SOFT RBF NEURAL NETWORK MODELS AND AN SVM MODEL FOR WAGES TIME SERIES MODELING AND FORECASTING

Dusan Marcek

Faculty of Philosophy and Science, Silesian University, 746 01 0pava, Czech Republic & Faculty of Management Science and Informatics, University of Zilina
010 26 Zilina, Slovak Republic, dusan.marcek@fpf.slu.cz, dusan.marcek@fri.uniza.sk

Abstract

The wages time series are in fact stochastic in which successive observations are dependent and can be represented by a linear combination of independent random variables $\varepsilon_t, \varepsilon_{t-1}$, ... If the successive observations are highly dependent, we should use in model past values of the time series variable and (or) current and past values of the error terms $\{\varepsilon_t\}$. There are available techniques which are designed to exploit this dependency and which will generally produce superior forecasts. Many of these techniques are based on developments in time series analysis recently presented by Box and Jenkins. This article offers computational algorithm used in SVM method and in soft RBF neural networks and conducts experiments using these algorithms for wages time series modelling. Section 1 firstly describes the framework of SVM´s methods and support vector (SV) regressions and briefly describes construction of NNW for the comparison and verification SV regression wages forecast. Section 2 introduces the soft RBF neural network. In Section 3 the approximation and prediction abilities of SVM method is compared with the soft RBF NNW approach. A section of conclusions will close the paper.

Keywords:
RBF Neural Networks, Mean Square Error, SVM Method.

1 Support Vector Machine and NNW for Functional Approximation

The SV regression approach is based on defining a loss function that ignores errors that are within a certain distance of the true value. This type of function is referred to as an ε-insensitive loss function (see Fig. 2).

Fig. 1 shows an example of one dimensional function with an ε-insensitive band. The variables ξ, ξ^* measure the cost of the errors on the training points. These are zero for all points inside the ε-insensitive band, and only the points outside the ε-tube are penalized by the so called Vapnik´s ε-insensitive loss function.

In regression, there are different error (loss) functions in use and that each one results from a different final model. Fig. 2 shows the typical shapes of three loss functions [1], [4]. Left: quadratic 2-norm. Middle: absolute error 1-norm. Right: Vapnik´s ε-insensitive loss function.

Formally this results from solving the following Quadratic Programming problem

$$\min_{\mathbf{w},b,\xi,\xi^*} R(w,\xi,\xi^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \qquad (1)$$

$$\text{subject to } \begin{cases} y_i - \mathbf{w}^T\varphi(\mathbf{x}) - b \le \varepsilon + \xi_i & i = 1,2,...,n, \\ \mathbf{w}^T\varphi(\mathbf{x}) + b - y_i \le \varepsilon + \xi_i^* & i = 1,2,...,n, \\ \xi_i, \xi_i^* \ge 0 & i = 1,2,...,n \end{cases} \tag{2}$$

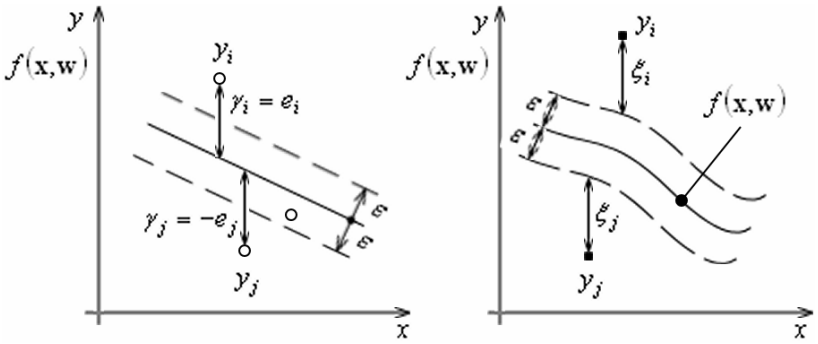

Figure 1: The insensitive band for one dimensional linear (left), non-linear (right) function

where *C* is the value of capacity degree

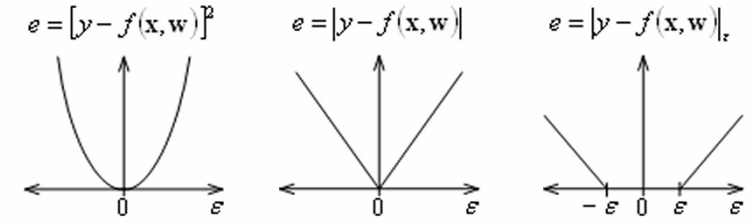

Figure 2: Error (loss) functions

To solve (1), (2) one constructs the Lagrangian

$$L_p(\mathbf{w}, b, \xi_i, \xi_i^*, \alpha_i, \alpha_i^*, \beta_i, \beta_i^*) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) - \sum_{i=1}^{n}\alpha_i\,(\varepsilon + \xi_i - y_i + \mathbf{w}^T\varphi(\mathbf{x}_i) + b)$$

$$- \sum_{i=1}^{n}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \mathbf{w}^T\varphi(\mathbf{x}_i) - b) - \sum_{i=1}^{n}(\beta_i\xi_i + \beta_i^*\xi_i^*) \tag{3}$$

by introducing Lagrange multipliers $\alpha_i, \alpha_i^* \ge 0$, $\xi_i, \xi_i^* \ge 0$, i = 1, 2, ..., n. The solution is given by the saddle point of the Lagrangian [2]

$$\max_{\alpha,\alpha_i^*,\beta_i,\beta_i^*} \min_{\mathbf{w},b,\xi_i,\xi_i^*} L_p(\mathbf{w}, b, \xi_i, \xi_i^*, \alpha_i, \alpha_i^*, \beta_i, \beta_i^*) \tag{4}$$

subject to constrains

$$\begin{cases} \dfrac{\partial L_p}{\partial w} = 0 & \rightarrow \mathbf{w} = \sum_{i=1}^{n} (\alpha_i^* - \alpha_i)\varphi(\mathbf{x}_i), \\ \dfrac{\partial L_p}{\partial b} = 0 & \rightarrow \sum_{i=1}^{n} (\alpha_i^* - \alpha_i) = 0, \\ \dfrac{\partial L_p}{\partial \xi_i} = 0, \dfrac{\partial L_p}{\partial \beta_i} = 0 & \rightarrow 0 \le \alpha_i \le C, \ \ i = 1,...,n, \\ \dfrac{\partial L_p}{\partial \xi_i^*} = 0, \dfrac{\partial L_p}{\partial \beta_i^*} = 0 & \rightarrow 0 \le \alpha_i^* \le C, \ \ i = 1,...,n, \end{cases} \tag{5}$$

which leads to the solution of the QP problem:

$$\max_{\alpha, \alpha_i^*} \ -\frac{1}{2} \sum_{i,j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\varphi(\mathbf{x}_i^T \mathbf{x}_j) \ - \varepsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*) \tag{6}$$

subject to (5).

After computing Lagrange multipliers α_i and α_i^*, one obtains of the form of (2), i.e.

$$f(\mathbf{x}) = \sum_{\substack{i=1 \\ j=1}}^{n} (\alpha_i - \alpha_i^*)\psi(\mathbf{x}_i, \mathbf{x}_j) + b \tag{7}$$

By substituting the first equality constraint of (5) in (1), one obtains the regression hyperplane as

$$f(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \varphi(\mathbf{x}) + b \tag{8}$$

Finally, b is computed by exploiting the Karush-Kuhn-Trucker (KKT) conditions [1]. These conditions state that at the optimal solution the product between the dual variables and constrains has to vanish. For the SVM´s, we obtain from, $\beta_i, \xi_i = 0$ and $\beta_i^*, \xi_i^* = 0$, i = 1, 2, ..., n that $(C - \alpha_i)\xi_i = 0$ and $(C - \alpha_i^*)\xi_i^* = 0$, respectively. Hence, $\xi_i = 0 \ \ (\xi_i^* = 0)$ if and only if $C = \alpha_i \ \ (C = \alpha_i^*)$. From this and KKT conditions, we obtain

$$\begin{cases} b = y_k - \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\psi(\mathbf{x}_i, \mathbf{x}_k) - \varepsilon & \text{for} \ \ \alpha_k \in (0, C), \\ b = y_k - \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)\psi(\mathbf{x}_i, \mathbf{x}_k) + \varepsilon & \text{for} \ \ \alpha_k^* \in (0, C). \end{cases} \tag{9}$$

When solving the QP-problem with an interior point method, it is more convenient to exploit primal-dual properties and to obtain b as a by-product of the optimization algorithm. Also note that for all samples inside the ε - band, α_i and α_i^* are zero. Therefore, we have a *sparse* expansion (2) of **w** in terms of $\mathbf{x}_i$. The samples that come with non-vanishing coefficients are called Support Vectors.

2 Soft RBF Neural Network Approach

The structure of soft RBF NNWs is defined by its architecture, its activation function and learning algorithm. In Fig 3 the its soft or fuzzy logic version is shown. The output layer neuron is linear and has a scalar output given by

$$\hat{y}_t = G(\mathbf{x}_t, \mathbf{c}, \mathbf{v}) = \sum_{j=1}^{s} v_{j,t} \frac{o_{j,t}}{\sum_{j=1}^{s} o_{j,t}} = \sum_{j=1}^{s} v_{j,t} \frac{\psi_2(x_t, c_j)}{\sum_{j=1}^{s} \psi_2(x_t, c_j)}, \ t = 1, 2, ..., N \tag{10}$$

where N is the size of data samples, s denotes the number of the hidden layer neurons, v_j are the trainable weights connecting the component of the output vector $\mathbf{o}$. The hidden layer neurons receive the Euclidian distances $(\|\mathbf{x} - \mathbf{c}_j\|)$ and compute the scalar values $o_{j,t}$ of the Gaussian function $\psi_2(\mathbf{x}_t, \mathbf{c}_j)$ that form the hidden layer output vector $\mathbf{o}_t$, where $\mathbf{x}_t$ is a k-dimensional neural input vector, $\mathbf{w}_j$ represents the hidden layer weights, ψ_2 are radial basis (Gaussian) activation functions. Note that for an RBF network, the hidden layer weights $\mathbf{w}_j$ represent the centres $\mathbf{c}_j$ of activation functions ψ_2.

The frequently used learning technique uses clustering to find a set of centres which more accurately reflect the distribution of the data points. For example by using K-means clustering algorithm, the member of K centres must be decided in advances. After choosing the centres **w**, the standard deviations σ_j can be selected as $\sigma_j \sim \Delta c_j$ where c_j denotes the average distance among the centres w_j. To train the weights v_j, the first-order gradient procedure is used. These weights can be adapted by the error back-propagation algorithm.

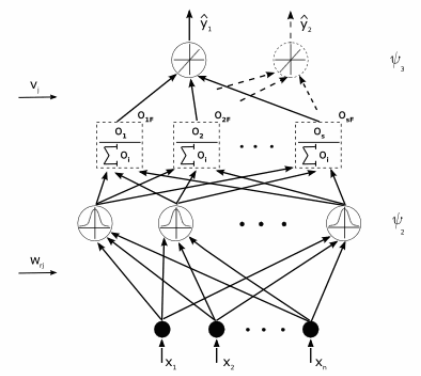

Fig. 3: Fully connected soft RBF neural network

3 Results from the Conducted Experiments

To learn the SV regression machine we used partly modified software developed in Matlab by Steve R. Gunn (http://www.isis.ecs.soton.ac.uk/resources/svminfo). One crucial design choice in constructing an SV machine is to decide on a kernel. Choosing of good kernels often requires lateral thinking: many measures of similarity between inputs have been developed in different contexts, and understanding which of them can provide good kernels depends on the insight into the application´s domains. Tab. 1 shows SVM´s learning of the historical period illustrating the actual and the fitted values by using various kernels.

Tab. 1 presents the results for finding the proper model by using the quantities MSEs. As shown in Tab. 1, the structural model that generates the "best" forecasts is the model with $MSE_E = 0.00117$.

Table 1: SV regression results of three different choice of the kernel on the training set (1991Q1-2007Q4). In two last columns the fit to the data and forecasting performance respectively are analyzed. See text for details.

SVM LEARNING MACHINE					
MODEL	KERNEL	σ	LOSS FUNCTION	MSE_A	MSE_E
causal	RBF	1.6	qadratic	0.0016	0.00117
causal	RBF	0.52	quadratic	10251	0.0018
causal	Exp. RBF	1	quadratic	3315	0.0038
time series	RBF	1	quadratic	0.421	none

In the soft (fuzzy logic) RBF neural network representation of model (7), the non-linear function was estimated according to the expressions (2) and (3). The fuzzy logic RBF neural network was extended towards estimation with (a priori known) noise levels of the entropy. We select, for practical reasons, that the noise level is a multiple, say 0.015 of entropy. Then, the non-linear input – output approximation function was estimated according to the formula (3) by substituting the Gaussian function $\psi_2(x_t, c_j)$ with Eq. (5). In Table 1, we give the achieved results of approximation ability in dependence on various number of RBF neurons. The mean square error (MSE_A) was used to measure the approximation ability.

The detailed computational algorithm for the MSE_A and MSE_E values in Table 2, the weight update rule for the soft network and detailed computational algorithm used for this type of neural networks are shown in [3]. The mean (centre) and standard deviation of clusters (RBF neurons) are computed using K-means algorithm. The accuracy of our one quarter forecasts is presented in Table 2.

Table 2: MSE_A and MSE_E results of the soft RBF NNW based model for different number hidden neurons

SOFT RBF NEURAL NETWORK		
Number of the hidden layer neurons	MSE_A	MSE_E
5	0.0671	0.000120
10	0.0670	0.000185

4 Conclusions

In this paper, we have examined the SVM´s and soft RBF NNW approach to study non-linear models on the time series of wages in the Slovak Republic. As it visually is clear from Table 1 and Table 2, too many model parameters results in overfitting, i.e. a curve fitted with too many parameters follows all the small fluctuations, but is poor for generalization.

Acknowledgement: This work was supported by Slovak grant foundation under the grant No. 1/0024/08 and from the Grant Agency of the Czech Republic under the grant No. 402/08/0022.

References

[1] CRISTIANINI, N., SHAWE-TAYLOR, J.: *An introduction to support vector machines*. Cambridge University Press, 2000

[2] FLETCHER, R.: *Practical methods of optimization*. John Wiley and Sons, Chichester and New York, 1987

[3] Marček, M., Marček, D.: *Granular RBF Neural Network Implementation of Fuzzy Systems: Application to Time Series Modeling*. Journal of Multi Valued Logic and Soft Computing, Vol. 12, pp. 101-114, Old City Publishing Product, Philadelphia, 2007

[4] VAPNIK, V.: *The nature of statistical learning theory*. Springer Verlag, New-York, 1995

Dusan Marcek
Faulty of Management Science and Informaticd, University of Žilina
Vysokoskolakov 1
010 26 Zilina
Slovak Republic