Foundation of Machine Learning (IT 582)

Autumn 2022

Pritam Anand

L_1-norm loss kernel regression model

For the given training set $T = \{(x_i, y_i) : x_i \in \mathbf{R}^n, y_i \in \mathbf{R}, i = 1, 2, ..., l\}$, the kernel regression model estimates $f(x) = \sum\limits_{i=1}^{l} k(x_i, x)\alpha_i + b$. The L_1-norm loss function based kernel regression model solves the optimization problem

$$\min_{(\alpha,b)} \; J(\alpha, b) = \frac{\lambda}{2}\alpha^T\alpha + \frac{1}{2}\sum_{i=1}^{l}\Big|y_i - \Big(\sum_{j=1}^{l} k(x_j, x_i)\alpha_j + b\Big)\Big| \tag{1}$$

We can realize that $J(\alpha, b)$ is a convex but, not a smooth function of α and b. For given data point (x_k, y_k), the L_1-norm loss can be given using

$$g(x_k, y_k) = \Big|y_k - \Big(\sum_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big)\Big| \tag{2}$$

We can obtain the sub-gradients for $g(x_k, y_k)$ as follows.

$$\delta_\alpha g(x_k, y_k) = \begin{cases} -\begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix}, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) > 0. \\ \begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix}, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) < 0. \\ r\begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix}, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) = 0, \text{ where } r \in (-1, 1). \end{cases}$$

$$\delta_b g(x_k, y_k) = \begin{cases} -1, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) > 0. \\ 1, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) < 0. \\ r, \text{ if } \; y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b\Big) = 0, \text{where } r \in (-1, 1). \end{cases}$$

After computing the sub gradients for $g(x_k, y_k)$, we can easily obtain the sub gradients for $J(\alpha, b)$ as

$$\delta_\alpha(J(\alpha,b)) = \lambda\alpha + \frac{1}{2}\sum_{i=1}^{l}\delta_\alpha g(x_i,y_i)$$

$$\delta_b(J(\alpha,b)) = \frac{1}{2}\sum_{i=1}^{l}\delta_b g(x_i,y_i)$$

Now, we present the steps of the sub gradient descent algorithm which can obtain the solution for the L_1-norm loss regression model (1).

Algorithm 1 Sub gradient method for L_1-norm loss kernel regression model

Input :- Training set T, λ, tolerance tol, kernel parameter (if any).
Intailize $\alpha^0 \in \mathbb{R}^l$, $b^0 \in \mathbb{R}$
Repeat
$\alpha^{k+1} = \alpha^k$ - $\eta(\delta_\alpha(J(\alpha^k, b^k)))$
$b^{k+1} = b^k$ - $\eta(\delta_b(J(\alpha^k, b^k)))$
until $|| \begin{bmatrix} \delta_w(J(\alpha^k,b^k)) \\ \delta_b(J(\alpha^k,b^k)) \end{bmatrix} || \leq tol.$

ϵ-Support Vector Regression model

The L_1-norm kernel regression model is a robust regression model but, it fails to obtain the sparse solution vector. Vapnik et al., have proposed the ϵ- Support Vector Regression (ϵ-SVR) [1] [2] [3] model which can obtain robust as well as sparse solution. The ϵ-insensitive loss function is given by

$$|u|_\epsilon = \max\ (|u| - \epsilon, 0) = \begin{cases} 0, & \text{if } |u| \leq \epsilon. \\ |u| - \epsilon, & \text{otherwise.} \end{cases} \tag{3}$$

The ϵ-insensitive loss function has been plotted in Figure (1). It can tolerate the error up to ϵ. For regression problem, the ϵ-insensitive loss can be given by

$$|((y_i - f(x_i)))|_\epsilon = \begin{cases} 0, & \text{if } |(y_i - f(x_i))| \leq \epsilon. \\ |(y_i - f(x_i))| - \epsilon, & \text{otherwise.} \end{cases} \tag{4}$$

The ϵ-insensitive loss function allows the estimated function $f(x)$ to deviate from the response y up to ϵ.

The ϵ-SVR model minimizes the ϵ-insensitive loss function along with L_2-norm regularization. It solves the problem

$$\min_{(\alpha,b)}\ J(\alpha,b) = \frac{\lambda}{2}\alpha^T\alpha + \frac{1}{2}\sum_{i=1}^{l}\Big|y_i - \Big(\sum_{j=1}^{l}k(x_j,x_i)\alpha_j + b\Big)\Big|_\epsilon, \tag{5}$$

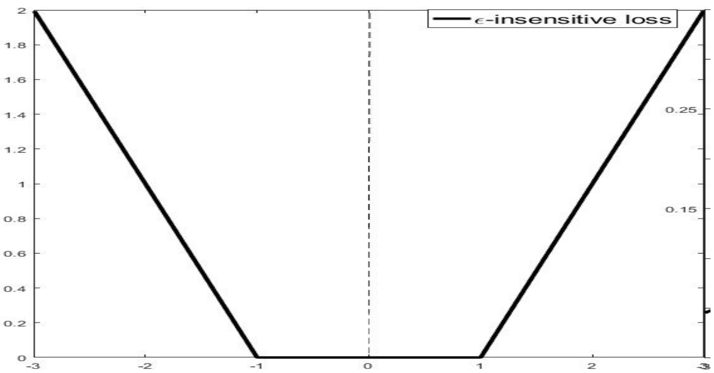

Figure 1: ϵ-insensitive loss function

where $\epsilon \geq 0$ and $\lambda \geq 0$ are user defined parameters.

The ϵ-SVR problem (5) can be converted to a Quadratic Programming Problem (QPP) which can be solved efficiently. But, in this class, we shall solve the problem (5) using sub gradient descent method. For this, let us consider

$$g_\epsilon(x_k, y_k) = \Big| y_k - \Big(\sum_{j=1}^{l} k(x_j, x_k)\alpha_j + b \Big) \Big|_\epsilon . \tag{6}$$

We can obtain the sub-gradients for $g_\epsilon(x_k, y_k)$ as follows.

$$\delta_\alpha g_\epsilon(x_k, y_k) = \begin{cases} \mathbf{0} & \text{, if } \ |y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b \Big)| < \epsilon \\ -\begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix} & \text{, if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b \Big) > \epsilon. \\ \begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix} & \text{, if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b \Big) < -\epsilon. \\ r \begin{bmatrix} k(x_1, x_k) \\ k(x_1, x_k) \\ .... \\ k(x_l, x_k) \end{bmatrix} & \text{, if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j, x_k)\alpha_j + b \Big) = \epsilon, \ \text{where } r \in (-1, 1). \end{cases}$$

$$\delta_b g_\epsilon(x_k,y_k) = \begin{cases} 0 \text{ , if } \ |y_k - \Big(\sum\limits_{j=1}^{l} k(x_j,x_k)\alpha_j + b\Big)| < \epsilon \\ -1, \text{ if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j,x_k)\alpha_j + b\Big) > 0. \\ 1, \text{ if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j,x_k)\alpha_j + b\Big) < 0. \\ r, \text{ if } \ y_k - \Big(\sum\limits_{j=1}^{l} k(x_j,x_k)\alpha_j + b\Big) = 0, \text{where } r \in (-1,1). \end{cases}$$

After computing the sub gradients for $g_\epsilon(x_k, y_k)$, we can easily obtain the sub gradients for $J(\alpha, b)$ as

$$\delta_\alpha(J(\alpha,b)) = \lambda\alpha + \frac{1}{2}\sum_{i=1}^{l}\delta_\alpha g_\epsilon(x_i,y_i)$$

$$\delta_b(J(\alpha,b)) = \frac{1}{2}\sum_{i=1}^{l}\delta_b g_\epsilon(x_i,y_i)$$

Now, we present the steps of the sub gradient descent algorithm which can obtain the solution for the ϵ-Support Vector Regression model (2).

Algorithm 2 Sub gradient method for ϵ- Support Vector Regression model

Input :- Training set T, ϵ , λ, tolerance tol, kernel parameter(if any).
Intailize $\alpha^0 \in \mathbb{R}^l$, $b^0 \in \mathbb{R}$.
Repeat
$\alpha^{k+1} = \alpha^k$ - $\eta(\delta_\alpha(J(\alpha^k, b^k)))$
$b^{k+1} = b^k$ - $\eta(\delta_b(J(\alpha^k, b^k)))$
until $|| \begin{bmatrix} \delta_w(J(\alpha^k, b^k)) \\ \delta_b(J(\alpha^k, b^k)) \end{bmatrix} || \leq tol.$

References

[1] Drucker, H., Burges, C. J., Kaufman, L., Smola, A., and Vapnik, V. Support vector regression machines. Advances in neural information processing systems, (1996) 9.

[2] Vapnik, Vladimir N. "An overview of statistical learning theory." IEEE transactions on neural networks 10.5 (1999): 988-999.

[3] Smola, Alex J., and Bernhard Schölkopf. "A tutorial on support vector regression." Statistics and computing 14.3 (2004): 199-222.