PSR: PSO-Based Signomial Regression Model

SeJoon Park¹, NagYoon Song², Wooyeon Yu², and Dohyun Kim²

¹Division of Energy Resources Engineering and Industrial Engineering, Kangwon National University, Chuncheon, Korea

²Department of Industrial and Management Engineering, Myongji University, Yongin, Korea

ljfis

Abstract

Regression analysis can be used for predictive and descriptive purposes in a variety of business applications. However, the successive existing regression methods such as support vector regression (SVR) have the drawback that it is not easy to derive an explicit function description that expresses the nonlinear relationship between an output variable and input variables. To resolve this issue, developed in this article is a nonlinear regression algorithm using particle swarm optimization (PSO) which is PSR. The output variables of PSR allow to obtain the explicit function description of the output variable using input variables. Three PSRs are proposed based on infeasible-particle update rules. Their experimental results show that the proposed approach performs similarly to and slightly better than the existing methods regardless of the data sets, implying that it can be utilized as a useful alternative when obtaining the explicit function description of the output variable using input variables and interpreting which of the original input variables are more important than others in the obtained regression model.

Keywords: Signomial regression, Particle swarm optimization, Meta-heuristic optimization

1. Introduction

Supervised learning is one of the representative machine learning methods that attempt to reveal hidden relationships between input data and output data. The main problems of supervised learning are largely classification and regression. Especially, regression analysis can be used for predictive and descriptive purposes in a variety of business applications. For this reason, many regression methods have been developed, including ordinary least squares (OLS) and support vector regression (SVR).

OLS aims to estimate the unknown parameters in a linear regression model in order to minimize the sum of squared differences between the observed and predicted values. OLS, one of the simplest methods, has the advantage that the output variable can be expressed as a function expression of the input variables. However, to express the nonlinear relationship between the input variables and the output variable, the analyst must manually generate the necessary variables.

SVR, proposed originally by Vapnik et al. in 1997 [1], is one of the most popular supervised learning algorithms widely used in regression analysis. Using kernel tricks to efficiently perform nonlinear regression and implicitly map inputs to higher-dimensional space, SVR's

Received: Nov. 11, 2019
 Revised : Dec. 10, 2019
 Accepted: Dec. 12, 2019

Correspondence to: Dohyun Kim
 (ftgog@mju.ac.kr)
 ©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

performance is very superior. However, there are some problems. First, it is difficult to obtain a clear function representation of the output variable using input variables. Therefore, it is not easy to understand the relationship between the output variable and the input variables. Second, the selection of many parameters is important to the performance of the SVR, directly affecting its generalization and regression efficiency.

In this paper, we propose an algorithm that combines the advantages of OLS and SVR using the particle swarm optimization (PSO) method. The chief advantages of this PSO-based algorithm are that it can first derive an explicit function description that expresses the nonlinear relationship between the output variable and the input variables without kernel tricks, and that, simultaneously, it helps to select the optimal parameters for the generalized regression model.

This paper is organized as follows. Section 2 explains PSO-based non-linear regression algorithm. Section 3 discusses the performance of the proposed method. Finally, Section 4 draws conclusions.

2. PSO-Based Signomial Regression (PSR) Model

2.1 PSR Model

In this section, we present the signomial function. The signomial function $f(x)$ is the sum of the monomial $g_i(x)$, which is the product of a number of positive variables $(x_1, \dots, x_n)$ with exponents of real numbers $(r_{i1}, \dots, r_{in})$ [2].

$$g_i(x) = x_1^{r_{i1}} x_2^{r_{i2}} \dots x_n^{r_{in}},$$

$$f(x) = \sum_{i=1}^m \beta_i g_i(x), \quad (1)$$

where the coefficients β_i is the real number.

Our goal is to obtain the function $f(x)$ by estimating the parameters $(\beta_i$ and $r_{i1}, r_{i2}, \dots, r_{in})$ using PSO.

2.2 Particle Swarm Optimization

In this research, the basic PSO proposed by Kennedy and Eberhart [3] was applied. In 2007, many implementations of the basic PSO were cataloged and defined by Bratton and Kennedy [4]. PSO has been applied to many optimization problems such as the traveling salesman problem [5] and scheduling problems [6]. PSO is also well known as a metaheuristic algorithm that can be utilized to optimize continuous nonlinear problems [7]. This

is why PSO also has been widely used to select optimal parameters of support vector machine (SVM) for classification and regression [8–10]. In this paper, we will present a new, PSO-based supervised learning method for regression problems. To the best of our knowledge, there have been no previous studies on regression problems using PSO.

PSO is an iterative procedure maintaining multiple “particles”(solutions) and their “directions and speeds,” which are referred to simply as particles and velocities, respectively. In this research, PSO was used to estimate r_{ij} for all i, j in Eq. (1). The objective of PSO is to find the estimated r_{ij} and the coefficients in Eq. (1) in order to minimize the mean square error (MSE) of Eq. (1). First, the particle and velocity is defined as below:

$$\text{Particle: } R_k = (r_{11}^k, r_{12}^k, \dots, r_{mn}^k),$$

$$\text{Velocity: } V_k = (v_{11}^k, v_{12}^k, \dots, v_{mn}^k), \quad (2)$$

where k is the particle index. The overall PSO procedure is schematized in Figure 1.

Figure 1. PSO procedure.

Table 1. Pseudo-code for initialization

	Pseudo-code
Particle	<p>Step 1: Set $r_{ij}^k = 0$ for $\forall i, j$. Set $i = 1, S = \{1, 2, 3, \dots, n\}$.</p> <p>Step 2: Randomly select q in S.</p> <p>Step 3: Generate uniform $[-1 + \sum_{\forall j} r_{ij}^k , 1 - \sum_{\forall j} r_{ij}^k]$ and round up two decimal places and assign it to r_{iq}^k.</p> <p>Step 4: $S = S - \{q\}$. If $S \neq \emptyset$, go to Step 2.</p> <p>Step 5: If $i < m$, $i = i + 1$, set $S = \{1, 2, 3, \dots, n\}$ and go to Step 2.</p>
Velocity	Generate uniform $[-0.01, 0.01]$ and assign it to v_{ij}^k for $\forall i, j$

In the beginning, multiple particles and their velocities are randomly generated. The initial r_{ij}^k is between -1 and 1 , and $\sum_{j=1}^n |r_{ij}^k| \leq 1$ for $\forall k$. The initial v_{ij}^k is between -0.01 and 0.01 . Table 1 shows how to initialize the k th particle and velocity. All r_{ij}^k are initialized from $i = 1$ to $i = m$ term by term. In each term the independent variable index, j , is randomly selected one by one. In each selection, a uniform $[-1 + \sum_{\forall j} |r_{ij}^k|, 1 - \sum_{\forall j} |r_{ij}^k|]$ random variable is generated and assigned to r_{ij}^k . This allows $\sum_{j=1}^n |r_{ij}^k|$ to be less than or equal to one.

In each iteration, each particle and each velocity is updated based on the current particle, current velocity, local best particle, and global best particle. If R_k^t and V_k^t are the particle and velocity for the t^{th} iteration, each particle is randomly changed by Eq. (3).

$$\begin{aligned}
 V_k^{t+1} &= V_k^t + 2 \cdot R1 \cdot (\text{global best particle} - R_k^t) \\
 &\quad + 2 \cdot R2 \cdot (\text{local best particle} - R_k^t), \\
 R_k^{t+1} &= R_k^t + V_k^{t+1},
 \end{aligned} \tag{3}$$

where, $R1$ and $R2$ are uniform $[0, 1]$ random variables. The local best particle is the best particle (solution) over all iterations for particle k , and the global best particle is the best particle (solution) over all iterations for all particles. The local and global best particles are updated when the coefficients and MSE are updated in Figure 1. The coefficients in Eq. (1) and its MSE are determined for each particle by the least square method. As shown in Figure 1, if the best global particle is not changed over TR iterations, the PSO procedure is terminated. This is the

Table 2. Three infeasible-particle update rules

Method	Update rule
PSR1	<p>If $\sum_{j=1}^n r_{ij}^k > 1$, set $r_{pq}^k = 0$, where</p> $(p, q) = \arg \min_{\forall (p, q), r_{pq}^k > 0} r_{pq}^k , \text{ until } \sum_{j=1}^n r_{ij}^k \leq 1.$
PSR2	If $r_{ij}^k > 1$, set $r_{ij}^k = 1$ or if $r_{ij}^k < -1$, set $r_{ij}^k = -1$.
PSR3	Add a penalty to MSE when $\sum_{j=1}^n r_{ij}^k > 1$ for any k .

termination rule of the PSO procedure.

2.3 Three Particle and Velocity Update Methods

When a particle and its velocity are updated, the particle can obey constraint $\sum_{j=1}^n |r_{ij}^k| \leq 1$ (Figure 1). Three different update methods are proposed in this research, namely PSR1, PSR2, and PSR3, as shown in Table 2. In PSR1, if $\sum_{j=1}^n |r_{ij}^k| > 1$, set $r_{ij}^k = 0$ one by one until $\sum_{j=1}^n |r_{ij}^k| \leq 1$. A smaller $|r_{ij}^k|$ is selected, and set to zero, in order to select a less important r_{ij}^k . In PSR 2, the constraint is relaxed as each $|r_{ij}^k| \leq 1$. Therefore, if $|r_{ij}^k| > 1$, set r_{ij}^k to 1 or -1 . In PSR3, PSO is able to keep infeasible particles (solutions) such as $|r_{ij}^k| > 1$. However, this method adds sufficient penalty to the associated MSE to prevent selection of an infeasible particle as the local or global best solution. In the next section, the results of the PSRs and other approaches are compared.

3. Experimental Design and Results

3.1 Experimental Design

In this section, the method by which the regression models are determined by the PSRs is explained, and then their respective experimental results are compared with OLS, multivariate adaptive regression splines (MARS), and SVR. In order to compare the OLS, MARS, SVR, and 3 PSRs, the four datasets shown in Table 3 were used. These data sets are usually used in machine learning area. Each data set has 10 train, valid, and test datasets, which are used for model training, model validation, and model testing, respectively. For optimal objectivity, various types of data were tested. Abalone had a larger number of data than the

Table 3. Parameters for each dataset

Dataset	Number of independent variables (<i>n</i>)	Number of independent runs	Number of observations		
			Train	Valid	Test
Abalone	7	10	2,507	835	304
Auto	7	10	304	101	101
Machine	6	10	125	42	42
Triazines	60	10	112	37	37

Table 4. Average MSEs and their standard deviations for each method and dataset

Method		Dataset											
		Abalone			Auto			Machine			Triazines		
		Train	Valid	Test	Train	Valid	Test	Train	Valid	Test	Train	Valid	Test
OLS	MSE	4.90	4.93	5.11	10.24	11.82	12.57	2854	8388	6215	0.011	0.042	0.058
	SD	0.10	0.29	0.41	0.51	1.63	1.57	846	6467	4488	0.001	0.009	0.043
MARS	MSE	4.45	4.53	4.66	6.97	8.72	9.36	1476	11212	10710	0.013	0.026	0.023
	SD	0.09	0.23	0.28	0.55	0.95	1.93	446	12416	15231	0.002	0.005	0.005
SVR	MSE	4.28	4.46	4.67	4.11	7.50	7.19	1156	5600	5092	0.013	0.023	0.023
	SD	0.13	0.26	0.36	0.46	1.36	1.24	250	4409	4976	0.001	0.006	0.006
PSR1	MSE	4.42	4.38	4.62	6.77	8.05	8.93	1324	5625	4231	0.013	0.021	0.019
	SD	0.12	0.22	0.31	0.43	1.05	1.75	243	4688	3795	0.001	0.004	0.004
PSR2	MSE	4.36	4.38	4.63	6.42	7.88	8.19	680	3507	4656	0.012	0.020	0.020
	SD	0.13	0.23	0.32	0.53	1.13	1.48	204	2585	4204	0.001	0.003	0.006
PSR3	MSE	4.43	4.39	4.66	7.06	7.92	8.89	1676	4484	4039	0.015	0.022	0.026
	SD	0.13	0.21	0.37	0.21	1.29	1.61	446	3550	2904	0.001	0.004	0.016

other datasets, and Triazines a larger number of independent variables.

A PSR, with its random search characteristics, provides different regression models over multiple runs with the same data. In order to find better regression models by the PSRs, five replication runs for each data set were performed. Five replication models were determined by each PSR with the test data. Then, the model providing the minimum MSE by valid data was selected as the regression model for the PSR. Finally, the selected model was tested by the test data. Ten terms ($m = 10$) were used for the PSR models in Eq. (1). The other parameters were pre-tested with data, as indicated in Table 3. Three numbers of particles were examined: 100, 200 and 300. The termination rule (TR) was also tested with 100, 200, and 300 particles. Two hundred particles and TR 200 were selected for a comparison experiment, since increasing them did not improve the overall MSEs. The R package version 3.4.0 was used for the PSR procedures, which were performed on an Intel Core i5-4590

CPU @ 3.30 GHz, 4 GB RAM.

3.2 Experimental Results

Table 4 compares the average MSEs and their average standard deviations for ten independent runs. For each method, the first row shows the average MSE, and the second row shows the average standard deviation. The bold values in Table 4 are the top 3 results for the 6 methods and test data. The 3 PSRs provided better test results than the other methods for the Abalone and Machine data sets. PSR1 and PSR2 provided better average MSEs for Abalone, Machine, and Triazines than OLS, MARS or SVR. SVR provided the best average MSE for the Auto data. These results show that the PSR methods provide competitive regression models.

Table 5 shows the test MSE error, (Method MSE – Best MSE) / Best MSE, between each method and the best method. PSR1 provided the best MSE for Abalone and Triazines, and

Table 5. Test MSE error

Method	Abalone	Auto	Machine	Triazines	Average
OLS	0.105856	0.748477	0.538925	1.969327	0.840646
MARS	0.008258	0.301676	1.651818	0.187167	0.537230
SVR	0.009596	0	0.260881	0.155416	0.106473
PSR1	0	0.241871	0.047562	0	0.072358
PSR2	0.002454	0.139036	0.152940	0.047548	0.085495
PSR3	0.007278	0.237266	0	0.340737	0.146320

MSE error = (Method MSE – Best MSE) / Best MSE.

Figure 2. Test MSE errors: (a) overall test MSE errors in Table and (b) a zoom-in of the most notable area of Figure 2(a).

PSR3 provided the best MSE for Machine. PSR1 was the best in terms of the average test MSE error. The second- and third-best methods were PSR2 and SVR, respectively. Tables 4 and 5 provide the evidence of the competitiveness of the PSR regression models.

Figure 2 represents the Table 5 data graphically. Figure 2(a) shows overall test MSE errors in Table 5. Figure 2(b) provides a zoom-in of the most notable area of Figure 2(a). Most of the methods provided smaller MSE errors for the Abalone dataset. OLS and MARS provided larger average errors than the others. Average MSE errors of PSR1 and PSR2 are smaller than SVR. All of the MSE errors of PSR1, PSR2, and PSR3 were less than 25%. Most significantly, all of the MSE errors of PSR2 were less than 20%. This result shows that PSR1 was the best method for minimization of average test MSE errors. However, PSR2 showed robustness with all of the data sets.

The advantage of the PSR models is their provision of a model description using independent variables. Tables 6-8 show

PSR1, PSR2, and PSR3 model examples, respectively. With PSR1, the most valid MSE was shown at the first replication for the first run of Avalone (Table 6); with PSR2, the most valid MSE was shown at the third replication for the first run of Auto (Table 7), and with PSR3, the most valid MSE was shown at the third replication for the first run of Machine (Table 8). Each PSR model provided the minimum test MSE error. No model for Triazines is provided herein, due to the large number of independent variables.

Eq. (4) shows the model in Table 6 in mathematical form.

$$\begin{aligned}
 y = & -3.4941 - 15.5727 \cdot x_1^{0.3} \cdot x_2^{0.06} \cdot x_3^{0.02} \cdot x_4^{0.14} \cdot x_5^{0.11} \\
 & \cdot x_6^{0.04} \cdot x_7^{-0.11} + 3.4747 \cdot x_1^{0.15} \cdot x_2^{-0.03} \cdot x_3^{-0.17} \cdot x_4^{-0.2} \\
 & \cdot x_6^{-0.1} \cdot x_7^{0.12} - 4.6892 \cdot x_1^{0.26} \cdot x_2^{-0.18} \cdot x_3^{0.03} \cdot x_4^{0.01} \\
 & \cdot x_5^{-0.06} \cdot x_6^{-0.05} \cdot x_7^{-0.3} - 1.8090 \cdot x_1^{-0.1} \cdot x_2^{-0.29} \\
 & \cdot x_3^{0.02} \cdot x_4^{-0.05} \cdot x_5^{-0.12} \cdot x_6^{-0.07} \cdot x_7^{0.01} + 15.6752 \\
 & \cdot x_1^{-0.11} \cdot x_2^{0.3} \cdot x_3^{0.25} \cdot x_4^{-0.01} \cdot x_5^{0.07} \cdot x_6^{-0.03} \cdot x_7^{-0.01}
 \end{aligned}$$

Table 6. PSR1 model for first dataset of Avalone

Term index (<i>i</i>)	Coefficient (β_i)	r_{ij}						
		Independent variable index (<i>j</i>)						
		1	2	3	4	5	6	7
1	-15.5727	0.3	0.06	0.02	0.14	0.11	0.04	-0.11
2	3.4747	0.15	-0.03	-0.17	-0.2	0	-0.1	0.12
3	-4.6892	0.26	-0.18	0.03	0.01	-0.06	0.05	-0.3
4	-1.8090	-0.1	-0.29	0.02	-0.05	-0.12	-0.07	0.01
5	15.6752	-0.11	-0.3	0.25	-0.01	0.07	-0.03	-0.01
6	-3.8564	-0.03	0.27	0	0.07	0.05	-0.33	0.07
7	7.0208	0	-0.02	0	-0.23	0.21	0.09	-0.37
8	28.3323	-0.11	-0.02	0	0.49	-0.27	-0.01	0.04
9	-6.8067	-0.1	-0.05	0.13	0.11	0.22	-0.09	-0.16
10	-14.3795	-0.37	-0.29	0	-0.01	0.02	0.2	-0.01
0	-3.4941				-			

Table 7. PSR2 model for first dataset of Auto

Term index (<i>i</i>)	Coefficient (β_i)	r_{ij}						
		Independent variable index (<i>j</i>)						
		1	2	3	4	5	6	7
1	5.64E+11	-0.96	-1	-1	-1	-1	-1	-1
2	-140.3110	-0.29	-0.61	-1	1	1	-1	-0.59
3	0.08789	-0.1	-1	-1	1	1	1	0
4	-0.0291	1	0.44	0.73	-0.41	1	-0.36	0.8
5	4.31E-06	1	-0.91	0.81	0.76	1	1	-0.75
6	0.0851	0.79	-0.7	-1	1	-0.61	1	0.85
7	3946841	-1	-0.5	-0.55	-1	-1	1	0.88
8	0.0008	1	1	0.54	0.04	-0.35	-0.51	1
9	-4714388	-1	-1	-1	-0.38	-1	0.81	1
10	2.9175	-0.42	1	-0.45	-1	0.99	1	-1
0	8.5401				-			

$$\begin{aligned}
& -3.8564 \cdot x_1^{-0.03} \cdot x_2^{0.27} \cdot x_4^{0.07} \cdot x_5^{0.05} \cdot x_6^{-0.33} \cdot x_7^{0.07} \\
& + 7.0208 \cdot x_2^{-0.02} \cdot x_4^{-0.23} \cdot x_5^{0.21} \cdot x_6^{0.09} \cdot x_7^{-0.37} \\
& + 28.3323 \cdot x_1^{-0.11} \cdot x_2^{-0.02} \cdot x_4^{0.49} \cdot x_5^{-0.27} \\
& \cdot x_6^{-0.01} \cdot x_7^{0.04} - 6.8067 \cdot x_1^{-0.1} \cdot x_2^{-0.05} \cdot x_3^{0.13} \cdot x_4^{0.11} \\
& \cdot x_5^{0.22} \cdot x_6^{-0.09} \cdot x_7^{-0.16} - 14.3795 \cdot x_1^{-0.37} \cdot x_2^{-0.29} \\
& \cdot x_4^{-0.01} \cdot x_5^{0.02} \cdot x_6^{0.2} \cdot x_7^{-0.01}.
\end{aligned} \quad (4)$$

Tables 6-8 represents the mathematical model from PSR1, PSR2, and PSR3. Eq. (4) shows example mathematical model

as a mathematical form.

The r_{ij} in Tables 6-8 are all between -1 and 1 . All $\sum_{j=1}^n |r_{ij}| \leq 1$ in Tables 6 and 8. The regression models differ based on the PSR method. More than half of r_{ij} are bounded at 1 or -1 in Table 7 but not in Table 6. Some r_{ij} are zeros in Tables 6-8. The ratios of $r_{ij} = 0$ for all of the developed models are provided in Table 9. The ratio was calculated by the total number of $r_{ij} = 0$ over the total number of estimated r_{ij} . When the number of independent variables increased, the number of $r_{ij} = 0$ also increased. For Abalone, 3,500 r_{ij} (7 independent variables

Table 8. PSR3 model for first dataset of Machine

Term index (<i>i</i>)	Coefficient (β_i)	r_{ij}					
		Independent variable index (<i>j</i>)					
		1	2	3	4	5	6
1	-1145.7200	-0.73	0	0	0.16	0	0.01
2	-318.298	-0.01	-0.4	-0.02	0.19	0.01	0.33
3	0.1497	0.83	0.02	-0.01	0.02	0.02	0.05
4	0.2105	0.01	0.05	-0.01	-0.07	0.65	0.18
5	0.7012	0.16	0.71	-0.01	0.02	-0.01	0
6	1679.2390	0.05	-0.17	-0.22	-0.23	-0.02	-0.29
7	-1.8570	0.32	0.52	-0.02	0.07	0.03	-0.01
8	0.4133	0.47	0.06	0.32	-0.01	0.03	-0.06
9	307.1087	-0.36	-0.04	-0.01	0.28	0.01	0.27
10	-28.4306	-0.23	0.23	0	-0.25	0.1	0.08
0	26.2379				-		

Table 9. Ratio of $r_{ij} = 0$ (total number of $r_{ij} = 0$)

Method	Abalone	Auto	Machine	Triazines
PSR1	0.294 (1,029)	0.611 (2,140)	0.527 (1,582)	0.952 (2,8571)
PSR2	0.012 (41)	0.034 (115)	0.018 (54)	0.467 (14,009)
PSR3	0.119 (415)	0.192 (671)	0.113 (338)	0.906 (27,181)

$\times 10$ terms $\times 5$ replications $\times 10$ independent data sets) were estimated for each PSR model. Totals of 3,500, 3,000, and 30,000 r_{ij} were estimated for Auto, Machine, and Triazines, respectively, based on the number of independent variables.

PSR1 and PSR3 estimated more r_{ij} to be zero than PSR2, since they have more bounded constraints. For Abalone and Triazines, PSR1 was the best and PSR2 second best. Users can choose their model in consideration of minimum test MSE error (PSR1), robustness (PSR2), or ratio of $r_{ij} = 0$ (PSR1), according to their specific purpose. All of these models were shown to be competitive with existing methods (i.e., OLS, MARS, and SVR).

4. Conclusion

In this paper, we proposed new PSRs to solve the nonlinear regression problem. The PSR methods provide the signomial functions using the PSO method. The PSR regression models offer the benefit of explicit functions descriptions representing the nonlinearity of the given data without kernel tricks.

Also, with the PSR regression models, sparse signomial functions that can be used for prediction purposes as well as inter-

pretation purposes can be obtained. The present experimental results revealed that this approach can detect a sparse signomial nonlinear function; indeed, PSR may be considered as a useful alternative when interpreting which original input variables are more important than others in an obtained regression model. The development of nonlinear sparse regression models using other metaheuristic optimization methods will be an important focus of future studies.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgements

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (NRF-2017R1E1A1A01077375).

References

- [1] H. Drucker, C. J. Burges, L. Kaufman, A. J. Smola, and V. Vapnik, "Support vector regression machines," *Advances in Neural Information Processing Systems*, vol. 9, pp. 155-161, 1996.
- [2] C. D. Maranas and C. A. Floudas, "Global optimization in generalized geometric programming," *Computers & Chemical Engineering*, vol. 21, no. 4, pp. 351-369, 1997. [https://doi.org/10.1016/S0098-1354\(96\)00282-7](https://doi.org/10.1016/S0098-1354(96)00282-7)
- [3] J. Kennedy and R. Eberhart, "Particle swarm optimization (PSO)," in *Proceedings of IEEE International Conference on Neural Networks*, Perth, Australia, 1995, pp. 1942-1948. <https://doi.org/10.1109/ICNN.1995.488968>
- [4] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of 2007 IEEE Swarm Intelligence Symposium*, Honolulu, HI, 2007, pp. 120-127. <https://doi.org/10.1109/SIS.2007.368035>
- [5] K. P. Wang, L. Huang, C. G. Zhou, and W. Pang, "Particle swarm optimization for traveling salesman problem," in *Proceedings of the 2003 international Conference on Machine Learning and Cybernetics (IEEE cat. no. 03ex693)*, Xian, China, 2003, pp. 1583-1585. <https://doi.org/10.1109/ICMLC.2003.1259748>
- [6] C. Akjiratikarl, P. Yenradee, and P. R. Drake, "PSO-based algorithm for home care worker scheduling in the UK," *Computers & Industrial Engineering*, vol. 53, no. 4, pp. 559-583, 2007. <https://doi.org/10.1016/j.cie.2007.06.002>
- [7] M. Schwaab, E. C. Biscaia, J. L. Monteiro, and J. C. Pinto, "Nonlinear parameter estimation through particle swarm optimization," *Chemical Engineering Science*, vol. 63, no. 6, pp. 1542-1552, 2008. <https://doi.org/10.1016/j.ces.2007.11.024>
- [8] Y. Bao, Z. Hu, and T. Xiong, "A PSO and pattern search based memetic algorithm for SVMs parameters optimization," *Neurocomputing*, vol. 117, no. 98-106, 2013. <https://doi.org/10.1016/j.neucom.2013.01.027>
- [9] D. Wang and X. Wu, "The Optimization of SVM Parameters Based on PSO," *Computer Applications*, vol. 28, no. 1, pp. 134-139, 2008.
- [10] X. Yuan and A. Liu, "The research of SVM parameter selection based on PSO algorithm," *Techniques of Automation and Application*, vol. 26, no. 5, pp. 5-8, 2007.

SeJoon Park received the Industrial Engineering Master and Ph.D. degrees from University of Florida in 2005 and the Ph.D. from Oregon State University in 2012, respectively. Currently, he is an Assistant Professor, Division of Energy Resources Engineering and Industrial Engineering at Kangwon National University. His research interests include big data analysis and optimization.

E-mail: sonmupsj@kangwon.ac.kr

NagYoon Song received his M.S. degree in Industrial and Management Engineering from Myongji University in 2016. His research interests anomaly detection, explainable artificial intelligence (XAI) and architecture search based deep learning.

E-mail: nysong@mju.ac.kr

Wooyeon Yu is currently a professor of Industrial and Management Engineering at Myongji University. He received his Ph.D. degree in the Department of Industrial and Manufacturing Systems Engineering from Iowa State University. Before he joined Myongji University, he worked as a senior consultant at Samsung SDS. His primary research interests are in material handling, logistics and supply chain management.

E-mail: wyyuie@mju.ac.kr

Dohyun Kim received his M.S. and Ph.D. degrees in industrial engineering from KAIST, Korea, in 2002 and 2007, respectively. Currently, he is an Associate Professor with the Department of Industrial and Management Engineering, Myongji University. His research interests include statistical data mining, deep learning and graph data analysis.

E-mail: ftgog@mju.ac.kr