

RESEARCH ARTICLE

Novel Statistical Regularized Extreme Learning Algorithm to Address the Multicollinearity in Machine Learning

HASAN YILDIRIM 

Kamil Özdağ Faculty of Science, Department of Mathematics, Karamanoğlu Mehmetbey University, 70100 Karaman, Türkiye

e-mail: hasanyildirim@kmu.edu.tr


ABSTRACT The multicollinearity problem is a common phenomenon in data-driven studies, significantly affecting the performance of machine learning algorithms during the process of extracting information from data. Despite its widespread use across various fields, the extreme learning machine (ELM) also suffers from multicollinearity issues. To address this challenge, the ridge and Liu estimators, drawn from statistics literature, have been integrated into ELM theory, resulting in a notable advancement. This study aims to further enhance the capabilities of ridge and Liu estimators within the ELM framework by introducing two innovative two-parameter algorithms (TP1-ELM and TP2-ELM) that simultaneously incorporate both estimators. The proposed algorithms undergo comprehensive benchmarking against ELM, ELM-based algorithms, and other commonly used machine learning techniques across seven diverse datasets. Benchmark results demonstrate that the proposed algorithms consistently outperform both ELM-focused approaches and traditional machine learning algorithms on most datasets, yielding more generalizable and stable results. These findings suggest that the proposed algorithms offer a promising alternative to traditional machine learning techniques for regression and classification tasks, particularly in scenarios where multicollinearity is a concern.

INDEX TERMS Extreme learning machine, Liu estimator, machine learning, multicollinearity, ridge estimator, Tikhonov regularization.

I. INTRODUCTION

In parallel with the facilitation of the acquisition of data, the need to make sense of it and transform it into valuable insights has emerged. For this purpose, numerous algorithms have been developed in the field of artificial intelligence, especially in the branch of machine learning. Artificial neural network-oriented algorithms stand out among these algorithms and continue to be intensively developed in the academic field.

Artificial neural networks have been used in practically every field due to its capabilities such as i) capturing complex patterns, ii) creating non-linear models, iii) flexibility. In general, artificial neural networks are classified into feed-forward and feed-back neural networks, with feed-forward networks leading the way with its simplicity [1],

The associate editor coordinating the review of this manuscript and approving it for publication was Diego Bellan .

[2]. Feed-forward networks, studied first in the literature, have a number of parameters to be determined, such as learning rate, momentum, period, stopping criteria, input weights and biases. Huang et al. [3], [4], proposed the extreme learning machine (ELM) algorithm as a single-layer feed-forward neural network capable of being used in both regression and classification tasks, essentially eliminating the process of determining such parameters and avoiding risks related to slowness, local optimum and over-fitting.

The ELM algorithm assigns input layer weights and hidden layer parameters (e.g. biases) randomly and provides a closed-form solution, thereby achieving i) faster trainability, ii) reasonable generalization performance, iii) universal approximation capability, iv) applicability in machine learning domains such as regression, classification and clustering, and v) relatively low human intervention [5]. Due to the consequence of such capabilities, it has been used in a wide range of fields serving varied objectives. The main

prominent studies can be presented as telecommunication [6], [7], [8], [9], [10], environmental sciences [11], [12], [13], robotics [14], [15], [16], computer sciences [17], [18], [19], [20], [21], agriculture [22], [23], management [24], [25], medical sciences [26], [27], [28], economics [29], [30], psychology [31], chemistry [32], [33], [34], [35] in Table 1. Therefore, the ELM algorithm, as a part of the feed-forward neural networks under artificial neural networks (also machine learning field), still appears to be an important algorithm in the process of transition from data to knowledge in this information age.

A. LITERATURE REVIEW OF RELATED WORKS

The ELM algorithm has attracted considerable attention and been actively studied in the statistics literature due to the following reasons: i) linear model form, ii) closed-form model structure, iii) relying on classical least squares for model parameter estimation, iv) possession of the previously mentioned capabilities and v) a wide range of applications. However, the basic ELM algorithm remains highly appealing, it has a number of shortcomings arising from structural risk minimization [36]. The shortcomings can be summarized as follows: i) The risk of over-fitting models, ii) over-dependence on hidden layer structure, iii) lack of sparsity capability, iv) non-stable performance on new data. The main reason for the over-fitting and lack of stability is that the columns of the hidden layer matrix in the ELM algorithm are linearly dependent (i.e. correlated). In the statistics literature, this problem, known as multicollinearity, leads to extreme variability in the model predictions and weakening of the predictions obtained with new data (overfitting model).

Deng et al. [37] proposed a regularized ELM algorithm based on weighted least squares to overcome the difficulties related to structural risk minimization. Feng et al. [38] introduced the error minimized extreme learning machine (EM-ELM) algorithm based on incremental adjustment of hidden layer weights. Yuan et al. [39] improved the convergence performance of the basic ELM model by deriving an optimal model by examining the rank of the hidden layer matrix under different rank conditions. Lu et al. [40] presented a comparative study of different solutions of the Moore-Penrose matrix calculus used in the basic ELM solution, which is directly related to the structure of the hidden layer matrix. These studies mostly emphasize either derivation of the hidden layer matrix or optimization of the computations based on it. However, there are alternative efficient solutions to the multicollinearity problem for linear models based on a slight adjustment to the structure of the hidden layer matrix that have been proposed in the statistical literature. The most important of these approaches are the ridge estimator proposed by Hoerl and Kennard [41] and the Liu estimator proposed by Liu [42].

The two estimators were initially proposed by Deng et al. [37] and later presented in a unified framework by Huang et al. [43], [44] in order to strengthen the generalization performance and obtain a more robust result

in the ELM context. Huang et al. [44] comprehensively described the theoretical and practical properties of ELM models based on the ridge estimator, along with its use in both classification and regression models with real-life data. Miche et al. [45] introduced a novel algorithm called OP-ELM to eliminate correlated or redundant components by considering the ridge estimator (a.k.a Tikhonov or l_2 norm regularization). Later, Miche et al. [46] developed the TROP-ELM algorithm, which is a modification of OP-ELM and incorporates LARS and Tikhonov regularization approaches into the model simultaneously. Martínez et al. [47] applied the ridge, lasso and elastic net approaches to the ELM domain and benchmarked the performances comparatively. Li and Niu [48] introduced novel models by extending ridge and alternative ridge-based estimators (such as almost unbiased ridge) to the ELM domain. Fakhr et al. [49] provided a comparative study of the basic ELM algorithm and its variants based on l_1 and l_2 norms. Luo et al. [50] also provide a detailed overview of the theoretical and practical properties of l_1 and l_2 form-based ELM algorithms for classification and regression problems. He et al. [51] proposed a novel Elm algorithm based on the $l_{1/2}$ norm to determine the number of hidden layer nodes and prune redundant components. Shan et al. [52] introduced a novel algorithm integrating the interval Lasso (i.e. l_2 norm) approach in the ELM algorithm, in order to achieve a more compact model selection. Yıldırım and Özkale [5] proposed ELM algorithms based on ridge, almost unbiased ridge and generalized ridge estimators, and also presented parameter selection methods (including cross validation (CV), Akaike information (AIC) and Bayesian information (BIC) criteria) with real life data.

The first implementation of the Liu estimator in the ELM domain is the Liu-ELM algorithm proposed by Yıldırım and Özkale [53], which is based on an improved form of ridge-based ELM algorithms. In order to address the lack of sparsity in the Liu estimator, a novel ELM algorithm called as LL-ELM based on simultaneous implementation with the Lasso approach has been presented by Yıldırım and Özkale [54]. Li and Zhao [55] present performance benchmarks by incorporating the Liu estimator into the Tensor-Based Type-2 Elm algorithm. The Ridge and Liu estimators based models represent a highly competitive ELM algorithm for real-world problems due to their different levels of model improvements. Therefore, the two-parameter estimator proposed by Özkale and Kaciranlar [56] in the statistics literature has been applied to the ELM domain and the OK-ELM algorithm has been proposed by Yıldırım and Özkale [57]. The proposed GO-ELM with this approach yields a better performance compared to the basic ELM, standalone ridge-based ELM and Liu-based ELM algorithms.

B. RESEARCH MOTIVATION AND ENTHUSIASM

While the ridge estimator has seen extensive use in data-driven studies, particularly in machine learning, the Liu estimator is gaining recognition. The proposed GO-ELM integrates the capabilities of both estimators, offering a

robust alternative for classification and regression tasks. Two-parameter estimators are a prominent and thoroughly researched topic in statistical theory, with various alternative models available. These include the k-d estimator by Sakallıoğlu and Kaçranlar [58] and the y-c estimator by Yang and Chang [59]. In this paper, we introduce the first novel implementations of these two new two-parameter estimators within the ELM framework, providing the following properties:

- The proposed algorithms combine classical least squares, ridge, and Liu approaches to address the limitations of the ELM algorithm effectively.
- We present a comparative analysis of the performance of the two new proposed algorithms with the highly competitive GO-ELM algorithm using real-life applications.
- Detailed comparisons with the most popular machine learning algorithms demonstrate the relative superiority of the proposed algorithms.

The structure of the study is organized as follows: Section II summarizes the basic ELM algorithm and its properties. Section III briefly presents the algorithm and related models. Mathematical details and properties of the proposed algorithms are described in Section IV. Section V discusses application details and results with real-life data within a comprehensive framework. The last section presents the main conclusions of the study.

II. THE PRELIMINARY ELM AND RELATED ALGORITHMS

ELM algorithm has been proposed for SLFNs and is effectively used in regression and classification tasks. Supposed that $(\mathbf{x}_i^T, t_i^T), i = 1, \dots, N$ are randomly taken samples from input space and $H = (h_1, h_2, \dots, h_m)$ is the function set of explanatory variables (i.e basis functions), then a regression function can be defined as follows:

$$\hat{f}(\mathbf{x}_i) = \sum_{l=1}^m \hat{\beta}_l h_l(\mathbf{x}_i) \tag{1}$$

where \hat{f} is the estimated function obtained from the calculation of the observed data points and $\hat{\beta} = (\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_m)$ is the estimated coefficients vector. From the view of neural networks, the main difference between the classic regression and the neural networks is the structure of the $h(\cdot)$ vector. In neural networks, $h(\cdot)$ consists of the input values, weights and biases in neural networks. Actually $h(\cdot)$ is the output of the activation function, which is the sum of the input weights and input values referring to explanatory variables and biases. Therefore, Eq. (1) can be rewritten in neural networks context as:

$$\hat{f}(\mathbf{x}_i) = \sum_{l=1}^m \hat{\beta}_l h(\mathbf{w}_l \cdot \mathbf{x}_i + b_l) \tag{2}$$

where \mathbf{w}_l corresponds the input weight vector linking the l th hidden node and the input nodes, b_l is the bias value of the l th hidden node and $\hat{\beta}_l$ is the output weight vector linking the l th hidden node and the output nodes. Here m is the number of nodes in hidden layer and h is the activation

function. $\mathbf{w}_l \cdot \mathbf{x}_i$ corresponds the inner product between \mathbf{w}_l and \mathbf{x}_i [3], [4]. By considering new parameters in estimation function given by Eq. (2), the problem can be defined in matrix form for SLFNs as follows [3], [4], [43]:

$$\begin{bmatrix} h(\mathbf{w}_1 \cdot \mathbf{x}_1 + b_1) & \dots & h(\mathbf{w}_m \cdot \mathbf{x}_1 + b_m) \\ \vdots & \dots & \vdots \\ h(\mathbf{w}_1 \cdot \mathbf{x}_N + b_1) & \dots & h(\mathbf{w}_m \cdot \mathbf{x}_N + b_m) \end{bmatrix}_{N \times m} \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_m \end{bmatrix}_{m \times 1} = \begin{bmatrix} \mathbf{t}_1 \\ \vdots \\ \mathbf{t}_N \end{bmatrix}_{N \times 1} \tag{3}$$

Eq. (3) can be written in matrix notation as

$$\mathbf{H}\beta = \mathbf{T} \tag{4}$$

where \mathbf{H} is called as the output matrix of hidden layer in the neural network by Huang et al. [3], [4]. For appropriate \mathbf{w}_l, b_l and β parameters, the solution of Eq. (4) is obtained by minimizing the following system in SLFNs:

$$\|\mathbf{H}(\hat{\mathbf{w}}_1, \dots, \hat{\mathbf{w}}_m, \hat{b}_1, \dots, \hat{b}_m) \hat{\beta} - \mathbf{T}\| = \min_{\mathbf{w}_l, b_l, \beta} \|\mathbf{H}\beta - \mathbf{T}\|. \tag{5}$$

The solution of Eq. (5) is calculated by the least squares method as $\hat{\beta}_{ELM} = (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T \mathbf{T}$ unless $m \neq N$ and $\mathbf{H}^T \mathbf{H}$ is singular. Otherwise, the Moore–Penrose inverse of matrix \mathbf{H} , say \mathbf{H}^+ , should be used to get the optimal solution as $\hat{\beta}_{ELM} = \mathbf{H}^+ \mathbf{T}$.

Orthogonal projection method, iterative methods and singular value decomposition (SVD) are well-known and effective ways to calculate the Moore–Penrose inverse [60], [61]. According to the orthogonal projection method [61], the Moore–Penrose inverse of \mathbf{H} is found as follows:

$$\mathbf{H}^+ = \begin{cases} (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{H}^T, & \text{rank}(\mathbf{H}) = m \\ \mathbf{H}^T (\mathbf{H} \mathbf{H}^T)^{-1}, & \text{rank}(\mathbf{H}) = N \end{cases} \tag{6}$$

The steps of ELM algorithm proposed by Huang et al. [4] can be summarised in Algorithm 1. Besides, the visual representation of the algorithm is given in Figure 1:

Algorithm 1 ELM Algorithm

- 1: Randomly assign input weights and bias parameters.
 - 2: Calculate the hidden layer output matrix \mathbf{H} .
 - 3: Calculate the output weights via $\hat{\beta}_{ELM} = \mathbf{H}^+ \mathbf{T}$ by using Eq. (6).
-

When there is a multicollinearity problem, inverting the matrix $(\mathbf{H}^T \mathbf{H})^{-1}$ can sometimes be impossible and sometimes unstable. Therefore, alternative methods to ordinary ELM are recommended. On the other hand, ordinary ELM does not have sparsity property; that is, does not do variable selection. The methods proposed on the basis of ELM as a solution to these two problems are as follows:

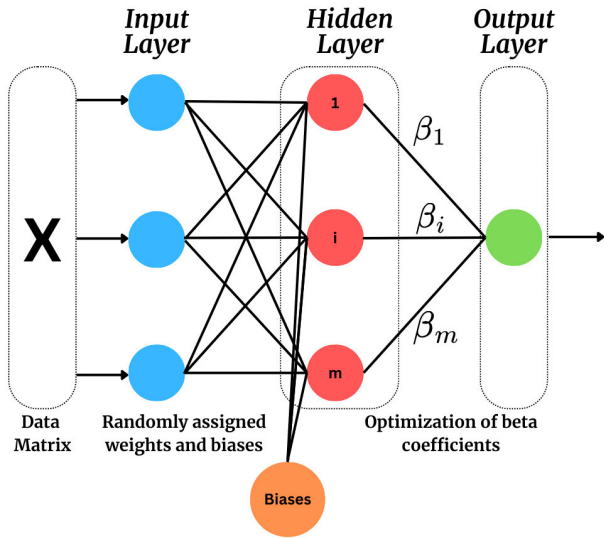


FIGURE 1. The general representation of the basic ELM model.

Reference [48] Li and Niu considered ridge regression in the context of ELM which was originally proposed by Hoerl and Kennard [41] and defined the Ridge-ELM as

$$\hat{\beta}_{Ridge-ELM}^k = (\mathbf{H}^T \mathbf{H} + k \mathbf{I}_m)^{-1} \mathbf{H}^T \mathbf{Y}, \quad k \geq 0$$

where k is the ridge tuning parameter and \mathbf{I}_φ is the identity matrix with dimension φ . Although the value of k affects the performance of Ridge-ELM, there is no single best and suitable method for selecting the ridge tuning parameter. Regression studies literatures [48] and [62] indicates that Ridge-ELM can provide smaller error than ELM if k is correctly determined.

Yildirim and Özkale [53] considered the Liu regression in the context of ELM which was originally defined by Liu [42] and defined the Liu-ELM as

$$\hat{\beta}_{Liu-ELM}^d = (\mathbf{H}^T \mathbf{H} + \mathbf{I}_m)^{-1} (\mathbf{H}^T \mathbf{Y} + d \hat{\beta}_{ELM}) \quad (7)$$

where $0 < d < 1$ is the Liu tuning parameter which shrinks each element of $\hat{\beta}_{ELM}$ by the same d value. Since $\hat{\beta}_{Liu-ELM}^d$ is in linear form of d , computing $\hat{\beta}_{Liu-ELM}^d$ is much easier and faster than $\hat{\beta}_{Ridge-ELM}^k$ which provides computationally effective and cost minimized solutions for machine learning. $\hat{\beta}_{Liu-ELM}^d$ is also a convex combination of $\hat{\beta}_{ELM}$ and $\hat{\beta}_{Ridge-ELM}^k$ for $k = 1$ which claims that $\hat{\beta}_{Liu-ELM}^d$ provides solution paths between $\hat{\beta}_{ELM}$ and $\hat{\beta}_{Ridge-ELM}^{(k=1)}$ as d goes from 1 to 0:

$$\hat{\beta}_{Liu-ELM}^d = d \hat{\beta}_{ELM} + (1 - d) \hat{\beta}_{Ridge-ELM}^{(k=1)}$$

Yildirim and Özkale [57] proposed OK-ELM which takes the advantages of ridge and Liu regressions as

$$\hat{\beta}_{OK-ELM} = (\mathbf{H}^T \mathbf{H} + k \mathbf{I}_m)^{-1} (\mathbf{H}^T \mathbf{Y} + kd \hat{\beta}_{ELM})$$

where $0 < d < 1$ and $k > 0$ are the tuning parameters. OK-ELM is a convex combination of Ridge-ELM and ELM:

$$\hat{\beta}_{OK-ELM} = d \hat{\beta}_{ELM} + (1 - d) \hat{\beta}_{Ridge-ELM}^{(k)}$$

where d serves as a balance parameter between ELM and Ridge-ELM and refers the relative contributions of them. Basically, while the higher d towards to 1 yields more contribution in favor of ELM, the lower d increases the effect of Ridge-ELM to the solution.

III. METHODOLOGY

A. THE PROPOSED ALGORITHMS

The ridge and Liu estimator stand-alone is directly relying on classical least squares and therefore has the tendency to yield inadequate or misleading information in dealing with multicollinearity. Therefore, two-parameter estimators have come to the prominence in order to retain the existing capabilities and make more accurate estimates through holding the estimators in a simultaneous model. Although these estimators possess the skills of the two estimators to an extent in terms of the mathematical structure, yet there is no single superior model in real-life applications. Motivated by this point, we propose the following two-parameters ELM (named as TP1-ELM) algorithm by incorporating the k -d estimator [58] from the two-parameter estimators into the ELM algorithm:

$$\hat{\beta}_{TP1-ELM}(k, d) = (\mathbf{H}^T \mathbf{H} + k \mathbf{I}_m)^{-1} (\mathbf{H}^T \mathbf{Y} + d \beta_{Ridge-ELM}^{\hat{}}) \quad (8)$$

where $-\infty < d < \infty$ and $k > 0$ are the tuning parameters. The proposed TP1-ELM algorithm aims to eliminate the stability problem in the classical least squares by incorporating Liu and Ridge approaches simultaneously. As a result of this property, the ridge or Liu estimator is considered to outperform the standalone ridge or standalone Liu estimator by adjusting the parameters in a tradeoff by making the parameter selection more flexible. The TP2-ELM algorithm is a general implementation of the classical least squares, ridge and Liu estimators, which can be derived in special cases:

$$\begin{aligned} \hat{\beta}(0, 1) &= \hat{\beta}_{ELM} \\ \hat{\beta}(k, 1 - k) &= \hat{\beta}_{Ridge-ELM} \\ \hat{\beta}(0, d) &= \hat{\beta}_{Liu-ELM}, \quad 0 \leq d \leq 1 \end{aligned}$$

An alternative two-parameters estimator originally introduced by Yang and Chang [59] can be proposed for first utilization in the field of ELM named as TP2-ELM and defined as follows:

$$\begin{aligned} \hat{\beta}_{TP2-ELM}(k, d) &= (\mathbf{H}^T \mathbf{H} + \mathbf{I}_m)^{-1} (\mathbf{H}^T \mathbf{H} + d \mathbf{I}_m) (\mathbf{H}^T \mathbf{H} + k \mathbf{I}_m)^{-1} (\mathbf{H}^T \mathbf{Y}) \end{aligned} \quad (9)$$

where $0 < d < 1$ and $k > 0$ are the tuning parameters. Analogously to the TP1-ELM algorithm, the OLS, ridge and

Liu estimators can be extracted based on specific selections of the parameters:

$$\begin{aligned}\widehat{\beta}(0, 1) &= \widehat{\beta}_{ELM} \\ \widehat{\beta}(k, 1) &= \widehat{\beta}_{Ridge-ELM} \\ \widehat{\beta}(0, d) &= \widehat{\beta}_{Liu-ELM}, 0 \leq d \leq 1\end{aligned}$$

In the parameter selection of the proposed algorithms, a grid search via cross-validation approach within a defined value range or the analytical selection methods proposed by Sakalioğlu and Kaçranlar [58] and Yang and Chang [59] can be implemented. A pseudo code implementation of the proposed TP1-ELM and TP2-ELM algorithms is presented in Algorithm 2.

Algorithm 2 Pseudo Code for the Proposed Algorithms

- 1: **Input:** Generate the input layer weights $\widehat{\mathbf{w}}_1, \dots, \widehat{\mathbf{w}}_m$ and biases $\widehat{b}_1, \dots, \widehat{b}_m$ based on a certain number of the hidden layer neurons (m), k range and d range for the tuning parameters space, φ as the trial number.
 - 2: **Output:** The β coefficient vector.
 - 3: Calculate the hidden layer output matrix \mathbf{H} based on inputs.
 - 4: Set the seed value for reproducibility
 - 5: **for** $trial = 1 : \varphi$ **do**
 - 6: **for** k in k range **do**
 - 7: **for** d in d range **do**
 - 8: Obtain TP1-ELM coefficient vector via Eq. 8
 - 9: Obtain TP2-ELM coefficient vector via Eq. 9
 - 10: Calculate performance metrics for CV split
 - 11: **end for**
 - 12: **end for**
 - 13: **end for**
 - 14: Determine the parameters with the lowest mean error values across the trials.
 - 15: Compute the error scores on the test data with the optimal parameters
-

B. DATA SOURCE

In this study, we utilized seven datasets with structural and qualitative diversity across various domains to compare the performance of the statistical and machine learning models under consideration. These datasets were sourced from the UCI Machine Learning Repository [63], Kaggle platform [64], several universities databases [65], [66]. The specifications of the data sets and target variables covered in the study can be briefly summarized as follows: Auto Price data to predict vehicle prices, Boston Housing data to predict the median value of a real estate price, Fish data to predict quantitative acute aquatic toxicity, Forests data to predict the burned area of forest fires, Machine CPU data to predict relative cpu performance, Servo data to predict the rise time of a servomechanism, Slump data to predict concrete slump and Strikes data to predict the level of strike volume (days

lost due to industrial disputes per 1000 wage salary earners). The details of dimensions are listed in Table 1.

C. EXPERIMENTAL SETTINGS

In the process of benchmarking model performances, experimental settings are crucial for ensuring reproducibility. In this context, the following steps were followed in training and testing statistical models and machine learning algorithms:

- Prior to training, the data is pre-processed by standardizing it to have a mean of zero and unit variance, using the following formula:

$$\mathbf{x} = \left(\frac{x_i - \bar{x}}{sd(\mathbf{x})} \right).$$

- The dataset is partitioned into three-fourths for training and the remaining one-fourth for testing.
- Models are developed using five-times repeated five-fold cross-validation on the training data, and their generalization performances are evaluated on the test data.
- To mitigate the effects of randomness, thirty trials are conducted, and the root mean square error (RMSE) is calculated for each trial. Mean values are reported using the following equation:

$$RMSE = \sqrt{\frac{1}{N} \sum_{j=1}^N (\mathbf{o}_j - \mathbf{t}_j)^2}$$

where $(\mathbf{o}_j - \mathbf{t}_j)$ corresponds to the error between the actual and output values of the target variable. RMSE is defined in the interval $[0, \infty]$ and the lower the value of this criterion, the better the model performance.

- The number of hidden layer nodes for ELM-based models is kept constant at 100.
- A grid search approach is employed to assess the tuning parameters of the ELM-based statistical models. The δ for Ridge-ELM, d for Liu-ELM, and both δ and d parameters for OK-ELM, TP1-ELM, and TP2-ELM models are calculated via cross-validation, considering the following ranges:

$$\begin{aligned}k &\in [10^{-5}, 10^{-4}, \dots, 10^{-2}, 0.02, \dots, 0.1, 0.2, \dots, 1, 1.5, 2, \dots, 5] \\ d &\in [10^{-5}, 10^{-4}, \dots, 10^{-2}, 0.02, 0.03, \dots, 0.1, 0.15, 0.2, \dots, 1].\end{aligned}$$

- Optimal parameter determination for machine learning algorithms is similarly conducted by analyzing a space of thirty possible parameters for each model based on cross-validation data. The parameters yielding the lowest error value are then assigned.
- The reduction rate (RR), representing the relative improvement of the proposed algorithms over other algorithms, is evaluated in percentage terms using the following equation for clearer interpretation:

$$RR = \frac{(\text{any algorithm}) - (TP1,2 - ELM)}{(\text{any algorithm})} \times 100$$

- The details of this process are given as a visual representation in Figure 2.

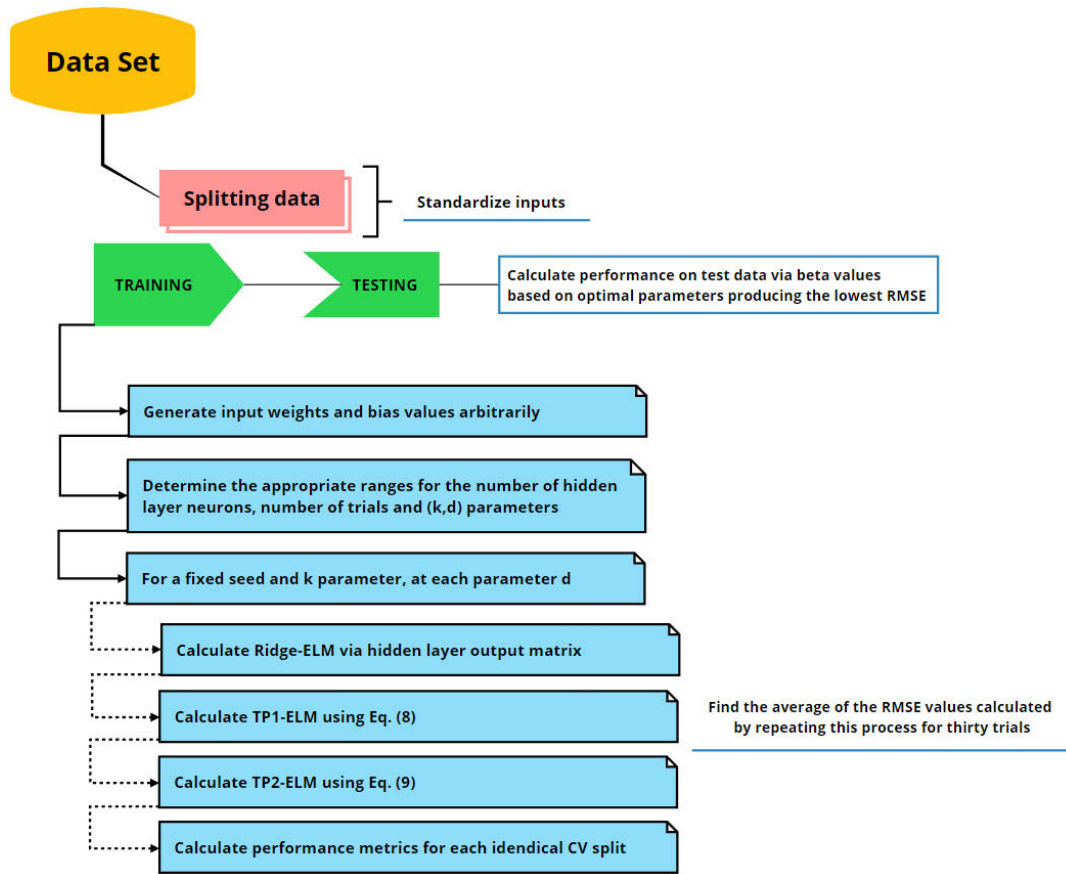


FIGURE 2. The flow chart of the calculation for the TP-ELMs.

TABLE 1. The properties of data sets used in this study.

Data sets	Sample Size	Attributes
Auto Price	159	13
Boston Housing	506	14
Fish	908	7
Forest	517	11
Machine CPU	209	7
Servo	167	5
Strikes	625	5

D. THE ASSESSMENT OF MULTICOLLINEARITY

The datasets were examined for multicollinearity using the Variance Inflation Factor (VIF) and eigenvalue analysis criteria, derived from the following equations:

$$VIF_j = C_{ij} = (1 - R_j^2)^{-1}$$

where $C = (X'X)^{-1}$ is the coefficient of determination calculated via the regression fit of x_j over the rest $p-1$ variables. In the dataset, the presence of a VIF value greater than 5 belonging to the one of the variables indicates a strong multicollinearity problem [67].

As an effective and widely used method in addition to the VIF approach, the eigenvalue analysis is based on

the decomposition of the $X'X$ matrix into eigenvalues and eigenvectors and calculated as follows:

$$X'X = T \Lambda T'$$

where $\Lambda_{p \times p}$ is the diagonal matrix, whose diagonal elements correspond to the eigenvalues ($\lambda_i, i = 1, 2, \dots, p$) and $T_{p \times p}$ is the orthogonal matrix whose columns correspond to the eigenvectors of $X'X$ matrix. Small-valued eigenvalues can be indicative of the presence of multicollinearity across columns of data. Rather than focusing on each eigenvalue, the condition number (CN), which is basically a representation of the spread of eigenvalues, is commonly used and is calculated as follows:

$$CN = \frac{\lambda_{\max}}{\lambda_{\min}}$$

Typically, a Condition Number (CN) exceeding 1000 is considered evidence of severe multicollinearity, while a CN between 100 and 1000 implies strong multicollinearity between the columns (i.e., variables) of the data matrix [67].

The VIF, eigenvalue, and condition number criteria were calculated for the seven datasets in the study, and the results are reported in Table 2. According to the findings, the Auto Price and Slump datasets exhibit multicollinearity issues. Additionally, the Boston Housing dataset shows

TABLE 2. The results of multicollinearity diagnostics for each data set.

Data set	Criterion	Variables ID												
		1	2	3	4	5	6	7	8	9	10	11	12	13
Auto Price	VIF	6,1088	7,7759	5,6902	2,2876	16,0658	9,0702	2,2690	1,4417	2,5073	7,1937	1,8893	6,1194	
	Eigenvalue	6,2844	1,8772	1,2903	0,8847	0,5191	0,3635	0,2743	0,1444	0,1240	0,1043	0,0871	0,0466	
	CN	134,7581												
Boston Housing	VIF	1,7922	2,2988	3,9916	1,0740	4,3937	1,9337	3,1008	3,9559	7,4845	9,0086	1,7991	1,3485	2,9415
	Eigenvalue	6,1268	1,4333	1,2426	0,8576	0,8348	0,6574	0,5354	0,3961	0,2769	0,2202	0,1860	0,1693	0,0635
	CN	96,4717												
Fish	VIF	2,1315	1,3977	1,6058	1,0786	1,2319	2,3537							
	Eigenvalue	1,6422	1,4729	1,1253	0,9636	0,5916	0,2043							
	CN	8,0382												
Forests	VIF	1,4334	1,4456	1,6981	2,3616	2,1259	1,5794	2,6667	1,9138	1,1428	1,0478			
	Eigenvalue	2,8597	1,5594	1,2961	1,2125	0,9315	0,6926	0,4756	0,4631	0,2947	0,2148			
	CN	13,3122												
Machine CPU	VIF	1,2016	2,9020	3,2740	1,8583	1,9662	1,8914							
	Eigenvalue	3,3567	0,8294	0,7392	0,4963	0,4044	0,1739							
	CN	19,3004												
Slump	VIF	2,5147	88,6201	100,2531	110,9583	54,2953	2,3860	156,4193	84,3636	7,4860	10,0768			
	Eigenvalue	3,0246	2,0919	1,1848	1,1284	0,8833	0,6927	0,6329	0,3021	0,0576	0,0017			
	CN	1784,4887												
Strikes	VIF	1,0557	1,0572	1,0556	1,0021									
	Eigenvalue	1,3522	1,0073	0,8239	0,8166									
	CN	1,6559												

a borderline condition number value, indicating potential multicollinearity concerns.

E. THE LIST OF MACHINE LEARNING ALGORITHMS

The machine learning models evaluated in this study can be categorized into five subgroups:

- Splines-based models: MARS (Multivariate Adaptive Regression Splines), Linear Regression (LR), Rulefit and Partial Least Squares (PLS) belong to this category.
- Kernel-based models: Support Vector Machines (SVM) fall into this category.
- Tree-based models: These include Classification and Regression Trees (CART), Bagging with different base learners including CART, MARS, MLP etc., Random Forests (RF), LightGBM and Extreme Gradient Boosting.
- Instance-based models: K-Nearest Neighbors (KNN) is an example of this subgroup. There are ensemble models incorporating different regression (e.g. stepwise regression) [68] or classification approaches [69], [70] within the KNN algorithm, but the classical KNN algorithm is employed in this study.
- Neural Networks-Based models: Multilayer Perceptron (MLP) and Bayesian Additive Trees (BAT) are representative of this subgroup.

IV. RESULTS AND DISCUSSION

In this section, we present detailed performance comparison results of both statistical models and machine learning algorithms. A comprehensive analysis of the ELM-based statistical models, including the two newly proposed algorithms, is provided separately, both among themselves and in comparison to machine learning algorithms.

A. THE PERFORMANCE COMPARISON OF ELM-BASED ALGORITHMS AND THE PROPOSED ALGORITHMS

Initially, comprehensive comparison results of the performance of the proposed TP1-ELM and TP2-ELM algorithms

TABLE 3. Performance comparisons of each statistical algorithms based on RMSE criterion.

Dataset	Model	d	delta	RMSE (Train)	SD	RMSE (Test)	SD
Auto Price	ELM	*	*	0,0671	0,0094	1,6947	0,5818
	Ridge-ELM	*	1,96	0,2967	0,0112	0,4576	0,0302
	Liu-ELM	0,01208	*	0,2943	0,0116	0,4710	0,0338
	OK-ELM	0,00624	1,96	0,2951	0,0111	0,4573	0,0311
	TP1-ELM	0,40804	2,24	0,2912	0,0115	0,4465	0,0318
	TP2-ELM	0,84	1,82	0,2917	0,0110	0,4274	0,0280
Boston Housing	ELM	*	*	0,2869	0,0131	0,4481	0,0481
	Ridge-ELM	*	0,26	0,3141	0,0111	0,4128	0,0297
	Liu-ELM	0,39	*	0,3140	0,0112	0,4119	0,0300
	OK-ELM	0,258	1,54	0,3153	0,0112	0,4088	0,0295
	TP1-ELM	0,224	0,4	0,3152	0,0111	0,4027	0,0283
	TP2-ELM	0,65	0,0762	0,3123	0,0112	0,4101	0,0273
Fish	ELM	*	*	0,5167	0,0072	0,6987	0,0454
	Ridge-ELM	*	2,18	0,5761	0,0047	0,6148	0,0088
	Liu-ELM	0,08002	*	0,5625	0,0044	0,6170	0,0101
	OK-ELM	0,114	2,88	0,5684	0,0045	0,6137	0,0092
	TP1-ELM	0,21604	2,38	0,5761	0,0047	0,6148	0,0089
	TP2-ELM	0,49	1,904002	0,5761	0,0048	0,6145	0,0087
Forests	ELM	*	*	0,8613	0,0110	1,1832	0,0736
	Ridge-ELM	*	4,8	0,9543	0,0046	1,0214	0,0093
	Liu-ELM	0,0001	*	0,9101	0,0071	1,0294	0,0184
	OK-ELM	0,0001	4,8	0,9443	0,0046	1,0714	0,0073
	TP1-ELM	0,17008	4,9	0,9421	0,0036	1,0414	0,0065
	TP2-ELM	0,0001	4,3	0,9334	0,0031	1,0099	0,0063
Machine CPU	ELM	*	*	0,1019	0,0029	0,7533	0,2449
	Ridge-ELM	*	0,0642	0,1508	0,0039	0,3189	0,0501
	Liu-ELM	0,15002	*	0,2261	0,0073	0,3501	0,0558
	OK-ELM	0,08602	0,166	0,1644	0,0044	0,3114	0,0486
	TP1-ELM	0,22202	0,1642	0,1591	0,0043	0,3149	0,0499
	TP2-ELM	0,72	0,02222	0,1640	0,0041	0,3068	0,0490
Slump	ELM	*	*	0,0611	0,0098	0,3321	0,1460
	Ridge-ELM	*	0,632	0,1672	0,0155	0,3019	0,0450
	Liu-ELM	0,138	*	0,1878	0,0138	0,3089	0,0406
	OK-ELM	0,08204	0,714	0,1616	0,0143	0,2982	0,0471
	TP1-ELM	0,0302	0,658	0,1669	0,0156	0,2619	0,0353
	TP2-ELM	0,64	0,4222	0,1613	0,0131	0,2981	0,0466
Strikes	ELM	*	*	0,8272	0,0039	1,0840	0,2298
	Ridge-ELM	*	0,108	0,8626	0,0038	0,8968	0,0096
	Liu-ELM	0,3	*	0,8735	0,0030	0,9064	0,0124
	OK-ELM	0,17402	0,13	0,8573	0,0034	0,8923	0,0094
	TP1-ELM	0,19044	0,208	0,8617	0,0038	0,8666	0,0066
	TP2-ELM	0,68	0,0046	0,8568	0,0033	0,8714	0,0105

with Ridge-ELM, Liu-ELM, and OK-ELM on both training and test data are presented in Table 3. A visual comparison of performance using calculated reduction rate values, based on the results in Table 3, is shown in Figure 3. Due to large variations in scale, percentages based on ELM performance are not included in Figure 3 but are discussed in the text to enhance the interpretability and readability of visual comparisons. The following conclusions can be drawn from Table 3 and Figure 3:

- The training performances of TP1-ELM and TP2-ELM are competitive with other algorithms. At least one of the proposed algorithms shows

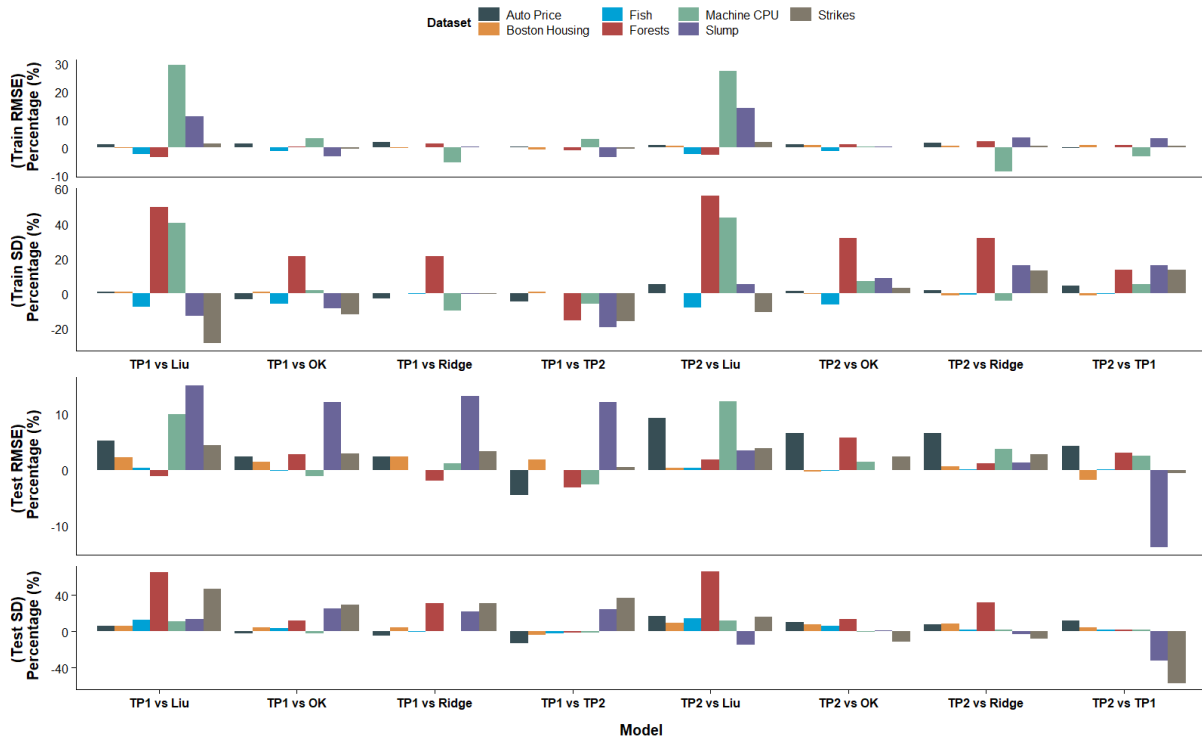


FIGURE 3. The overall comparison of ELM-based statistical models for both train and testing performance based on reduction rates.

TABLE 4. Coefficient norm comparison for each statistical algorithm.

Dataset	Model	Norm	SD	Dataset	Model	Norm	SD
Auto Price	ELM	57.7752	10.4343	Machine CPU	ELM	57.1196	7.0261
	Ridge-ELM	2.5481	0.0905		Ridge-ELM	9.5113	0.4728
	Liu-ELM	2.5706	0.1341		Liu-ELM	9.8764	1.0567
	OK-ELM	2.6324	0.0897		OK-ELM	9.0178	0.6254
	TP1-ELM	2.7082	0.1029		TP1-ELM	9.3592	0.4686
	TP2-ELM	2.6340	0.0820		TP2-ELM	10.0332	0.5347
Boston Housing	ELM	12.3740	1.2719	Slump	ELM	12.8130	2.3867
	Ridge-ELM	6.0931	0.3340		Ridge-ELM	2.7237	0.1397
	Liu-ELM	6.5575	0.4605		Liu-ELM	2.8096	0.2668
	OK-ELM	6.5181	0.4611		OK-ELM	3.0404	0.2132
	TP1-ELM	6.0056	0.3305		TP1-ELM	2.7196	0.1394
	TP2-ELM	6.3418	0.3818		TP2-ELM	3.0155	0.1946
Fish	ELM	40.6888	6.1286	Strikes	ELM	171.8604	23.2722
	Ridge-ELM	3.4479	0.1491		Ridge-ELM	26.3884	1.0555
	Liu-ELM	5.7294	0.4603		Liu-ELM	52.6346	6.9376
	OK-ELM	6.0678	0.5996		OK-ELM	42.3485	3.9297
	TP1-ELM	3.4178	0.1470		TP1-ELM	27.0780	1.0861
	TP2-ELM	4.0299	0.2216		TP2-ELM	38.5599	2.1897
Forests	ELM	18.8127	2.0581				
	Ridge-ELM	1.6162	0.0923				
	Liu-ELM	3.9754	0.1841				
	OK-ELM	1.5970	0.0829				
	TP1-ELM	1.5907	0.0823				
	TP2-ELM	1.2163	0.0786				

considerable superiority (improvement rates of up to 29.64%) over Liu-ELM in most of the data (including auto-price, boston housing, machine cpu, slump and strikes) and the difference is especially obvious in the Slump and Machine CPU datasets. However, the basic ELM algorithm outperforms all models in terms of training performance, consistent with expectations.

- Regarding deviation scores on the training data, at least one of the proposed algorithms displays greater stability on most datasets to some extent (up to 56.22%).
- In terms of generalization performance, a primary focus of this study, the proposed algorithms clearly outperform other models on almost all datasets (up to

15.22%). Although they show slight weakness on Forest and Machine CPU datasets, the scores remain highly competitive. Notably, TP1-ELM and TP2-ELM significantly outperform the ELM algorithm, with TP1-ELM surpassing it by up to 73.65% and TP2-ELM by up to 74.78%.

- Similarly, standard deviations of test performance suggest that the proposed algorithms exhibit much lower variability (up to 65.61%), indicating more stable and generalizable results across almost all datasets.
- While the proposed algorithms generally perform comparably, TP2-ELM demonstrates slightly superior testing performance compared to TP1-ELM.

The norm means and deviations of coefficient estimates for the ELM-based statistical models and proposed algorithms are calculated and presented in Table 4, providing insights into coefficient volatility. Results indicate that the TP1-ELM algorithm yields smaller and more stable norm values than all other algorithms on Boston, Fish, and Slump datasets. However, TP2-ELM outperforms on the Forest dataset. Conversely, Ridge-ELM performs slightly better on Auto Price and Strikes datasets, while OK-ELM shows superiority on Machine CPU data.

B. THE PERFORMANCE COMPARISON OF MACHINE LEARNING ALGORITHMS AND THE PROPOSED ALGORITHMS

The performance of the proposed algorithms, along with seventeen widely used machine learning algorithms, was

TABLE 5. Performance results of machine learning algorithms for both training and testing data sets.

Data Set	Split	Model	Bag-Cart	Bag-Mars	Bag-MLP	BAT	LightGBM	XGBoost	Cart	KNN	LR	MARS	MLP	PLS	RF	RuleFit	SMV-Linear	SVM-Poly	SVM-Radial
Auto Price	Train	RMSE	0.3666	0.3698	0.3926	0.3410	0.3639	0.3001	0.3152	0.2790	0.3748	0.3545	0.3315	0.3082	0.2819	0.2720	0.3078	0.3679	0.3671
		SD	0.0698	0.0553	0.0414	0.0847	0.1670	0.2510	0.0876	0.0399	0.0442	0.1090	0.2270	0.0212	0.0523	0.1540	0.0673	0.2050	0.2048
		RMSE	0.3777	0.5082	0.4514	0.3974	0.3989	0.5609	0.3930	0.3526	0.4912	0.4533	0.4468	0.4555	0.3592	0.4523	0.4489	0.4540	0.4260
Boston Housing	Train	RMSE	0.4223	0.4207	0.3942	0.3722	0.3963	0.3295	0.3285	0.2944	0.4644	0.3250	0.3293	0.4226	0.3270	0.2992	0.3242	0.4649	0.3950
		SD	0.0669	0.0726	0.0260	0.0536	0.2330	0.1180	0.0533	0.0487	0.0529	0.0931	0.0559	0.0418	0.0294	0.0957	0.0447	0.1350	0.1255
		RMSE	0.4678	0.4556	0.4311	0.4159	0.4161	0.3782	0.5419	0.4388	0.5722	0.4618	0.3964	0.5655	0.3932	0.3833	0.5595	0.5612	0.4571
Fish	Train	RMSE	0.6194	0.5644	0.6170	0.6272	0.6253	0.5722	0.5809	0.5662	0.6189	0.6181	0.5964	0.6361	0.5600	0.5692	0.5725	0.6191	0.6192
		SD	0.0486	0.0455	0.0103	0.0337	0.1477	0.0652	0.0430	0.0474	0.0849	0.0363	0.3037	0.0290	0.0095	0.1118	0.0185	0.1064	0.1062
		RMSE	0.6447	0.6609	0.7072	0.6154	0.6142	0.6145	0.7177	0.6129	0.6686	0.6801	0.6383	0.6685	0.6200	0.6303	0.6699	0.6704	0.6534
Forest	Train	RMSE	0.9886	0.9884	1.0002	0.9876	0.9853	0.9853	1.0193	1.0365	0.9921	1.0262	0.9855	1.0026	0.9853	1.0306	1.0307	1.0307	1.0307
		SD	0.0200	0.0184	0.1489	0.0790	0.0326	0.0740	0.0850	0.0997	0.4852	0.0148	1.4575	0.0415	0.0090	0.2615	0.0050	0.0531	0.0534
		RMSE	1.0310	1.0848	1.0307	1.1207	1.1417	1.0978	1.0282	1.0265	1.0966	1.0307	1.0350	1.0403	1.0437	1.0307	1.1092	1.0994	1.0675
Machine CPU	Train	RMSE	0.3503	0.3649	0.4242	0.3194	0.3669	0.2619	0.4343	0.3934	0.3549	0.3099	0.3759	0.2516	0.3971	0.3171	0.4166	0.3528	0.3615
		SD	0.0573	0.0401	0.0327	0.0325	0.1120	0.1095	0.0686	0.0250	0.0690	0.0292	0.2140	0.0349	0.0392	0.0907	0.0623	0.1083	0.1109
		RMSE	0.4256	0.5501	0.5893	0.4153	0.5099	0.4901	0.5434	0.3988	0.4963	0.4090	0.3944	0.4765	0.4072	0.5997	0.6277	0.4226	0.4228
Slump	Train	RMSE	0.3266	0.3032	0.2370	0.3207	0.3647	0.2919	0.1503	0.4248	0.3478	0.3264	0.1400	0.3061	0.1774	0.2583	0.3531	0.3335	0.2831
		SD	0.0763	0.2263	0.0434	0.2181	0.1748	0.2039	0.0493	0.0546	0.0299	0.2085	0.1161	0.1263	0.0850	0.0595	0.2248	0.0902	0.0885
		RMSE	0.4606	0.5912	0.5120	0.3426	0.4753	0.4554	0.5665	0.6048	0.3853	0.3609	0.4461	0.3717	0.4699	0.3936	0.3614	0.5186	0.3295
Strikes	Train	RMSE	0.7580	0.8031	0.7787	0.8250	0.7621	0.7107	0.8421	0.7763	0.9601	0.8456	0.8537	0.9126	0.7578	0.6955	0.9304	0.9542	0.9508
		SD	0.0463	0.0501	0.4255	0.0458	0.0827	0.0661	0.0309	0.0599	0.1101	0.0497	1.0372	0.0043	0.0202	0.1081	0.0089	0.0095	0.0063
		RMSE	0.9579	0.8809	1.0061	0.9485	0.9566	0.9345	1.0224	0.8894	0.9935	0.9563	0.8834	0.9917	0.8086	0.8002	1.0073	1.0173	0.9774

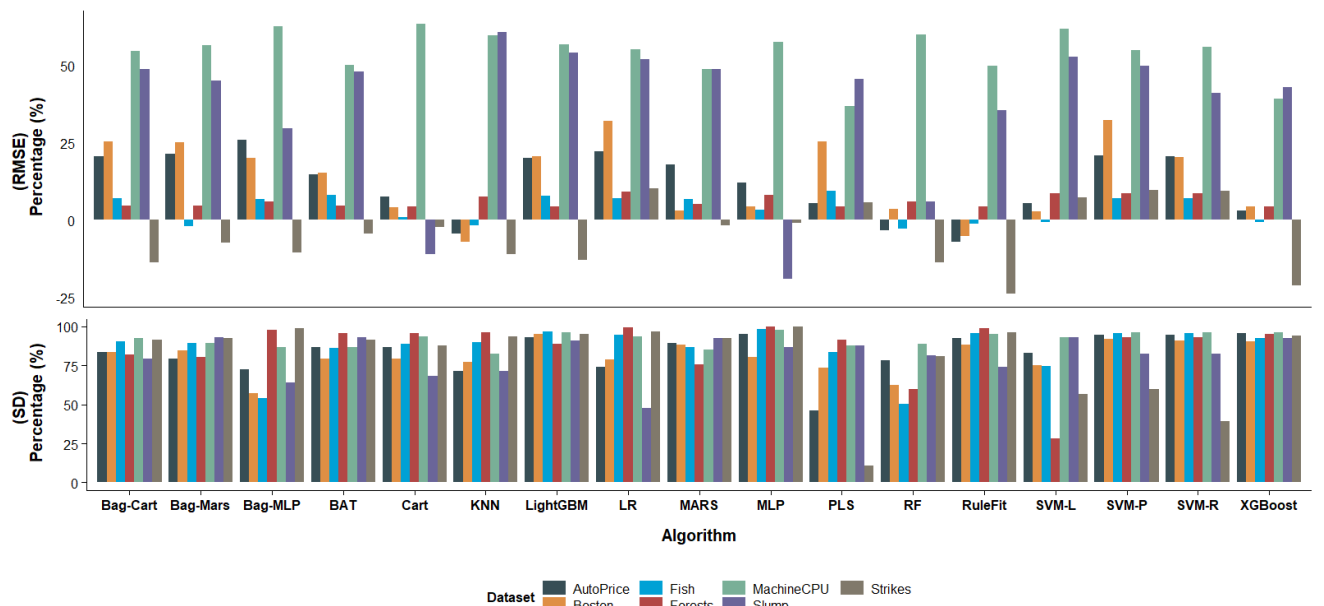


FIGURE 4. The training performance comparison between TP1-ELM and machine learning algorithms based on reduction rates.

benchmarked in the second step of the modeling process. Training and testing performances of TP1-ELM and TP2-ELM algorithms are summarized in Table 5, with training results of TP1-ELM visualized in Figure 4 and TP2-ELM in Figure 5. Additionally, reduction rates of test performances are displayed in Figure 6 for both proposed algorithms. Conclusions drawn from Table 5 and Figures 4-6 can be simplified as follows:

- In terms of training performance, both TP1-ELM and TP2-ELM algorithms achieved lower RMSE scores (up to 63.38% for TP1-ELM and 62.25% for TP2-ELM) compared to machine learning algorithms across most datasets. However, Rulefit, XGBoost, and Bagging (with Cart learner) performed better on the Strikes dataset. It appears that Rulefit and KNN algorithms are more competitive with the proposed algorithms compared to others.

- Regarding variability (standard deviation) of training performances, TP1-ELM and TP2-ELM algorithms exhibit lower deviation values (up to 99.75% for TP1-ELM and 99.78% for TP2-ELM) than all models across all datasets, indicating greater stability.
- Test performances (generalization performance) were found to be superior to machine learning models in most datasets, except for the Auto Price dataset (for some algorithms such as Random Forests, Bag-CART, etc.). Both proposed algorithms perform similarly, with TP1-ELM showing improvement of up to 56.79% and TP2-ELM achieving reduction of up to 51.11% in RMSE scores.
- In summary, the proposed algorithms offer more stable and generalizable results, with lower RMSE values, compared to both ELM-based and machine learning

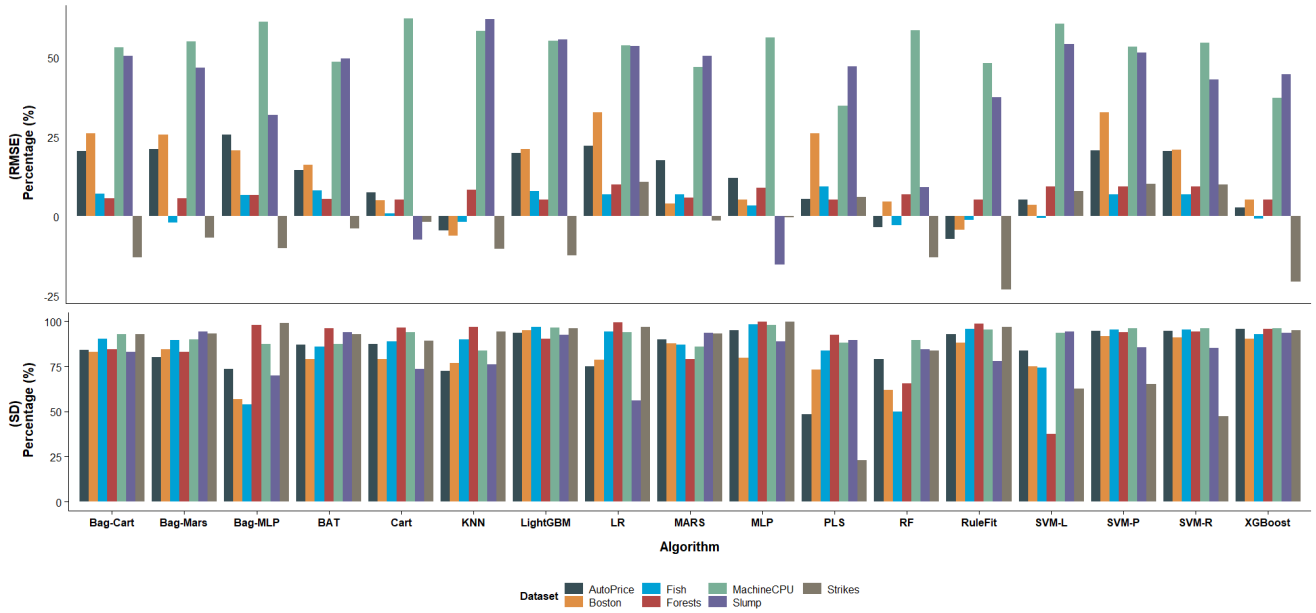


FIGURE 5. The training performance comparison between TP2-ELM and machine learning algorithms based on reduction rates.

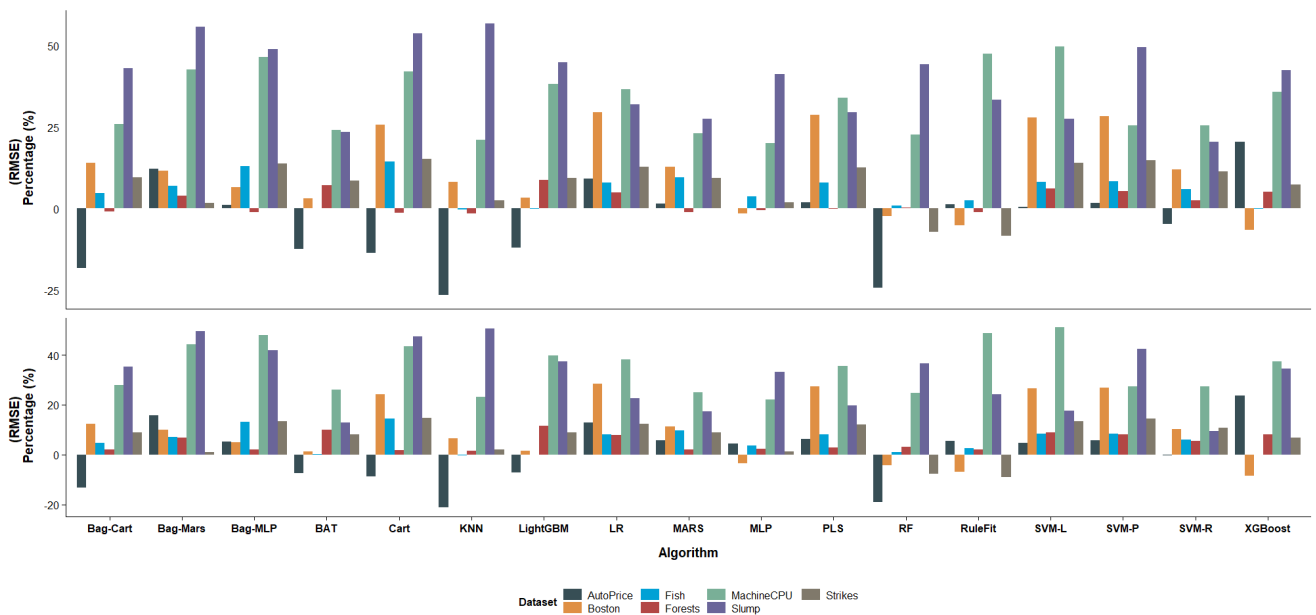


FIGURE 6. The testing performance comparison between the proposed models and machine learning algorithms based on reduction rates.

algorithms, especially for datasets with multicollinearity problems.

To facilitate more effective comparison of test performances for Slump and Boston datasets, the distribution of residual values calculated on the validation data is depicted in Figure 7 and Figure 8, respectively. The graphs clearly illustrate that the proposed algorithms exhibit more stable distributions of residual values around zero within narrower ranges compared to other algorithms. Therefore,

it should be noted that the proposed algorithms can be considered as a robust alternative to ELM-based algorithms for all regression-oriented areas and problems, particularly in real-life applications dealing with multicollinearity. More specifically, it has important application areas in economic [71], [72], [73] and health sciences [72], [74] in which statistical estimators developed for linear models are successful.

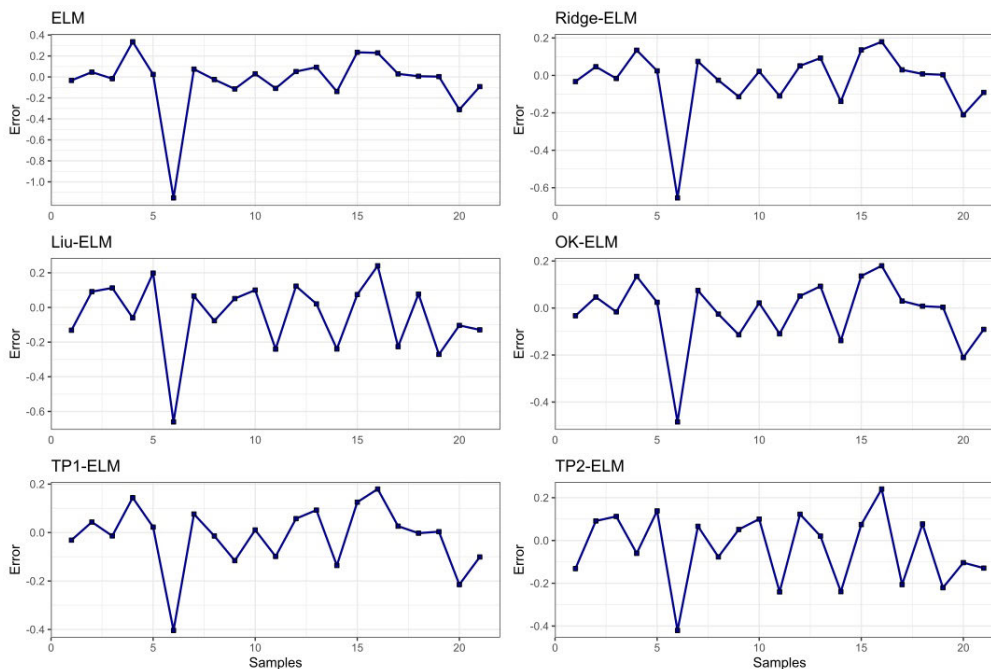


FIGURE 7. The testing error of the proposed algorithms based on slump testing data.

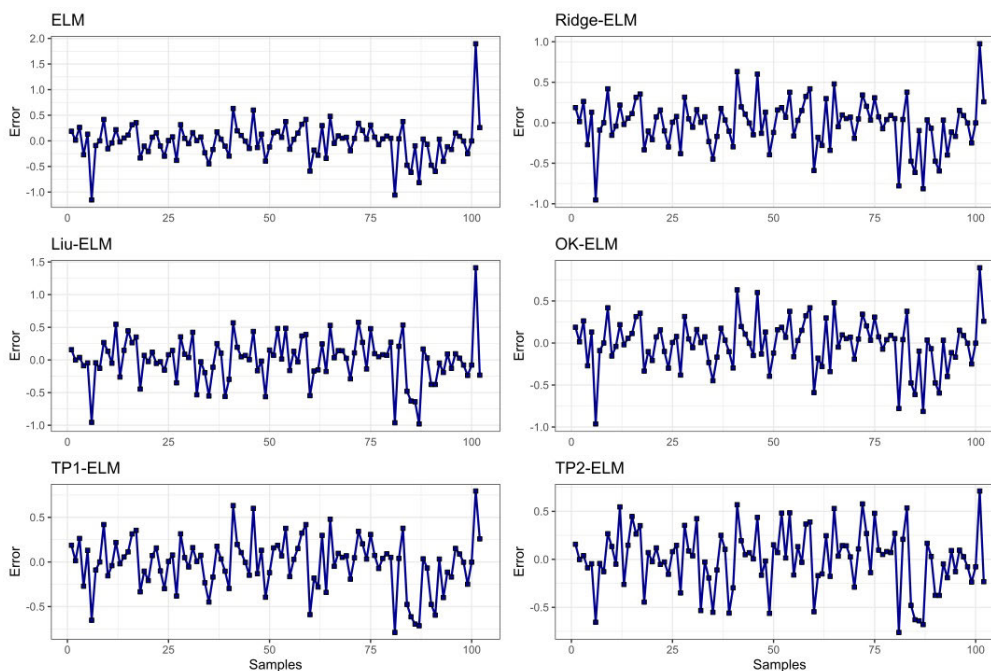


FIGURE 8. The testing error of the proposed algorithms based on boston testing data.

V. CONCLUSION

In this paper, we introduce two novel statistical estimators (TP1-ELM and TP2-ELM) aimed at addressing the multicollinearity problem within the Extreme Learning Machine (ELM) algorithm. They stand out as an effective alternative to ridge regression (aka Tikhonov

regularization), which is widely used in the field of machine learning, and are the foremost to be implemented in the context of feed-forward neural networks. Our proposed algorithms demonstrate superior performance over existing ELM-based and commonly used machine learning algorithms when applied to real-life datasets across

various domains, providing more generalizable and stable results.

It should also be noted that the proposed algorithms can be competitive with well-known machine learning algorithms on regular data apart from the data with multicollinearity problem, which is the focus of this study. Given their applicability to both classification and regression tasks in data-driven studies, these algorithms show promise as effective solutions to the multicollinearity problem.

Although our proposed algorithms exhibit satisfactory performance, a notable limitation of this study is the absence of various analytical methods from the literature for tuning parameter selection. Therefore, the development of analytical or heuristic methodologies for parameter estimation in algorithms involving statistical estimators is an open problem in the field of artificial neural networks (especially ELM). Additionally, future studies will focus on enhancing the proposed algorithms by incorporating feature selection capabilities to achieve more compact solutions, particularly in high-dimensional data settings where multicollinearity is prevalent. In conclusion, we plan to conduct a comprehensive comparison of the proposed algorithms on classification tasks as part of our future research goals by incorporating the limitations and identified gaps.

REFERENCES

- [1] S. Ding, H. Zhao, Y. Zhang, X. Xu, and R. Nie, "Extreme learning machine: Algorithm, theory and applications," *Artif. Intell. Rev.*, vol. 44, no. 1, pp. 103–115, Jun. 2015.
- [2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
- [3] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: A new learning scheme of feedforward neural networks," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Jul. 2004, pp. 985–990.
- [4] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: Theory and applications," *Neurocomputing*, vol. 70, nos. 1–3, pp. 489–501, Dec. 2006.
- [5] H. Yildirim and M. Revan Özkale, "The performance of ELM based ridge regression via the regularization parameters," *Expert Syst. Appl.*, vol. 134, pp. 225–233, Nov. 2019.
- [6] L. Ma, W. Xie, Y. Zhang, and X. Feng, "Extreme learning machine based defect detection for solder joints," *J. Internet Technol.*, vol. 21, no. 5, pp. 1535–1543, 2020.
- [7] P. Kansal, A. Kumar, and M. Gangadharappa, "Optimized extreme learning machine for intelligent spectrum sensing in 5G systems," *J. Commun. Technol. Electron.*, vol. 66, no. 3, pp. 322–332, Mar. 2021.
- [8] M. K. Giri and S. Majumder, "Cooperative spectrum sensing using extreme learning machines for cognitive radio networks," *IETE Tech. Rev.*, vol. 39, no. 3, pp. 698–712, May 2022.
- [9] G. Aldehim, M. A. Arasi, M. Khalid, S. S. Aljameel, R. Marzouk, H. Mohsen, I. Yaseen, and S. S. Ibrahim, "Gauss-mapping black widow optimization with deep extreme learning machine for Android malware classification model," *IEEE Access*, vol. 11, pp. 87062–87070, 2023.
- [10] J. V. Kumar and S. M. Shaby, "Design and optimization of triangular microstrip patch antenna using extreme learning machine (ELM)-based improved crystal structure algorithm (CryStAl) for C-band application," *Int. J. Commun. Syst.*, vol. 37, no. 7, May 2024, Art. no. e5721.
- [11] Z.-F. Liu, L.-L. Li, M.-L. Tseng, and M. K. Lim, "Prediction short-term photovoltaic power using improved chicken swarm optimizer-Extreme learning machine model," *J. Cleaner Prod.*, vol. 248, Mar. 2020, Art. no. 119272.
- [12] W. Sun and C. Huang, "Predictions of carbon emission intensity based on factor analysis and an improved extreme learning machine from the perspective of carbon emission efficiency," *J. Cleaner Prod.*, vol. 338, Mar. 2022, Art. no. 130414.
- [13] M. Ehteram, F. B. Banadkooki, and M. A. Nia, "Gaussian mutation-alpine skiing optimization algorithm-recurrent attention unit-gated recurrent unit-extreme learning machine model: An advanced predictive model for predicting evaporation," *Stochastic Environ. Res. Risk Assessment*, vol. 38, no. 5, pp. 1803–1830, May 2024.
- [14] T. Hussain, S. M. Siniscalchi, H. S. Wang, Y. Tsao, V. M. Salerno, and W.-H. Liao, "Ensemble hierarchical extreme learning machine for speech dereverberation," *IEEE Trans. Cogn. Develop. Syst.*, vol. 12, no. 4, pp. 744–758, Dec. 2020.
- [15] A. Miao and F. Liu, "Application of human motion recognition technology in extreme learning machine," *Int. J. Adv. Robot. Syst.*, vol. 18, no. 1, Jan. 2021, Art. no. 172988142098321.
- [16] Z. Zhou, J. Ji, Y. Wang, Z. Zhu, and J. Chen, "Hybrid regression model via multivariate adaptive regression spline and online sequential extreme learning machine and its application in vision servo system," *Int. J. Adv. Robot. Syst.*, vol. 19, no. 3, May 2022, Art. no. 172988062211086.
- [17] Z. Jin, G. Zhou, D. Gao, and Y. Zhang, "EEG classification using sparse Bayesian extreme learning machine for brain-computer interface," *Neural Comput. Appl.*, vol. 32, no. 11, pp. 6601–6609, Jun. 2020.
- [18] M. Jiao, D. Wang, Y. Yang, and F. Liu, "More intelligent and robust estimation of battery state-of-charge with an improved regularized extreme learning machine," *Eng. Appl. Artif. Intell.*, vol. 104, Sep. 2021, Art. no. 104407.
- [19] L. Sun, T. Wang, W. Ding, and J. Xu, "Partial multilabel learning using fuzzy neighborhood-based ball clustering and kernel extreme learning machine," *IEEE Trans. Fuzzy Syst.*, vol. 31, no. 7, pp. 2277–2291, Jul. 2023.
- [20] S. Govindaraj, L. Raja, S. Velmurugan, and K. Vijayalakshmi, "Extreme learning machine optimized by artificial cell swarm optimization for the data fusion modal in WSNs," *Peer-to-Peer Netw. Appl.*, vol. 17, no. 3, pp. 1344–1357, May 2024.
- [21] Z. Wang, S. Huo, X. Xiong, K. Wang, and B. Liu, "A maximally split and adaptive relaxed alternating direction method of multipliers for regularized extreme learning machines," *Mathematics*, vol. 11, no. 14, p. 3198, Jul. 2023.
- [22] L. Wu, G. Huang, J. Fan, X. Ma, H. Zhou, and W. Zeng, "Hybrid extreme learning machine with meta-heuristic algorithms for monthly pan evaporation prediction," *Comput. Electron. Agricult.*, vol. 168, Jan. 2020, Art. no. 105115.
- [23] M. Dogan, Y. S. Taspinar, I. Cinar, R. Kursun, I. A. Ozkan, and M. Koklu, "Dry bean cultivars classification using deep CNN features and salp swarm algorithm based extreme learning machine," *Comput. Electron. Agricult.*, vol. 204, Jan. 2023, Art. no. 107575.
- [24] P. V. de Campos Souza and L. C. B. Torres, "Extreme wavelet fast learning machine for evaluation of the default profile on financial transactions," *Comput. Econ.*, vol. 57, no. 4, pp. 1263–1285, Apr. 2021.
- [25] C. Zhang, A. Hu, and Y. Tian, "Daily tourism forecasting through a novel method based on principal component analysis, grey wolf optimizer, and extreme learning machine," *J. Forecasting*, vol. 42, no. 8, pp. 2121–2138, Dec. 2023.
- [26] F. Özyurt, E. Sert, and D. Avci, "An expert system for brain tumor detection: Fuzzy C-means with super resolution and convolutional neural network with extreme learning machine," *Med. Hypotheses*, vol. 134, Jan. 2020, Art. no. 109433.
- [27] S. Govindarajan and R. Swaminathan, "Extreme learning machine based differentiation of pulmonary tuberculosis in chest radiographs using integrated local feature descriptors," *Comput. Methods Programs Biomed.*, vol. 204, Jun. 2021, Art. no. 106058.
- [28] U. Ayman, M. S. Zia, O. D. Okon, N.-U. Rehman, T. Meraj, A. E. Ragab, and H. T. Rauf, "Epileptic patient activity recognition system using extreme learning machine method," *Biomedicines*, vol. 11, no. 3, p. 816, Mar. 2023.
- [29] J. Wang, G. Athanasopoulos, R. J. Hyndman, and S. Wang, "Crude oil price forecasting based on internet concern using an extreme learning machine," *Int. J. Forecasting*, vol. 34, no. 4, pp. 665–677, Oct. 2018.
- [30] S. Melina, H. Napitupulu, and N. Mohamed, "A conceptual model of investment-risk prediction in the stock market using extreme value theory with machine learning: A semisystematic literature review," *Risks*, vol. 11, no. 3, p. 60, Mar. 2023.
- [31] F. Peng, C. Chen, D. Lv, N. Zhang, X. Wang, X. Zhang, and Z. Wang, "Gesture recognition by ensemble extreme learning machine based on surface electromyography signals," *Frontiers Hum. Neurosci.*, vol. 16, Jun. 2022, Art. no. 911204.

- [32] Y. Ma, L. Wu, Y. Guan, and Z. Peng, "The capacity estimation and cycle life prediction of lithium-ion batteries using a new broad extreme learning machine approach," *J. Power Sources*, vol. 476, Nov. 2020, Art. no. 228581.
- [33] Z. Chen, H. Yu, L. Luo, L. Wu, Q. Zheng, Z. Wu, S. Cheng, and P. Lin, "Rapid and accurate modeling of PV modules based on extreme learning machine and large datasets of I-V curves," *Appl. Energy*, vol. 292, Jun. 2021, Art. no. 116929.
- [34] W. Dang, J. Guo, M. Liu, S. Liu, B. Yang, L. Yin, and W. Zheng, "A semi-supervised extreme learning machine algorithm based on the new weighted kernel for machine smell," *Appl. Sci.*, vol. 12, no. 18, p. 9213, Sep. 2022.
- [35] N. M. Eldabah, A. Shoukry, W. Khair-Eldeen, S. Kobayashi, and M. A.-H. Gepreel, "Design and characterization of low Young's modulus Ti-Zr-Nb-based medium entropy alloys assisted by extreme learning machine for biomedical applications," *J. Alloys Compounds*, vol. 968, Dec. 2023, Art. no. 171755.
- [36] V. Vapnik, *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer, 2013.
- [37] W. Deng, Q. Zheng, and L. Chen, "Regularized extreme learning machine," in *Proc. IEEE Symp. Comput. Intell. Data Mining*, Mar. 2009, pp. 389–395.
- [38] G. Feng, G.-B. Huang, Q. Lin, and R. Gay, "Error minimized extreme learning machine with growth of hidden nodes and incremental learning," *IEEE Trans. Neural Netw.*, vol. 20, no. 8, pp. 1352–1357, Aug. 2009.
- [39] Y. Yuan, Y. Wang, and F. Cao, "Optimization approximation solution for regression problem based on extreme learning machine," *Neurocomputing*, vol. 74, no. 16, pp. 2475–2482, Sep. 2011.
- [40] S. Lu, X. Wang, G. Zhang, and X. Zhou, "Effective algorithms of the Moore–Penrose inverse matrices for extreme learning machine," *Intell. Data Anal.*, vol. 19, no. 4, pp. 743–760, Jul. 2015.
- [41] A. E. Hoerl and R. W. Kennard, "Ridge regression: Applications to nonorthogonal problems," *Technometrics*, vol. 12, no. 1, pp. 69–82, Feb. 1970.
- [42] L. Kejian, "A new class of biased estimate in linear regression," *Commun. Statist., Theory Methods*, vol. 22, no. 2, pp. 393–402, Jan. 1993.
- [43] G.-B. Huang, D. H. Wang, and Y. Lan, "Extreme learning machines: A survey," *Int. J. Mach. Learn. Cybern.*, vol. 2, no. 2, pp. 107–122, Jun. 2011.
- [44] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.
- [45] Y. Miche, A. Sorjamaa, P. Bas, O. Simula, C. Jutten, and A. Lendasse, "OP-ELM: Optimally pruned extreme learning machine," *IEEE Trans. Neural Netw.*, vol. 21, no. 1, pp. 158–162, Jan. 2010.
- [46] Y. Miche, M. van Heeswijk, P. Bas, O. Simula, and A. Lendasse, "TROP-ELM: A double-regularized ELM using LARS and Tikhonov regularization," *Neurocomputing*, vol. 74, no. 16, pp. 2413–2421, Sep. 2011.
- [47] J. M. Martínez-Martínez, P. Escandell-Montero, E. Soria-Olivas, J. D. Martín-Guerrero, R. Magdalena-Benedito, and J. Gómez-Sanchis, "Regularized extreme learning machine for regression problems," *Neurocomputing*, vol. 74, no. 17, pp. 3716–3721, Oct. 2011.
- [48] G. Li and P. Niu, "An enhanced extreme learning machine based on ridge regression for regression," *Neural Comput. Appl.*, vol. 22, nos. 3–4, pp. 803–810, Mar. 2013.
- [49] M. W. Fakhr, E. S. Youssef, and M. S. El-Mahallawy, "L1-regularized least squares sparse extreme learning machine for classification," in *Proc. Int. Conf. Commun. Technol. Res. (ICTRC)*, May 2015, pp. 222–225.
- [50] X. Luo, X. Chang, and X. Ban, "Regression and classification using extreme learning machine based on L₁-norm and L₂-norm," *Neurocomputing*, vol. 174, pp. 179–186, Jan. 2016.
- [51] B. He, T. Sun, T. Yan, Y. Shen, and R. Nian, "A pruning ensemble model of extreme learning machine with L_{1/2} regularizer," *Multidimensional Syst. Signal Process.*, vol. 28, pp. 1051–1069, Jul. 2017.
- [52] P. Shan, Y. Zhao, X. Sha, Q. Wang, X. Lv, S. Peng, and Y. Ying, "Interval LASSO regression based extreme learning machine for nonlinear multivariate calibration of near infrared spectroscopic datasets," *Anal. Methods*, vol. 10, no. 25, pp. 3011–3022, 2018.
- [53] H. Yildirim and M. R. Özkale, "An enhanced extreme learning machine based on liu regression," *Neural Process. Lett.*, vol. 52, no. 1, pp. 421–442, Aug. 2020.
- [54] H. Yildirim and M. R. Özkale, "LL-ELM: A regularized extreme learning machine based on L₁-norm and Liu estimator," *Neural Comput. Appl.*, vol. 33, no. 16, pp. 10469–10484, Aug. 2021.
- [55] J. Li and G. Zhao, "Tensor-based type-2 extreme learning machine with L₁-norm and Liu regression," in *Proc. 41st Chin. Control Conf. (CCC)*, Jul. 2022, pp. 6327–6333.
- [56] M. R. Özkale and S. Kaçiranlar, "The restricted and unrestricted two-parameter estimators," *Commun. Statist., Theory Methods*, vol. 36, no. 15, pp. 2707–2725, Nov. 2007.
- [57] H. Yildirim and M. R. Özkale, "A combination of ridge and Liu regressions for extreme learning machine," *Soft Comput.*, vol. 27, no. 5, pp. 2493–2508, Mar. 2023.
- [58] S. Sakalhoğlu and S. Kaçiranlar, "A new biased estimator based on ridge estimation," *Stat. Papers*, vol. 49, no. 4, pp. 669–689, Oct. 2008.
- [59] H. Yang and X. Chang, "A new two-parameter estimator in linear regression," *Commun. Statist., Theory Methods*, vol. 39, no. 6, pp. 923–934, Mar. 2010.
- [60] C. R. Rao and M. B. Rao, *Matrix Algebra and Its Applications to Statistics and Econometrics*. Singapore: World Scientific, 1998.
- [61] J. R. Schott, *Matrix Analysis for Statistics*. Hoboken, NJ, USA: Wiley, 2016.
- [62] G. Tutz and H. Binder, "Boosting ridge regression," *Comput. Statist. Data Anal.*, vol. 51, no. 12, pp. 6044–6059, Aug. 2007.
- [63] A. Asuncion and D. Newman. (2007). *UCI Machine Learning Repository*. [Online]. Available: <https://archive.ics.uci.edu>
- [64] Kaggle. (2024). *Datasets*. [Online]. Available: <https://www.kaggle.com/datasets>
- [65] L. Torgo. (2024). *Datasets*. [Online]. Available: <https://www.dcc.fc.up.pt/~ltorgo/Regression/price.html>
- [66] Carnegie Mellon University. (2024). *Datasets*. [Online]. Available: <https://lib.stat.cmu.edu/datasets/strides>
- [67] D. C. Montgomery, E. A. Peck, and G. G. Vining, *Introduction to Linear Regression Analysis*. Hoboken, NJ, USA: Wiley, 2021.
- [68] A. Ali, M. Hamraz, P. Kumam, D. M. Khan, U. Khalil, M. Sulaiman, and Z. Khan, "A k-nearest neighbours based ensemble via optimal model selection for regression," *IEEE Access*, vol. 8, pp. 132095–132105, 2020.
- [69] A. Ali, M. Hamraz, N. Gul, D. M. Khan, S. Aldahmani, and Z. Khan, "A k nearest neighbour ensemble via extended neighbourhood rule and feature subsets," *Pattern Recognit.*, vol. 142, Oct. 2023, Art. no. 109641.
- [70] A. Ali, Z. Khan, D. M. Khan, and S. Aldahmani, "An optimal random projection k nearest neighbors ensemble via extended neighborhood rule for binary classification," *IEEE Access*, vol. 12, pp. 61401–61409, 2024.
- [71] K. C. Arum, F. I. Ugwuowo, H. E. Oranye, T. O. Alakija, T. E. Ugah, and O. C. Asogwa, "Combating outliers and multicollinearity in linear regression model using robust Kibria–Lukman mixed with principal component estimator, simulation and computation," *Sci. Afr.*, vol. 19, Mar. 2023, Art. no. e01566.
- [72] I. S. Dar, S. Chand, M. Shabbir, and B. M. G. Kibria, "Condition-index based new ridge regression estimator for linear regression model with multicollinearity," *Kuwait J. Sci.*, vol. 50, no. 2, pp. 91–96, Apr. 2023.
- [73] S. Yasin, S. Kamal, and M. Suhail, "Performance of some new ridge parameters in two-parameter ridge regression model," *Iranian J. Sci. Technol., Trans. A, Sci.*, vol. 45, no. 1, pp. 327–341, Feb. 2021.
- [74] A. F. Lukman, B. M. G. Kibria, C. K. Nziku, M. Amin, E. T. Adewuyi, and R. Farghali, "K-L estimator: Dealing with multicollinearity in the logistic regression model," *Mathematics*, vol. 11, no. 2, p. 340, Jan. 2023.



HASAN YILDIRIM received the Bachelor of Science degree from Anadolu University and the master's and Ph.D. degrees in statistics from Çukurova University, Türkiye. Since 2012, he has been a Faculty Member with Karamanoğlu Mehmetbey University, where he is currently an Assistant Professor Doctor. His research interests include statistical learning, artificial intelligence, challenges associated with high-dimensional data, and biostatistics.

...