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & \cdots & \begin{matrix}-\phi(x_1)^T\phi(x_n)+(1-H)[\phi(x_1)^T\phi(|e_{xn}|) \\ -\phi(|e_{x1}|)^T\phi(x_n)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix} \\
\vdots & \vdots & \vdots \\
\begin{matrix}-\phi(x_n)^T\phi(x_1)+(1-H)[\phi(x_n)^T\phi(|e_{x1}|) \\ -\phi(|e_{xn}|)^T\phi(x_1)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & \cdots & \begin{matrix}-\phi(x_n)^T\phi(x_n)+(1-H)[\phi(x_n)^T\phi(|e_{xn}|) \\ -\phi(|e_{xn}|)^T\phi(x_n)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix}
\end{bmatrix}
\begin{bmatrix} b \\ \alpha_{11} \\ \vdots \\ \alpha_{1n} \\ \alpha_{21} \\ \vdots \\ \alpha_{2n} \end{bmatrix}
=
\begin{bmatrix} 0 \\ y_1+(1-H)e_1 \\ \vdots \\ y_n+(1-H)e_n \\ -y_1+(1-H)e_1 \\ \vdots \\ -y_n+(1-H)e_n \end{bmatrix}
$$

$$
\begin{bmatrix}
\begin{matrix}\phi(x_1)^T\phi(x_1)+(1-H)[-\phi(x_1)^T\phi(|e_{x1}|) \\ -\phi(|e_{x1}|)^T\phi(x_1)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{x1}|)]+(1-H)^2+1/\gamma\end{matrix} & \cdots & \begin{matrix}\phi(x_1)^T\phi(x_n)+(1-H)[-\phi(x_1)^T\phi(|e_{xn}|) \\ -\phi(|e_{x1}|)^T\phi(x_n)+(1-H)\phi(|e_{x1}|)^T\phi(|e_{xn}|)]+(1-H)^2\end{matrix} \\
\vdots & \ddots & \vdots \\
\begin{matrix}\phi(x_n)^T\phi(x_1)+(1-H)[-\phi(x_n)^T\phi(|e_{x1}|) \\ -\phi(|e_{xn}|)^T\phi(x_1)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{x1}|)]+(1-H)^2\end{matrix} & \cdots & \begin{matrix}\phi(x_n)^T\phi(x_n)+(1-H)[-\phi(x_n)^T\phi(|e_{xn}|) \\ -\phi(|e_{xn}|)^T\phi(x_n)+(1-H)\phi(|e_{xn}|)^T\phi(|e_{xn}|)]+(1-H)^2+1/\gamma\end{matrix}
\end{bmatrix}
$$

(4)

$\alpha_{1i}, \alpha_{2i}, b$ can be solved by a single matrix inversion. Finally, the regularized least squares support vector regression model for fuzzy input-fuzzy output system can be obtained as follows:

$$
\begin{aligned}
&y(x) = W^T\phi(x)+B = (w^T\phi(x)+b, w\phi(|e_{xi}|)+d) \\
&= \sum_{i=1}^{n}(\alpha_{1i}-\alpha_{2i})k(x_i,x)+(\alpha_{1i}+\alpha_{2i})(1-H)k(|e_{xi}|,|x|)+(\alpha_{1i}+\alpha_{2i}), \qquad (5) \\
&\sum_{i=1}^{n}(\alpha_{1i}-\alpha_{2i})k(x_i,|e_x|)+(\alpha_{1i}+\alpha_{2i})(1-H)k(|e_{xi}|,|e_x|)+(\alpha_{1i}+\alpha_{2i})(1-H)
\end{aligned}
$$

3. Incremental Learning Regularized Least Squares Support Vector Fuzzy Regression

Assume samples $X_i, Y_i, i = 1\cdots n$. $X_i = (x_i, e_{xi}) \in T(R^l)$, $Y_i = (y_i, e_i) \in T(R)$. In incremental learning, the initial training samples can be obtained as following:

$$
X = \begin{bmatrix} (x_1, e_{x1}) \\ (x_2, e_{x2}) \\ \vdots \\ (x_{n-p}, e_{xn-p}) \end{bmatrix}, Y = \begin{bmatrix} (y_1, e_1) \\ (y_2, e_2) \\ \vdots \\ (y_{n-p}, e_{n-p}) \end{bmatrix} \qquad (6)
$$

At the initial moment, $\partial_{n-p} = (a_{11}, a_{12}, \cdots a_{1n-p}, a_{21}, a_{22}, \cdots a_{2n-p})^T, b_{n-p} = b$.

The Eq.(4) can be rewritten as following:

$$\begin{bmatrix} 0 & e1^T \\ e1 & U_{n-p} \end{bmatrix} \begin{bmatrix} b_{n-p} \\ a_{n-p} \end{bmatrix} = \begin{bmatrix} 0 \\ Y \end{bmatrix} \tag{7}$$

Assume $M_{n-p} = U_{n-p}^{-1}$, from Eq(4), it can be deduced:

$$b_{n-p} = \frac{e1^T M_{n-p} Y}{e1^T M_{n-p} e1}$$

$$\partial_{n-p} = M_{n-p}(Y - \frac{e1e1^T M_{n-p} Y}{e1^T M_{n-p} e1}) \tag{8}$$

new data points are obtained continuously. To track the dynamics of the system, the new data points should be included into training data. Then LS_SVM starts with $n-p+1$ training data points and U_{n-p+1} can be presented as following:

$$U_{n-p+1} = \begin{bmatrix} U_{n-p} & V_{n-p+1}^T \\ V_{n-p+1} & d_{n-p+1} \end{bmatrix}$$

$$d_{n-p+1} = \phi(x_n)^T \phi(x_n) + (1-H)[-\phi(x_n)^T \phi(|e_{xn}|)$$
$$-\phi(|e_{xn}|)^T \phi(x_n) + (1-H)\phi(|e_{xn}|)^T \phi(|e_{xn}|)] + (1-H)^2 + 1/\gamma$$

$$V_{n-p+1} = \begin{bmatrix} -\phi(x_1)^T \phi(x_{n-p+1}) + (1-H)[\phi(x_1)^T \phi(|e_{xn-p+1}|) \\ -\phi(|e_{x1}|)^T \phi(x_{n-p+1}) + (1-H)\phi(|e_{x1}|)^T \phi(|e_{xn-p+}|)] + (1-H)^2 \\ \vdots \\ -\phi(x_{n-p+})^T \phi(x_{n-p+}) + (1-H)[\phi(x_{n-p+})^T \phi(|e_{xn-p+}|) \\ -\phi(|e_{xn-p+}|)^T \phi(x_{n-p+}) + (1-H)\phi(|e_{xn-p+}|)^T \phi(|e_{xn-p+}|)] + (1-H)^2 \\ \phi(x_1)^T \phi(x_{n-p+}) + (1-H)[-\phi(x_1)^T \phi(|e_{xn-p+}|) \\ -\phi(|e_{x1}|)^T \phi(x_{n-p+}) + (1-H)\phi(|e_{x1}|)^T \phi(|e_{xn-p+}|)] + (1-H)^2 \\ \vdots \\ \phi(x_{n-p+})^T \phi(x_{n-p+}) + (1-H)[-\phi(x_{n-p+})^T \phi(|e_{xn-p+}|) \\ -\phi(|e_{xn-p+}|)^T \phi(x_{n-p+}) + (1-H)\phi(|e_{xn-p+}|)^T \phi(|e_{xn-p+}|)] + (1-H)^2 + 1/\gamma \end{bmatrix} \tag{9}$$

In order to compute ∂_{n-p+1} *and* b_{n-p+1}, U_{n-p+1}^{-1} must be calculated. But large-scale matrix inverse needs much time and memory to be calculated. the training of SVIRM can be placed in a way of incremental chunk avoiding computing large-scale matrix inverse but maintaining the precision. So the $M_{n-p+1} = U_{n-p+1}^{-1}$ should be deduced as following:

$$M_{n-p+1} = U_{n-p+1}^{-1} = \begin{bmatrix} U_{n-p} & V_{n-p+1}^T \\ V_{n-p+1} & d_{n-p+1} \end{bmatrix}^{-1}$$
$$= \begin{bmatrix} M_{n-p} & 0 \\ 0 & 0 \end{bmatrix} + F_{n-p+1} F_{n-p+1}^T W_{n-p+1} \tag{10}$$

Where $F_{n-p+1} = [v_{n-p+1}^T M_{n-p}, -1]^T$

$$W_{n-p+1} = \frac{1}{d_{n-p+1} - V_{n-p+1}^T M_{n-p} V_{n-p+1}} \tag{11}$$

From Eq.(10), M_{n-p+1} can be deduced without computing U_{n-p+1} inverse directly. Then b_{n-p+1} and ∂_{n-p+1} can be calculated through . This iteration process will save much time and memory.

4. EXPERIMENTS

In this section, the proposed incremental learning regularized least squares support vector fuzzy regression is applied to the fuzzy input- fuzzy output system. The fuzzy data is shown in Table I.

Table I: Fuzzy input-fuzzy output data

No.(i)	Fuzzy input (x_i, e_{xi})	Fuzzy output (y_i, e_i)
1	(1 ,0.5)	(-1.6 ,0.5)
2	(3 ,0.5)	(-1.8 ,0.5)
3	(4 ,0.5)	(-1 ,0.5)
4	(5.6 ,0.8)	(1.2 ,0.5)
5	(7.8 ,0.8)	(2.2 ,1.0)
6	(10.2 ,0.8)	(6.8 ,1.0)
7	(11 ,1.0)	(10 ,1.0)
8	(11.5 ,1.0)	(10 ,1.0)
9	(12.7 ,1.0)	(10 ,1.0)

From (5, the proposed linear fuzzy regression model is $y(x) = W^T \phi(x) + B = (w^T x + b, w|e_x| + d)$. The linear fuzzy regression model is $y(x) = W^T \phi(x) + B = (0.6191x + 0.5997, 0.6716|e_x| + 0.2998)$

Fig. 1 -Fig. 3 illustrate the results obtained by the proposed fuzzy regression with linear kernel function, polynomial kernel function and RBF kernel function. As seen from Fig. 1 one know that the nonlinear fuzzy regression model with RBF kernel is more appropriate for estimating fuzzy input-fuzzy output system. The experiment results in TableII . It confirms that the nonlinear fuzzy regression with RBF kernel function is very effective to estimate fuzzy input-fuzzy output system because the observations are all inside the estimation interval.

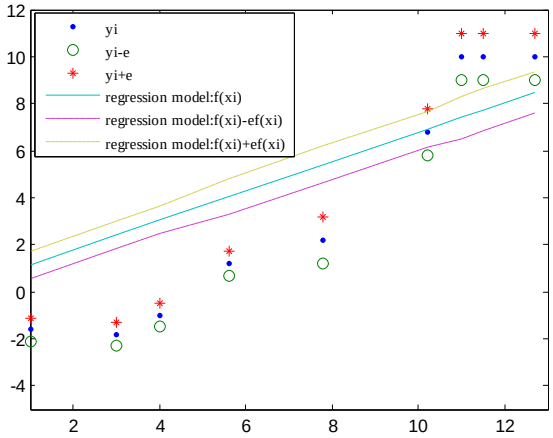

Fig.1 Linear regularized least squares support vector fuzzy regression model for fuzzy input-fuzzy output system

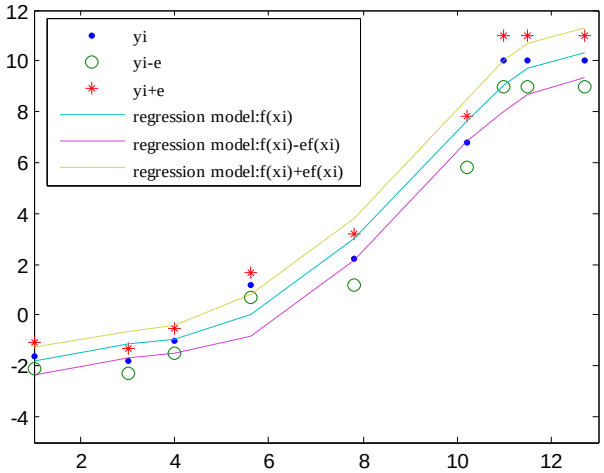

Fig.2 Nonlinear regularized least squares support vector fuzzy regression model with polynomial kernel fuzzy input-fuzzy output system

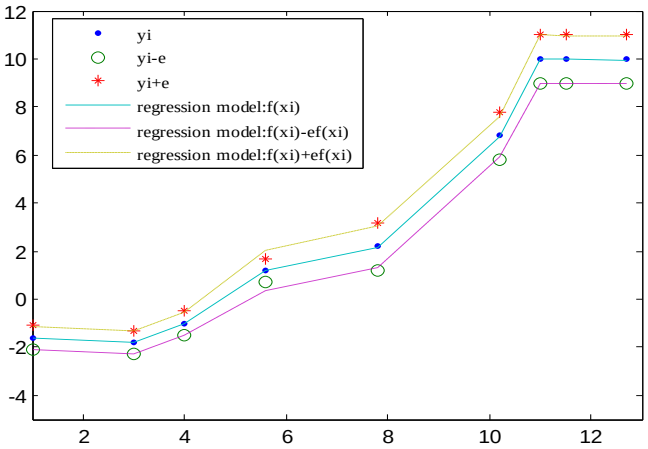

Fig.3 Nonlinear regularized least squares support vector fuzzy regression model with RBF kernel fuzzy input-fuzzy output system

Table II: COMPARISON RESULTS WITH VARIOUS KERNEL FUNCTION FOR FUZZY INPUT-FUZZY OUTPT SYSTEM

Method	Err_c	Err_u	Err_l
fuzzy regression(linear kernel)	2.8985	2.9747	2.8292
fuzzy regression (polynomial kernel)	0.6992	0.6048	0.8064
Fuzzy regression (RBF kernel)	0.024	0.1397	0.1375

5. Conclusions

In this note, a incremental learning least squares support vector fuzzy regression model is proposed to estimate fuzzy system. Compared with the previous fuzzy support vector regression approaches, the proposed fuzzy regression is achieved by a set of linear equations instead of by quadratic programming. These linear equations are calculated by increment learning. It reduces memory and calculates time. Numerical examples are given to demonstrate the effectiveness of the proposed model.

6. Acknowledgment

This work was supported by SRIP project of Ningbo University in 2011, natural science foundation of Ningbo(No. 2009A610074) , A Project Supported by Scientific Research Fund of Zhejiang Provincial Education Department (Y200803444).

7. References

[1] V. Vapnik, The Nature of Statistical Learning Theory[M].Springer, NewYork, 1995.

[2] V. Vapnik, Statistical Learning Theory[M]. Wiley, New York, 1998.

[3] C. Cortes, V. N. Vapnik, Support vector network[J]. Mach. Learn, 1995(20): 1–25.

[4] D. H. Hong, C. Hwang. Interval regression analysis using quadratic loss support vector machine[J]. IEEE Trans. Fuzzy Syst., 2005,13(4):229−237.

[5] Chih-Chia Yao, Pao-Ta Yu. Fuzzy regression based on asymmetric support vector machines[J]. Applied Mathematics and Computation. 2006,182:175−193.

[6] Zhili zhang, Hanggeng guo, Web Prediction Using Online Support Vector Machine, Proceedings of the17th IEEE international Conference on Tools with Artificial Intelligence