Neural Computing and Applications (2021) 33:10469–10484
https://doi.org/10.1007/s00521-021-05806-0

ORIGINAL ARTICLE

# LL-ELM: A regularized extreme learning machine based on $L_1$-norm and Liu estimator

Hasan Yıldırım[1] · M. Revan Özkale[2]

Received: 10 August 2020 / Accepted: 5 February 2021 / Published online: 1 March 2021


**Abstract**
In this paper, we proposed a novel regularization and variable selection algorithm called Liu–Lasso extreme learning machine (LL-ELM) in order to deal with the ELM's drawbacks like instability, poor generalizability and underfitting or overfitting due to the selection of inappropriate hidden layer nodes. Liu estimator, which is a statistically biased estimator, is considered in the learning phase of the proposed algorithm with Lasso regression approach. The proposed algorithm is compared with the conventional ELM and its variants including ELM forms based on Liu estimator (Liu-ELM), $L_1$-norm (Lasso-ELM), $L_2$-norm (Ridge-ELM) and elastic net (Enet-ELM). Convenient selection methods for the determination of tuning parameters for each algorithm have been used in comparisons. The results show that there always exists a $d$ value such that LL-ELM overperforms either Lasso-ELM or Enet-ELM in terms of learning and generalization performance. This improvement in LL-Lasso over Lasso-ELM and Enet-ELM in the sense of testing root mean square error varies up to 27% depending on the proposed $d$ selection methods. Consequently, LL-ELM can be considered as a competitive algorithm for both regressional and classification tasks because of easy integrability property.

**Keywords** Extreme learning machine · Regularized extreme learning machine · Liu estimator · Ridge regression · Lasso · Elastic net

## 1 Introduction

Since extreme learning machine (ELM) has been proposed by Huang et al. [18, 19], there has been great attention by researchers in many disciplines and real-world applications such as medical/biomedical [1, 48, 56], computer vision [9, 34], image/video processing [2, 4, 44], text classification [36, 59], system modeling and prediction [6, 33, 51, 52, 58], control and robotics [39], chemical process [7, 50], fault detection and diagnosis [10], time-series analysis [3] and remote sensing [17, 45]. For some comprehensive review studies on ELM theory and applications, the readers are referred to Huang et al. [22], Huang et al. [24] and Deng et al. [11]. The underlying reasons for attention are its superior properties like extremely fast learning speed, simplicity and convincing performance in different learning problems including supervising [22], unsupervising [20] and semisupervising tasks [20]. Although ELM produces successful results in many real-world applications, it has several drawbacks caused by its learning nature. As a result of structural risk minimization approach [47], ELM may provide poor results in terms of stability, generalization performance and sparsity perspectives. In some situations, even the ELM presents efficient convergence in theory, its results may be overfitted or underfitted compared to what it should be in reality. Besides, the performance of the ELM significantly depends on the number of the hidden layer nodes. The selection of optimal node number is a complicated issue because of the possibility of overfitting or underfitting [30].

In order to deal with the ELM drawbacks, alternative variants of the ELM have been proposed to improve

✉ Hasan Yıldırım
hasanyildirim@kmu.edu.tr

M. Revan Özkale
mrevan@cu.edu.tr

[1] Department of Mathematics, Kamil Özdağ Faculty of Science, Karamanoğlu Mehmetbey University, 70100 Karaman, Turkey

[2] Department of Statistics, Faculty of Science and Letters, Çukurova University, 01330 Adana, Turkey

performance. Some of them have been mentioned in this study. A majority of these studies are based on regularization methods including the norms like $L_2$ (i.e., Tikhonov, ridge) and $L_1$ (i.e., Lasso).

When collinearity exists among the columns of the output matrix of hidden layer, multicollinearity exists. In case of multicollinearity, ELM results cannot be obtained or they will be unstable. The ridge estimator was proposed by Hoerl and Kennard [18] to deal with multicollinearity problem in linear models. The first appearance of ridge estimator in the context of ELM has been presented by Deng et al. [12] and later proposed by Li and Niu [25] in a regular form. Li and Niu [25] proposed an enhanced ELM algorithm based on the ridge regression (RR-ELM) and showed that RR-ELM may provide more stable and generalizable results than the conventional ELM. Cao et al. [5] proposed a suitable method for both classification and regressional studies called as stacked sparse denoising autoencoder—ridge regression (SSDAE-RR) to obtained more stable and generalizable network model. Shao et al. [43] developed a new regularized ELM based on leave-one-out cross-validation (LOO-CV) to determine the optimal model and get better learning performance. Chen et al. [8] proposed a two-stage method based on uninformative variable elimination and ridge regression to carry on a ridge-based ELM with the most informative feature. Yu et al. [57] developed a novel dual adaptive regularized online sequential extreme learning machine (DA-ROS-ELM) to deal with ill-posing and over-fitting on problems related to network intrusion. Wang and Li [49] presented one of the first studies on ELM for survival data and proposed a new method called as ELM-CoxBAR (a kernel ELM Cox model regularized by an L-0-based broken adaptive ridge) to reduce the computational cost for high-dimensional survival data. Yıldırım and Özkale [54] have found that the performance of ELM algorithms based on the ridge and almost unbiased ridge estimators [35] depends on the selection method of the parameter and they proposed different criteria including Akaike information criterion (AIC), Bayesian information criterion (BIC) and cross-validation (CV) for the selection of the tuning parameter in the context of ELM. Luo et al. [29] presented a weighted extreme learning machine with distribution optimization using ridge regression to classify user behavior prediction. Yan et al. [53] proposed a new algorithm called as artificial bee colony algorithm-based kernel ridge regression to provide more efficient results on insurance fraud identification. Raza et al. [38] presented k-sparse ELM to select the most informative features to obtain more compact model. In this study, it is shown that the selection method of parameter is effective on the performance of ridge-based algorithms. ELM algorithms based on ridge regression improve ELM performance in a certain extent.

Although ELM algorithms in regression studies based on $L_2$-norm have been widely adapted by researchers, there are alternatives to the ridge estimator. Liu estimator [27] is one of the alternatives which is effective for dealing with multicollinearity problem. The advantage of Liu estimator over the ridge estimator is to have a linear form of the tuning parameter unlike the nonlinear form in the ridge estimator. This property gives the possibility to determine the tuning parameter easier and faster than ridge. Therefore, the superiority of Liu estimator can be beneficial to dealing with the instability and poor generalization performance of ELM. Hence, Yıldırım and Özkale [55] proposed an enhanced ELM algorithm based on Liu (Liu-ELM) with different selection methods for tuning parameter and showed that the proposed algorithm generally provides more stable and generalizable results than ELM and RR-ELM.

RR-ELM has been widely used in different fields due to its superiorities like stability, predictive performance and functionality on high-dimensionality settings over ELM. However, it does not present a sparse (i.e., compact) model which is quite important to deal with high-dimensional data sets including irrelevant or noisy features. In other words, RR-ELM does not carry out a variable selection process and affects on the compactness of the model. This yields less interpretable model compared to its competitors. Lasso regression proposed by Tibshirani [46] could be a remedy for these drawbacks of RR-ELM.

To make further, particularly in the presence of irrelevant features, $L_1$-norm has been considered in learning process of ELM. Miche et al. [31] proposed a pruning algorithm called optimally pruned extreme learning machine (OP-ELM) to obtained more robust results than ELM. Later on, in order to benefit from the advantages of $L_1$ and $L_2$ regularizations, several studies have been carried out. Firstly, Miche et al. [32] proposed a double-regularized ELM called Tikhonov-regularized OP-ELM (TROP-ELM) and obtained better results than OP-ELM [31] and the basic ELM. Afterward, Martinez et al. [30] proposed a regularized ELM to select the optimal number of hidden layer nodes and they adapted the elastic net method in ELM training phase and the results have been given comparatively. Luo et al. [28] proposed a unified form of ELM algorithm based on $L_1$ and $L_2$ forms to be used in regression and classification tasks. Shan et al. [42] proposed a new method called as interval LASSO-based ELM to determine the appropriate nodes of network and avoid from the overfitting problem. Li et al. [26] presented a new regularized algorithm based on $L_{2,1}$-norm for deal with noises and outliers to obtain more robust and compact networks. Preti et al. [37] developed novel online sequential extreme learning machine based on $L_{2,1}$-norm to

improve the processing time and memory usage for process real-time sequential data. Fan et al. [15] proposed two different algorithms based on $L_{1/2}$-norm having both group and smoothing group regularization to present more compact networks with effective learning capability.

The advantage of Lasso method is that it shrinks some of the weights exactly to zero. Thus, a much sparser model is obtained. Both RR-ELM and lasso-ELM shrink the weights matrix with proportional to the tuning parameters, but Lasso also carries out variable selection by setting some weights to zero. Due to these properties, Lasso and its variants have been attracted in many disciplines. Due to the impossibility of closed-form solution, several algorithms [14, 16, 40] have been proposed to obtain solution in Lasso-type problems. Although Lasso is well in high-dimensional data, there are some drawbacks of Lasso regression which are pointed out by Zou and Hastie [60]. In order to overcome the drawbacks of Lasso, Zou and Hastie [60] proposed elastic net as a regularization and variable selection method. In elastic net, the superiorities of both ridge and Lasso methods have been used in a unified model. Thus, an effective variable selection process can be carried out by considering the grouping effect (the relationships between variables).

Liu-ELM was proposed as an alternative to RR-ELM, and it has the disadvantage of that it does not do variable selection. However, when we combine Liu-ELM with $L_1$-norm, it does variable selection because of the feature of $L_1$-norm penalization. Thus, it may be more efficient and convenient than Liu-ELM and Lasso-based ELM algorithms. With the sparsity property of $L_1$-norm, the results may be more convincing for dealing with ELM's drawbacks. Taken together, the drawbacks mentioned above can be briefly summarized as follows:

The performance results may present poor generalization and stability performance.
Some of algorithms such as Liu-ELM and RR-ELM do not carry a variable (i.e., node in the context of ELM) selection process. Therefore, they do not have the sparsity property.
Ridge-type algorithms are based on a parameter selection which do not have a generally accepted way and could be adversely effective on the speed of algorithm.

In this study, we present a new algorithm which is a cascade of two regularization types including Liu estimator and $L_1$-norm to improve the ELM and its variants based on $L_1$- and $L_2$-norms in terms of stability and generalization performance. We called this novel algorithm as Liu–Lasso extreme learning machine (LL-ELM) algorithm. The main contributions of the proposed algorithm can be listed as follows:

The new method produces a solution to the multicollinearity problem like Liu-ELM and RR-ELM as well as its superiority compared to Liu-ELM and RR-ELM is that it does variable selection.
The new method does variable selection like Lasso; however, instead of shrinking the components toward the origin as Lasso does, it shrinks toward $d$ times ELM results where $d$ is a parameter between 0 and 1.
The new method, like Enet-ELM, does variable selection and depends on two tuning parameters, but the selection of the new tuning parameter is mathematically easier than that of Enet-ELM.

In the rest of the paper, a brief review of algorithms is presented in Sect. 2. The details of the proposed algorithm are given in Sect. 3. In Sect. 4, the details of parameter and model selection are provided. An experimental study is carried out to investigate the performances of all algorithms on the real data sets in Sect. 5. The discussions and conclusions are summarized in Sect. 6.

## 2 A brief review of algorithms

### 2.1 Extreme learning machine (ELM)

This section shortly presents ELM algorithm proposed by Huang et al. [30, 31]. The main property of ELM is to give a chance to select the number of hidden layer neurons and biases arbitrary (i.e., randomly). As a result of random selection, the estimation problem of the neural network is reduced to find a solution of a linear system which can be easily obtained analytically. Therefore, ELM provides faster and simpler learning and estimation processes than other learning algorithms like backpropagation. Consider a group of samples $(x_i, t_i)$, where $x_i = (x_{i1}, x_{i2}, ..., x_{in})^{\mathrm{T}} \in \mathbb{R}^n$ and $t_i = (t_{i1}, t_{i2}, ..., t_{im})^{\mathrm{T}} \in \mathbb{R}^m$ to be estimated in a problem. For a given activation function $(g)$ and a number of hidden layer neurons $(K)$, the mathematical model of estimation based on a single-layer feedforward neural network (SLFN) can be written as follows:

$$\mathbf{t}_j^{\mathrm{T}} = \sum_{i=1}^{K} \boldsymbol{\beta}_i^{\mathrm{T}} g\left(\mathbf{w}_i.\mathbf{x}_j + b_i\right), j = 1, 2, \ldots, N, \tag{1}$$

where $\mathbf{x}_j^{\mathrm{T}} = \left[x_{j1}, x_{j2,\ldots,}x_{jn}\right]$ is the input values corresponding to the original data points, $\mathbf{t}_j^{\mathrm{T}} = \left[t_{j1}, t_{j2,\ldots,}t_{jm}\right]$ is the outputs of neural network, $\mathbf{w}_i = [w_{i1}, w_{i2}, \ldots, w_{im}]^T$ and $b_i$ are the learning parameters between the $i$th hidden node and the input nodes. $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, ..., \beta_{im}]^{\mathrm{T}}$ is the weight vector linking the $i$th hidden node and the output nodes. $\mathbf{w}_i.\mathbf{x}_j$ corresponds to the inner product between $\mathbf{w}_i$ and $\mathbf{x}_j$ [30, 31]. Generally, $n$ is equal to the number of explanatory

variables and $m$ is equal to the number of response variables which is commonly taken as one in most practical applications. ELM algorithm as a SLFN claims that it can converge with zero error to the actual values of $N$ samples. In other words, there are always a group of optimal parameters (i.e., weight and bias values) to provide $\sum_{i=1}^{K} \left\|\mathbf{y}_j - t_j\right\| = 0$. Equation (1) can be written more compactly as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{T}, \tag{2}$$

where

$$\mathbf{H} = \begin{bmatrix} g(\mathbf{w}_1.\mathbf{x}_1 + b_1) & \ldots & g(\mathbf{w}_K.\mathbf{x}_1 + b_K) \\ \vdots & \ldots & \vdots \\ g(\mathbf{w}_1.\mathbf{x}_N + b_1) & \ldots & g(\mathbf{w}_K.\mathbf{x}_N + b_K) \end{bmatrix}_{N\times K}$$

is the output matrix of hidden layer in the neural network,

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1^{\mathrm{T}} \\ \vdots \\ \beta_K^{\mathrm{T}} \end{bmatrix}_{K\times m}$$

corresponds to the weights and

$$\mathbf{T} = \begin{bmatrix} \mathbf{t}_1^{\mathrm{T}} \\ \vdots \\ \mathbf{t}_N^{\mathrm{T}} \end{bmatrix}_{N\times m}$$

is the output values of neural network.

By selecting the learning parameters randomly, the weight matrix $(\beta)$ can be obtained via a classical least squares approach. Therefore, the estimation of $\boldsymbol{\beta}$ in ELM, say $\hat{\beta}_{\mathrm{ELM}}$, is equivalent to obtain the solution of the following objective function:

$$\hat{\boldsymbol{\beta}}_{\mathrm{ELM}} = \arg\min_{\boldsymbol{\beta}} \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_2^2, \tag{3}$$

where $\|.\|_2^2$ denotes the $L_2$-norm.

In order to find the solution of Eq. (3), the classical inverse can be used if $\mathbf{H}$ matrix has full-column rank. In this case, $\boldsymbol{\beta}$ is estimated as $\hat{\boldsymbol{\beta}}_{\mathrm{ELM}} = \left(\mathbf{H}^{\mathrm{T}}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathrm{T}$.

### 2.2 The variants of ELM based on $L_1$- and $L_2$-Norms

In the presence of multicollinearity, $H$ will not be full-column rank. The RR-ELM proposed by Li and Niu [25] as a solution of this problem has the closed-form solution

$$\hat{\boldsymbol{\beta}}_{\mathrm{RR-ELM}} = \left(\mathbf{H}^{\mathrm{T}}\ \mathbf{H} + k\mathbf{I}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{T},$$

where $\mathbf{I}$ is the identity matrix and $k > 0$ refers to the tuning parameter for RR-ELM

The $L_1$ minimization of system (2) that Miche et al. [30, 31] presented is the solution of

$$\hat{\boldsymbol{\beta}}_{\mathrm{Lasso-ELM}} = \arg\min_{\boldsymbol{\beta}} \left\{ \|\mathbf{H}\boldsymbol{\beta} - \ \mathbf{T}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \right\},$$

where $\|.\|_1$ denotes the $L_1$-norm and $\lambda$ is the tuning parameter.

The elastic net is the solution of [30, 32]:

$$\hat{\boldsymbol{\beta}}_{\mathrm{Enet-ELM}} = \arg\min_{\boldsymbol{\beta}} \left\{ \|\mathbf{H}\boldsymbol{\beta} - \ \mathbf{T}\|_2^2 + k\|\boldsymbol{\beta}\|_2^2 + \lambda\|\boldsymbol{\beta}\|_1 \right\},$$

where $k$ and $\lambda$ are the tuning parameters representing the size of $L_1$- and $L_2$-norms, respectively.

If the parameters of elastic net ($k$ and$\lambda$) are tuned carefully, both more predictive and sparse results can be obtained than Lasso or ridge regression. One of the parameters which should be tuned in elastic net is the ridge tuning parameter $(k)$. However, there is no consensus on the appropriate selection of $k$ parameter. There has been an extensive study for determination of optimal $(k)$ parameter. As a remedy of this problem, an alternative method called Liu estimator was proposed by Liu [27]. Similar to ridge regression, Liu can deal with multicollinearity problem by using a different parameter on the learning process. The difference between Liu and ridge is the form of tuning parameter. Although ridge includes a nonlinear form of $k$ biasing parameter, Liu offers a linear form of its $d$ parameter. The linear form makes Liu better and easier than ridge in terms of calculations and speed. Therefore, Liu estimator can be considered as an alternative to ridge in elastic net model. In this study, we propose a new regularization and variable selection method which is based on Liu and Lasso methods. In the following section, we present some details of Liu and the form of the proposed method called Liu–Lasso ELM (LL-ELM).

## 3 The proposed algorithm: LL-ELM

Liu and ridge deal with multicollinearity by shrinking coefficient with a tuning parameter and provide more stable and generalizable results than classical ELM model. Although both estimators are effective on accounting the variables with high correlations, Liu estimator is faster and easier than ridge in terms of selecting tuning parameters because of having linear form of tuning parameter. The objective function of Liu estimator can be defined as [55]

$$\hat{\boldsymbol{\beta}}_{\mathrm{Liu-ELM}} = \arg\min_{\boldsymbol{\beta}} \left\{ \|\mathbf{H}\boldsymbol{\beta} - \mathbf{T}\|_2^2 + \left\|d\hat{\boldsymbol{\beta}}_{\mathrm{ELM}} - \boldsymbol{\beta}\right\|_2^2 \right\}, 0 < d < 1, \tag{4}$$

where $d$ refers to the Liu tuning parameter. The solution of Eq. (4) is obtained in a closed form as follows:

$$\hat{\boldsymbol{\beta}}_{\text{Liu}-\text{ELM}}=\left(\mathbf{H}^{T}\ \mathbf{H}+\boldsymbol{I}\right)^{-1}\left(\mathbf{H}^{T}\mathbf{H}+d\hat{\boldsymbol{\beta}}_{\text{ELM}}\right),0<d<1.$$

Similar to elastic net method, the solution of Eq. (4) can be obtained via an augmented data set, which is defined as follows:

$$\tilde{\mathbf{H}}=\begin{pmatrix}\mathbf{H}\\ \mathbf{I}\end{pmatrix}\quad\text{and}\ \tilde{\mathbf{T}}=\begin{pmatrix}\mathbf{T}\\ d\hat{\boldsymbol{\beta}}_{\text{ELM}}\end{pmatrix}. \tag{5}$$

Then, the Liu estimator in Eq. (4) can be redefined in augmented form as

$$\hat{\boldsymbol{\beta}}_{\text{Liu}-\text{ELM}}=\arg\min_{\boldsymbol{\beta}}\left\{\left\|\tilde{\mathbf{H}}\boldsymbol{\beta}-\tilde{\mathbf{T}}\right\|_2^2\right\}.$$

When multicollinearity exists, the ELM estimates often have low bias but large variance, which results in prediction difficulty. Besides, when there exist a large number of nodes, interpolation problem occurs with the ELM estimates. Shrinkage estimation methods and variable selection methods are the standard techniques for improving the ELM in these cases. RR-ELM and Liu-ELM are such shrinkage estimation methods. Although RR-ELM and Liu-ELM give more stable results, they do not set any weight to zero and do not give an easily interpretable model. Lasso-ELM was proposed as a competitor to RR-ELM which shrinks some weights and sets others to zero. Similar to RR-ELM, Liu-ELM shrinks the weights, resulting in good prediction accuracy, but does not select weights; in other words, it does not set some weights to zero. To obtain more interpretable estimates, we here aim to combine the idea of Lasso-ELM and Liu-ELM and propose a new algorithm called Liu–Lasso ELM (LL-ELM). By inspiring the objective functions of Enet-ELM and Liu-ELM, the objective function of our proposed method is as

$$\hat{\boldsymbol{\beta}}_{LL-ELM}=\arg\min_{\boldsymbol{\beta}}\left\{\|\mathbf{H}\boldsymbol{\beta}-\mathbf{T}\|_2^2+\left\|d\hat{\boldsymbol{\beta}}_{ELM}-\boldsymbol{\beta}\right\|_2^2+\lambda\|\boldsymbol{\beta}\|_1\right\}. \tag{6}$$

The objective function given by Eq. (6) is proposed in such a way that the new method has length closer to the true parameter vector than ELM and carries the properties of Enet-ELM. Furthermore, it has the sparsity property of Lasso in virtue of Eq.(6). The objective function given by Eq. (6) can also be defined with augmented form as:

$$\hat{\boldsymbol{\beta}}_{LL-ELM}=\arg\min_{\boldsymbol{\beta}}\left\{\left\|\tilde{\mathbf{H}}\boldsymbol{\beta}-\ \tilde{\mathbf{T}}\right\|_2^2+\lambda\|\boldsymbol{\beta}\|_1\right\}. \tag{7}$$

where $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{T}}$ are defined in Eq (5) and $\lambda$ is any fixed non-negative parameter.

By defining the proposed method (LL-ELM) in Eq. (7), the problem is reduced to a Lasso problem, so that similar to the approach of elastic net, LARS-EN algorithm [60] can be used to estimate the $\boldsymbol{\beta}$ . In our study, we adopted the approach of Sjöstrand et al. [31] and LARS-EN algorithm with piecewise linear regularization path proposed by Rosset and Zhu [40]. For fixed $d$, LL-ELM problem is equivalent to a Lasso problem on the augmented data set. So LARS which is originally proposed by Efron et al. [11] and its variants can be directly used to compute the weights of LL-ELM.

In order to present the sparsity property of LL-ELM, a simple experiment has been carried out on the body fat data set which is also used in the experiment section. The solution paths of the coefficients (i.e., weights) of LL-ELM are given in Fig. 1 where $s$ corresponds to the fraction of $L_1$-norm of coefficients $(s)$ which is defined as $\left\|\hat{\boldsymbol{\beta}}_{\text{ELM}}\right\|/\max\left(\hat{\boldsymbol{\beta}}_{\text{ELM}}\right)$ and has a range in [0, 1]. The $d$ tuning parameter and the number of hidden layer nodes have been arbitrarily selected as 0.5 and 20, respectively. The node number is deliberately chosen to be small for a better visuality. It is expected that the solution paths are piecewise linear because of the property of the LARS-EN algorithm which is explained by Zou and Hastie [60]. Figure 1 shows the points at which the variables enter into the model. Therefore, according to Fig. 1, it can be said that the proposed algorithm can be considered a beneficial tool to obtain sparse models. The following algorithm can be used for experiments:

Algorithm 1

LL-ELM.

**Input:** Training set $\{(x_i,t_i)\}$, the maximum number of the hidden neurons$\{K\}$, an activation function $\{g(.)\}$, the number of trials $\{L\}$.

**Output:** The $\beta$ weight matrix.

1 Generate the initial parameters $\mathbf{w}_i$ and $b_i$, $1\le i\le K$, randomly.
2 Calculate the hidden layer output matrix $\{\mathbf{H}\}$ via Eq. (3) and obtain the ELM solution.
3 Find the optimal Liu parameters $\{\hat{d}\}$ via Eq. (8) or Eq. (9), respectively.
4 Solve the equation $\boldsymbol{\beta}_{LL-ELM}=\arg\min_{\boldsymbol{\beta}}\left\{\left\|\tilde{\mathbf{H}}\boldsymbol{\beta}-\tilde{\mathbf{T}}\right\|_2^2+\lambda\|\boldsymbol{\beta}\|_1\right\}$ by using LARS-EN algorithm as $\boldsymbol{\beta}=\text{LARS-EN}\left(\tilde{H},\tilde{T},\hat{d}\right)$
5 for $1\le t\le size(\beta)$ do
6 Calculate BIC$(t)$ using each possible model using Eq. (10).
7 end
8 Find the $t^*$ corresponding to the minimum BIC value among all possible models of $\beta$ vector.
9 Select the optimum weights vector as $\beta_{Best}=\beta(t^*)$.

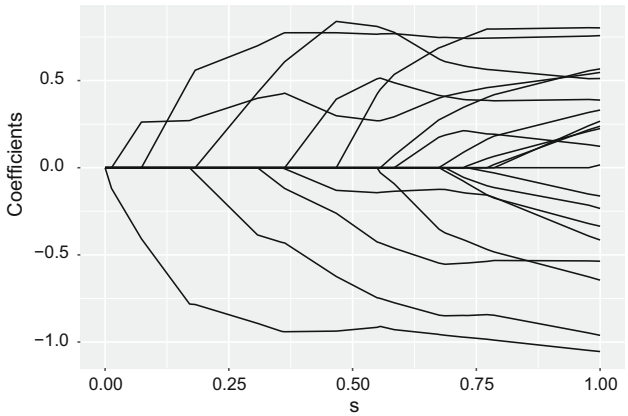


**Fig. 1** The coefficient estimates of LL-ELM based on the range of $s$

## 4 Parameter and model selection

As seen from Eq. (7) that LL-ELM depends on two tuning parameters $d$ and $\lambda$ and the objective function given by Eq. (7) is same with that of Lasso-ELM when $d$ is fixed. Therefore, the method for the selection of $\lambda$ can be same with that of Lasso-ELM when $d$ is fixed. Since first two terms in Eq. (7) are same with that of Eq. (6) which corresponds to the objective function of Liu-ELM, the LL-ELM could follow with Liu-ELM for the initialization of tuning parameter $d$.

It is clear that the selection of Liu tuning parameter is effective on the performance of Liu-ELM and LL-ELM. In the context of ELM, Yıldırım and Özkale [55] proposed the following methods:

$$\hat{d}_1 = 1 - \hat{\sigma}^2 \left[ \left( \sum_{i=1}^{K} \frac{1}{\lambda_i(\lambda_i+1)} \right) / \left( \sum_{i=1}^{K} \frac{\hat{\alpha}_i^2}{(\lambda_i+1)^2} \right) \right] \tag{8}$$

and

$$\hat{d}_2 = 1 - \hat{\sigma}^2 \left[ \left( \sum_{i=1}^{\tilde{N}} \frac{1}{(\lambda_i+1)} \right) / \left( \sum_{i=1}^{\tilde{N}} \frac{\lambda_i \hat{\alpha}_i^2}{(\lambda_i+1)^2} \right) \right], \tag{9}$$

where $\lambda_1, \lambda_2, ..., \lambda_K$ correspond to the eigenvalues of the $\mathbf{H}^{\mathrm{T}}\mathbf{H}$ matrix. $\hat{\alpha}_i$ is the $i$th element of $\hat{\alpha} = \mathbf{P}^{\mathrm{T}}\hat{\boldsymbol{\beta}}_{\mathrm{ELM}}$ and $\mathbf{P}_{KxK}$ is the orthogonal matrix whose columns are the eigenvectors of $\mathbf{H}^T\mathbf{H}$. $\hat{\sigma}^2$ is the estimate of the variance of residuals which are the residents between actual and model output values. Yıldırım and Özkale [55] presented $\hat{d}_1$ as the minimizer of the scalar mean square error of $\hat{\boldsymbol{\beta}}_{Liu-ELM}$ and $\hat{d}_2$ as the minimizer of the $C_p$ statistic under $\hat{\boldsymbol{\beta}}_{Liu-ELM}$ which was defined as

$$C_p = \frac{\mathrm{SS}_{\mathrm{Res,d}}}{\hat{\sigma}^2} + 2\mathrm{tr}(\mathbf{M}_{\mathrm{d}}) - (\mathrm{n}-2),$$

**Table 1** The properties of data sets used in this study

| Data set | Attributes | Split size | |
|---|---|---|---|
| | | Training | Testing |
| Body fat | 14 | 179 | 73 |
| Energy | 8 | 540 | 228 |
| Fish | 6 | 637 | 271 |
| Bank domains | 8 | 5736 | 2456 |
| Abalone | 8 | 2925 | 1252 |

where $M_d = \mathbf{H}\left(\mathbf{H}^{\mathrm{T}}\mathbf{H}+I\right)^{-1}\left(\mathbf{H}^{\mathrm{T}}\mathbf{H}+\mathrm{dI}\right)\left(\mathbf{H}^T\mathbf{H}\right)^{-1}\mathbf{H}^T$ is the quasi-projection matrix and $\mathrm{SS}_{\mathrm{Res,d}}$ is the residual sum of squares using $\hat{\boldsymbol{\beta}}_{\mathrm{Liu-ELM}}$.

In Liu-ELM, $\hat{d}_1$ and $\hat{d}_2$ values given in Eq. (8) and Eq. (9) are obtained using training data and used for measuring testing performance. For the proposed algorithm (LL-ELM), same $\hat{d}_1$ and $\hat{d}_2$ values with Liu-ELM can be used for the initial parameters. For each fixed $\hat{d}_1$ and $\hat{d}_2$, the $\lambda$ parameter is needed to be tuned carefully. There are various ways like AIC, BIC and CV to determine it. All of these methods have been widely used in the literature. Among these methods, BIC tends to produce more parsimonious models (i.e., more compact). This property of BIC may guarantee an optimal model instead of underfitted or overfitted one. The optimal $\lambda$ value is determined via BIC, which is defined as follows:

$$\boldsymbol{BIC}_t = \|\mathbf{T} - \mathbf{H}\boldsymbol{\beta}_t\|_2^2 + \log(N)\hat{\sigma}^2 L_t, \tag{10}$$

where $\boldsymbol{\beta}_t$ is the $t$th model obtained with each possible combination of $(d, \lambda)$, $N$ is the size of training set, $\hat{\sigma}^2$ is mean of squares of residuals and $L_t$ is the number of positive elements in $\boldsymbol{\beta}_t$ vector. The $(d, \lambda)$ combination providing minimum BIC value is selected for each $d$ value, and an overall examination among all $(d)$ values is carried out. The best $(d, \lambda)$ pair is selected for final model and used for obtaining testing results.

## 5 Experimental study: real data sets

In this section, a performance comparison has been carried out on several benchmark data sets to investigate the effectiveness of the algorithms. All data sets have been obtained from UCI repository [13] and standardized to have zero mean and unit variance. For LL-ELM, the effect of standardization is to avoid from the adverse effect of magnitude of each variable and to get more reasonable constraints which is also effective on the performance of the model. Standardization has a common usage for

**Table 2** The results of optimal tuning parameters for each algorithm

| Data set | Algorithm | Data | $d$ | $SD$ | $k$ | $s$ | $Node$ |
|---|---|---|---|---|---|---|---|
| Body fat | ELM [24] | Hold-out | – | – | – | – | 100 |
| | | Fivefold CV | – | – | – | – | 100 |
| | Ridge-ELM [25] | Hold-out | – | – | $10^{-12}$ | – | 100 |
| | | Fivefold CV | – | – | $10^{-10}$ | – | 100 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.63 | 0.07 | – | – | 100 |
| | | Fivefold CV | 0.45 | 0.13 | – | – | 100 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.89 | 0.02 | – | – | 100 |
| | | Fivefold CV | 0.83 | 0.04 | – | – | 100 |
| | Lasso-ELM [30, 31] | Hold-out | – | – | – | 0.71 | 81.25 |
| | | Fivefold CV | – | – | – | 0.64 | 75.40 |
| | Enet-ELM [30, 32] | Hold-out | – | – | $10^{-2}$ | 0.91 | 93.30 |
| | | Fivefold CV | – | – | $10^{-2}$ | 0.88 | 90.41 |
| | LL-ELM ($d_1$) | Hold-out | 0.63 | 0.07 | – | 0.98 | 97.75 |
| | | Fivefold CV | 0.45 | 0.13 | – | 0.87 | 86.28 |
| | LL-ELM ($d_2$) | Hold-out | 0.89 | 0.02 | – | 0.98 | 98.70 |
| | | Fivefold CV | 0.83 | 0.04 | – | 0.98 | 97.76 |
| Energy | ELM [24] | Hold-out | – | – | – | – | 100 |
| | | Fivefold CV | – | – | – | – | 100 |
| | Ridge-ELM [25] | Hold-out | – | – | $10^{-12}$ | – | 100 |
| | | Fivefold CV | – | – | $10^{-10}$ | – | 100 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.84 | 0.065 | – | – | 100 |
| | | Fivefold CV | 0.82 | 0.064 | – | – | 100 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.99 | 0.007 | – | – | 100 |
| | | Fivefold CV | 0.99 | 0.006 | – | – | 100 |
| | Lasso-ELM [30, 31] | Hold-out | – | – | – | 0.08 | 43.45 |
| | | Fivefold CV | – | – | – | 0.073 | 51.85 |
| | Enet-ELM [30, 32] | Hold-out | – | – | $10^{-1}$ | 0.80 | 71.10 |
| | | Fivefold CV | – | – | $10^{-1}$ | 0.755 | 46.95 |
| | LL-ELM ($d_1$) | Hold-out | 0.84 | 0.065 | – | 0.998 | 99.60 |
| | | Fivefold CV | 0.82 | 0.064 | – | 0.998 | 99.68 |
| | LL-ELM ($d_2$) | Hold-out | 0.99 | 0.007 | – | 0.998 | 99.45 |
| | | Fivefold CV | 0.99 | 0.006 | – | 0.998 | 99.75 |
| Fish | ELM [24] | Hold-out | – | – | – | – | 100 |
| | | Fivefold CV | – | – | – | – | 100 |
| | Ridge-ELM [25] | Hold-out | – | – | $10^{-13}$ | – | 100 |
| | | Fivefold CV | – | – | $10^{-10}$ | – | 100 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.27 | 0.26 | – | – | 100 |
| | | Fivefold CV | 0.22 | 0.27 | – | – | 100 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.93 | 0.02 | – | – | 100 |
| | | Fivefold CV | 0.90 | 0.03 | – | – | 100 |
| | Lasso-ELM [30, 31] | Hold-out | – | – | – | 0.03 | 16.30 |
| | | Fivefold CV | – | – | – | 0.03 | 16.47 |
| | Enet-ELM [30, 32] | Hold-out | – | – | 0.1 | 0.13 | 16.60 |
| | | Fivefold CV | – | – | | 0.12 | 16.96 |
| | LL-ELM ($d_1$) | Hold-out | 0.27 | 0.26 | – | 0.61 | 62.60 |
| | | Fivefold CV | 0.22 | 0.27 | – | 0.52 | 55.38 |
| | LL-ELM ($d_2$) | Hold-out | 0.93 | 0.02 | – | 0.99 | 98.45 |
| | | Fivefold CV | 0.90 | 0.03 | – | 0.98 | 98.29 |

**Table 2** (continued)

| Data set | Algorithm | Data | $d$ | $SD$ | $k$ | $s$ | $Node$ |
|---|---|---|---|---|---|---|---|
| Bank Domains | ELM [24]] | Hold-out | – | – | – | – | 100 |
| | | Fivefold CV | – | – | – | – | 100 |
| | Ridge-ELM [25] | Hold-out | – | – | $10^{-13}$ | – | 100 |
| | | Fivefold CV | – | – | $10^{-13}$ | – | 100 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.10 | 0.29 | – | – | 100 |
| | | Fivefold CV | 0.16 | 0.16 | – | – | 100 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.26 | 0.16 | – | – | 100 |
| | | Fivefold CV | 0.27 | 0.14 | – | – | 100 |
| | Lasso-ELM [30, 31] | Hold-out | – | – | – | 0.94 | 94.30 |
| | | Fivefold CV | – | – | – | 0.92 | 94.10 |
| | Enet-ELM [30, 32] | Hold-out | – | – | $10^{-3}$ | 0.99 | 99.05 |
| | | Fivefold CV | – | – | $10^{-3}$ | 0.91 | 92.90 |
| | LL-ELM ($d_1$) | Hold-out | 0.10 | 0.29 | – | 0.94 | 93.65 |
| | | Fivefold CV | 0.16 | 0.16 | – | 0.99 | 99.10 |
| | LL-ELM ($d_2$) | Hold-out | 0.26 | 0.16 | – | 0.90 | 89.80 |
| | | Fivefold CV | 0.27 | 0.14 | – | 0.98 | 98.6 |
| Abalone | ELM [24] | Hold-out | – | – | – | – | 100 |
| | | Fivefold CV | – | – | – | – | 100 |
| | Ridge-ELM [25] | Hold-out | – | – | $10^{-12}$ | – | 100 |
| | | Fivefold CV | – | – | $10^{-11}$ | – | 100 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.15 | 0.19 | – | – | 100 |
| | | Fivefold CV | 0.19 | 0.13 | – | – | 100 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.83 | 0.04 | – | – | 100 |
| | | Fivefold CV | 0.80 | 0.04 | – | – | 100 |
| | Lasso-ELM [30, 31] | Hold-out | – | – | – | 0.11 | 37.20 |
| | | Fivefold CV | – | – | – | 0.099 | 35.40 |
| | Enet-ELM [30, 32] | Hold-out | – | – | $10^{-1}$ | 0.38 | 44.00 |
| | | Fivefold CV | – | – | $10^{-1}$ | 0.223 | 36.20 |
| | LL-ELM ($d_1$) | Hold-out | 0.15 | 0.19 | – | 0.34 | 35.80 |
| | | Fivefold CV | 0.19 | 0.13 | – | 0.576 | 60.66 |
| | LL-ELM ($d_2$) | Hold-out | 0.83 | 0.04 | – | 0.98 | 98.10 |
| | | Fivefold CV | 0.80 | 0.04 | – | 0.98 | 98.76 |

training Lasso-based models. That is why, it is deliberately preferred not to use the data sets including mostly categorical variables for testing LL-ELM algorithm and is aimed to compare the algorithms under same conditions which they are firstly proposed. Also, it is assumed that the dependent variable spans sufficient enough through the attribute space. The properties of data set are given in Table 1.

Both hold-out and k-fold cross-validation approaches have been separately conducted to validate the efficiency of the algorithms. For hold-out approach, each data set is split into training and testing data sets with the ratios 70 and 30%, respectively. On the other side, fivefold CV is applied for k-fold CV approach. Twenty trials have been conducted to eliminate the randomness effect of initial parameters assignments. The initial number of the hidden layer nodes is fixed as 100, and sigmoid activation function is used for all data sets and experiments.

The experiments have been carried out via R software. In order to train LL-ELM, ELM variants based on Lasso and elastic net algorithms, LARS-EN algorithm with piecewise linear regularization path proposed by Rosset and Zhu [40] has been used. All coding processes of each algorithm have been carried out from scratch in the R platform.

**Table 3** Comparison of all algorithms in terms of training and testing RMSE

| Data set | Algorithm | Data | Training | SD | Testing | RR(%) $d_1$ | RR(%) $d_2$ | SD | RR(%) $d_1$ | RR(%) $d_2$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Body fat | ELM [24] | Hold-out | 0.05924 | 0.00810 | 0.26786 | 8.68 | 2.87 | 0.02831 | 19.82 | 4.69 |
| | | Fivefold CV | 0.09121 | 0.00688 | 0.24579 | 14.89 | 6.54 | 0.04622 | 38.36 | 10.71 |
| | Ridge-ELM [25] | Hold-out | 0.05920 | 0.00810 | 0.26785 | 8.68 | 2.87 | 0.02828 | 19.82 | 4.69 |
| | | Fivefold CV | 0.09121 | 0.09121 | 0.24579 | 14.89 | 6.54 | 0.04622 | 38.36 | 10.71 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.06821 | 0.00962 | 0.24545 | 0.35 | −6.00 | 0.02318 | 2.05 | −16.43 |
| | | Fivefold CV | 0.10661 | 0.01085 | 0.20758 | −0.78 | −10.67 | 0.03027 | 5.88 | −36.34 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.06018 | 0.00828 | 0.26040 | 6.07 | 0.09 | 0.02670 | 14.97 | −1.08 |
| | | Fivefold CV | 0.09277 | 0.00722 | 0.23061 | 9.28 | 0.39 | 0.04181 | 31.86 | 1.29 |
| | Lasso-ELM [30, 31] | Hold-out | 0.07899 | 0.01808 | 0.25684 | 4.76 | −1.30 | 0.02666 | 14.86 | −1.22 |
| | | Fivefold CV | 0.13182 | 0.02931 | 0.23722 | 11.81 | 3.16 | 0.03717 | 23.35 | −11.03 |
| | Enet-ELM [30, 32] | Hold-out | 0.06059 | 0.01480 | 0.25794 | 5.17 | −0.87 | 0.02799 | 18.90 | 3.59 |
| | | Fivefold CV | 0.10313 | 0.01670 | 0.22909 | 8.68 | −0.28 | 0.04016 | 29.06 | −2.76 |
| | LL-ELM ($d_1$) | Hold-out | 0.06564 | 0.00944 | 0.24460 | – | −6.37 | 0.02270 | – | −18.88 |
| | | Fivefold CV | 0.11719 | 0.02256 | 0.20920 | – | −9.81 | 0.02849 | – | −44.86 |
| | LL-ELM ($d_2$) | Hold-out | 0.05092 | 0.00742 | 0.26018 | 5.99 | – | 0.02699 | 15.88 | – |
| | | Fivefold CV | 0.09404 | 0.00767 | 0.22972 | 8.93 | – | 0.04127 | 30.97 | – |
| Energy | ELM [24] | Hold-out | 0.16976 | 0.00991 | 0.21523 | 2.82 | −3.58 | 0.01617 | 6.19 | −162.36 |
| | | Fivefold CV | 0.17226 | 0.00855 | 0.21029 | 1.07 | −0.56 | 0.01303 | 9.67 | −27.78 |
| | Ridge-ELM [25] | Hold-out | 0.16976 | 0.00988 | 0.21522 | 2.81 | −3.59 | 0.01613 | 6.18 | −162.39 |
| | | Fivefold CV | 0.17226 | 0.00855 | 0.21029 | 1.07 | −0.56 | 0.01303 | 9.67 | −27.78 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.17248 | 0.01095 | 0.20932 | 0.07 | −6.51 | 0.01498 | −1.24 | −183.14 |
| | | Fivefold CV | 0.17493 | 0.00948 | 0.20796 | −0.03 | −1.68 | 0.01185 | 0.68 | −40.51 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.16977 | 0.00992 | 0.21466 | 2.56 | −3.86 | 0.01598 | 5.06 | −165.51 |
| | | Fivefold CV | 0.17227 | 0.00856 | 0.20993 | 0.91 | −0.73 | 0.01291 | 8.83 | −28.97 |
| | Lasso-ELM [30, 31] | Hold-out | 0.24887 | 0.02330 | 0.27770 | 24.68 | 19.72 | 0.01627 | 6.79 | −160.69 |
| | | Fivefold CV | 0.26956 | 0.02421 | 0.28482 | 26.96 | 25.76 | 0.02270 | 48.15 | 26.65 |
| | Enet-ELM [30, 32] | Hold-out | 0.21968 | 0.00861 | 0.26511 | 21.10 | 15.91 | 0.00957 | −58.54 | −343.40 |
| | | Fivefold CV | 0.24727 | 0.01159 | 0.26817 | 22.43 | 21.15 | 0.01306 | 9.88 | −27.49 |
| | LL-ELM ($d_1$) | Hold-out | 0.15944 | 0.01042 | 0.20917 | – | −6.58 | 0.01517 | – | −179.67 |
| | | Fivefold CV | 0.17519 | 0.00948 | 0.20803 | – | −1.65 | 0.01177 | – | −41.46 |
| | LL-ELM ($d_2$) | Hold-out | 0.16673 | 0.03764 | 0.22294 | 6.18 | – | 0.04242 | 64.24 | – |
| | | Fivefold CV | 0.17377 | 0.01297 | 0.21146 | 1.62 | – | 0.01665 | 29.31 | – |
| Fish | ELM [24] | Hold-out | 0.49160 | 0.00698 | 0.77594 | 13.35 | 2.27 | 0.06101 | 55.91 | 5.61 |
| | | Fivefold CV | 0.51388 | 0.00693 | 0.68299 | 7.27 | 1.72 | 0.04328 | 54.48 | 8.09 |
| | Ridge-ELM [25] | Hold-out | 0.49158 | 0.00690 | 0.77591 | 13.35 | 2.26 | 0.06100 | 55.90 | 5.60 |
| | | Fivefold CV | 0.51388 | 0.00693 | 0.68299 | 7.27 | 1.72 | 0.04328 | 54.48 | 8.09 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.52827 | 0.02571 | 0.66426 | −1.22 | −14.17 | 0.02918 | 7.83 | −97.32 |
| | | Fivefold CV | 0.54898 | 0.02421 | 0.62138 | −1.92 | −8.03 | 0.01786 | −10.30 | −122.73 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.49194 | 0.00707 | 0.75992 | 11.53 | 0.21 | 0.05782 | 53.47 | 0.40 |
| | | Fivefold CV | 0.51443 | 0.00710 | 0.67003 | 5.48 | −0.18 | 0.03914 | 49.67 | −1.64 |
| | Lasso-ELM [30, 31] | Hold-out | 0.55793 | 0.01405 | 0.67020 | −0.32 | −13.15 | 0.00983 | −173.56 | −485.61 |
| | | Fivefold CV | 0.61914 | 0.02035 | 0.64065 | 1.14 | −4.78 | 0.01593 | −23.67 | −149.72 |
| | Enet-ELM [30, 32] | Hold-out | 0.55376 | 0.01901 | 0.67272 | 0.06 | −12.73 | 0.01126 | −138.98 | −411.59 |
| | | Fivefold CV | 0.61383 | 0.02080 | 0.63838 | 0.79 | −5.15 | 0.01179 | −67.09 | −237.40 |
| | LL-ELM ($d_1$) | Hold-out | 0.52782 | 0.05457 | 0.67233 | – | −12.79 | 0.02690 | – | −114.07 |
| | | Fivefold CV | 0.58100 | 0.04826 | 0.63334 | – | −5.99 | 0.01970 | – | −101.93 |
| | LL-ELM ($d_2$) | Hold-out | 0.45960 | 0.00785 | 0.75835 | 11.34 | – | 0.05758 | 53.29 | – |
| | | Fivefold CV | 0.51779 | 0.01681 | 0.67126 | 5.65 | – | 0.03978 | 50.48 | – |

**Table 3** (continued)

| Data set | Algorithm | Data | Training | SD | Testing | RR(%) $d_1$ | RR(%) $d_2$ | SD | RR(%) $d_1$ | RR(%) $d_2$ |
|---|---|---|---|---|---|---|---|---|---|---|
| Bank domains | Elm [24] | Hold-out | 0.21702 | 0.00401 | 0.21852 | −0.26 | −0.67 | 0.00511 | 3.82 | −3.01 |
| | | Fivefold CV | 0.21776 | 0.00203 | 0.22200 | 0.03 | −0.06 | 0.00218 | 2.75 | 16.97 |
| | Ridge-ELM [25] | Hold-out | 0.21702 | 0.00400 | 0.21849 | −0.26 | −0.67 | 0.00510 | 3.82 | −3.01 |
| | | Fivefold CV | 0.21776 | 0.00203 | 0.22200 | 0.03 | −0.06 | 0.00218 | 2.75 | 16.97 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.21738 | 0.00413 | 0.21843 | −0.29 | −0.71 | 0.00493 | 0.25 | −6.83 |
| | | Fivefold CV | 0.21795 | 0.00183 | 0.22195 | 0.00 | −0.09 | 0.00210 | −0.95 | 13.81 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.21725 | 0.00406 | 0.21838 | −0.32 | −0.73 | 0.00495 | 0.71 | −6.34 |
| | | Fivefold CV | 0.21794 | 0.00187 | 0.22193 | 0.00 | −0.09 | 0.00210 | −0.95 | 13.81 |
| | Lasso-ELM [30, 31] | Hold-out | 0.21624 | 0.00460 | 0.21948 | 0.19 | −0.23 | 0.00551 | 10.75 | 4.42 |
| | | Fivefold CV | 0.21915 | 0.00196 | 0.22325 | 0.59 | 0.50 | 0.00227 | 6.61 | 20.26 |
| | Enet-ELM [30, 32] | Hold-out | 0.21576 | 0.00421 | 0.21895 | −0.06 | −0.47 | 0.00524 | 6.17 | −0.50 |
| | | Fivefold CV | 0.21857 | 0.00174 | 0.22259 | 0.29 | 0.20 | 0.00239 | 11.30 | 24.27 |
| | LL-ELM ($d_1$) | Hold-out | 0.21835 | 0.00509 | 0.21907 | – | −0.41 | 0.00492 | – | −7.10 |
| | | Fivefold CV | 0.21800 | 0.00183 | 0.22194 | – | −0.09 | 0.00212 | – | 14.62 |
| | LL-ELM ($d_2$) | Hold-out | 0.21829 | 0.00475 | 0.21998 | 0.41 | – | 0.00526 | 6.63 | – |
| | | Fivefold CV | 0.21809 | 0.00167 | 0.22214 | 0.09 | – | 0.00181 | −17.13 | – |
| Abalone | ELM [24] | Hold-out | 0.63230 | 0.00221 | 0.64882 | −1.75 | 0.64 | 0.00455 | −187.38 | 11.57 |
| | | Fivefold CV | 0.62897 | 0.00105 | 0.67080 | 1.00 | 0.99 | 0.01482 | 41.16 | 26.99 |
| | Ridge-ELM [25] | Hold-out | 0.63230 | 0.00220 | 0.64880 | −1.75 | 0.64 | 0.00454 | −187.39 | 11.57 |
| | | Fivefold CV | 0.62897 | 0.00105 | 0.67080 | 1.00 | 0.99 | 0.01482 | 41.16 | 26.99 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 0.64343 | 0.00561 | 0.63915 | −3.29 | −0.86 | 0.00243 | −438.88 | −65.82 |
| | | Fivefold CV | 0.63761 | 0.00281 | 0.65599 | −1.23 | −1.24 | 0.00306 | −184.97 | −253.59 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 0.63277 | 0.00228 | 0.64449 | −2.44 | −0.02 | 0.00380 | −243.58 | −5.72 |
| | | Fivefold CV | 0.62950 | 0.00110 | 0.66413 | 0.01 | 0.00 | 0.01100 | 20.73 | 1.64 |
| | Lasso-ELM [30, 31] | Hold-out | 0.65770 | 0.01374 | 0.65823 | −0.30 | 2.06 | 0.01016 | −28.71 | 60.40 |
| | | Fivefold CV | 0.66662 | 0.00911 | 0.67677 | 1.87 | 1.87 | 0.00732 | −19.13 | −47.81 |
| | Enet-ELM [30, 32] | Hold-out | 0.65028 | 0.01082 | 0.65149 | −1.33 | 1.06 | 0.00845 | −52.80 | 52.98 |
| | | Fivefold CV | 0.65922 | 0.00567 | 0.67115 | 1.05 | 1.05 | 0.00594 | −46.80 | −82.15 |
| | LL-ELM ($d_1$) | Hold-out | 0.66915 | 0.02135 | 0.66020 | – | 2.36 | 0.01307 | – | 69.23 |
| | | Fivefold CV | 0.65155 | 0.01193 | 0.66409 | – | −0.01 | 0.00872 | – | −24.08 |
| | LL-ELM ($d_2$) | Hold-out | 0.62302 | 0.00281 | 0.64465 | −2.41 | – | 0.00402 | −224.98 | – |
| | | Fivefold CV | 0.62985 | 0.00107 | 0.66413 | 0.01 | – | 0.01082 | 19.41 | – |

In Lasso-ELM, instead of tuning $\lambda$, it is suggested to use the fraction of $L_1$-norm of coefficients $(s)$ which is defined as $\left\|\hat{\boldsymbol{\beta}}_{\mathrm{ELM}}\right\| / \max\left(\hat{\boldsymbol{\beta}}_{\mathrm{ELM}}\right)$. Here, $\max\left(\hat{\boldsymbol{\beta}}_{\mathrm{ELM}}\right)$ actually corresponds to the ELM solution which is the $L_1$-norm of low-bias model solution. From the point of optimization perspective, there is a $\lambda$ value corresponding to each $s$ value and the solutions obtained via any form of this optimization problem are exactly the same each other. In other words, the $\hat{\boldsymbol{\beta}}$ solution corresponding to any $\lambda$ value in Lagrangian form solves the problem which has the bound of $s = \left\|\hat{\boldsymbol{\beta}}_{\lambda}\right\|$. The advantage of $s$ over $\lambda$ is to have values within $[0, 1]$. Similar to the process in LL-ELM, BIC is used to determine optimal $s$ value by using fixed $\hat{d}_1$ and $\hat{d}_2$ parameters as initial parameters. On the other hand, the ridge tuning parameter $k$ is selected from the sequence of $\left(10^{-15}, 10^{-14}, ..., 10^{1}\right)$. In RR-ELM, the $k$ value minimizing the training error is used to obtain training and testing results. In order to get the ELM results based on elastic net, for each fixed $k$ value, the optimal $s$ value minimizing BIC is calculated and the $(k, s)$ values giving the global minimum among all possible combinations are used for the final model's performance.

In addition to the performance results of each algorithm, the optimal parameters for both hold-out and fivefold CV

**Table 4** Norm comparison of Liu-ELM and LL-ELM algorithms

| Data set | Algorithm | Data | Norm | SD |
|---|---|---|---|---|
| Body fat | ELM [24] | Hold-out | 3.9955 | 0.4202 |
| | | Fivefold CV | 4.6458 | 0.4148 |
| | Ridge-ELM [25] | Hold-out | 3.9954 | 0.4202 |
| | | Fivefold CV | 4.6458 | 0.4148 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 2.9124 | 0.3426 |
| | | Fivefold CV | 2.7591 | 0.5416 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 3.6674 | 0.3904 |
| | | Fivefold CV | 4.0380 | 0.4478 |
| | Lasso-ELM [30, 31] | Hold-out | 3.1337 | 0.6508 |
| | | Fivefold CV | 3.3164 | 0.8600 |
| | Enet-ELM [30, 32] | Hold-out | 3.2759 | 0.4468 |
| | | Fivefold CV | 3.7298 | 0.6212 |
| | LL-ELM ($d_1$) | Hold-out | 2.8741 | 0.3457 |
| | | Fivefold CV | 2.5880 | 0.6461 |
| | LL-ELM ($d_2$) | Hold-out | 3.6368 | 0.4137 |
| | | Fivefold CV | 3.9905 | 0.4567 |
| Energy | ELM [24] | Hold-out | 48.6403 | 10.3119 |
| | | Fivefold CV | 44.0252 | 11.1549 |
| | Ridge-ELM [25] | Hold-out | 48.6116 | 10.2888 |
| | | Fivefold CV | 44.0252 | 11.1549 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 41.4017 | 11.8085 |
| | | Fivefold CV | 37.0551 | 11.8767 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 48.1587 | 10.4254 |
| | | Fivefold CV | 43.5191 | 11.2823 |
| | Lasso-ELM [30, 31] | Hold-out | 4.0826 | 2.2249 |
| | | Fivefold CV | 3.7841 | 1.8710 |
| | Enet-ELM [30, 32] | Hold-out | 4.1892 | 1.4575 |
| | | Fivefold CV | 4.9504 | 1.3993 |
| | LL-ELM ($d_1$) | Hold-out | 41.3467 | 11.8364 |
| | | Fivefold CV | 37.0280 | 11.8784 |
| | LL-ELM ($d_2$) | Hold-out | 48.1413 | 10.4214 |
| | | Fivefold CV | 43.4872 | 11.2717 |
| Fish | ELM [24] | Hold-out | 47.7400 | 9.5774 |
| | | Fivefold CV | 40.6623 | 7.1378 |
| | Ridge-ELM [25] | Hold-out | 47.7307 | 9.5728 |
| | | Fivefold CV | 40.6623 | 7.1378 |
| | Liu-ELM [55] ($d_1$) | Hold-out | 17.3667 | 13.5730 |
| | | Fivefold CV | 14.0094 | 9.1434 |
| | Liu-ELM [55] ($d_2$) | Hold-out | 44.6700 | 9.9829 |
| | | Fivefold CV | 37.1280 | 7.6645 |
| | Lasso-ELM [30, 31] | Hold-out | 2.5862 | 1.2739 |
| | | Fivefold CV | 2.5090 | 1.4640 |
| | Enet-ELM [30, 32] | Hold-out | 2.7503 | 1.4320 |
| | | Fivefold CV | 2.5880 | 1.3110 |
| | LL-ELM ($d_1$) | Hold-out | 15.3633 | 14.9289 |
| | | Fivefold CV | 11.3677 | 10.6316 |
| | LL-ELM ($d_2$) | Hold-out | 44.3644 | 10.1091 |
| | | Fivefold CV | 36.7401 | 7.9080 |

**Table 4** (continued)

| Data set | Algorithm | Data | Norm | SD |
|---|---|---|---|---|
| Bank domains | ELM [24] | Hold-out | 3.025553 | 0.2816 |
| | | Fivefold CV | 3.081977 | 0.0517 |
| | Ridge-ELM [25] | Hold-out | 3.025552 | 0.2816 |
| | | Fivefold CV | 3.081977 | 0.0517 |
| | Liu-ELM [55] $(d_1)$ | Hold-out | 2.60761 | 0.2594 |
| | | Fivefold CV | 2.779690 | 0.2781 |
| | Liu-ELM [55] $(d_2)$ | Hold-out | 2.67404 | 0.2454 |
| | | Fivefold CV | 2.775343 | 0.2324 |
| | Lasso-ELM [30, 31] | Hold-out | 2.886881 | 0.3804 |
| | | Fivefold CV | 2.894868 | 0.1547 |
| | Enet-ELM [30, 32] | Hold-out | 3.002505 | 0.301 |
| | | Fivefold CV | 2.898261 | 0.1497 |
| | LL-ELM $(d_1)$ | Hold-out | 2.521507 | 0.347 |
| | | Fivefold CV | 2.768495 | 0.2751 |
| | LL-ELM $(d_2)$ | Hold-out | 2.513822 | 0.406 |
| | | Fivefold CV | 2.750076 | 0.2579 |
| Abalone | Elm [24] | Hold-out | 36.4415 | 4.9751 |
| | | Fivefold CV | 34.0607 | 3.9836 |
| | Ridge-ELM [25] | Hold-out | 36.4395 | 4.9744 |
| | | Fivefold CV | 34.0607 | 3.9836 |
| | Liu-ELM [55] $(d_1)$ | Hold-out | 10.1211 | 6.0219 |
| | | Fivefold CV | 10.6560 | 3.7127 |
| | Liu-ELM [55] $(d_2)$ | Hold-out | 30.6996 | 5.6065 |
| | | Fivefold CV | 28.0417 | 4.5573 |
| | Lasso-ELM [30, 31] | Hold-out | 6.1214 | 2.1496 |
| | | Fivefold CV | 5.3479 | 1.2502 |
| | Enet-ELM [30, 32] | Hold-out | 6.7838 | 2.2010 |
| | | Fivefold CV | 5.8900 | 1.0210 |
| | LL-ELM $(d_1)$ | Hold-out | 6.3967 | 6.8261 |
| | | Fivefold CV | 8.3154 | 4.0896 |
| | LL-ELM $(d_2)$ | Hold-out | 30.4090 | 5.8022 |
| | | Fivefold CV | 27.6888 | 4.7145 |

approaches are presented in Table 2. In Table 2, the third and fourth columns correspond to the average $d$ values and their standard deviations which are calculated based on all trials. The $k$ and $s$ columns refer to the best parameters corresponding to the optimal $(k, s)$ or $(d, s)$ combination giving the overall minimum value of BIC. In the last column, the mean node number throughout all trials is reported.

When the sparsity results are examined, Lasso-ELM and Enet-ELM generally give more parsimonious models than LL-ELM. For bank domains and abalone data, LL-ELM based on $\hat{d}_2$ provides more sparse models. Table 3 shows the training and testing results of all algorithms based on data sets given in Table 1. The training and testing performance with their standard deviations is calculated by taking the averages of 20 trials. Based on the results for hold-out approaches in Table 3, the following interpretations can be said:

LL-ELM for at least one Liu tuning parameter overperforms to all algorithms in terms of training performance except bank domains data set.

According to the testing RMSE values, the proposed algorithm with $\hat{d}_1$ parameter is more generalizable than other algorithms for body fat and energy data sets. Similarly, LL-ELM is seen as stable in terms of standard deviation of testing performance. It provides best results

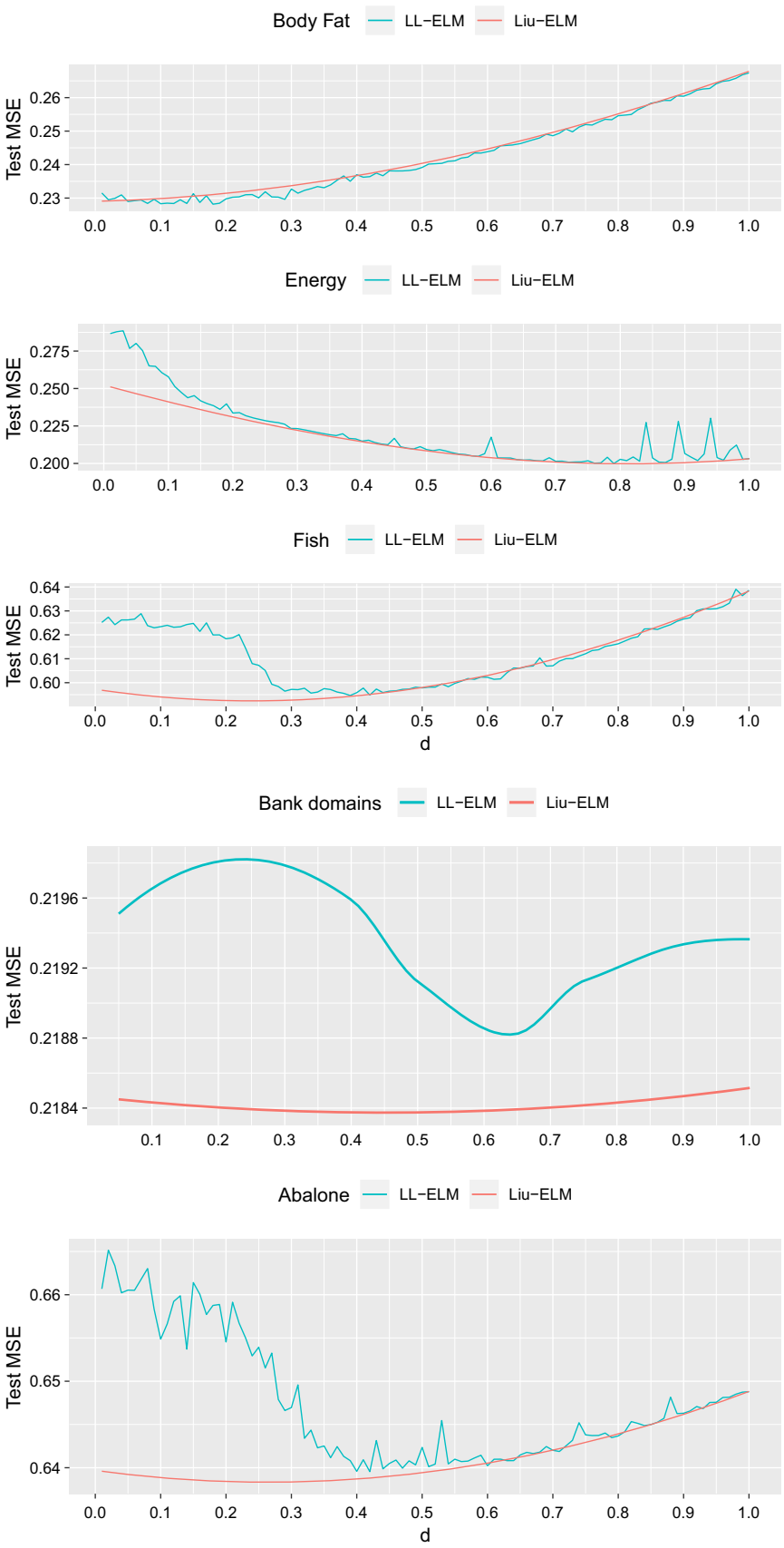


**Fig. 2** The performance comparison of Liu-ELM and LL-ELM based on random biasing parameter

for body fat and bank domains data sets. Liu-ELM is better than LL-ELM in terms of testing performance for fish, bank domains and abalone data sets. Additionally, Liu-ELM and LL-ELM are compared based on 100 trials with the random assignments of Liu tuning parameters within range $[0, 1]$, and the results are given in Fig. 1. In Fig. 1, it is seen that there is at least one Liu tuning parameter where LL-ELM overperforms Liu-ELM for all data sets except Bank Domains.

On the other side, fivefold CV approaches contribute some additional insights about the performances of the proposed algorithms. These insights can be listed as follows:

- Considering LL-ELM and Liu-ELM in all data sets, there is one $d$ value which gives better fivefold CV results in terms of either RMSE or SD than hold-out results.
- In all data sets, fivefold CV results for LL-ELM according to $d_1$ give higher reduction rate in testing RMSE values than Lasso-ELM and Enet-ELM when compared to hold-out results. Additionally, this is also true for the SD criterion with one exception of Lasso-ELM results for Bank Domains data set.
- In all data sets, according to SD criterion, LL-ELM for $d_2$ gives better reduction rate values than ELM and RR-ELM in fivefold CV criterion than hold-out criterion.

In order to present an insight into the regularization level of each algorithm, the norm values of coefficients obtained via Liu-ELM and LL-ELM are calculated for these data sets and are given in Table 4. Although Liu-ELM has lower testing RMSE value, the mean norm value for Liu-ELM is higher than that for LL-ELM for all data sets for both hold-out and fivefold CV approaches. This means that the proposed algorithm shrinks more severe than Liu-ELM.

## 6 Discussion and conclusions

In this paper, we proposed a novel regularization and variable selection algorithm to improve the conventional extreme learning machine and its variants. The proposed algorithm combines the benefits of Liu and Lasso regression methods to deal with the drawbacks of ELM like the instability, poor generalizability and under or overfitting problems. The experimental studies based on both hold-out and k-fold cross-validation approaches show that LL-ELM generally improves the training and testing performance of ELM and overperforms the well-known competitors. It is seen that LL-ELM has a notable shrinkage property compared with other algorithms, particularly Liu-ELM. Although LL-ELM does not carry out a hard variable selection (i.e., node selection) process like Lasso or elastic net, the level of shrinkage with an amount of sparsity is better to give good generalization performance. The norm of estimated coefficients is lower than other algorithms. This means that LL-ELM may guarantee lower norm estimated which can provide more stable and accurate results in terms of generalization performance. It should be noted that the level of LL-ELM's sparsity property can be improved by considering alternative ways of parameter selection methods. Besides, the proposed algorithm is a tool for both regressional and classification tasks in data-oriented studies.

The limitation LL-ELM is that it cannot be applied to high-dimensional data. Therefore, our next study will focus on solving this limitation.

## Compliance with ethical standards

**Conflict of interest** All authors declare that they have no conflict of interest.

## References


1. Barea R, Boquete L, Ortega S, López E, Rodríguez-Ascariz J-M (2012) EOG-based eye movements codification for human computer interaction. Expert Syst Appl 39:2677–2683. https://doi.org/10.1016/j.eswa.2011.08.123
2. Bazi Y, Alajlan N, Melgani F, AlHichri H, Malek S, Yager R-R (2014) Differential evolution extreme learning machine for the classification of hyperspectral images. IEEE Geosci Remote Sens Lett 11:1066–1070. https://doi.org/10.1109/LGRS.2013.2286078
3. Butcher JB, Verstraeten D, Schrauwen B, Day C-R, Haycock P-W (2013) Reservoir computing and extreme learning machines for non-linear time-series data analysis. Neural Netw 38:76–89. https://doi.org/10.1016/j.neunet.2012.11.011
4. Cao F, Liu B, Sun P-D (2013) Image classification based on effective extreme learning machine. Neurocomputing 102:90–97. https://doi.org/10.1016/j.neucom.2012.02.042
5. Cao L, Huang W, Sun F (2016) Building feature space of extreme learning machine with sparse denoising stacked-autoencoder. Neurocomputing 174:60–71. https://doi.org/10.1016/j.neucom.2015.02.096
6. Chen FL, Ou TY (2011) Sales forecasting system based on Gray extreme learning machine with Taguchi method in retail industry. Expert Syst Appl 38:1336–1345. https://doi.org/10.1016/j.eswa.2010.07.014
7. Chen W-R, Bin J, Lu H-M, Zhang Z-M, Liang Y-Z (2016) Calibration transfer via an extreme learning machine auto-encoder. Analyst 141:1973–1980. https://doi.org/10.1039/C5AN02243F
8. Chen Y-Y, Wang Z-B, Wang Z-B (2017) Novel variable selection method based on uninformative variable elimination and ridge extreme learning machine: CO gas concentration retrieval trial. Guang pu xue yu guang pu fen xi = Guang pu 37(1) 299–305. https://doi.org/10.3964/j.issn.1000-0593(2017)01-0299-07

9. Choi K, Toh K-A, Byun H (2012) Incremental face recognition for large-scale social network services. Pattern Recognition 45:2868–2883. https://doi.org/10.1016/j.patcog.2012.02.002
10. Creech G, Jiang F (2012) The application of extreme learning machines to the network intrusion detection problem. Kos, Greece, pp 1506–1511
11. Deng C, Huang G, Xu J, Tang J (2015) Extreme learning machines: new trends and applications. Sci China Inf Sci 58:1–16. https://doi.org/10.1007/s11432-014-5269-3
12. Deng W, Zheng Q, Chen L (2009) Regularized Extreme Learning Machine. 2009 IEEE Symposium on Computational Intelligence and Data Mining. IEEE, Nashville, TN, USA, pp 389–395
13. Dua D, Graff C (2020) UCI Machine Learning Repository. Irvine, CA: University of California, School, of Information and Computer Science.http://archive.ics.uci.edu/ml
14. Efron B, Hastie T, Johnstone I, Tibshirani R (2004) Least angle regression. Anna Stat 32:407–451. https://doi.org/10.1214/009053604000000067
15. Fan Q, Niu L, Kang Q (2020) Regression and Multiclass Classification Using Sparse Extreme Learning Machine via Smoothing Group L1/2 Regularizer. In: 2020 IEEE Access, pp 191482-191494. https://doi.org/10.1109/ACCESS.2020.3031647
16. Friedman J, Hastie T, Tibshirani R (2010) Regularization paths for generalized linear models via coordinate descent. J Stat Soft. https://doi.org/10.18637/jss.v033.i01
17. Haut JM, Liu Y, Paoletti ME, Xu X, Plaza J, Plaza A (2018) Evaluation of Different Regularization Methods for the Extreme Learning Machine Applied to Hyperspectral Images. IGARSS 2018–2018 IEEE International Geoscience and Remote Sensing Symposium. IEEE, Valencia, pp 3603–3606
18. Hoerl AE, Kennard RW (1970) Ridge regression: applications to nonorthogonal problems. Technometrics 12:69–82. https://doi.org/10.1080/00401706.1970.10488635
19. Huang G, Huang G-B, Song S, You K (2015) Trends in extreme learning machines: a review. Neural Netw 61:32–48. https://doi.org/10.1016/j.neunet.2014.10.001
20. Huang G, Song S, Gupta JND, Wu C (2014) Semi-supervised and unsupervised extreme learning machines. IEEE Trans Cybern 44:2405–2417. https://doi.org/10.1109/TCYB.2014.2307349
21. Huang G-B, Wang DH, Lan Y (2011) Extreme learning machines: a survey. Int J Mach Learn Cyber 2:107–122. https://doi.org/10.1007/s13042-011-0019-y
22. Huang G-B, Zhou H, Ding X, Zhang R (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst, Man, Cybern B 42:513–529. https://doi.org/10.1109/TSMCB.2011.2168604
23. Huang G-B, Zhu Q-Y, Siew C-K (2004) Extreme learning machine: a new learning scheme of feedforward neural networks. In: 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No.04CH37541). IEEE, Budapest, Hungary, pp 985–990
24. Huang G-B, Zhu Q-Y, Siew C-K (2006) Extreme learning machine: theory and applications. Neurocomputing 70:489–501. https://doi.org/10.1016/j.neucom.2005.12.126
25. Li G, Niu P (2013) An enhanced extreme learning machine based on ridge regression for regression. Neural Comput Appl 22:803–810. https://doi.org/10.1007/s00521-011-0771-7
26. Li R, Wang X, Lei L, Song Y (2019) L21-Norm Based Loss Function and Regularization Extreme Learning Machine. In: 2019 IEEE International Joint Conference on Neural Networks. IEEE Access, pp 6575-6586
27. Liu K (1993) A new class of blased estimate in linear regression. Commun Stat - Theor Methods 22:393–402. https://doi.org/10.1080/03610929308831027
28. Luo X, Chang X, Ban X (2016) Regression and classification using extreme learning machine based on L1-norm and L2-norm. Neurocomputing 174:179–186. https://doi.org/10.1016/j.neucom.2015.03.112
29. Luo X, Jiang C, Wang W, Xu Y, Wang J-H, Zhao W (2019) User behavior prediction in social networks using weighted extreme learning machine with distribution optimization. Future Gener Comput Syst 93:1023–1035. https://doi.org/10.1016/j.future.2018.04.085
30. Martínez-Martínez J-M, Escandell-Montero P, Soria-Olivas E, Martín-Guerrero J-D, Magdalena-Benedito R, Gómez-Sanchis J (2011) Regularized extreme learning machine for regression problems. Neurocomputing 74:3716–3721. https://doi.org/10.1016/j.neucom.2011.06.013
31. Miche Y, Sorjamaa A, Bas P, Simula O, Jutten C, Lendasse A (2010) OP-ELM: optimally pruned extreme learning machine. IEEE Trans Neural Netw 21:158–162. https://doi.org/10.1109/TNN.2009.2036259
32. Miche Y, van Heeswijk M, Bas P, Simula O, Lendasse A (2011) TROP-ELM: a double-regularized ELM using LARS and Tikhonov regularization. Neurocomputing 74:2413–2421. https://doi.org/10.1016/j.neucom.2010.12.042
33. Naik J, Satapathy P, Dash PK (2018) Short-term wind speed and wind power prediction using hybrid empirical mode decomposition and kernel ridge regression. Appl Soft Comput 70:1167–1188. https://doi.org/10.1016/j.asoc.2017.12.010
34. Nian R, He B, Lendasse A (2013) 3D object recognition based on a geometrical topology model and extreme learning machine. Neural Comput Appl 22:427–433. https://doi.org/10.1007/s00521-012-0892-7
35. Nomura M (1988) On the almost unbiased ridge regression estimator. Commun Stat - Simul Comput 17:729–743. https://doi.org/10.1080/03610918808812690
36. Poria S, Cambria E, Winterstein G, Huang G-B (2014) Sentic patterns: dependency-based rules for concept-level sentiment analysis. Know-Based Syst 69:45–63. https://doi.org/10.1016/j.knosys.2014.05.005
37. Preeti Bala R, Dagar A, Singh RP (2020) A novel online sequential extreme learning machine with L2,1-norm regularization for prediction problems. Appl Intell. https://doi.org/10.1007/s10489-020-01890-2
38. Raza N, Tahir M, Ali K (2020) k-Sparse extreme learning machine. Electron Lett 56:1277–1280. https://doi.org/10.1049/el.2020.1840
39. Rong H-J, Suresh S, Zhao G-S (2011) Stable indirect adaptive neural controller for a class of nonlinear system. Neurocomputing 74:2582–2590. https://doi.org/10.1016/j.neucom.2010.11.029
40. Rosset S, Zhu J (2007) Piecewise linear regularized solution paths. Ann Stat 35:1012–1030. https://doi.org/10.1214/009053606000001370
41. Sjöstrand K, Clemmensen LH, Larsen R, Einarsson G, Ersbøll B-K (2018) SpaSM: a MATLAB toolbox for sparse statistical modeling. J Stat Soft. https://doi.org/10.18637/jss.v084.i10
42. Shan P, Zhao Y, Sha X, Wang Q, Lv X, Peng S, Ying Y (2018) Interval LASSO regression based extreme learning machine for nonlinear multivariate calibration of near infrared spectroscopic datasets. Anal Methods 10:3011–3022. https://doi.org/10.1039/C8AY00466H
43. Shao Z, Er MJ (2016) Efficient leave-one-out cross-validation-based regularized extreme learning machine. Neurocomputing 194:260–270. https://doi.org/10.1016/j.neucom.2016.02.058
44. Suresh S, Venkatesh Babu R, Kim HJ (2009) No-reference image quality assessment using modified extreme learning machine classifier. Appl Soft Comput 9:541–552. https://doi.org/10.1016/j.asoc.2008.07.005
45. Tang J, Deng C, Huang G-B, Zhao B (2015) Compressed-domain ship detection on spaceborne optical image using deep neural network and extreme learning machine. IEEE Trans Geosci

Remote Sens 53:1174–1185. https://doi.org/10.1109/TGRS.2014.2335751
46. Tibshirani R (1996) Regression shrinkage and selection Via the Lasso. J R Stat Soc: Series B (Methodol) 58:267–288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x
47. Vapnik V-N (1995) The nature of statistical learning theory. Springer, New York
48. Várkonyi DT, Buza K (2019) Extreme learning machines with regularization for the classification of gene expression data. In: Petra B, Martin H, Tomáš H, Matúš P, Rudolf R (eds) Proceedings of the 19th Conference Information Technologies – Applications and Theory (ITAT 2019) Dóval, Czechoslovakia: CEUR Workshop Proceedings, pp 99–103
49. Wang H, Li G (2019) Extreme learning machine Cox model for high-dimensional survival analysis. Stat Med 38:2139–2156. https://doi.org/10.1002/sim.8090
50. Wang J-N, Jin J-L, Geng Y, Sun S-L, Xu H-L, Lu Y-H, Su Z-M (2013) An accurate and efficient method to predict the electronic excitation energies of BODIPY fluorescent dyes. J Comput Chem 34:566–575. https://doi.org/10.1002/jcc.23168
51. Wong WK, Guo ZX (2010) A hybrid intelligent model for medium-term sales forecasting in fashion retail supply chains using extreme learning machine and harmony search algorithm. Int J Prod Econ 128:614–624. https://doi.org/10.1016/j.ijpe.2010.07.008
52. Yan Z, Wang J (2014) Robust model predictive control of nonlinear systems with unmodeled dynamics and bounded uncertainties based on neural networks. IEEE Trans Neural Netw Learning Syst 25:457–469. https://doi.org/10.1109/TNNLS.2013.2275948
53. Yan C, Li Y, Liu W, Li M, Chen J, Wang L (2020) An artificial bee colony-based kernel ridge regression for automobile insurance fraud identification. Neurocomputing 393:115–125. https://doi.org/10.1016/j.neucom.2017.12.072
54. Yildirim H, Özkale MR (2019) The performance of ELM based ridge regression via the regularization parameters. Expert Syst Appl 134:225–233. https://doi.org/10.1016/j.eswa.2019.05.039
55. Yildirim H, Özkale MR (2020) An enhanced extreme learning machine based on Liu regression. Neural Process Lett 52:421–442. https://doi.org/10.1007/s11063-020-10263-2
56. You Z-H, Lei Y-K, Zhu L, Xia J, Wang B (2013) Prediction of protein-protein interactions from amino acid sequences with ensemble extreme learning machines and principal component analysis. BMC Bioinform 14:S10. https://doi.org/10.1186/1471-2105-14-S8-S10
57. Yu Y, Kang S, Qiu H (2018) A new network intrusion detection algorithm: DA-ROS-ELM: intrusion detection algorithm DA-ROS-ELM. IEEJ Trans Elec Electron Eng 13:602–612. https://doi.org/10.1002/tee.22606
58. Zhao J, Wang Z, Park D-S (2012) Online sequential extreme learning machine with forgetting mechanism. Neurocomputing 87:79–89. https://doi.org/10.1016/j.neucom.2012.02.003
59. Zheng W, Qian Y, Lu H (2013) Text categorization based on regularization extreme learning machine. Neural Comput Appl 22:447–456. https://doi.org/10.1007/s00521-011-0808-y
60. Zou H, Hastie T (2005) Regularization and variable selection via the elastic net. J Royal Stat Soc B 67:301–320. https://doi.org/10.1111/j.1467-9868.2005.00503.x