Iraqi Journal for Computer Science and Mathematics

Journal Homepage: http://journal.esj.edu.iq/index.php/IJCM
e-ISSN: 2788-7421 p-ISSN: 2958-0544

Kernel semi-parametric model improvement based on quasi-oppositional learning pelican optimization algorithm

Firas Ahmed Yonis AL-Taie[1,*], Omar Saber Qasim[1], Zakariya Yahya Algamal[2*]

[1]Department of Mathematics, University of Mosul, Mosul, 41014, IRAQ
[2]Department of Statistics and Informatics, University of Mosul, Mosul, 41014, IRAQ

*Corresponding Author: Firas Ahmed Yonis AL-Taie

DOI: https://doi.org/10.52866/ijcsm.2023.02.02.013
Received December 2022 ; Accepted March 2023 ; Available online April 2023

ABSTRACT: Statistical modeling plays a critical role in various scientific fields as it offers an understanding of how the response variable of interest is linked to a range of explanatory variables. However, selecting the best model beforehand is not easy. To address this issue, we conducted a study to explore the selection of an appropriate hyper-parameter for kernel semi-parametric regression. We used a quasi-oppositional learning pelican optimization algorithm technique to choose the smoothness parameter. Our simulation results showed that our proposed methodology outperformed its competitors in terms of mean squared error (MSE). Moreover, our suggested quasi-oppositional learning pelican optimization technique demonstrated superior performance to the CV and GCV methods in computational time, as evidenced by experimental results and statistical analyses.

Keywords: semi-parametric regression model; Pelican optimization algorithm (POA); cross-validation; quasi-oppositional learning.

1. INTRODUCTION

Because it clarifies the connection that exists between both the response variable of interest and a number of additional variables serving as explanatory factors, statistical modeling is an important tool in a variety of scientific study fields. To study and analyze the relationship between variables, one of the most common types of statistical analysis is regression analysis, which is also one of the most extensively used statistical techniques. There is a diverse array of fields in which regression may be utilized, covering a wide range of disciplines, from engineering and the hard sciences to business and sociology.

Numerous disciplines, including economics and business, have paid close attention to the semi-parametric regression model because of its efficacy in extracting insights from data, and because of the versatility with which they can mimic occurrences [1, 2]. For example, take the model:

$$y_j = x_j\alpha + g(t_j) + \varepsilon_j, j = 1,2,\ldots,N, \qquad (0)$$

Where $\alpha = (\alpha_1,\alpha_2,\ldots,\alpha_f)'$ is a vector of unknown f-dimensional parameters, $x_j = (x_{j1}, x_{j2},\ldots,x_{if})$ is a vector of explanatory variables, the t_j's are known and non-random in some bounded domain $D \subset \mathbb{R}$, $g(t_j)$ is a vector of unknown smooth functions, and ε_j's are independent and identically distributed random errors with mean 0 and variance σ^2 that are unrelated to (x_j, t_j) [1]. Many semiparametric regression methods borrow ideas from other, more generalized methods of estimating regression parameters.

The rest of the study is laid out as follows: in Section 2, we introduce the kernel semi-parametric regression model. As shown in Section 3, the proposed method is demonstrated. The fourth section involves a simulated investigation and a subsequent application. Section 5 offers some parting thoughts.

2. Kernel semi-parametric regression model

The kernel estimate method is one of the significant statistical tools, and it has a wide variety of applications in the real world [4-7]. It has been effectively utilized, both to univariate and multivariate situations, with positive results. Kernel estimation is a method of data smoothing that involves estimating a density function with the use of independent sample data that is dispersed in a same fashion. Exploratory data analysis and visualization are two applications that make use of the data mining approach known as kernel estimation [8].

Both parametric and nonparametric kernel estimators are wide categories that may be used to organize kernel estimators. Estimators that use parametric kernels begin with the assumption of a distribution's functional form, which reduces the complexity of the task to just estimating the distribution's parameters. It is not necessary to use a functional form in order to estimate the density using a nonparametric kernel estimator. However, nonparametric estimators often demand more computing effort than their parametric counterparts despite providing more flexibility [9-11].

In other words, we can take the $Y = (Y_1, \ldots, Y_N)^T$ be a (N × 1) continuous outcomes as a vector, with *Yj* being the value for the subject$j = 1, \ldots, N$, and Z and X are (N × f) and (N × p) variable predictor matrices , respectively, where Y is shown below to be dependent on both X and Z. This is the kernel semi-parametric model:

$$Y = X\alpha + o(Z) + e \qquad (2)$$

where α represents the linear parametric regression coefficients with a size of (f × 1), o(.) is a function with unknown smoothness, and e is a vector of independent and homoscedastic errors with a size of (N × 1), such as $e_i \sim N(0, \sigma^2)$. The linear component of the model is denoted by the Xα, and it is presumed that element X includes an intercept. The "kernel component" is o(Z). The function o(.) may be found in the function space denoted by the notation O(k), which was formed by the kernel function k (.,.). K is the (N × N) Gram matrix of k (.,.), and it is constructed in such a way that j indicates the degree to which subjects i and Z are alike $K_{ji} = k(Z_j, Z_i)$. The procedure for estimating O() involves expressing it as a "linear combination" of k (.,.) in such a way that:

$$o = K\beta \qquad (3)$$

The kernel method's primary objective is to map $R = [Y_1, Y_2, \ldots, Y_n]$ into a higher-dimensional space of features through nonlinear transformations. Cover's theorem states that problems that were previously intractable in the original low-dimensional space are now simply decomposable into linear subproblems and may be solved with relative ease in the new space [2]. The kernel method calculates the dot products of the input data into the Hilbert space as a function of the input data. A function of this kind is known as Mercer's kernel [3-5]. The result of the inner product of two dots z_j ? z_i, is mapped as. It is possible to compute the dot product in the modified space using the formula $K(z_j, z_i) = \theta(z_j)$? $\theta(z_i)$ in the original space$\mathbb{R}^d$. So,

$$d^2(z_j, z_i) = \|\theta(z_j) - \theta(z_i)\|^2 = (\theta(z_j), \theta(z_j)) + (\theta(z_i), \theta(z_i)) - 2(\theta(z_j), \theta(z_i))$$
$$= K(z_j, z_j) + K(z_i, z_i) - 2K(z_j, z_i) \qquad (4)$$

The "Gaussian kernel function" are $k(Z_j, Z_i) = exp(-\| Z_j - Z_i \|^2/\delta)$ where $\delta > 0$ is typically employed.

This kernel requires tuning, namely of the settings noted δ. In actual practice, the user decides on these tuning parameters or they can be calculated using techniques like cross-validation (CV) or generalized cross-validation (GCV).

3. The proposed method

The optimization solution for the objective function is highly dependent on the values of the hyper-parameters that define the kernel type, as stated in Equation (3). At present, there is no mathematically rigorous method for determining the precise values that should be used for these hyper-parameters.[6, 7]. despite the fact that their selection has a profound impact on the performance of the kernel function.

The selection of these hyper-parameters is thus crucial to the study of kernel semi-parametric regression models. The CV and GCV method is the most used approach to determining hyper-parameters in the literature. It's been pointed out, nevertheless, that this process is not always quick. [8].

The natural world has served as a source of motivation for the development of several meta-heuristic algorithms. In the realm of scientific study, swarm intelligence is indeed a crucial instrument for the solution of many difficult issues. [9, 10]. There has been a lot of research done on swarm intelligence algorithms, and they have been effectively applied to a range of difficult optimization issues. [7, 11-26]. Pelican optimization algorithm (POA) proposed by Trojovsky and

Dehghani [27] is based on the manner in which pelicans go about their hunts and the tactics they use, which is a thoughtful adaptation that has enabled these birds to become proficient hunters.

The POA is an algorithm that is built on a population, and in this algorithm, pelicans are considered to be members of this population. In algorithms that are based on populations, each member of the population represents a possible solution. Every individual in the population has a location inside the search space, and this determines the values that they suggest for the variables that make up the optimization issue. In the beginning, population members are chosen at random and then initialized in accordance with the problem's lower bound and upper bound by employing eq. (5)

$$x_{j,i} = I_i + rand.(U_i - h_i), j = 1,2,\ldots,n, i = 1,2,\ldots m \quad (5)$$

where $x_{j,i}$ is the *ith* variable's value as stated by the *jth* "candidate solution", n determines how big the populace, m denotes the total number of variables to consider, $rand$is a random and $\in R$ so that it is between 0 and 1,h_i and $U_i$is the *ith* lower and upper bound respectively of the variables problem.

In Eq. 6, a matrix referred as the population matrix is used to identify the individuals of the population of pelicans that live in the POA. The suggested solutions to the problem variables are listed in the columns of this matrix, while the rows of this matrix each indicate a potential answer to the problem.

$$X = \begin{bmatrix} X_1 \\ . \\ . \\ X_2 \\ . \\ . \\ X_n \end{bmatrix}_{n\times M} = \begin{bmatrix} x_{1,1}\ldots x_{1,i}\ldots x_{1,M} \\ \ldots \\ \ldots \\ x_{j,1}\ldots x_{j,i}\ldots x_{J,M} \\ \ldots \\ \ldots \\ x_{n,1}\ldots x_{n,i}\ldots x_{n,M} \end{bmatrix}_{n\times M} \quad (6)$$

Where X_j is the *jth* pelican and X is the pelican population matrix.

One of the potential answers to the problem at hand is for the whole population of POA to assume the shape of pelicans; this is the short way to solve the problem. As a consequence of this, the potential solutions may individually be used to conduct an analysis of the objective function associated with the given problem. In the equation, there is a vector that is referred to as the objective function vector. This vector is used to calculate the values that are produced for the objective function as seen in Eq. (7)

$$F = \begin{bmatrix} F_1 \\ . \\ . \\ F_2 \\ . \\ . \\ F_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} G(X_1) \\ . \\ . \\ G(X_2) \\ . \\ . \\ G(X_3) \end{bmatrix}_{n\times 1} \quad (7)$$

Where: F is the vector of the "objective function", and F_j is the value of the objective function for the *jth* candidate solution. In order to improve potential solutions, the POA runs simulations that model the actions and strategies that pelicans employ as they attack and pursue their prey. The simulation of this hunting method takes place in two stages:

(i) Getting closer to the prey (exploration phase).

(ii) Floating on the surface of the sea with wings (exploitation phase).

3.1 Phase 1: Getting closer to the prey (exploration phase)

The initial step of the process involves the pelicans determining the position of the prey and then moving toward the region that they have determined. The modeling of this pelican's technique leads to the scanning of the search space, which demonstrates the exploration capacity of the suggested point of attack in locating various regions of the search space. The fact that the position of the prey is chosen at random inside the search space is one of the most significant

aspects of POA. This increases POA's ability for exploration in the exact search for the space that may hold viable answers to the issue. Eq. (8) provides a mathematical representation of the aforementioned ideas as well as the approach used by pelicans as they go to their hunting grounds

$$x_{\mathrm{j,i}}^{f_1} = \begin{cases} x_{\mathrm{j,i}} + rand.(\mathrm{p}_i - \mathrm{I}.\mathrm{x}_{\mathrm{j,i}}), \mathrm{G}_f < G_j; \\ x_{\mathrm{j,i}} + rand.(x_{\mathrm{j,i}} - f_i), else, \end{cases} \quad (8)$$

I is a random integer that can be either one or two only, f_i is the position of prey in the *ith* dimension, and G_f is the value of its objective function. $x_{j,i}^{f_1}$ represents the new state of the *ith* pelican in the *jth* dimension based on phase 1. The value of the parameter I, which can either be 1 or 2, will be chosen at random. This parameter is chosen at random for each iteration as well as for each member of the group. It delivers additional displacement for a member whenever the value of the parameter is exactly two, which might lead that member to discover newer locations of the search space. As a result, the POA exploration power is affected by parameter I, which makes it more difficult to correctly scan the search space. If the value of the goal function is better in a new position for a pelican, then that new position will be considered acceptable. When doing this kind of updating, which is known as efficient updating, the algorithm is stopped from going to places that are not optimal. The equation used to represent this process is shown below in Eq (9).

$$X_j = \begin{cases} X_j^{f_1}, G_j^{f_1} < G_j \\ X_j, else, \end{cases} \quad (9)$$

where $X_j^{f_1}$ represents the upgraded standing of the *jth* pelican and $G_j^{f_1}$ represents the objective function value attained in Stage 1.

3.2 Phase 2: Floating on the surface of the sea with wings (exploitation phase)

Following the first step, which consists of swimming to the surface of the water, for the next part of the process, the pelicans use the water's surface to lift the fish using their wings, after which they gather the prey in the pouch in their throat. Pelicans are able to capture a greater number of fish in the region that is being assaulted as a result of this tactic. The POA is made to approach to better places in the hunt area as a result of modeling this movement of pelicans. This procedure enhances both the power of the local search and the capability of exploitation offered by POA. In order to arrive at a more optimal answer from a mathematical perspective, the algorithm will need to conduct an investigation into the spots that are located close to where the pelican is located. This hunting strategy utilized by pelicans is modeled mathematically in the form of an Eq (10).

$$x_{j,i}^{f_2} = x_{j,i} + C.(1 - \frac{t}{T}).(2.rand - 1)x_{j,i}, \quad (10)$$

where $x_j^{p_2}$ represents the new status of the *jth* pelican in the *ith* dimension based on this phase, C represents a constant that has a value of 0.2, $C.(1 - \frac{t}{T})$ represents the neighborhood radius of $x_{j.i}$ while t represents the iteration counter, and T represents the maximum number of iterations that can be performed. The value $C.(1 - \frac{t}{T})$ indicates the distance of the neighborhood in which the members of the population are situated in order to search locally near every member in an attempt to converge on a solution that is more optimum. This is done in order to find the optimal solution as quickly as possible. . This parameter has an influence on the power of exploitation of the POA, which brings us that much closer to the ideal global solution. The value of this coefficient is set to be high at the beginning of the iterations, and as a consequence, a bigger region surrounding each member is taken into consideration. The "$C.(1 - \frac{t}{T})$" coefficient will drop as the number of replicated copies of the method rises, which will lead to lower radii for the neighborhoods of every member. Because of this, we are able to scan the region surrounding each individual in the population with steps that are both smaller and more precise, which enables the POA to converge on solutions that are closer to the global solutions.

At this point, effective updating has also been utilized to either accept or reject the newly proposed pelican posture, which is depicted in Eq. (11)

$$X_j = \begin{cases} X_j^{f_2}, G_j^{f_2} < G_j; \\ X_j, else, \end{cases} \quad (11)$$

where $x_j^{f_2}$ is the new position of the *jth* pelican and $G_j^{f_2}$ is the value of its objective function based on phase 2 of the analysis.

The suggested strategy will assist in effectively locating the optimal value while maintaining a high level of prediction performance. In addition, the use of "quasi-oppositional" learning is suggested as a method for enhancing POA's performance in determining the optimal value of the hyper-parameter.

By using the O-BL method [28], Tizhoosh was the first to successfully improved the evolutionary trajectory. In order to deal with this issue, academics proposed a variety of asymmetrical learning strategies, such as the ones listed below:

Let $a \in [x, y]$ where $a \in R$, and $\overline{a}$ be an opposite number defined as:

$$\overline{a} = x + y - a \quad (12)$$

if $\{x, y\} = \{0,1\}$ we have:

$$\overline{a} = 1 - a \quad (13)$$

we may define the inverse in a multidimensional instance in a manner similar to that used here. When we have a point $p(a_1, a_2, \ldots\ldots\ldots, a_m)$ in m-dimensional. where $a_1, a_2, \ldots\ldots\ldots, a_m \in R$ and $x_i \in [x_i, y_i]$, Then an opposite point $\overline{P}$ is $\overline{a_1}, \overline{a_2}, \ldots\ldots\ldots, \overline{a_m}$ as follow:

$$\overline{a_i} = x_i + y_i - a_i i = 1, \ldots\ldots\ldots, m \quad (14)$$

In conclusion, we assume $S(a)$ to be the primary function and $Q(.)$ to be an assessment function for determining optimality. From the above, we get two values, a and $\overline{a}$, where a indicates a random beginning value in $[x, y[]]$ and $\overline{a}$ indicates the opposite value; we use these values to calculate $S(a)$ and $S(\overline{a})$ in each iteration, which we then apply to the evaluation function. $Q(.)$, if $Q(S(a)) \geq Q(S(\overline{a}))$ we'll go with a ; otherwise, we'll go with $\overline{a}$. Rahnamayan et al. [29] created the first-ever "Quasi Oppositional" (QO) based learning strategy to solve the problem at hand by contrasting the actual population with the ideal one. Additionally, it was shown that a quasi-opposite number is often more near to the answer than an opposite number [30]. the quasi-opposite number, $\overline{a_p}$ is provided for any real number such as $P(a_1, a_2, \ldots\ldots\ldots, a_m)$, subject to $a_i \in [x_i, y_i]$ in the same way as the opposite of a real number in D-dimensional space was defined in O-BL:

$$\overline{a_{pi}} = rand(\frac{x_i + y_i}{2}, \overline{a}_i) i = 1, \ldots\ldots\ldots, m. \quad (15)$$

Below, we detail the many parameters sets that can be used with our suggested technique:

(1) 25 pelicans have been allotted, and $t{=}500_{max}$ iterations have been planned out.

(2) Each pelican's location is chosen at random. The hyper-parameter f is depicted here as the pelican in its typical position. Initial pelican locations are drawn from a U-distribution in the interval [0.5,12].

(3) The "fitness function" is characterized by the formula which is as follows:

$$\text{fitness} = min \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2, \quad (16)$$

(4) Eq. (8) and (10) are used to recalculate the locations.

(5) Iterate Step (3) and Step (4) to $t_{\max}$ is achieved.

4. Experimental results

Here we compare our proposed method, POA, to the CV and GCV, as well as two other similar algorithms: the quasi-oppositional learning pelican optimization algorithm (QOL-POA) and the oppositional learning pelican optimization algorithm (OL-POA). Within Eq (10). When using $n = 30,50,100,150.$ four sample sizes are considered. In order to boost the efficiency of the existing algorithms, four distinct models are investigated. There are 25 iterations of the simulation.

Model 1 [31]:

$$y_j = \sum_{i=1}^{6} x_{ji}\, \alpha_i + g(t_j) + \varepsilon_j, j = 1, \ldots, N, \quad (17)$$

where $\beta = (3,1,-3,2,-5,4)'$ and $g(t) = \frac{1}{3}[\Phi(t; -3, 0.81) + \Phi(t; 0, 0.36) + \Phi(t; 3, 0.81)]$, Which is a mixture of normal densities for $t \in [-5,5]$ with a normal density function $\Phi(x; \mu, \sigma^2)$ with mean μ and

variance σ^2. Nonlinear components with this particular structure were chosen primarily to test the efficacy of nonparametric estimation for wavy functions.

Model 2 [32]:

$$y_j = sin\left(2\pi x_{1,i}()\left(1 - x_{2,i}()\left(1 + x_{2,i}()\frac{2x_{3,j}}{1+0.8x_{3,i}^2+\varepsilon_j \sim N(0,0.09^2)}\right)\right)\right) \quad (18)$$

All of the independent variables, ranging in value from 0 to 1, are produced by a normal distribution.

Model 3:

$$y_j = \sum_{i=1}^{5\Sigma} z_{ji}\alpha_i + g(u_j) + \varepsilon_j, j = 1,2,\ldots,N \quad (19)$$

Where$\alpha = (1.5,2,3,-5,4)^T$, $u_j = \frac{(j-0.5)}{n}$, and

$$g(u_j) = \sqrt{u_j(1-u_j)} * sin\left(\frac{2.1\pi}{u_j+0.05}\right) \quad (20)$$

With$\sigma^2 = 0.01$.

Model 4:

$y_j = \sum_{i=1}^{6\Sigma} z_{ji}\alpha_i + g(u_j) + \varepsilon_j, j = 1,2,\ldots,N$ (21)

where $\beta = (-1.-1,2,3,-5,4)^T$ and $g(u_j) = sin(2\,u_j) * cos(5\,u_j)$, $\varepsilon_j \sim N_n(0,0.09)$.

Tables 1 through 4 provide the MSE-based comparison findings of the proposed POA, QOL-POA, OL-POA, and other algorithms. After examining the data, we conclude that QOL-POA performed best, with the lowest outcomes compared to the other algorithms. This is then followed by the OL-POA, which had the least MSE value, while the CV and GCV approaches were the worst. The suggested approach, QOL-POA, reduces MSE by a larger margin than CV and GCV do, and by a smaller margin than OL-POA and POA do. The MSE was reduced by 20.95% and 17.68%, respectively, when using the QOL-POA over the CV and GCV approaches for Model 2 at n=150. When compared to POA and OL-POA, however, it was only 6.41 percent and 5.58 percent, respectively. The MSE values, in relation to the sample size, decrease as the sample size grows. QOL-POA, the suggested technique, continues to outperform the competition across all models.

Table 1: The average MSE of Model 1.

Method	n=30	n=50	n=100	n=150
CV	1.8081	1.5975	1.4971	1.2942
GCV	1.7724	1.5618	1.4614	1.2583
POA	1.5161	1.3055	1.2051	1.0022
OL-POA	1.4558	1.2452	1.1448	0.9417
QOL-POA	1.2871	1.0765	0.9761	0.7731

Table 2: The average MSE of Model 2.

Methods	n=30	n=50	n=100	n=150
CV	2.7688	2.5582	2.4578	2.2547
GCV	2.6791	2.4685	2.3681	2.1652
POA	2.4183	2.2077	2.1073	1.9042
OL-POA	2.4019	2.1913	2.0909	1.8878
QOL-POA	2.2964	2.0858	1.9854	1.7823

Table 3: The average MSE of Model 3.

Methods	n=30	n=50	n=100	n=150
CV	3.7942	3.5836	3.4832	3.2801
GCV	3.7045	3.4939	3.3935	3.1904
POA	3.4437	3.2331	3.1327	2.9296
OL-POA	3.4273	3.2167	3.1163	2.9132
QOL-POA	3.3218	3.1112	3.0108	2.8077

Table 4: The average MSE of Model 4.

Methods	n=30	n=50	n=100	n=150
CV	3.6601	3.4495	3.3491	3.1464
GCV	3.5704	3.3598	3.2594	3.0563
POA	3.3096	3.0992	2.9986	2.7955
OL-POA	3.2932	3.0826	2.9822	2.7791
QOL-POA	3.1877	2.9771	2.8767	2.6736

To provide additional evidence of the effectiveness of our proposed method, time averages for 25 iterations are presented in Tables 5-8. A more efficient algorithm is indicated by a shorter time to complete. We suggested QOL-POA algorithm demonstrated superior performance to its competitors in Tables 5-8. It outperformed the other three models and was the fastest overall. The OL-POA method was the second fastest overall, while the CV and GCV methods were the slowest. Wilcoxon's rank sum test (a nonparametric statistical test) p-values (*) are used, with a 5% significance threshold. Statistical analysis is required to show that QOL-POA is demonstrably superior than competing algorithms. Across all models, a statistically significant gap can be detected between QOL-POA and both CV and GCV.

The findings of the aforementioned comparisons suggest that the proposed QOL- POA has the potential to outperform its main rivals, CV and GCV.

Table 5: Model 1 computational time in seconds.

Methods	n=30	n=50	n=100	n=150
CV	88*	101*	109*	122*
GCV	75*	87*	96*	110*
POA	68*	80*	89	102*
OL-POA	60	73*	81	92
QOL-POA	51	63	72	85

Table 6: Model 2 computational time in seconds.

Methods	n=30	n=50	n=100	n=150
CV	111*	121*	130*	143*
GCV	96*	108*	117*	130*
POA	88	101*	110*	123*
OL-POA	83	93	102*	115
QOL-POA	72	84	93	106

Table 7: Model 3 computational time in seconds.

Methods	n=30	n=50	n=100	n=150
CV	122*	134*	143*	156*
GCV	109*	121*	130*	143*
POA	102*	114*	123*	136*
OL-POA	94	106	115	128
QOL-POA	85	97	106	119

Table 8: Model 4 computational time in seconds.

Methods	n=30	n=50	n=100	n=150
CV	113*	125*	134*	147*
GCV	100*	112*	121*	134*
POA	93*	105*	114*	127*
OL-POA	85	97	106	119
QOL-POA	76	88	97	110

5. Conclusion

The objective of this study is to investigate the selection of a hyper-parameter for a kernel semi-parametric regression model. To achieve this, we developed a quasi-oppositional learning pelican optimization technique. The simulation results indicated that our proposed approach outperformed other competing methods in terms of mean squared error. Furthermore, the practical results and statistical analysis demonstrate that our quasi-oppositional learning pelican optimization method is more efficient in terms of running time than the CV and GCV methods.

Funding

None

ACKNOLEDGEMENT

The author we would like to thank the reviewers for their valuable contribution in the publication of this paper.

CONFLICTS OF INTEREST

The author declares no conflict of interest.

REFERENCES

[1] H. Emami, "Local influence for Liu estimators in semiparametric linear models," *Statistical Papers*, vol. 59, pp. 529-544, 2016.

[2] H. Tamiminia, S. Homayouni, H. McNairn, and A. Safari, "A particle swarm optimized kernel-based clustering method for crop mapping from multi-temporal polarimetric L-band SAR observations," *International Journal of Applied Earth Observation and Geoinformation*, vol. 58, pp. 201-212, 2017.

[3] F. d. A. T. de Carvalho, E. C. Simões, L. V. C. Santana, and M. R. P. Ferreira, "Gaussian kernel c-means hard clustering algorithms with automated computation of the width hyper-parameters," *Pattern Recognition*, vol. 79, pp. 370-386, 2018.

[4] S. Srivastava and M. Tushir, "Exploring different kernel functions for kernel-based clustering," *International Journal of Artificial Intelligence and Soft Computing*, vol. 5, 2016.

[5] B. Tavakkol and Y. Son, "Fuzzy kernel K-medoids clustering algorithm for uncertain data objects," *Pattern Analysis and Applications*, vol. 24, pp. 1287-1302, 2021.

[6] Z. Y. Algamal, M. K. Qasim, M. H. Lee, and H. T. Mohammad Ali, "Improving grasshopper optimization algorithm for hyperparameters estimation and feature selection in support vector regression," *Chemometrics and Intelligent Laboratory Systems*, vol. 208, 2021.

[7] O. M. Ismael, O. S. Qasim, and Z. Y. Algamal, "Improving Harris hawks optimization algorithm for hyperparameters estimation and feature selection in v-support vector regression based on opposition-based learning," *Journal of Chemometrics*, vol. 34, 2020.

[8] N. A. Al-Thanoon, O. S. Qasim, and Z. Y. Algamal, "Tuning parameter estimation in SCAD-support vector machine using firefly algorithm with application in gene selection and cancer classification," *Computers in Biology and Medicine*, vol. 103, pp. 262-268, 2018.

[9] Z. Y. Algamal, "A new method for choosing the biasing parameter in ridge estimator for generalized linear model," *Chemometrics and Intelligent Laboratory Systems*, vol. 183, pp. 96-101, 2018.

[10] M. A. Kahya, S. A. Altamir, and Z. Y. Algamal, "Improving firefly algorithm-based logistic regression for feature selection," *Journal of Interdisciplinary Mathematics*, vol. 22, pp. 1577-1581, 2020.

[11] A. A. Ewees, M. A. A. Al-qaness, L. Abualigah, D. Oliva, Z. Y. Algamal, A. M. Anter, *et al.*, "Boosting Arithmetic Optimization Algorithm with Genetic Algorithm Operators for Feature Selection: Case Study on Cox Proportional Hazards Model," *Mathematics*, vol. 9, 2021.

[12] M. Mijwil, A. Mohammad, and ChatGpt, "Towards Artificial Intelligence-Based Cybersecurity: The Practices and ChatGPT Generated Ways to Combat Cybercrime," Iraqi Journal For Computer Science and Mathematics, vol. 4, no. 1, pp. 65-70, 01/19 2023.

[13] O. S. Qasim, N. A. Al-Thanoon, and Z. Y. Algamal, "Feature selection based on chaotic binary black hole algorithm for data classification," *Chemometrics and Intelligent Laboratory Systems*, vol. 204, 2020.

[14] O. S. Qasim and Z. Y. Algamal, "Feature selection using particle swarm optimization-based logistic regression model," *Chemometrics and Intelligent Laboratory Systems*, vol. 182, pp. 41-46, 2018.

[15] S. Atheel Sabih, F. Y. Omar, A. Mohammad, Z. Abbood, and S. M. Mahdi, "SEEK Mobility Adaptive Protocol Destination Seeker Media Access Control Protocol for Mobile WSNs," Iraqi Journal For Computer Science and Mathematics, vol. 4, no. 1, pp. 130-145, 01/07 2023.

[16] O. S. Qasim and Z. Y. J. T. S. Algamal, "Improving Feature Selection for Credit Scoring Classification Using a Novel Hybrid Algorithm," vol. 19, pp. 593-605, 2021.

[17] G. T. Basheer and Z. Y. Algamal, "Improving Flower Pollination Algorithm for Solving 0–1 Knapsack Problem," *Journal of Physics: Conference Series*, vol. 1879, 2021.

[18] A. M. N. Al Radhwani and Z. Y. Algamal, "Improving K-means clustering based on firefly algorithm," *Journal of Physics: Conference Series*, vol. 1897, 2021.

[19] S. G. Mahmood Al-Kababchee, O. S. Qasim, and Z. Y. Algamal, "Improving penalized regression-based clustering model in big data," *Journal of Physics: Conference Series*, vol. 1897, 2021.

[20] M. A. Kahya, S. A. Altamir, and Z. Y. Algamal, "Improving whale optimization algorithm for feature selection with a time-varying transfer function," *Numerical Algebra, Control & Optimization*, vol. 11, 2021.

[21] M. Al-Janabi and M. A. Ismail, "Improved intrusion detection algorithm based on TLBO and GA algorithms," Int. Arab J. Inf. Technol., vol. 18, no. 2, pp. 170-179, 2021.
[22] G. Y. Abdallh and Z. Y. J. E. J. o. A. S. A. Algamal, "A QSAR classification model of skin sensitization potential based on improving binary crow search algorithm," vol. 13, pp. 86-95, 2020.
[23] A. M. Alharthi, M. H. Lee, Z. Y. Algamal, and A. M. Al-Fakih, "Quantitative structure-activity relationship model for classifying the diverse series of antifungal agents using ratio weighted penalized logistic regression," *SAR QSAR Environ Res*, vol. 31, pp. 571-583, Aug 2020.
[24] Z. Y. Algamal, "Shrinkage parameter selection via modified cross-validation approach for ridge regression model," *Communications in Statistics - Simulation and Computation*, vol. 49, pp. 1922-1930, 2018.
[25] Z. A. Basheer and Z. Y. J. I. J. O. S. S. Algamal, "Smoothing parameter selection in Nadaraya-Watson kernel nonparametric regression using nature-inspired algorithm optimization," vol. 17, pp. 62-75, 2020.
[26] N. A. Al-Thanoon, O. S. Qasim, and Z. Y. Algamal, "Tuning parameter estimation in SCAD-support vector machine using firefly algorithm with application in gene selection and cancer classification," *Comput Biol Med*, vol. 103, pp. 262-268, Dec 1 2018.
[27] P. Trojovsky and M. Dehghani, "Pelican Optimization Algorithm: A Novel Nature-Inspired Algorithm for Engineering Applications," *Sensors (Basel)*, vol. 22, Jan 23 2022.
[28] H. R. Tizhoosh, "Opposition-based learning: a new scheme for machine intelligence," in *International conference on computational intelligence for modelling, control and automation and international conference on intelligent agents, web technologies and internet commerce (CIMCA-IAWTIC'06)*, 2005, pp. 695-701.
[29] A.Sleem , I. Elhenawy, Survey of Artificial Intelligence of Things for Smart Buildings: A closer outlook, Journal of Intelligent Systems and Internet of Things, Vol. 8 , No. 2 , (2023) : 63-71 (Doi : https://doi.org/10.54216/JISIoT.080206)
[30] I. M. H. Naveh, E. Rakhshani, H. Mehrjerdi, and M. A. Elsaharty, "A Quasi-Oppositional Method for Output Tracking Control by Swarm-Based MPID Controller on AC/HVDC Interconnected Systems With Virtual Inertia Emulation," *IEEE Access*, vol. 9, pp. 77572-77598, 2021.
[31] E. Akdeniz, F. Akdeniz, and M. Roozbeh, "A new difference-based weighted mixed Liu estimator in partially linear models," *Statistics*, vol. 52, pp. 1309-1327, 2018.
[32] H. L. Shang, X. Zhang, and M. H. L. Shang, "Package 'bbemkr'," 2014.