The Korean Communications
in Statistics Vol. 6, No. 3, 1999
pp. 813-819

Bayesian Model Selection for Support Vector Regression Using the Evidence Framework

Changha Hwang[1], Kyungha Seok[2]

Abstract

Support vector machine(SVM) is a new and very promising regression and classification technique developed by Vapnik and his group at AT&T Bell Laboratories. In this paper we provide a brief overview of SVM for regression. Furthermore, we describe Bayesian model selection based on MacKay's evidence framework for SVM regression.

1. Support Vector Regression

The foundations of SVM have been developed by Vapnik(1995) and are gaining popularity due to many attractive features, and promising empirical performance. In SVM there are two types, i.e., support vector classification(SVC) and support vector regression(SVR). SVM was developed to solve the classification problem, but recently it has been extended to the domain of regression problems. However, SVC can be viewed as a special case of SVR. This section gives an overview of SVR for readers new to this rapidly developing field of research. For more details, see Gunn(1998) and Smola & Scholkopf(1998). First, we review linear SVR.

Suppose we are given training data $\{(\boldsymbol{x}_i, y_i),\ i = 1, \cdots, n\} \subset \mathfrak{X} \times R$, where $\mathfrak{X}$ denotes the space of the input vectors, R^d. Our goal is to find a function $f(\boldsymbol{x})$ 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. We now take the form

$$f(\boldsymbol{x}) = \boldsymbol{w}^t \boldsymbol{x} + b \text{ with } \boldsymbol{w} \in \mathfrak{X},\ b \in R$$

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

$$\text{minimize } \frac{1}{2} \| \boldsymbol{w} \|^2,$$

$$\text{subject to } y_i - \boldsymbol{w}^t \boldsymbol{x}_i - b \leq \varepsilon \text{ and } \boldsymbol{w}^t \boldsymbol{x}_i + b - y_i \leq \varepsilon$$

The underlying assumption here is that the convex optimization problem is feasible.

1) Dept. of Statistical Information, Catholic University of Taegu-Hyosung, Kyungbuk, Korea.
2) Dept. of Data Science, Inje University, Kyungnam, Korea.

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, ξ_i^*. Hence we arrive at the formulation stated in Vapnik(1995, 1998).

$$\text{minimize} \quad \frac{1}{2} \| \boldsymbol{w} \|^2 + C \sum_{i=1}^{n} (\xi_i + \xi_i^*), \tag{1}$$

$$\text{subject to} \quad \begin{cases} y_i - \boldsymbol{w}^t \boldsymbol{x}_i - b \leq \varepsilon + \xi_i \\ \boldsymbol{w}^t \boldsymbol{x}_i + b - y_i \leq \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

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, ξ_i^* are slack variables representing upper and lower constraints on the outputs. The formulation above corresponds to dealing with a so called ε-insensitive loss function $| \xi |_\varepsilon$ described by

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

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

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

We notice that the positivity constraints $\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0$ should be satisfied. Hence we arrive at the optimization problem below.

$$\text{maximize} \quad -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) \boldsymbol{x}_i^t \boldsymbol{x}_j - \varepsilon \sum_{i=1}^{n} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*),$$

$$\text{subject to} \quad \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) = 0 \quad \text{and} \quad \alpha_i, \alpha_i^* \in [0, C]$$

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

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n} (\hat{\alpha}_i - \hat{\alpha}_i^*) \boldsymbol{x}_i, \quad \hat{b} = -\frac{1}{2} \hat{\boldsymbol{w}}^t [\boldsymbol{x}_r + \boldsymbol{x}_s],$$

where $\boldsymbol{x}_r$ and $\boldsymbol{x}_s$ are support vectors. Therefore, the optimal regression function can be rewritten in the form of

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} (\hat{\alpha}_i - \hat{\alpha}_i^*) \boldsymbol{x}_i^t \boldsymbol{x} + \hat{b}. \tag{2}$$

The Karush-Kuhn-Tucker(KKT) conditions that are satisfied by the solution are,

$$\hat{\alpha}_i \hat{\alpha}_i^* = 0, \quad i = 1, \cdots, n.$$

Hence the support vectors are points where exactly one of the Lagrange multipliers is greater than zero.

Next, we consider nonlinear SVR. How can the above methods be generalized to the case where the regression function is not a linear function of the data? A rather old trick can be used to accomplish this in an astonishingly starightforward way. First notice that the only way in which the data appears in the training problem is in the form of dot products $\boldsymbol{x}_i^t \boldsymbol{x}_j$. Now suppose we first mapped the data to some other (possibly infinite dimensional) Euclidean space E, using a mapping which we will call $\Phi: R^d \rightarrow E$.

Then the training algorithm would only depend on the data through dot products in E, i.e. on functions of the form $\Phi(\boldsymbol{x}_i)^t \Phi(\boldsymbol{x}_j)$. Now if there were a "kernel function" K such that $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \Phi(\boldsymbol{x}_i)^t \Phi(\boldsymbol{x}_j)$, we would only need to use K in the training algorithm, and would never need even to know explicitly what Φ is. The kernels often used are given below.

$$K(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x}^t \boldsymbol{y} + 1)^p, \quad K(\boldsymbol{x}, \boldsymbol{y}) = e^{-\frac{|\boldsymbol{x}-\boldsymbol{y}|^2}{2\sigma^2}}.$$

Here, p and σ^2 are kernel parameters. The kernel approach is again employed to address the curse of dimensionality. The nonlinear SVR solution, using an ε-insensitive loss function, is given by

$$\text{maximize} \quad -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(\boldsymbol{x}_i, \boldsymbol{x}_j) - \varepsilon\sum_{i=1}^{n}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{n} y_i(\alpha_i - \alpha_i^*),$$

$$\text{subject to} \quad \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \quad \text{and} \quad \alpha_i, \alpha_i^* \in [0, C]$$

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

$$\hat{\boldsymbol{w}} = \sum_{i=1}^{n}(\hat{\alpha}_i - \hat{\alpha}_i^*)\Phi(\boldsymbol{x}_i), \quad \hat{b} = -\frac{1}{2}\sum_{i=1}^{n}(\hat{\alpha}_i - \hat{\alpha}_i^*)[K(\boldsymbol{x}_r, \boldsymbol{x}_i) + K(\boldsymbol{x}_s, \boldsymbol{x}_i)]$$

Therefore, the optimal regression function can be rewritten in the form of

$$f(\boldsymbol{x}) = \sum_{i=1}^{n}(\hat{\alpha}_i - \hat{\alpha}_i^*)K(\boldsymbol{x}_i, \boldsymbol{x}) + \hat{b} \tag{3}$$

The difference to the linear case is that $\boldsymbol{w}$ 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 feature space, not in input space.

For regression problem, there are three parameters need to be tuned, the size ε of the insensitive zone, the constraint parameter C and kernel parameter. In SVR, it is known that kernel selection as well as the appropriate kernel parameters gives a significant effect on the SVR. Three parameters will affect the final model complexity, and therefore should be used in model selection. Model selection in SVM is still an open issue.

There are actually three situations for each coefficient α_i: $0 < \alpha_i < C$, $\alpha_i = 0$ or $\alpha_i = C$. In the last two situations, the coefficient hits the boundary and does not provide much information. In terms of free parameters, only when $0 < \alpha_i < C$ it counts as free.

2. The Evidence Framework

The evidence framework divides our inferences into three distinct levels of infrence. For details, see Bishop(1995) and MacKay(1992, 1995). A model H is assumed to have a m -dimensional parameter vector $\boldsymbol{w}$. This model is defined by its functional form and two probability distributions: the distribution $P(D \mid \boldsymbol{w}, \lambda, H)$ that the model makes about the data D when its parameters have a particular value $\boldsymbol{w}$; and a prior parameter distribution $P(\boldsymbol{w} \mid \lambda, H)$ which usually has the form given by $P(\boldsymbol{w} \mid \lambda, H) = \exp(-\lambda E_W(\boldsymbol{w}))/Z_W(\lambda)$, where λ is a regularization parameter.

2.1 Level 1 Inference

At the first level of inference, we assume that one model H is true and λ is given, and we infer what the model's parameters $\boldsymbol{w}$ might be given the data D. For a given value of λ, the posterior distribution of $\boldsymbol{w}$ is given by

$$P(\boldsymbol{w} \mid D, \lambda, H) = P(D \mid \boldsymbol{w}, \lambda, H) P(\boldsymbol{w} \mid \lambda, H)/P(D \mid \lambda, H)$$

Substituting in $P(\boldsymbol{w} \mid \lambda, H)$ above, it can be shown that finding the most probable estimate $\boldsymbol{w}_{MP}$ of $\boldsymbol{w}$ is the same as minimizing

$$M(\boldsymbol{w}) \equiv \lambda E_W(\boldsymbol{w}) - \log P(D \mid \boldsymbol{w}, \lambda, H). \tag{4}$$

2.2 Level 2 Inference

The second level of inference determines the value of λ by maximizing the posterior distribution of λ given by

$$P(\lambda \mid D, H) = P(D \mid \lambda, H) P(\lambda \mid H)/P(D \mid H).$$

When $P(\lambda \mid H)$ is flat, the evidence $P(D \mid \lambda, H)$ can be used to assign a preference to alternative values of λ. By approximating the posterior distribution of $\boldsymbol{w}$ by a single Gaussian at $\boldsymbol{w}_{MP}$, it can be shown that

$$\log P(D \mid \lambda, H) = -\lambda E_W^{MP} + E_D^{MP} - \frac{1}{2} \log det(\boldsymbol{A}) + \frac{m}{2} \log \lambda, \tag{5}$$

where $\boldsymbol{A} = \nabla^2 M$, $E_D(\boldsymbol{w}) \equiv \log P(D \mid \boldsymbol{w}, \lambda, H)$, and E_W^{MP} and E_D^{MP} are the values of E_W and E_D evaluated at $\boldsymbol{w}_{MP}$.

2.3 Level 3 Inference

At the third level of inference, we wish to infer which model is most plausible given the data. The posterior probability of each model is given by

$$P(H \mid D) \propto P(D \mid H)P(H).$$

The third level of inference ranks different models by examining their posterior probabilities $P(H \mid D)$. Assuming a flat $P(H)$ for all models, different models can be rated by their evidence $P(D \mid H)$. We now assume that the evidence maximum can be well approximated by a Gaussian. Then we have

$$P(D \mid H) = P(D \mid \lambda_{MP}, H)/\sqrt{\gamma},$$

where γ is the effective number of parameters.

3. Applying the Evidence Framework to SVR

In this section we describe Bayesian model selection based on the evidence framework. Namely, we illustrate that training of SVR can be considered as the level 1 inference of the evidence framework. Furthermore, we illustrate how levels 2 and 3 can be applied to SVR.

3.1 Level 1 Inference for SVR

The first level of inference determines $\boldsymbol{w}$. To determine $\boldsymbol{w}$, we minimize (1). For a fixed C, minimizing (1) is equivalent to minimizing

$$\frac{\| \boldsymbol{w} \|^2}{2C} + \sum_{i=1}^{n} (\xi_i + \xi_i^*), \tag{6}$$

We now use the probability model with the prior distribution $P(\boldsymbol{w} \mid C) \propto \exp(-\frac{\| \boldsymbol{w} \|^2}{2C})$ for $\boldsymbol{w}$ and the probability distribution $\exp(-\xi_i - \xi_i^*)$ for each pattern. Then the log data likelihood is $\log P(D \mid \boldsymbol{w}) = -\sum_{i=1}^{n} (\xi_i + \xi_i^*)$ by assuming that the patterns are i.i.d. Thus, putting $\lambda = 1/C$ leads (6) to the form of (4). In fact, we notice that minimizing (1) can be regarded as performing the level 1 of inference under this probability model.

3.2 Evaluation of the Hessian

We now compute the hessian matrix $\boldsymbol{A} = \nabla^2 M = \nabla^2(\lambda E_W + \sum_{i=1}^{n}(\xi_i + \xi_i^*))$. We know that computing the hessian matrix for neural network is very complicated. But, we will see it is much simpler to compute the hessian matrix for SVR than for neural network. We know that ξ_i, ξ_i^* measures the difference between y_i and $\boldsymbol{w}^t \boldsymbol{z}_i + b$. Here, we use $\boldsymbol{z}_i$ to cover both linear and nonlinear cases. Thus, we have

$$\begin{aligned} \xi_i &= step(y_i - \hat{y}_i)(y_i - \hat{y}_i - \varepsilon) \\ \xi_i^* &= step(\hat{y}_i - y_i)(\hat{y}_i - y_i - \varepsilon) \end{aligned}$$

where $\hat{y}_i \equiv \boldsymbol{w}^t \boldsymbol{z}_i + b$ and $step(x)$ is the step function. By the way, $step(x)$ is not differentiable. Thus, we approximate it by the sigmoid function $g(x) = 1/(1+e^{-x})$. Since $\nabla \hat{y}_i = \boldsymbol{z}_i$ and $\nabla^2 \hat{y}_i = \boldsymbol{O}$, we obtain $\nabla^2 \xi_i = d(y_i - \hat{y}_i)\boldsymbol{z}_i \boldsymbol{z}_i^t$ and $\nabla^2 \xi_i = d(\hat{y}_i - y_i)\boldsymbol{z}_i \boldsymbol{z}_i^t$. Here, $\boldsymbol{O}$ is zero matrix and $d(x) = (x-\varepsilon)g''(x) + 2g'(x)$. Finally, we have $\boldsymbol{A} = \lambda \boldsymbol{I} + \boldsymbol{B}$, where $\boldsymbol{B} = \sum_{i=1}^{n} d_i \boldsymbol{z}_i \boldsymbol{z}_i^t$ and $d_i \equiv d(|y_i - \hat{y}_i|)$.

We now compute the eigenvalues ρ_k of $\boldsymbol{B}$. For the linear case, we put $\boldsymbol{z}_i = \boldsymbol{x}_i$. Then, computing eigenvalues ρ_k is straightforward. However, we need something more for the nonlinear case. Thus, we will explain how to compute eigenvalues ρ_k for nonlinear case. We know the training algorithm depends on some kernel function. Thus, we should put $\boldsymbol{z}_i = \Phi(\boldsymbol{x}_i)$ for some Φ. Let $\boldsymbol{e}_k$ be the eigenvectors of $\boldsymbol{B}$. Then we have $\boldsymbol{e}_k = \sum_{i=1}^{n} u_{ki} \boldsymbol{z}_i$ for some constants u_{ki} and $\rho_k \boldsymbol{z}_j^t \boldsymbol{e}_k = \boldsymbol{z}_j^t \boldsymbol{B} \boldsymbol{e}_k$. This leads $\rho_k \boldsymbol{K} \boldsymbol{u}_k = \boldsymbol{K}\boldsymbol{K}^* \boldsymbol{u}_k$, where $\boldsymbol{u}_k = (u_{k1}, \cdots, u_{kn})^t$, $\boldsymbol{K}$ is the $n \times n$ matrix with elements $\boldsymbol{z}_i^t \boldsymbol{z}_j = K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ and $\boldsymbol{K}^*$ is another $n \times n$ matrix with elements $d_i \boldsymbol{z}_i^t \boldsymbol{z}_j = d_i K(\boldsymbol{x}_i, \boldsymbol{x}_j)$. In general, we can assume that $\boldsymbol{K}$ is invertible. Hence, we have $\rho_k \boldsymbol{u}_k = \boldsymbol{K}^* \boldsymbol{u}_k$, and obtain the eigenvalues $\{\gamma_k\}$ of $\boldsymbol{A}$ as $\gamma_k = \lambda + \rho_k$.

3.3 Level 2 and 3 Inference for SVR

At the level 2 of inference we determine the value of λ by maximizing $P(D \mid \lambda, H)$ in (5). Recall that $P(D \mid \lambda, H)$ is computed by approximating the posterior distribution of $\boldsymbol{w}$ by a single m-dimensional Gaussian at $\boldsymbol{w}_{MP}$. Here, m is usually very large, depending on the chosen kernel function. In this case, only a subspace of $\boldsymbol{w}$ will be affected by the data likelihood, and we argue that the Gaussian approximation is valid only in this subspace. Thus, we replace m in (5) by the number of significant eigenvalues, N in $\boldsymbol{K}^*$. Moreover, $det(\boldsymbol{A})$ in (5) can be readily computed given the eigenvalues γ_k of $\boldsymbol{A}$, and we get

$$\log P(D \mid \lambda, H) = -\lambda E_W^{MP} + E_D^{MP} - \frac{1}{2}\log \prod_{i=1}^{N} \gamma_i + \frac{N}{2}\log \lambda$$

At the level 3 of inference we compute $P(D \mid H)$. By the way, we know $P(D \mid H) = P(D \mid \lambda_{MP}, H)/\sqrt{\gamma}$. Thus, to obtain the model evidence $P(D \mid H)$, we have to calculate the effective number of parameters, γ. It can be written in terms of the eigenvalues of $\boldsymbol{B}$, ρ_i, where the subscript i runs over the N eigenvectors. The eigenvalues of $\boldsymbol{A}$ are $\lambda + \rho_i$, so we have:

$$\gamma = N - \lambda\, trace(\boldsymbol{A}^{-1}) = N - \sum_{i=1}^{N} \lambda/(\rho_i + \lambda) = \sum_{i=1}^{N} \rho_i/(\rho_i + \lambda).$$

4. Choosing the Kernel Parameter

This section discusses how to determine the kernel parameter h, for example polynomial kernel p. We usually use the model evidence $P(D \mid H)$ and mean square error to determine h. For fixed h, we compute model evidence and mean square error in the iterative manner as described above. We can choose h which maximizes model evidence and minimize mean square error. The evidence framework allows automatic adjustment of the kernel parameter to their near-optimal value.

References

[1] Bishop, C.M. (1995). Neural Networks for Pattern Recognition, Oxford.

[2] Gunn, S. (1998). Support Vector Machines for Classification and Regression, ISIS Technical Report, U. of Southampton.

[3] MacKay, D.J.C. (1992). Bayesian Interpolation, *Neural Computation* 4, 3,s 415-447.

[4] MacKay, D.J.C. (1995). Bayesian Methods for Neural Networks: Theory and Applications. Neural Networks Summer School.

[5] Shao, X. (1999). Model Selection Using Statistical Learning Theory, Ph. D. Thesis, U. of Minnesota.

[6] Smola, A.J. and Scholkopf, B. (1998). A Tutorial on Support Vector Regression, NeuroCOLT2 Technical Report, NeuroCOLT.

[7] Vapnik, V. (1995). The Nature of Statistical Learning Theory, Springer.

[8] Vapnik, V. (1998). Statistical Learning Theory, Springer.