

Weighted Solution Path Algorithm of Support Vector Regression for Abnormal Data

Wen-tao MAO Long-lei Dong Gang ZHANG

The Key Laboratory of Strength and Vibration of Ministry of Education

Xi'an Jiaotong University, Xi'an, China

maowt.mail@gmail.com dongll@mail.xjtu.edu.cn zidane029@gmail.com

Abstract

In the solution path algorithm of support vector regression, the penalty for violation of the required error is considered equally for every training sample, which means every training sample affects the generalization ability equally. Considering the existing of abnormal samples among the training data, for example noises with different variances, the weighted solution path algorithm of support vector regression is proposed. To reduce the negative effect of abnormal samples, different weighting coefficients are set on the error penalty parameter of corresponding samples. The whole solution path can be adjusted correspondingly. So the effects of abnormal samples on regression model have been reduced by setting lower coefficient. Experiments demonstrate that accuracy of prediction and the generalization of regression model can be improved.

1. Introduction

In ϵ -SVR, the ϵ -insensitive loss function

$$|y - f(x)|_\epsilon = \max\{0, |y - f(x)| - \epsilon\}$$

is used as an empirical risk functional[1]. The primal optimization problem for ϵ -SVR can be stated as follows:

$$\begin{aligned} \min \quad & \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - (\mathbf{w} \cdot \phi(\mathbf{x}_i) + w_0) \leq \epsilon + \xi_i \\ & (\mathbf{w} \cdot \phi(\mathbf{x}_i) + w_0) - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \quad i = 1, 2, \dots, l \end{aligned} \quad (1)$$

The $\lambda > 0$ is error penalty parameter that controls the degree of regularization and punishes the samples

which violate the ϵ -tube and incur loss. The $\epsilon \geq 0$ and λ have to be chosen a priori in advance by the users[1]. In practice, users often use the method of grid-search to determine the most optimal parameters, which requires re-training the model many times under different parameter settings.

More recently, the solution path algorithm has emerged that seeks to explore the entire solution path for all parameter values without having to re-train the model multiple times. The property of this algorithm is: any model with an L_1 regularization and a quadratic, piecewise quadratic, piecewise linear, or linear loss function has a piecewise linear coefficient path[2]. Based on this property, Zhu et al.[3] proposed an algorithm to compute the entire regularization path for the L_1 -norm SVC and Hastie et al.[4] proposed one for the standard L_2 -norm SVC. Gunter and Zhu[5] derived the entire solution path algorithm for ϵ -SVR with respect to λ with fixed ϵ . Wang et al.[6] proved that the solution path for ϵ -SVR is also piecewise linear with respect to ϵ and proposed one algorithm named ϵ -path.

However, the solution path algorithm has not good performance if there exist abnormal samples in training data. The abnormal data are often caused by noise in practice. The hypothesis of constructing primal problem is that samples follow the same distribution, that is, error of samples has the same variance independently. The same λ and ϵ for every sample mean the same demand of model precision and error penalty to every sample. In fact, there often exists heterogeneity of variance in practical applications. If we treat impartially to various samples, the regression model often could not have good prediction performance.

As pointed in [7, 8], weighting on regularization parameter λ can improve the prediction performance more greatly than ϵ . The ϵ -path algorithm explores the correspondence between every ϵ value and the corresponding solution $\alpha_i^{(*)}(\epsilon)$ for a fixed λ . So, we can get the bet-

ter regression model easily by weighting λ_i rather than ϵ_i of the corresponding abnormal sample \mathbf{x}_i in ϵ -path algorithm.

2. ϵ -Path algorithm

Applying the method of Lagrange multipliers to (1), the regression function can be obtained as:

$$f(\mathbf{x}) = \frac{1}{\lambda} \sum_{i=1}^n (\alpha_i - \alpha_i^*) K(\mathbf{x}_i, \mathbf{x}) + w_0 \quad (2)$$

From the Karush-Kuhn-Tucker(KKT) conditions, there is only a subset of data points for which either α_i or α_i^* are nonzero. These points are referred to as support vectors. According to the value of α_i and α_i^* , the whole set of data points could be partitioned into five subsets[6]:

$$\begin{aligned} R &= \{i : y_i - f(\mathbf{x}_i) > \epsilon, \alpha_i = 1, \alpha_i^* = 0\} \\ E_R &= \{i : y_i - f(\mathbf{x}_i) = \epsilon, \alpha_i \in [0, 1], \alpha_i^* = 0\} \\ E_C &= \{i : |y_i - f(\mathbf{x}_i)| < \epsilon, \alpha_i = 0, \alpha_i^* = 0\} \\ E_L &= \{i : y_i - f(\mathbf{x}_i) = -\epsilon, \alpha_i = 0, \alpha_i^* \in [0, 1]\} \\ L &= \{i : y_i - f(\mathbf{x}_i) < -\epsilon, \alpha_i = 0, \alpha_i^* = 1\} \end{aligned} \quad (3)$$

The algorithm starts from a big ϵ and decreases it gradually, which will shrink the ϵ -tube. Some events may occur during this process. An event is said to occur when a point enters or leaves an elbow, i.e. in $E_R \cup E_L$, causing some point sets to change. For the points inside or outside the tube, i.e. in $R \cup C \cup L$, their α_i and α_i^* values remain fixed until an event occurs. As a point passes through E_R or E_L , its α_i or α_i^* value will change from 0 to 1 or from 1 to 0. Hence, the algorithm only needs to focus on the points at the elbow, and monitor the change of α_i and α_i^* to fix the breakpoints in the path.[6]

As argued above, the path between two breakpoints can be computed easily. Let $\alpha_i^{(*)}$, α_0^l and λ^l denote the parameter values right after the l th event has occurred and $f^l(x)$ is the regression function at this point. For $\epsilon^{l+1} < \epsilon < \epsilon^l$, the algorithm only need to think of those points $i \in E_R \cup E_L$ whose $\alpha_i^{(*)}$ values change with λ , not fixing at 0 or 1 any more. The final linear equations are[6]:

$$\begin{aligned} \alpha_0 &= \alpha_0^l + \lambda(\epsilon^l - \epsilon)c_0 \\ \alpha_i &= \alpha_i^l + \lambda(\epsilon^l - \epsilon)c_i, \quad \forall i \in E_R^l \\ \alpha_j^* &= \alpha_j^{*l} + \lambda(\epsilon^l - \epsilon)c_j, \quad \forall j \in E_L^l \end{aligned} \quad (4)$$

where $\mathbf{c} = A_l^{-1} \mathbf{1}^a$, $A_l = \begin{pmatrix} 0 & \mathbf{1}^T \\ \mathbf{1} & K_l \end{pmatrix}$. Hence, $\alpha_i^{(*)}$ ($\forall i \in E_R^l \cup E_L^l$) proceeds linearly with ϵ , which

cause the solution to be computed quickly.

Then Wang[6] obtained the regression function as

$$\begin{aligned} f(\mathbf{x}) &= (\epsilon^l - \epsilon)h^l(\mathbf{x}) + f^l(\mathbf{x}) \\ h^l(\mathbf{x}) &= \sum_{i \in E_R^l} c_i K(\mathbf{x}_i, \mathbf{x}) - \sum_{j \in E_L^l} c_j K(\mathbf{x}_j, \mathbf{x}) + c_0 \end{aligned} \quad (5)$$

As ϵ decreases, the algorithm monitors the occurrence of two events:

- The α_i or α_i^* value of a point $i \in E_R^l \cup E_L^l$ reaches 0 or 1. Then the value of ϵ can be calculated from (4).
- A point $i \in R^l \cup C^l \cup L^l$ hits an elbow, i.e. $|y_i - f(x_i)| = \epsilon$. Then the value of ϵ for which this event occurs can be calculated from (5).

The largest $\epsilon < \epsilon^l$ is assigned as ϵ^{l+1} and the point sets (3) are updated accordingly. After multiple iteration steps along this process, the entire solution path could be obtained. Wang[6] gave the detailed derivation of this algorithm.

3. Weighted ϵ -Path algorithm

In ϵ -Path algorithm, λ could be initialized independently. But once λ has been fixed, the regression model will give the same penalty to every sample when violation of ϵ -tube occurs. In practice, there are many reasons, for example noise, to bring forth abnormal samples. In geometry, the regression line will be close to the samples with bigger variance rather than lower variance. Therefore, the λ according to different samples should be weighted for higher generalization.

From the primal optimization problem (1), we have λ weighted:

$$\begin{aligned} \min &= \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n s_i (\xi_i + \xi_i^*) \\ \text{s.t.} \quad & y_i - (\mathbf{w} \cdot \phi(\mathbf{x}_i) + w_0) \leq \epsilon + \xi_i \\ & (\mathbf{w} \cdot \phi(\mathbf{x}_i) + w_0) - y_i \leq \epsilon + \xi_i^* \\ & \xi_i, \xi_i^* \geq 0, \quad i = 1, 2, \dots, l \end{aligned}$$

where s_i is weighting coefficient of λ for i th sample. Then we get

$$\begin{aligned} \max_{\alpha, \alpha^*, \beta, \beta^*} \min_{\mathbf{w}, w_0} L_p &= \frac{\lambda}{2} \|\mathbf{w}\|^2 + \sum_{i=1}^n s_i (\xi_i + \xi_i^*) \\ &- \sum_{i=1}^n \alpha_i (\epsilon + \xi_i - y_i + \mathbf{w} \cdot \phi(\mathbf{x}_i) + w_0) \\ &- \sum_{i=1}^n \alpha_i^* (\epsilon + \xi_i^* + y_i - \mathbf{w} \cdot \phi(\mathbf{x}_i) - w_0) \\ &- \sum_{i=1}^n (\beta_i \xi_i + \beta_i^* \xi_i^*) \end{aligned}$$

After partial derivative, we have

$$\begin{aligned}\frac{\partial L_p}{\partial \mathbf{w}} = 0 &\Rightarrow \mathbf{w} = \frac{1}{\lambda} \sum_{i=1}^n (\alpha_i - \alpha_i^*) \phi(x_i) \\ \frac{\partial L_p}{\partial w_0} = 0 &\Rightarrow \sum_{i=1}^n (\alpha_i - \alpha_i^*) = 0 \\ \frac{\partial L_p}{\partial \xi_i^{(*)}} = 0 &\Rightarrow 0 \leq \alpha_i, \alpha_i^* \leq s_i\end{aligned}\quad (6)$$

As a consequence of (6), α_i can be adjusted by s_i . From (2), we can see that the bigger the absolute value of α_i is, the greater the corresponding role of sample in the regression model is. So the effect of generalization can be modified by adjusting the value of s_i .

From (6), we need to modify the ϵ -Path algorithm locally. Redefine the set of (3):

$$\begin{aligned}R &= \{i : y_i - f(\mathbf{x}_i) > \epsilon, \alpha_i = s_i, \alpha_i^* = 0\} \\ E_R &= \{i : y_i - f(\mathbf{x}_i) = \epsilon, \alpha_i \in [0, s_i], \alpha_i^* = 0\} \\ E_C &= \{i : |y_i - f(\mathbf{x}_i)| < \epsilon, \alpha_i = 0, \alpha_i^* = 0\} \\ E_L &= \{i : y_i - f(\mathbf{x}_i) = -\epsilon, \alpha_i = 0, \alpha_i^* \in [0, s_i]\} \\ L &= \{i : y_i - f(\mathbf{x}_i) < -\epsilon, \alpha_i = 0, \alpha_i^* = s_i\}\end{aligned}\quad (7)$$

Because the coefficient s_i is decided in advance, it suffices to focus on the points at the elbows, i.e., in $E_R \cup E_L$ as before. Therefore, (4) and (5) will be in effect on the samples in $E_R \cup E_L$.

When ϵ decreases gradually, samples will move among five sets in (7), and two events will occur during this process. When the samples in $E_R \cup E_L$ enter other sets, i.e., in $R^l \cup C^l \cup L^l, \alpha_i^{(*)}$ will become 0 or s_i . Therefore, the equations become

$$\begin{aligned}\epsilon &= (\alpha_i^l - s_i + \lambda \epsilon^l c_i) / (\lambda c_i) \quad \text{when samples enter } R \\ \epsilon &= (s_i - \alpha_i^{*l} + \lambda \epsilon^l c_i) / (\lambda c_i) \quad \text{when samples enter } L\end{aligned}$$

When samples in other sets enter $E_R \cup E_L$, the range of $\lambda_i^{(*)}$ will change to $[0, s_i]$ from 0 or s_i , which satisfies $|y_i - f(x_i)| = \epsilon$. Under such conditions, ϵ can be computed by:

$$\begin{aligned}\epsilon &= (y_i - \epsilon^l h^l(\mathbf{x}) - f^l(\mathbf{x})) / (1 - h^l(\mathbf{x})) \\ &\quad \text{when samples enter } E_R \\ \epsilon &= (y_i + \epsilon^l h^l(\mathbf{x}) + f^l(\mathbf{x})) / (1 + h^l(\mathbf{x})) \\ &\quad \text{when samples enter } E_L\end{aligned}$$

We should assign the largest $\epsilon < \epsilon^l$ as ϵ^{l+1} and update the point sets (7) at last. The process of iteration is similar to the ϵ -Path algorithm.

It is worth noting that the variance of noise is commonly unknown in advance in practice. It needs to be decided by users according to the actual condition. There are many methods to detect the heterogeneity of variance, for example, Residual Plot and Rank Correlation Coefficient.

4. Experimental Results

We randomly generate a set of 160 data points $\{(x_i, y_i)\}$ as training set with x_i drawn uniformly from $[-4, +4]$ and $y_i = \sin(x_i)/x_i + e_i$, where e_i is a Gaussian noise term, with zero mean and a variance of 0.8 for the first N samples and a variance 0.1 for the others. We set N to be different values to test the performance of our proposed algorithm under different conditions. We randomly select other 40 data points as test set. The Gaussian RBF kernel $K(x, y) = \exp(-\|x - y\|^2/2\sigma)$ is used. The σ is set to be 0.2. To avoid over-fitting problem, the value of λ should not be set very small. Here λ is set to be 0.01. From [6], the algorithm will terminate when 70% of the points become support vectors. For each value of N, we compute the mean squared error(MSE): $\sqrt{\frac{1}{l} \sum_{i=1}^l |\hat{y}_i - y_i|^2}$ on the test set, where \hat{y}_i is the value of prediction, $y_i = \sin(x_i)/x_i$ with no noise. The weighting coefficients of first N samples are set to be s_i , while others' be 1.

Figure 1 shows the effect of different values of s_i

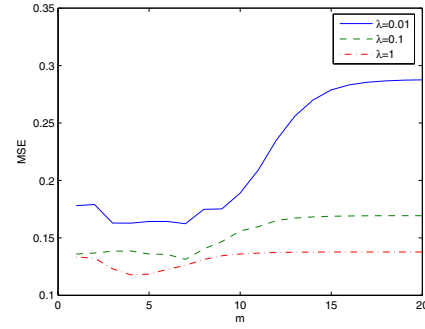


Figure 1. Relationships between MSE and weighting coefficient(represented by m)

on the weighted ϵ -Path algorithm, where s_i is set to be exponent of 1/2, i.e. $s_i = (1/2)^m$, and N is set to be 40. The value of MSE decreases when m increases during the first steps. But when m increases rapidly after some values, it always tends to give a curve leading to large MSE. This is mainly because that the effects of weighting coefficient are achieved through adjusting value of α of corresponding samples. The large value of m generating small weighting coefficient means α becomes smaller, which leads to little effect of corresponding samples to construct regression function. It will result in imperfect regression model.

Figure 2 shows the effect of the weighted ϵ -Path algorithm for different λ with different N. We take 14 values of N from 10 to 140 with the interval of 10. The effect of ϵ -Path algorithm is also plotted. In fact, the ϵ -

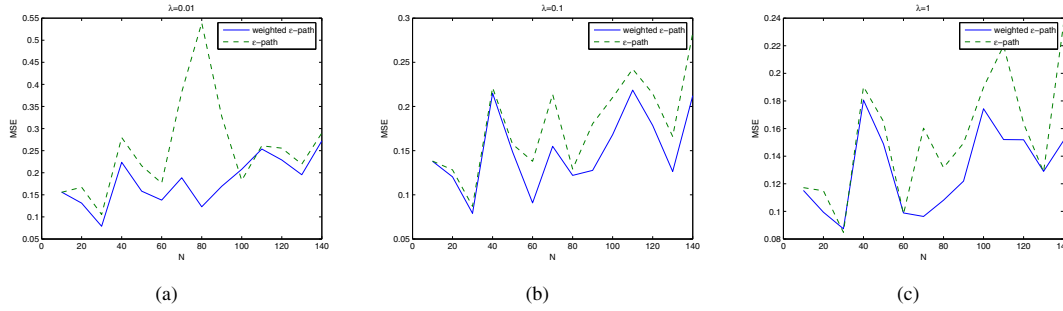


Figure 2. Relationships between MSE and N of two ϵ -Path algorithms for different values of λ (a) $\lambda=0.01$;(b) $\lambda=0.1$;(c) $\lambda=1$;

Path algorithm is special case that the weighting coefficient always takes value of 1. It can be seen obviously that the weighted ϵ -Path algorithm could improve the accuracy of prediction under different conditions.

We next examine the relationships between the

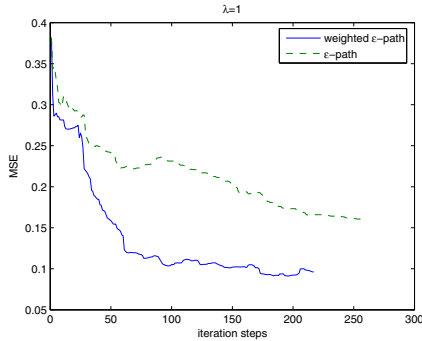


Figure 3. Relationships between MSE and Number of iteration steps in weighted ϵ -Path algorithm and ϵ -Path algorithm

MSE and the number of iteration steps of both weighted ϵ -Path algorithm and ϵ -Path algorithm, where $N=70$, $m=4$ and $\lambda=1$. The MSE decreases rapidly during the first few steps. The figure shows two ϵ -Path algorithms give similar MSE curves as ϵ decreases which causes the iteration. However, the performance of weighted ϵ -Path algorithm is always better than ϵ -Path algorithm.

5. Conclusion

In this paper, we have proposed an efficient solution path algorithm to solve the regression problem existing abnormal samples among the training data. In ϵ -Path algorithm, the solution path is the function of ϵ , so we take the regularization parameter λ weighted according to abnormal samples. The effect of weighting coefficient

can be achieved through adjusting value of α of corresponding samples in the regression model, so more accurate model can be obtained.

References

- [1] V N Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.
- [2] S Rosset and J Zhu. Piecewise linear regularized solution paths. *Annals of statistics*, 35(3):1012–1030, 2007.
- [3] J Zhu, S Rosset, and T Hastie. 1-norm support vector machines. In *Advances in Neural Information Processing Systems(NIPS) 16*, pages 49–56. Cambridge: MIT Press, 2004.
- [4] T Hastie, S Rosset, and R Tibshirani. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5:1391–1415, 2004.
- [5] L Gunter and J Zhu. Efficient computation and model selection for the support vector regression. *Neural Computation*, 19(6):1633–1655, 2007.
- [6] G Wang, D Y Yeung, and F H Lochovsky. Two dimensional solution path for support vector regression. In *Proceedings of the 23rd International Conference on Machine Learning(ICML)*, pages 993–1000. ACM Press, 2006.
- [7] F E H TOY and L J CAO. Modified support vector machines in financial time series forecasting. *Neurocomputing*, 48:847–861, 2002.
- [8] C F LIN and S D WANG. Fuzzy support vector machines. *IEEE Trans on Neural Networks*, 13(2):464–471, 2002.