Journal of Korean
Data & Information Science Society
2002, Vol. 13, No.2 pp. 65 75

On Approximate Prediction Intervals for Support Vector Machine Regression

Kyungha Seok[1] , Changha Hwang[2] , Daehyeon Cho[3]

Abstract

The support vector machine(SVM), first developed by Vapnik and his group at AT&T Bell Laboratories, is being used as a new technique for regression and classification problems. In this paper we present an approach to estimating approximate prediction intervals for SVM regression based on posterior predictive densities. Furthermore, the method is illustrated with a data example.

1. Introduction

The support vector machine(SVM), first developed by Vapnik and his group at AT&T Bell Laboratories, is being used as a new technique for regression and classification problems. SVM is gaining popularity due to many attractive features, and promising empirical performance. SVM has been successfully applied to a number of real world problems such as handwritten character and digit recognition, face detection, text categorization and object detection in machine vision. The aforementioned applications relate to classification problems. but SVM is also widely applicable in regression problems. SVM was initially developed to solve classification problems, but recently it has been extended to the domain of regression problems. However, SVM classification can be viewed as a special case of SVM regression. For tutorial introductions and overviews of recent developments of SVM regression, see Gunn(1998), Smola & Schölkopf(1998), and

1. Associate Professor, Data Science Dept., Inje University, Kyungnam, 621-749, Korea.
E-mail : skh@stat.inje.ac.kr
2. Dept. of Statistical Information, Catholic University of Daegu, Kyungbuk, 712-702, Korea.
3. Dept. of Data Science, Inje University, Kyungnam, 621-749, Korea

Vapnik(1995, 1998).

SVM is based on the structural risk minimization(SRM) principle, which has been shown to be superior to traditional empirical risk minimization(ERM) principle. SRM minimizes an upper bound on the expected risk unlike ERM minimizing the error on the training data. By minimizing this bound, high generalization performance can be achieved. Based on this observation we believe that SVM regression will perform better when calculating prediction intervals than other methods such as neural networks and MARS. As we demonstrate, prediction intervals for SVM regression can be slightly difficult to obtain. De Veaux el. al(1998) computed prediction intervals for neural networks and compared them with prediction intervals based on MARS. See Bishop(1995) for neural networks and MARS.

Despite its successes in many real world applications, SVM depends only on a one weight solution represented indirectly by the set of Lagrangian multipliers in making predictions. However, according to a Bayesian perspective, the weights of SVM, even after learning, still take a certain posterior distribution. Thus, utilizing just a one weight solution as representative neglects posterior uncertainty in the weights. This often leads to more extreme predicted outputs during testing, and in turn indicates an overly high confidence. The appropriate way to handle these weight parameters is to utilize predictive posterior densities often used in Bayesian Statistics. Kwok(1999) used this idea to get the moderated outputs for SVM classification under the name of marginalization represented by MacKay(1992). In this paper we apply this idea to computing prediction intervals for SVM regression. Sollich(2000) guarantees that we can apply Bayesian methods to SVM.

The purpose of this paper is to present a Bayesian approach to estimating prediction intervals for SVM regression and to illustrate the method with an example using the same real data as in De Veaux el. al(1998). The rest of this paper is organized as follows. Section 2 gives an overview of SVM regression. Section 3 briefly reviews the relationship between the likelihood principle and SVM regression. Section 4 discusses how to compute prediction intervals for SVM regression, and in Section 5 the method is illustrated with a data example .

2. Support Vector Machine Regression

Let the training data set D be denoted by $\{(\mathbf{x}_i, y), i = 1, \ldots, n\}$, with each input $\mathbf{x}_i \in R^d$ and the output $y_i \in R$. Our goal is to find a function $f(\mathbf{x})$ that has at most ε deviation from the actually obtained targets y_i's for all the training data, and at the same time, is as flat as possible. Flatness here means that we seek small $\mathbf{w}$. We first consider the case of linear regression. Then, we take the form

$$f(x) = \mathbf{w}^t \mathbf{x} + b \quad \text{with} \quad \mathbf{w} \in R^d, b \in R$$

where superscript t represents the transpose of a vector. One way to ensure this is to minimize the Euclidean norm $||\mathbf{w}||^2$. Formally we can write this problem as a convex optimization problem by requiring:

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}||\mathbf{w}||^2, \\ &\text{subject to} \quad y_i - \mathbf{w}^t \mathbf{x}_i - b \le \varepsilon \text{ and } \mathbf{w}^t \mathbf{x}_i + b - y_i \le \varepsilon \end{aligned}$$

The underlying assumption here is that the convex optimization problem is feasible. Sometimes, however, this may not be the case, or we also may want to allow for some errors. To make it feasible, we introduce slack variables ξ_i and ξ_i^*. Hence we arrive at the formulation stated in Vapnik (1995, 1998).

$$\begin{aligned} &\text{minimize} \quad \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*), \\ &\text{subject to} \quad \begin{cases} y_i - \mathbf{w}^t \mathbf{x}_i - b \le \varepsilon + \xi_i \\ \mathbf{w}^t \mathbf{x}_i + b - y_i \le \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \ge 0 \end{cases} \end{aligned} \tag{1}$$

The constant $C > 0$ determines the trade off between the flatness of f and the amount up to which deviations larger than ε are tolerated. Here, ξ_i and ξ_i^* are slack variables representing upper and lower constraints on the outputs. The formulation above corresponds to dealing with Vapnik's ε-insensitive loss function described by

$$|\xi|_\varepsilon = \begin{cases} 0 & \text{if } |\xi| \le \varepsilon \\ |\xi| - \varepsilon & \text{otherwise} \end{cases}$$

The key idea is to construct a Lagrange function. Hence we proceed as follows:

$$\begin{aligned} L = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) - \sum_{i=1}^{n}\alpha_i(\varepsilon + \xi_i - y_i + \mathbf{w}^t \mathbf{x}_i + b) \\ - \sum_{i=1}^{n}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \mathbf{w}^t \mathbf{x}_i - b) - \sum_{i=1}^{n}(\eta_i\xi_i + \eta_i^*\xi_i^*) \end{aligned} \tag{2}$$

We notice that the positivity constraints $\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \ge 0$ should be satisfied. After taking partial derivatives of equation (2) with regard to the primal variables

($\mathbf{w}$, b, ξ_i, ξ_i^*) and plugging them into (2), we get the optimization problem below.

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

with constraints $\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$ and $\alpha_i, \alpha_i^* \in [0, C]$.

Solving the above equation with these constraints determines the Lagrange multipliers, α_i, α_i^*, and the optimal regression function is given by

$$\mathrm{w} = \sum_{i=1}^{n}(\alpha_i - \alpha'^*_i)\,\mathbf{x}_i, \quad b = -\frac{1}{2}\,\mathbf{w}^t\,[\,\mathbf{x}_r + \mathbf{x}_s],$$

where $\mathbf{x}_r$ and $\mathbf{x}_s$ are support vectors. Here, the support vectors are data points where exactly one of the Lagrange multipliers is greater than zero. Therefore, the optimal regression function can be rewritten in the form of

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\,\mathbf{x}_i^t\,\mathbf{x} + b.$$

Actually, b can be computed by using the Karush-Kuhn-Tucker(KKT) conditions. See for details Smola & Schölkopf(1998).

We now consider the case of nonlinear SVM regression. A nonlinear model is usually required to adequately model data. In this case SVM regression first maps $\mathbf{x}$ from the input space R^d to $\mathbf{z} = \psi(\mathbf{x})$ in a high dimensional feature space where linear regression performed. The kernel approach is employed to address the curse of dimensionality. The nonlinear SVM regression solution, using an ε-insensitive loss function, is given by

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

with constraints $\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$ and $\alpha_i, \alpha_i^* \in [0, C]$.

Solving the above equation with these constraints determines the Lagrange multipliers, α_i, α_i^*, and the optimal regression function is given by

$$f(\mathbf{x}) = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + b \qquad (3)$$

where

$$b = -\frac{1}{2}\sum_{i=1}^{n}(\alpha_i - \alpha_i^*)[K(\mathbf{x}_r, \mathbf{x}_i) + K(\mathbf{x}_s, \mathbf{x}_i)],$$

The difference to the linear case is that $\mathbf{w}$ is no longer explicitly given. However, it is uniquely defined in the weak sense by the dot products. Also note that in the nonlinear setting, the optimization problem corresponds to finding the flattest function in the feature space, not in the input space. In this paper we only consider the nonlinear case. Thus, for a test vector $\mathbf{x} \in R^m$, we first need to compute

$$a(\mathbf{x}, \mathbf{w}) = \mathbf{w}^t \mathbf{z} + b = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}) + b,$$

where

$$\mathbf{w}^t \mathbf{z} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)K(\mathbf{x}_i, \mathbf{x}).$$

3. Likelihood Principle and SVM Regression

In this section, we describe the relationship between the likelihood principle and SVM regression. In order to determine $\mathbf{w}$, we minimize (1), which, for a fixed C, is the same as minimizing

$$\frac{\|\mathbf{w}\|^2}{2C} + \sum_{i=1}^{n}(\xi_i + \xi_i^*). \tag{4}$$

We put $\lambda = 1/C$. Then, a model H, with a k-dimensional parameter vector $\mathbf{w}$, consists of its functional form f, the distribution $p(D \mid \mathbf{w}, H)$, that the model makes about the data D, and a prior parameter distribution $p(\mathbf{w} \mid \lambda, H)$ with a regularization parameter of λ. Therefore, we have the posterior distribution of w for a given value of λ by using Bayes' rule:

$$p(\mathbf{w} \mid D, \lambda, H) \propto p(\mathbf{w} \mid \lambda, H)\, p(D \mid \mathbf{w}, H). \tag{5}$$

Now, consider the following probability model:

- The prior over w is the normal prior

$$p(\mathbf{w} \mid \lambda, H) \propto exp\left(-\frac{\lambda}{2}\|\mathbf{w}\|^2\right).$$

• The probability distribution is given by

$$p(y_i \mid \mathbf{x}_i, \mathbf{w}, H) = \frac{1}{2(1+\varepsilon)} exp\left(-|\xi_i|_\varepsilon\right).$$

Note that $p(y_i \mid \mathbf{x}_i, \mathbf{w}, H)$ is actually the density model for an ε-insensitive loss function. Substituting these probabilities into (5) and assuming that the observations are i.i.d., we obtain

$$-\log p(\mathbf{w} \mid D, \lambda, H) = \frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n}(\xi_i + \xi_i^*) - \sum_{i=1}^{n}\log p(\mathbf{x}_i) + constant. \quad (6)$$

The last two terms on the right do not depend on $\mathbf{w}$. Hence, by putting $\lambda = 1/C$, optimizing (1) can be regarded as finding the *maximum apposteriori* (MAP) estimate $\mathbf{w}_{MP}$ of $\mathbf{w}$. Moreover, traditional SVM regression can be considered as using $\mathbf{w}_{MP}$ as the sole representative of the whole posterior distribution upon prediction.

4. Posterior Predictive Distribution and Prediction Intervals

To compute prediction interval for each output y, we need to derive the posterior predictive distribution $p(y \mid D, \mathbf{x})$ of y given the training data set D and an input vector $\mathbf{x}$. We first assume that the posterior distribution of $\mathbf{w}$ can be approximated by a single normal distribution at $\mathbf{w}_{MP}$. Since $a(\mathbf{x}, \mathbf{w}) = \mathbf{w}^t \mathbf{z} + b$, the posterior distribution of $a(\mathbf{x}, \mathbf{w})$ will also be normal $N(a_{MP}, s^2)$, with mean $a_{MP}(\mathbf{x}) = \mathbf{a}(\mathbf{x}, \mathbf{w}_{MP})$ and variance

$$s^2(\mathbf{x}) = \mathbf{z}^t \mathbf{A}^{-1} \mathbf{z}, \quad (7)$$

where $A = \nabla^2 M = \nabla^2\left(\frac{\lambda}{2}\|\mathbf{w}\|^2 + \sum_{i=1}^{n}(\xi_i + \xi'^*_i)\right)$ is the Hessian. Thus, we can get the posterior predictive density given below

$$\begin{aligned} p(y \mid D, \mathbf{x}) &= \int p(y \mid a, D, \mathbf{x})\, p(a \mid D, \mathbf{x})\, da \\ &= \frac{1}{2(1+\varepsilon)\sqrt{2\pi}s} \int e^{-|y-a|_\varepsilon} e^{-(a-a_{MP})^2/2s^2} da \\ &= \frac{1}{2(1+\varepsilon)} \left[e^{-y+\varepsilon+\frac{s^3+2a_{MP}}{2}} \Phi\left(\frac{y-\varepsilon-a_{MP}-s^2}{s}\right) \right. \end{aligned}$$

$$+ e^{y+\varepsilon+\frac{s^2-2a_{MP}}{2}}\left(1-\Phi\left(\frac{y-\varepsilon-a_{MP}-s^2}{s}\right)\right)$$
$$+\Phi\left(\frac{y+\varepsilon-a_{MP}}{s}\right)-\Phi\left(\frac{y-\varepsilon-a_{MP}}{s}\right)\Bigg],$$

where Φ is the distribution function of a standard normal distribution. A plot of the posterior predictive distribution is shown in Figure 1. This plot can be made after we train SVM regression for fixed values of C, ε, and a kernel parameter. If an interval summary is desired, a central interval of posterior predictive probability, which corresponds, in the case of a $100(1-a)\%$ interval, to the range of values above and below which lies exactly $100(a/2)\%$ of the posterior predictive probability can be calculated. Such interval estimates are referred to as prediction intervals.

Now, it is left to illustrate how to compute $s^2(\mathbf{x})$. To compute $s^2(\mathbf{x})$ in (7), we have to determine the Hessian A. We already know that computing the Hessian matrix for neural network is very complicated. But, we will see that it is much simpler to compute the Hessian matrix for SVM regression. We know that ξ_i, ξ'^{*}_i measures the difference between y_i and $\mathbf{w}^t \mathbf{z}_i + b$. Here, we use $\mathbf{z}_i$ for both linear and nonlinear cases. Thus, we have

$$\xi_i = step(y_i - a_i)(y_i - a_i - \varepsilon)$$
$$\xi_i^* = step(a_i - y_i)(a_i - y_i - \varepsilon)$$

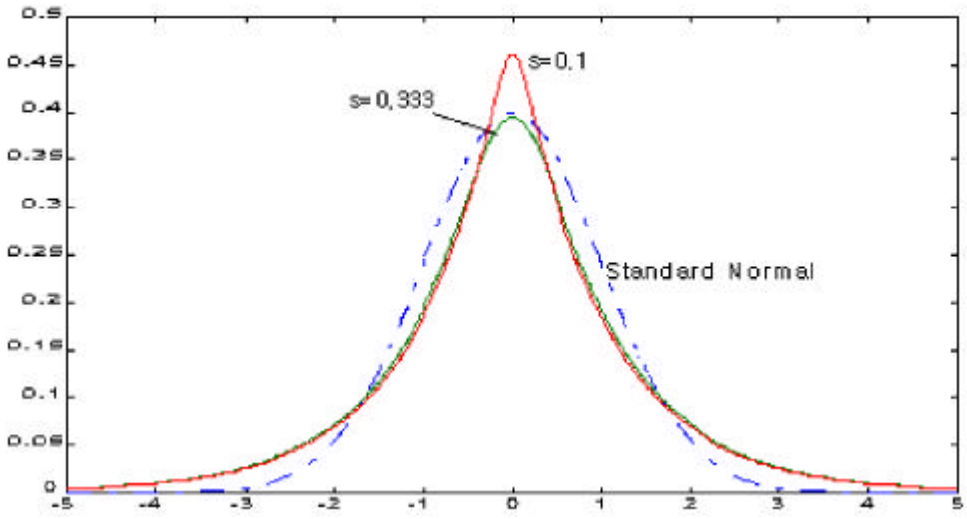


Figure 1: Poster predictive density and Standard normal density

where $a_i = \mathbf{w}^t \mathbf{z}_i + b$ and $step(x)$ is the step function. It should be noted that $step(x)$ is not differentiable. Thus, we replace it by the sigmoid function $\sigma(x) = 1/(1+e^{-\eta x})$. Since $\nabla a_i = \mathbf{z}_i$ and $\nabla^2 a_i = \mathbf{O}$, we obtain $\nabla^2 \xi_i = r(y_i - a_i)\,\mathbf{z}_i\,\mathbf{z}_i^t$ and $\nabla^2 \xi_i^* = r(a_i - y_i)\,\mathbf{z}_i\,\mathbf{z}_i^t$. Here, $\mathbf{O}$ is the zero matrix and $r(x) = (x - \varepsilon)\sigma''(x) + 2\sigma'(x)$. Finally, we have $\mathbf{A} = \lambda\mathbf{I} + \mathbf{B}$ where $\mathbf{B} = \sum_{i=1}^{n} r_i\,\mathbf{z}_i\,\mathbf{z}_i^t$ and $r_i \equiv r(|y_i - a_i|)$.

We now compute the eigenvalues ρ_k of $\mathbf{B}$. For the linear case, we put $\mathbf{z}_i = \mathbf{x}_i$.. Then, computing eigenvalues ρ_k is straightforward. However, we need something more for the nonlinear case. We now will explain in detail how to compute eigenvalues ρ_k for the nonlinear case. We know the training algorithm depends on some kernel function. We put $\mathbf{z}_i = \phi(\mathbf{x}_i)$ for some mapping ϕ. Let $\mathbf{v}_k$ be the eigenvectors of $\mathbf{B}$. Then we have $\mathbf{v}_k = \sum_{i=1}^{n} u_{ki}\,\mathbf{z}_i$ for some constants u_{ki} and $\rho_k \mathbf{z}_j^t \mathbf{v}_k = \mathbf{z}_j^t \mathbf{B}\,\mathbf{v}_k$.. This leads to $\rho_k \mathbf{K}\mathbf{u}_k = \mathbf{K}\widetilde{\mathbf{K}}\mathbf{u}_k$, where $\mathbf{u}_k = (u_{k1}, \ldots, u_{kn})^t$, $\mathbf{K}$ is the $n \times n$ matrix with elements $\mathbf{z}_i^t \mathbf{z}_j = K(\mathbf{x}_i, \mathbf{x}_j)$ and $\widetilde{\mathbf{K}}$ is another $n \times n$ matrix with elements $r_i \mathbf{z}_i^t \mathbf{z}_j = r_i K(\mathbf{x}_i, \mathbf{x}_j)$. In general, we can assume that $\mathbf{K}$ is invertible. Hence, we have $\rho_k \mathbf{u}_k = \widetilde{\mathbf{K}}\mathbf{u}_k$, and obtain the eigenvalues γ_k of $\mathbf{A}$ as $\gamma_k = \lambda + \rho_k$. Using these results, $\mathbf{A}$ can be approximated as

$$\mathrm{A} \simeq \mathrm{P}\,\mathrm{A}\,\mathrm{P}^t, \tag{8}$$

where

$$\mathbf{P} = (\mathbf{v}_1, \ldots, \mathbf{v}_m) = \left(\sum_{n=1}^{n} u_{1i}\,\mathbf{z}_i, \ldots, \sum_{i=1}^{n} u_{mi}\,\mathbf{z}_i\right), \tag{9}$$

and

$$\Lambda = diag\,(\lambda + \rho_1, \ldots, \lambda + \rho_m) \tag{10}$$

Here, the $\mathbf{v}_k$'s are orthogonal and m is the number of significant eigenvalues in the $n \times n$ matrix $\widetilde{\mathbf{K}}$.

Utilizing (8), (9), (10) and the fact that $\mathbf{P}^{-1} = \mathbf{P}^t$, $s^2(\mathbf{x})$ in (7) can then be computed as

$$s^2 = \mathbf{z}^t \mathbf{A}^{-1} \mathbf{z} = \mathbf{z}^t \mathbf{P}\,\Lambda^{-1}\,\mathbf{P}\,\mathbf{z} = (\mathbf{P}^t \mathbf{z})^t \Lambda^{-1} (\mathbf{P}^t \mathbf{z})$$

$$= \sum_{i=1}^{n} \frac{1}{\lambda + \rho_i} \left(\sum_{j=i}^{n} u_{ij} K(\mathbf{x}_j, \mathbf{x}) \right)^2 \tag{11}$$

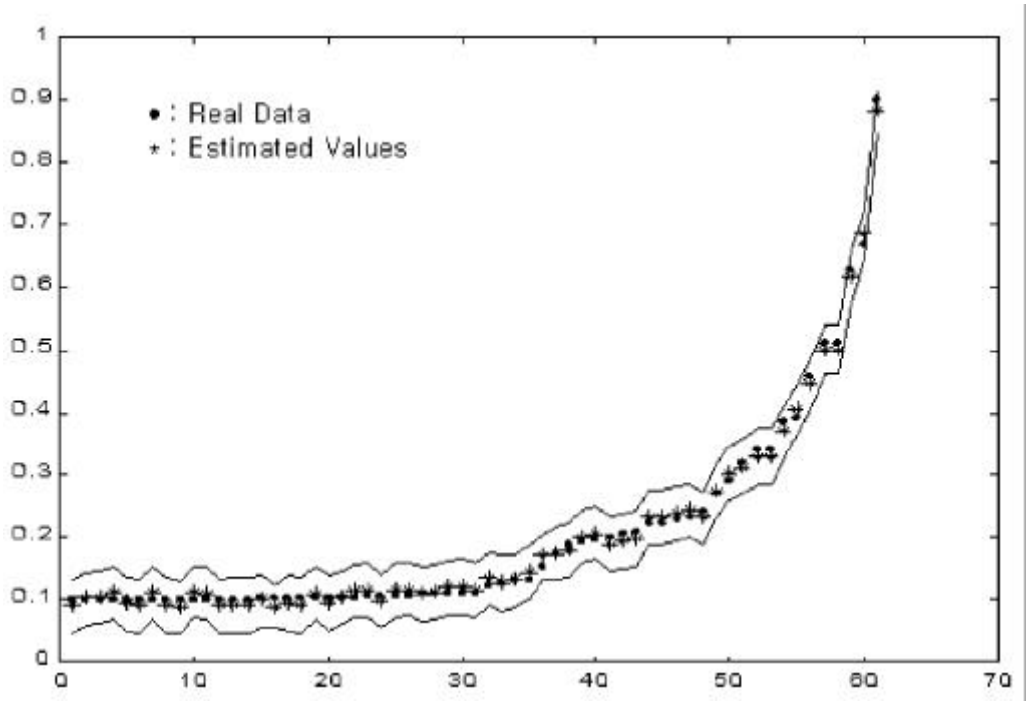
5. Data Example

As explained in Section 1, SVM regression is based on the SRM principle, which minimizes an upper bound on the expected risk, unlike ERM which minimizes the error on the training data. By minimizing this bound, SVM regression achieves high generalization performance. Thus, we can guess that SVM regression will perform better than prediction intervals based on other methods such as neural networks and MARS.

In this section we will illustrate how to compute approximate prediction intervals with the data set in De Veaux *et al.*(1993). For this data set De Veaux *et al.*(1998) computed prediction intervals for neural networks and compared them with prediction intervals based on MARS/GAM. They showed that the MARS/GAM fit appears to be a slightly smoother fit than the neural network model with correspondingly slightly smoother prediction intervals. From two figures for prediction intervals in this paper and De Veaux *et al.*(1998) we can presumably compare the performances of such methods. This data set is from a polymer process with 10 predictor variables $(x_1, \ldots, x_{10})$ and a single response variable y. Because the data are proprietary, no other information is available on the variables. The data set consist of 61observations and are available via ftp at ftp.cis.upenn.edu: pub/ungar/chemdata.

Figure2: Prediction Intervals for SVM Regression

Shao(1999) discusses several model selection criteria and finally argues the model selection criterion based on VC-dimension works best. However, in this paper we basically use cross validation method to determine model parameters since it generally works quite well. The Gaussian kernel $K(\mathbf{x}, \mathbf{y}) = \exp\{ \frac{-\|\mathbf{x} - \mathbf{y}\|^2}{2\sigma^2} \}$ is used in this experiment for SVM regression. For SVM regression C, ε and σ should be pre-specified. Here, we chose $C = 100$, $\varepsilon = 0.01$ and $\sigma = 0.541$. Based on our simulation studies, we found that SVM model is not sensitive to choices of C and ε that vary a little bit from our choices of $C = 100$ and $\varepsilon = 0.01$. The important parameter that does require careful consideration is the kernel parameter σ. To determine the kernel parameter, we performed 10-fold cross-validations. In each run, 50 points were used for training and the remaining 11 were used to compute the prediction sum of ε-insensitive losses. The best result on the prediction sum of ε-insensitive losses was obtained using $\sigma = 0.541$.



Two standard error prediction intervals for $\eta = 5{,}000$ are shown in Figure 2. Because there are multiple inputs, we have ordered the responses in this plot. As seen from Figure 2, the estimates of the prediction variance at almost every point seems to be stable and reasonable. From two figures for prediction intervals in this paper and De Veaux *et. al*(1998) we can presumably compare the performances of such methods. The SVM fit appears to be a much smoother fit than MARS/GAM and the neural network fit in De Veaux *et. al*(1998). In addition, SVM gives slightly smoother prediction intervals. The prediction intervals for SVM are somewhat narrower compared with the other two models in De Veaux *et. al*(1998). We checked the effect of varying η in the sigmoid function $\sigma(x)$ on the average size of the prediction intervals for SVM regression. When $\eta = 1{,}000$ the average size is 0.181 and decreases to 0.081 when $\eta = 5{,}000$ and further decreases to 0.057 when $\eta = 10{,}000$. The average size of the prediction intervals for the other two models is very close, 0.235 for the neural network and 0.253 for the GAM model. This experiment indicates that SVM compares favorably with other methods for regression modelling. To conclude, we recommend SVM as the technique for regression modelling.

References

1. Bishop, C.M. (1995). Neural Networks for Pattern Recognition, Oxford.
2. DeVeaux, R., Schumi, J., Schweinsberg, J., Shellington, D. and Ungar, L.H. (1998). Prediction Intervals for Neural Networks via Nonlinear Regression, Techno-metrics 40, 4, 273-282.
3. DeVeaux, R.D., Psichogios and Ungar, L.H. (1993). A comparison of two non-parametric estimation schemes: MARS and Neural Networks, Computers in Chemical Engineering, 17, 8, 819-837.
4. Gunn, S. (1998). Support Vector Machines for Classification and Regression, ISIS Technical Report, U. of Southampton.
5. Kwok, J.T. (1999). Moderating the Outputs of Support Vector Machine Classifiers, IEEE Transactions on Neural Networks 10, 5, 1018-1031.
6. MacKay, D.J.C. (1992). Bayesian Interpolation, Neural Computation 4, 3, 415-447
7. Shao, X. (1999). Model Selection Using Statistical Learning Theory, Ph. D. Thesis, U. of Minnesota.
8. Smola, A.J. and Schölkopf, B. (1998). A Tutorial on Support Vector Regression, NeuroCOLT2 Technical Report, NeuroCOLT.
9. Sollich, P. (2000). Probabilistic Methods for Support Vector Machines, Advances in Neural Information Processing Systems 12, to appear.
10. Vapnik, V. (1995). The Nature of Statistical Learning Theory, Springer.
11. Vapnik, V. (1998). Statistical Learning Theory, Springer.

[2002년 6월 접수, 2002년 9월 채택]