Transactions of the Korean Nuclear Society Virtual Autumn Meeting
December, 17-18, 2020

A Method to Reduce Data Dimensions in Machine Learning Programming

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

When programming a machine learning application, a large amount of input data can collected. It might be needed to segregate the data into several dimensions of data. While the data is being collected and segregated, some of the data might be unnecessarily collected or the data dimensions be unnecessarily segregated. In this case, we need to eliminate the unnecessary data or reduce the dimensions of the data. This paper deals with the reduction of data dimensions.

The machine learning programming for simple and multiple linear regressions were briefly reviewed in previous papers [1] and [2]. The machine learning method and the algebra method for the linear regressions were also compared in previous paper [3].

When we want to apply machine learning to nuclear facilities, we collect much data from field instruments, such as temperature, level, pressure, flow, etc. While the domain of a nuclear facility is analyzed, some of the field data are not needed for the machine learning. In this case, we want to determine how much data impacts the results of machine learning. If some of the data has a low impact, we can eliminate the data from the original data. This elimination will increase the computing time of the machine learning. In programming the linear regressions using the machine learning and/or algebra method, the reduction of data dimensions produces many benefits in computing time.

This paper introduces the principal component analysis (PCA) as a method to reduce the data dimensions and presents the test results of the PCA with three dimensional simple and arbitral sample data.

2. Principal Component Analysis

Data projection is one of ways to reduce data dimensions by orthogonally moving data in higher dimensional data space to lower dimensional data space [4]. The lower or higher dimensional data space then its original data space is called a hyper plane [5].

When we suppose that the data vector $\boldsymbol{x}$ is projected onto a unit vector $\boldsymbol{w}$, the projected data is represented in the form of $\boldsymbol{x^T w}$, which is a scalar value. The projected data $\boldsymbol{x^T w}$ lies on the $\boldsymbol{w}$ so it can be represented in the vector form of $(\boldsymbol{x^T w})\boldsymbol{w}$. When n numbers of $\boldsymbol{x}$ data are projected onto the $\boldsymbol{w}$, the mean of all projected data is represented in the form of $\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\,\boldsymbol{w}$. In this paper, we assume that all the $\boldsymbol{x}$ data are normalized so that the mean of them becomes zero. Thus, $\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\,\boldsymbol{w}$ is also zero because $\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T$ is zero. This makes many of our equations simple. The normalization of data enables the machining learning program to run fast and produce accurate outcomes.

When the data $\boldsymbol{x}$ is projected onto a hyperplane $\boldsymbol{w}$, we need to find the optimized vector to minimize the distance between the $\boldsymbol{x}$ and $\boldsymbol{w}$ and maximize the variance of the projected data $\boldsymbol{x^T w}$.

The distance between the $\boldsymbol{x}$ and $(\boldsymbol{x^T w})\boldsymbol{w}$ can be represented in the square of the L2-norm form of $||[\boldsymbol{x}-(\boldsymbol{x^T w})\boldsymbol{w}]||^2$. An optimal $\boldsymbol{w}$ can be obtained using the mean square error (MSE) called a least square fit for all $\boldsymbol{x}$ vectors projected onto the $\boldsymbol{w}$. The MSE with respect to $\boldsymbol{w}$ is represented in the form of

$$\begin{aligned}\text{MSE}(\boldsymbol{w}) &= \frac{1}{n}\sum_{i=1}^{n}[\boldsymbol{x}_i-(\boldsymbol{x}_i^T\boldsymbol{w})\boldsymbol{w}]^2\\ &= \frac{1}{n}\sum_{i=1}^{n}[\boldsymbol{x}_i^T\boldsymbol{x}_i-\boldsymbol{2}\boldsymbol{x}_i^T(\boldsymbol{x}_i^T\boldsymbol{w})\boldsymbol{w}+(\boldsymbol{x}_i^T\boldsymbol{w})\boldsymbol{w^T}(\boldsymbol{x}_i^T\boldsymbol{w})\boldsymbol{w}]\\ &= \frac{1}{n}\sum_{i=1}^{n}[\boldsymbol{x}_i^T\boldsymbol{x}_i-(\boldsymbol{x}_i^T\boldsymbol{w})^{\boldsymbol{2}}]\\ &= \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\boldsymbol{x}_i-\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{w})^{\boldsymbol{2}}. \qquad (1)\end{aligned}$$

From the Eq. (1), we know that the MSE($\boldsymbol{w}$) is minimized when the $\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{w})^{\boldsymbol{2}}$ is maximized because the $\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\boldsymbol{x}_i$ is not related to $\boldsymbol{w}$.

We represent the variance of projected data $\boldsymbol{x^T w}$ in the form of

$$\text{Var}(\boldsymbol{x^T w}) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{w})^2-\left(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\boldsymbol{w}\right)^2. \qquad (2)$$

In Eq. (2), the square of mean of $\boldsymbol{x^T w}$, $(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{x}_i^T\boldsymbol{w})^2$, is zero because we already assumed all the $\boldsymbol{x}$ data were normalized in this paper. Thus, the variance is represented in the simple form of

$$\text{Var}(\boldsymbol{x^T w}) = \frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{w})^2. \qquad (3)$$

In Eq. (3), when the $\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{x}_i^T\boldsymbol{w})^2$ is maximized, the Var($\boldsymbol{x^T w}$) is also maximized. Thus, we know that the MSE($\boldsymbol{w}$) is minimized when the Var($\boldsymbol{x^T w}$) is maximized. We need to find a way to maximize the Var($\boldsymbol{x^T w}$). In order to eliminate the Σ in Eq. (3) and make the equation a simple form, we expand the data vector $\boldsymbol{x}$ to an n-by-m or n-by-n matrix $\boldsymbol{X}$. In this case, the $\boldsymbol{w}$ will be n number of vectors. When the $\boldsymbol{X}$ is projected onto the $\boldsymbol{w}$, the variance of $\boldsymbol{Xw}$ is represented in the form of

$$\mathrm{Var}(\boldsymbol{Xw}) = \frac{1}{n}(\boldsymbol{Xw})^{\boldsymbol{T}}(\boldsymbol{Xw}) = \frac{1}{n}\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{Xw}$$
$$= \boldsymbol{w}^{\boldsymbol{T}}\frac{\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}}{n}\boldsymbol{w} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{Vw}, \qquad (4)$$

where $\boldsymbol{V}$ is $\frac{\boldsymbol{X}^{\boldsymbol{T}}\boldsymbol{X}}{n}$.

In Eq. (4), the $\boldsymbol{V}$ is a matrix that represents a variance of $\boldsymbol{X}$. When the matrix $\boldsymbol{X}$ is given, we want to find an optimal (i.e., stationary) $\boldsymbol{w}$ to maximize the $\mathrm{Var}(\boldsymbol{Xw})$. Thus, the function of maximizing the $\mathrm{Var}(\boldsymbol{Xw})$ is defined as an objective function. Along with the objective function, when $\boldsymbol{w}$ is a unit vector, we define an constraint function

$$\mathrm{g}(\boldsymbol{w}) = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} - 1 \qquad (5)$$

With Eqs. (4) and (5), we can adopt the Lagrange multiplier λ to find an optimal $\boldsymbol{w}$ so as to define a Lagrangian function with respect to $\boldsymbol{w}$ and λ

$$\mathcal{L}(\boldsymbol{w}, \lambda) = \mathrm{Var}(\boldsymbol{Xw}) - \lambda \mathrm{g}(\boldsymbol{w})$$
$$= \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{Vw} - \lambda(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} - 1) \qquad (6)$$

In order to solve the Lagrangian function, we take partial derivatives

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{w}} = 2\boldsymbol{Vw} - 2\lambda\boldsymbol{w} \qquad (7)$$
$$\frac{\partial \mathcal{L}}{\partial \lambda} = \boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} - 1 \qquad (8)$$

Thus, we obtain an optimal $\boldsymbol{w}$ value at zero from Eq. (7)

$$\boldsymbol{Vw} = \lambda\boldsymbol{w} \qquad (9)$$

In Eq. (9), the $\boldsymbol{w}$ is an eigenvector of matrix $\boldsymbol{V}$ and the λ is an eigenvalue of eigenvector $\boldsymbol{w}$. The linear transformation of $\boldsymbol{w}$ depends on the $\boldsymbol{V}$. The maximal variance of $\boldsymbol{w}$ is in λ. Thus, the λ is also the diagonal value of the $\boldsymbol{V}$. The λ is obtained by the eigenvalue decomposition for n-by-n matrix or singular value decomposition for n-by-m matrix of $\boldsymbol{V}$. The largest λ represents the largest variance of projected data.

3. Sample Test Cases

Sample test cases to reduce the data dimensions are arbitrarily chosen in this paper as shown in Table 1.

Table 1: Sample data and normalized data

Given data					
$\boldsymbol{x}_1$	1	3	5	8	10
$\boldsymbol{x}_2$	1	12	33	54	75
$\boldsymbol{x}_3$	1	1	33	33	99
y	1	2	3	4	5
Normalized data of the given $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$					
$\boldsymbol{x}_1$	-0.49	-0.27	-0.04	0.29	0.51
$\boldsymbol{x}_2$	-0.33	-0.33	-0.01	-0.01	0.67
$\boldsymbol{x}_3$	0.46	-0.31	-0.03	0.26	0.54

Using the Sckit-learn PCA library, we can obtain projected sample data of $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$ and eigenvalues of λ as shown in Table 2. In Table 2, the $\boldsymbol{x}_1$ is the most dominant characteristic sample data because its eignevalue is the largest which represents the largest variance.

Table 2: Results of PCA

Projected sample data after PCA					
$\boldsymbol{x}_1$	-0.74	-0.52	-0.04	0.32	0.99
$\boldsymbol{x}_2$	-0.11	0.04	-0.02	0.22	-0.13
$\boldsymbol{x}_3$	-0.01	0.02	-0.01	-0.01	0.01
λ	1.38	0.29	0.03		

When we perform the linear regression with the normalized data in Table 1, we obtain the values of $\boldsymbol{w}$ (i.e., coefficient of $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$) and the bias (i.e., y-intercept) for the regression linear line, and predicted **y** values and R-squared value as shown in Table 3.

Table 3: Linear regression with the normalized data in Table 1

$\boldsymbol{w}$	4.89	0.43	-1.38		
bias	3				
Predicted **y**	1.10	1.98	2.82	4.06	5.04
R^2	0.995				

When we perform the linear regression with the projected data in Table 2, we obtain the $\boldsymbol{w}$ and the bias for each data dimensions, and predicted **y** values and R-squared value as shown in Table 4.

Table 4: Linear regression with the projected data in Table 2

Case 1: Linear regression with $\boldsymbol{x}_1, \boldsymbol{x}_2$ and $\boldsymbol{x}_3$ (i.e., Data dimensions are 3.)					
$\boldsymbol{w}$	2.26	1.68	4.25		
bias	3				
Predicted **y**	1.10	1.98	2.82	4.06	5.04
R^2	0.995				
Case 2: Linear regression with $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ (i.e., Data dimensions are 2.)					
$\boldsymbol{w}$	2.26	1.68			
bias	3				
Predicted **y**	1.14	1.89	2.86	4.09	5.02
R^2	0.994				
Case 3: Linear regression with $\boldsymbol{x}_1$ (i.e., Data dimension is 1.)					
$\boldsymbol{w}$	2.26				
bias	3				
Predicted **y**	1.33	1.82	2.90	3.71	5.24
R^2	0.971				

In Table 4, Case 1 does not reduce the dimensions but Case 2 and 3 do. Case 1 generates the same results as the linear regression with the normalized sample data because the data dimensions are not reduced. Case 2 dose not significantly degrade the characteristic of the original data but Case 3 degrades it a little bit. If we allow the threshold of R-squared value is 9.5, Case 3 is also effective. We need to study or establish a formal method to measure how much the data dimensions can be reduced while still maintaining the characteristic of the original data. The programming is done using Spyder 3.3.6 in Anaconda 3 1.9.12, Python 3.7.4, and Sckit-learn library.

4. Conclusions

This paper introduced the principal component analysis (PCA), which we used to reduce data dimensionality, and presented test results of the reduction of data dimensions using the PCA method.

The PCA method showed an effective method for the reduction. There is a further need to establish a formal method to determine the level of reduction.

A further study will explore the application of singular value decomposition.

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] Yong Suk Suh, et al., Comparisons of Linear Regression Methods: Machine Learning Method and Linear Algebra Method, Transactions of the Korean Nuclear Society Virtual Spring Meeting, July 9-10, 2020.
[4] Davie C. Lay, et al., LINEAR ALGEBRA AND ITS APPLICATIONS, 5th Edition, PEARSON, 2014.
[5] http://en.wikipedia.org/wiki/Hyperplane