Transactions of the Korean Nuclear Society Virtual Spring Meeting
July 9-10, 2020

Comparisons of Linear Regression Methods: Machine Learning Method and Linear Algebra Method

Yong Suk Suh*, Seung Ki Shin, Dane Baang, Sang Mun Seo, Jong Bok Lee
KJRR Project Div., Korea Atomic Energy Research Institute (KAERI), 111, Daedeok-Daero 989Beon-Gil, Yuseong-Gu, Daejeon, 34057, Korea
**Corresponding author: yssuh@kaeri.re.kr*

1. Introduction

A simple linear regression (SLR) and a multiple linear regression (MLR) in machine learning programming (MLP) were briefly reviewed in previous papers [1], [2]. The results of the programming depended on the initial value, learning rate, and epoch (i.e., the number of trainings) that were determined by its programmer [1]. The normalization and variance inflation factor should be considered in the MLR [2]. If the nuclear facility domain can be modeled in a linear system, the linear regression is a useful method to predict an outcome (e.g., temperature, pressure, etc.) based on the data historically collected from the facility. We can collect process parameters as independent variables and predict a phenomenon as a dependent variable using the linear regression method. This paper does not deal with the domain, but shows some examples of the linear regression method itself. The linear regression problem can be solved with the MLP or linear algebra method. This paper compares the two methods.

2. Linear System

The linear system is represented in the form of

$$y = w_1x_1 + w_2x_2 + \dots + w_nx_n, \tag{1}$$

where y is the dependent variable, x is the independent variable, and w is the slope of y to be calculated using the linear regression.

The linear system in Eq. (1) is represented in the vector form of

$$\mathbf{y} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{x}. \tag{2}$$

We can get $\boldsymbol{w}$ by using the linear regression method when $\boldsymbol{x}$ and $\mathbf{y}$ are given. When n numbers of sample data are given to n number of x variables, x becomes an n-by-n matrix. In this case, the linear system (2) is represented in the matrix form of

$$\mathbf{y} = \boldsymbol{X}\boldsymbol{w}, \tag{3}$$

where $\boldsymbol{X}$ is an n-by-n matrix and $\mathbf{y}$ and $\boldsymbol{w}$ are respectively an n row vector.

This paper introduces two methods to get $\boldsymbol{w}$: machine learning method and linear algebra method.

3. Machining Learning Method

The machine learning method solves the linear regression by using the mean square error (MSE) equation and gradient descent method of the MSE.

The MSE of Eq. (3) with respect to $\boldsymbol{w}$ is represented in the form of

$$\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{n}(\mathbf{y} - \hat{\mathbf{y}})^2, \tag{4}$$

where $\mathbf{y}$ is the given data and $\hat{\mathbf{y}}$ is the predicted data by calculating $\boldsymbol{Xw}$.

Thus, Eq. (4) can be represented in the form of

$$\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{n}(\mathbf{y} - \boldsymbol{Xw})^2. \tag{5}$$

In the machine language programming, the MSE equation is called a loss or cost function. We can get the minimal value of the loss function by the partial derivative of the MSE with respect to $\boldsymbol{w}$ in the form of

$$\begin{aligned}\frac{\partial}{\partial w}\mathrm{MSE}(\boldsymbol{w}) &= \frac{\partial}{\partial w}\frac{1}{n}(\mathbf{y} - \boldsymbol{Xw})^2 \\ &= \frac{2}{n}\boldsymbol{X}^{\boldsymbol{T}}(\boldsymbol{Xw} - \mathbf{y}).\end{aligned} \tag{6}$$

The gradient descent method calculates $\boldsymbol{w}$ by repeating Eq. (6) until $\boldsymbol{w}_{(j+1)}$ is equal to $\boldsymbol{w}_{\boldsymbol{j}}$ as

$$\boldsymbol{w}_{(j+1)} = \boldsymbol{w}_j - \rho\frac{\partial}{\partial w}\mathrm{MSE}(\boldsymbol{w}_j), \tag{7}$$

where ρ is a step (or jump) of gradient descent, called a learning rate.

The number of repeat in Eq. (7) is called an epoch. The results of Eq. (7) depended on the initial value of $\boldsymbol{w}$, learning rate ρ, and epoch [1]. In addition, the normalization and variance inflation factors should be considered in the machine learning method of multiple linear regression [2]. The loss (or cost) value (i.e., mean square error) becomes zero when $\boldsymbol{w}_{(j+1)}$ is equal to $\boldsymbol{w}_j$.

4. Linear Algebra Method

The linear algebra method solves the linear regression using the inverse of matrix. When $\boldsymbol{X}$ and $\mathbf{y}$ are given and $\boldsymbol{X}$ is invertible, the $\boldsymbol{w}$ of Eq. (3) can be calculated in the form of

$$\boldsymbol{X}^{-\mathbf{1}}\mathbf{y} = \boldsymbol{w}. \tag{8}$$

However, we cannot solve Eq. (8) when $\boldsymbol{X}$ is not an invertible matrix. In the real world, the number of sample data (i.e., given data) is larger than the number of x variables. This is called an over-determined linear system. In this case, we cannot get the inverse of $\boldsymbol{X}$, but we can solve Eq. (8) by using a pseudo-inverse method.

The MSE of Eq. (4) can be represented in the form of

$$\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{n}(\mathbf{y}-\boldsymbol{X}\boldsymbol{w})^2 \\ = \frac{1}{n}(\mathbf{y}-\boldsymbol{X}\boldsymbol{w})^T(\mathbf{y}-\boldsymbol{X}\boldsymbol{w}) \\ = \frac{1}{n}(\mathbf{y}^T\mathbf{y} - 2\boldsymbol{w}^T\boldsymbol{X}^T\mathbf{y} + \boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\,\boldsymbol{w}). \qquad (9)$$

We calculate the partial derivative of the loss function (9) with respect to $\boldsymbol{w}$ in the form of

$$\frac{\partial}{\partial w}\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{n}\left(\frac{\partial}{\partial w}\mathbf{y}^T\mathbf{y} - \frac{\partial}{\partial w}2\boldsymbol{w}^T\boldsymbol{X}^T\mathbf{y} + \frac{\partial}{\partial w}\boldsymbol{w}^T\boldsymbol{X}^T\boldsymbol{X}\,\boldsymbol{w}\right) = \frac{2}{n}(-\boldsymbol{X}^T\mathbf{y} + \boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w}). \qquad (10)$$

We want to get a minimal value of the MSE. For this, the partial derivative should be zero in the form of

$$\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{w} = \boldsymbol{X}^T\mathbf{y}. \qquad (11)$$

Thus, we can calculate **w** in the form of

$$\boldsymbol{w} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\mathbf{y}, \qquad (12)$$

where $(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T$ is a pseudo-inverse of $\boldsymbol{X}$.

We can also get the pseudo-inverse of $\boldsymbol{X}$ using the singular value decomposition (SVD). It is known that an n-by-n square matrix $\boldsymbol{X}$ can be orthogonally diagonalizable if there exist an orthogonal matrix **P** and a diagonal matrix **D** in the form of

$$\mathbf{X} = \mathbf{PD}\boldsymbol{P}^{\boldsymbol{T}}. \qquad (13)$$

This is called an eigen value decomposition (EVD). If $\boldsymbol{X}$ is not a square matrix, it can be factorized in the form of

$$\mathbf{X} = \mathbf{U\Sigma}\boldsymbol{V}^{\boldsymbol{T}}, \qquad (14)$$

where **U** is a left singular vector, **Σ** is a diagonal matrix representing singular values of **X**, $\boldsymbol{V}$ is a right singular vector.

This is called a SVD [3]. The **U** is the eigen vector of $\boldsymbol{X}\boldsymbol{X}^T$, which can be determined from the EVD of $\boldsymbol{X}\boldsymbol{X}^T$. The $\boldsymbol{V}$ is the eigen vector of $\boldsymbol{X}^T\mathbf{X}$, which can be determined from the EVD of $\boldsymbol{X}^T\mathbf{X}$. When we can get the SVD of **X** as in Eq. (14), the pseudo-inverse of $\boldsymbol{X}$ is represented in the form of

$$\mathbf{X}^+ = \mathbf{U\Sigma}^+\boldsymbol{V}^{\boldsymbol{T}}, \qquad (15)$$

where $\mathbf{\Sigma}^+$ is a reciprocal matrix of **Σ**.

When we can get the $\mathbf{X}^+$, we can get the $\boldsymbol{w}$ in the form of

$$\mathbf{y}\boldsymbol{X}^+ = \boldsymbol{w}. \qquad (16)$$

5. Results of the Comparisons

In order to compare the two methods (i.e., the machine learning method and linear algebra method), arbitral values for x and y are determined as follows:

y =([1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20])
x_1=([1,3,5,8,9,10,11,13,20,25,30,41,44,49,58,60,66,78,80,99])
x_2=([1,2,3,4,5,11,12,13,14,15,21,22,23,24,25,31,32,33,34,35])
x_3=([1,1,1,1,1,30,30,30,30,30,50,50,50,50,50,70,70,70,70,70])

The simple linear regressions of each x value to y value using the SVD method are shown in Fig. 1.

The slopes (i.e., w values) of each regression graph in Fig. 1 are shown in Table 1. The w values in MS-Excel in Table 1 were calculated by setting the y-intercept to 0.

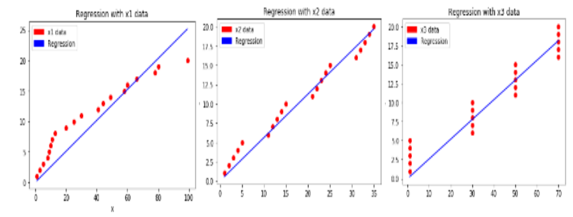

Figure 1 Simple linear regressions of each x value

Table 1 w values of each regression

w values (slopes)	Using SVD	Using MS-Excel
w_1 for x_1	0.25419	0.25419
w_2 for x_2	0.56208	0.56208
w_3 for x_3	0.25937	0.25937

Table 2 and Fig. 2 show the predicted y values that resulted from the linear regression by combining the x_1, x_2, and x_3 values in the form of

$$\mathrm{y} = w_1x_1 + w_2x_2 + w_3x_3\ . \qquad (17)$$

Table 2 Predicted y values per linear regression method

Method	Predicted y values
SVD	0.9,2.0,3.1,4.2,5.3,5.4,6.6,7.7,8.5,9.4,11.4,12.0, 13.0,13.9,14.6,16.8,17.7,18.1,19.2,19.3
Tensorflow pinv	0.9,2.0,3.1,4.2,5.3,5.4,6.6,7.7,8.5,9.4,11.4,12.0, 13.0,13.9,14.6,16.8,17.7,18.2,19.3,19.3
Numpy pinv	0.9,2.0,3.1,4.2,5.3,5.4,6.6,7.7,8.5,9.4,11.4,12.0, 13.0,13.9,14.6,16.8,17.7,18.2,19.3,19.3
Numpy lstsq	0.9,2.0,3.1,4.2,5.3,5.4,6.6,7.7,8.5,9.4,11.4,12.0, 13.0,13.9,14.6,16.8,17.7,18.2,19.3,19.3
Sckitlearn	1.4,2.3,3.3,4.2,5.2,5.7,6.7,7.7,8.4,9.3,11.4,12.0, 13.0,13.8,14.5,16.7,17.6,18.2,19.1,19.5
ML	1.9,2.8,3.3,3.9,4.3,6.4,6.8,7.3,8.2,8.9,11.3,12.5, 13.0,14.7,16.9,16.9,18.1,18.6,20.3

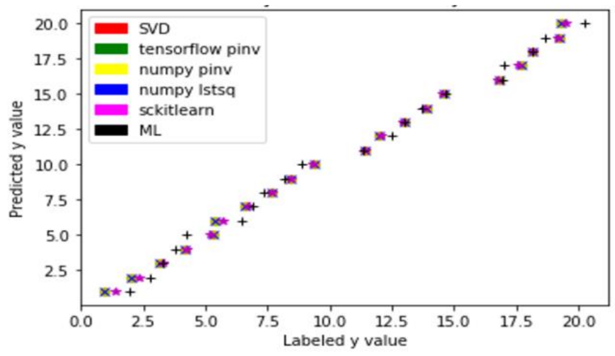

Figure 2 Predicted y value vs. Labeled y value

In Table 2 and Fig. 2, the SVD represents the regression using the SVD method. The tensorflow pinv represents the regression using the pseudo-inverse function provided in Tensorflow. The numpy pinv represents the regression using the pseudo-inverse function in Numpy. The numpy lstsq represents the regression using the least square function in Numpy. The sckitlearn represents the regression using the linear regression function in Scikitlearn. The ML represents the regression using the gradient descent function in machine learning programming. In the ML, the epoch was 100,000 and the learning rate was 0.001. The w values of Fig 2 are shown in Table 3 including the w value calculated by MS-Excel.

Table 3 Values of w_1, w_2 and w_3

Method	Values of w_1, w_2 and w_3
SVD	-0.063, 1,230, -0.250 without bias
Tensorflow pinv	-0.063, 1.230, -0.250 without bias
Numpy pinv	-0.063, 1.230, -0.250 without bias
Numpy lstsq	-0.063, 1.230, -0.250 without bias
Sckitlearn	-0.035, 1.034, -0.197 with bias=0.554
ML	6.616, 12.755, -0.023 with bias=10.662
MS-Excel	-0.035, 1.034, -0.197 with bias=0.554

The results of linear regression using the machine learning method and linear algebra method were found that they were not significantly different. This paper used Spyder 3.3.6 in Anaconda3 1.9.12, Python 3.7.4, and Tensorflow 2.0 as programming environment, language and library, respectively.

6. Conclusions

This paper showed some examples to solve the linear regression by using machine learning programming and linear algebra methods. If the independent variables have big data, we may not solve the linear regression by using the linear algebra method, but by using the machine learning programming method. The singular value decomposition can be used to reduce the dimension of big data. This will be studied further.

Acknowledgement
This work was supported by Korea MSIT (Grant Code: 2020M2D5A1078133).

REFERENCES

[1] Yong Suk Suh, et al., A Brief Review of a Machine Learning Programming of Simple Linear Regression, Transactions of the Korean Nuclear Society Spring Meeting, Jeju, Korea, May 17-18, 2018.
[2] Yong Suk Suh, et al., Considerations on Machine Learning Programming of Multiple Linear Regression, Transactions of the Korean Nuclear Society Spring Meeting, Jeju, Korea, May 23-24, 2019
[3] http://en.wikipedia.org/wiki/Singular_value_decomposition