Learning a distance metric for regression on small samples

Shi Binbin

September 20, 2017

In ridge regression, the mean squared error (MSE) and L_2 regularization term is minimized:

$$\text{minimize}||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_2^2 \tag{1}$$

However, for problems with small number of samples and large number of samples, the linear regression is under-determined and there is no unique solution to the ordinary least-squares problem. The L_2 regularization term solves the problem by penalize the regression coefficients to generate unique solution.

For datasets with small number of samples, the model easily overfits to the data and generalizes poorly.

For datasets with a large number of features, we can transform the features into a low-dimensional space by linear combination of original features:

$$\mathbf{x}'_i = \mathbf{A}\mathbf{x}_i \tag{2}$$

where $\mathbf{x}_i$ is a p-dimensional vector and $\mathbf{x}'_i$ is a q-dimensional vector. $\mathbf{A}$ is a $p \times q$ weight matrix.

The whole data matrix $\mathbf{X} \in \mathbb{R}^{N \times p}$ with rows as samples can be transformed to low dimension $\mathbf{X}' \in \mathbb{R}^{N \times q}$:

$$\mathbf{X}' = \mathbf{X}\mathbf{A} \tag{3}$$

Both training samples $\mathbf{X}_1 \in \mathbb{R}^{N_1 \times p}$ and test samples $\mathbf{X}_2 \in \mathbb{R}^{N_1 \times p}$ by the same set of weights:

$$\mathbf{X}'_1 = \mathbf{X}_1\mathbf{A} \tag{4}$$

$$\mathbf{X}'_2 = \mathbf{X}_2\mathbf{A} \tag{5}$$

On the transformed training samples $\mathbf{X}'_1$, a ridge regression model can be fitted by minimizing the cost function:

$$\text{minimize}||\mathbf{y}_1 - \mathbf{X}'_1\boldsymbol{\beta}||_2^2 + \lambda||\boldsymbol{\beta}||_2^2 \tag{6}$$

The coefficients $\boldsymbol{\beta}$ that minimize the cost function can be written as:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}_1'^T\mathbf{X}'_1 + \lambda\mathbf{I}_q)^{-1}\mathbf{X}_1'^T\mathbf{y}_1 \tag{7}$$

From the properties of pseudoinverse, the coefficients can also be written as:

$$\hat{\boldsymbol{\beta}} = \mathbf{X}_1'^T(\mathbf{X}_1'\mathbf{X}_1'^T + \lambda\mathbf{I}_{N_1})^{-1}\mathbf{y}_1 \tag{8}$$

This form can reduce computational complexity because the matrix $(\mathbf{X}_1'\mathbf{X}_1'^T + \lambda\mathbf{I}_q) \in \mathbb{R}^{N_1\times N_1}$ that needs to be inverted is usually much smaller than $(\mathbf{X}_1'^T\mathbf{X}_1' + \lambda\mathbf{I}_{N_1}) \in \mathbb{R}^{q\times q}$.

The prediction of target variable $\mathbf{y}_2$ is given by:

$$\hat{\mathbf{y}}_2 = \mathbf{X}_2'\mathbf{X}_1'^T(\mathbf{X}_1'\mathbf{X}_1'^T + \lambda\mathbf{I}_{N_1})^{-1}\mathbf{y}_1 \tag{9}$$

$\mathbf{X}_2'\mathbf{X}_1'^T$ and $\mathbf{X}_1'\mathbf{X}_1'^T$ are also called kernel matrices and we can denote them by $\mathbf{K}^*$ and $\mathbf{K}_1$: $\mathbf{K}^* = \mathbf{X}_2'\mathbf{X}_1'^T$ and $\mathbf{K}_1 = \mathbf{X}_1'\mathbf{X}_1'^T$.

Then the prediction of target variable on the test samples can also be written as:

$$\hat{\mathbf{y}}_2 = \mathbf{K}^*(\mathbf{K}_1 + \lambda\mathbf{I}_{N_1})^{-1}\mathbf{y}_1 \tag{10}$$

The objective is to find the best combination of weights that minimizes the mean squared error on the test samples:

$$\text{argmin}_{\mathbf{A}} L(\mathbf{A}; \mathbf{X}_1, \mathbf{y}_1, \mathbf{X}_2, \mathbf{y}_2) = \frac{1}{2}||\mathbf{y}_2 - \hat{\mathbf{y}}_2||_2^2 \tag{11}$$

The gradients of the loss function with respect to $\mathbf{A}$ is given by:

$$\begin{aligned}
\frac{\partial L}{\partial A} &= (\hat{\mathbf{y}}_2 - \mathbf{y}_2)^T\frac{\partial\hat{\mathbf{y}}_2}{\mathbf{A}} && (12)\\
&= (\hat{\mathbf{y}}_2 - \mathbf{y}_2)^T\frac{\partial}{\partial\mathbf{A}}(\mathbf{K}^*(\mathbf{K}_1 + \lambda\mathbf{I}_{N_1})^{-1}\mathbf{y}_1) && (13)\\
&= (\hat{\mathbf{y}}_2 - \mathbf{y}_2)^T\left(\frac{\partial\mathbf{K}^*}{\mathbf{A}}(\mathbf{K}_1 + \lambda\mathbf{I}_{N_1})^{-1} + \mathbf{K}^*\frac{\partial(\mathbf{K}_1 + \lambda\mathbf{I}_{N_1})^{-1}}{\partial\mathbf{A}}\right)\mathbf{y}_1 && (14)
\end{aligned}$$

Each element of $\mathbf{K}^*$ and $\mathbf{K}$ can be evaluated as a kernel function between two samples:

$$\begin{aligned}
k(\mathbf{x}_i, \mathbf{x}_j) &= \mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_j && (15)\\
[\mathbf{K}_1]_{ij} &= k([\mathbf{x}_1]_i, [\mathbf{x}_1]_j) && (16)\\
[\mathbf{K}^*]_{ij} &= k([\mathbf{x}_2]_i, [\mathbf{x}_1]_j) && (17)
\end{aligned}$$

The partial derivative of $k(\mathbf{x}_i, \mathbf{x}_j)$ with respect to $\mathbf{A}$ can be written as:

$$\begin{aligned}
\frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial\mathbf{A}} &= \frac{\partial(\mathbf{x}_i^T\mathbf{A}^T\mathbf{A}\mathbf{x}_j)}{\partial\mathbf{A}} && (18)\\
&= \mathbf{A}(\mathbf{x}_i\mathbf{x}_j^T + \mathbf{x}_j'\mathbf{x}_i'^T) && (19)
\end{aligned}$$

and the derivatives respect to each element in $\mathbf{A}$ is:

$$\left[\frac{\partial k(\mathbf{x}_i, \mathbf{x}_j)}{\partial\mathbf{A}}\right]_{kl} = \sum_{m=1}^{q} A_{km}(x_{im}x_{jl} + x_{jm}x_{il}) \tag{20}$$

The partial derivative of the matrix inverse $(\mathbf{K}_1 + \lambda \mathbf{I}_{N_1})^{-1}$ can be evaluated for each element in A:

$$\frac{\partial (\mathbf{K}_1 + \lambda \mathbf{I}_{N_1})^{-1}}{\partial A_{kl}} = (\mathbf{K}_1 + \lambda \mathbf{I}_{N_1})^{-1} \frac{\partial \mathbf{K}_1}{\partial A_{kl}} (\mathbf{K}_1 + \lambda \mathbf{I}_{N_1})^{-1} \tag{21}$$

Note that the partial derivative $\frac{\partial \mathbf{K}_1}{\partial A_{kl}}$ needs to be evaluated for every element of **A**.

The overall time complexity for computation of the gradients of the mean squared error is approximately $O(N_1^3 + N_1(N_1 + N_2)pq)$. This is much more computationally extensive than ordinary ridge regression, but can be reduced by choosing a smaller N_1 and N_2. The time complexity also grows proportionally with the number of features p and low-rank dimension q. For datasets with a large number of features, the time complexity can be reduced by use a sparse weight matrix for **A**. A certain portion of elements in **A** can be set to zeros which can be ignored during feature transformation and evaluation of the gradients with respect to **A**.

In the training process, a small number of samples (usually fewer than 20) is taken from the whole dataset and then splitted into a training dataset and a test dataset. Then a ridge regression model is fitted on the training dataset and the gradients of the mean squared error is evaluated on the test dataset. The weight matrix **A** is optimized by stochastic gradient descent (SGD) or other variants.

The overall training procedure can be viewed as minimizing the expected mean squared error on the test dataset:

$$\text{argmin}_{\mathbf{A}} \mathbb{E}_{(\mathbf{X}_1, \mathbf{y}_1, \mathbf{X}_2, \mathbf{y}_2)} ||\mathbf{y}_2 - \hat{\mathbf{y}}_2(\mathbf{X}_1, \mathbf{y}_1)||_2^2 \tag{22}$$