ARTICLE IN PRESS

Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

SMOTE based class-specific extreme learning machine for imbalanced learning☆

Bhagat Singh Raghuwanshi, Sanyam Shukla *

Department of Computer Science and Engineering, Maulana Azad National Institute of Technology (MANIT), Bhopal, Madhya Pradesh, 462003, India

A R T I C L E I N F O

Article history:
Received 25 December 2018
Received in revised form 20 June 2019
Accepted 24 June 2019
Available online xxxx

Keywords:
SMOTE
Imbalanced learning
Class-specific extreme learning machine
Class-specific regularization
Classification

A B S T R A C T

Imbalanced learning is one of the substantial challenging problems in the field of data mining. The datasets that have skewed class distribution pose hindrance to conventional learning methods. Conventional learning methods give the same importance to all the samples. This leads to biased accuracy, which favors the majority classes. Several classifiers have been designed to tackle the class imbalance problems. Weighted kernel-based SMOTE (WKSMOTE) is a recently proposed method, which employs the minority oversampling in kernel space to tackle the class imbalance problem. Motivated by WKSMOTE, this work proposes a novel SMOTE based class-specific extreme learning machine (SMOTE-CSELM), a variant of class-specific extreme learning machine (CS-ELM), which exploits the benefit of both the minority oversampling and the class-specific regularization. For minority oversampling, this work uses synthetic minority oversampling technique (SMOTE). It increases the significance of the minority class samples for determining the decision region of the classifiers. The proposed method has comparable computational complexity than the weighted extreme learning machine (WELM) for imbalanced learning. The extensive experimental results evaluated on the real-world benchmark datasets demonstrate the efficacy of our proposed method.

© 2019 Elsevier B.V. All rights reserved.

1. Introduction

The imbalanced classification problems have been widely reported in many real-world applications such as medical diagnoses [1], detection of oil spills [2], software defect prediction [3] and cancer malignancy grading [4]. The problem associated with the class imbalance learning is that the standard methods usually misclassify most of the positive class samples as the negative class samples. For problems like medical diagnosis, the detection of the minority class samples is more important. The imbalanced nature of such real-world application is one of the current challenges for the machine learning researchers. Due to this, imbalanced classification problem is currently drawing a lot of attention from the pattern recognition and the machine learning communities [5–7]. Almost all the classification problems in the real-world do not have uniform class distribution. The classes whose number of samples are below the average number of samples per class are termed as the minority classes. The classes whose number of samples are above the average number of samples per class are termed as the majority classes. Often, the positive (minority) class misclassification is much more expensive compared to the negative (majority) class. It is also harder for the classifier to learn the minority class, as it has fewer number of samples. The most of the conventional learning methods have been developed to work on the balanced training dataset. These methods usually focus on improving the overall accuracy which consequently deteriorate the detection rate of the positive class.

During the last decades, various methods have been developed to handle the class imbalance problem [5,8]. The methods available for the imbalanced classification [8] can be broadly categorized as the data level methods, the algorithmic level methods and the cost-sensitive methods. The data level methods like oversampling and undersampling [5,9] alter the data space to reduce the impact of the class imbalance. The undersampling method randomly selects a fraction of data from the majority class samples and balances the data distribution at the cost of information loss. For example, EasyEnsemble and BalanceCascade [9] algorithms employ undersampling for balancing the dataset. The oversampling method randomly duplicates the samples of the minority classes in order to enhance its cardinality. This may lead to over-fitting. The informed oversampling approach like synthetic minority over-sampling techniques (SMOTE) [10] generates synthetic minority class samples to balance the class distribution. It has received a lot of admiration and has extensive

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.06.022.

* Corresponding author.
E-mail addresses: bhagat.mnit@gmail.com (B.S. Raghuwanshi), sanyamshukla@gmail.com (S. Shukla).

https://doi.org/10.1016/j.knosys.2019.06.022
0950-7051/© 2019 Elsevier B.V. All rights reserved.

ARTICLE IN PRESS

range of practical applications. Many variants of SMOTE have been developed, such as adaptive synthetic sampling approach (AdaSyn) [11], borderline-SMOTE [12], majority weighted minority oversampling technique [13] and weighted kernel based SMOTE [14].

The algorithmic level methods [15–17] directly modify the classifier design to address the imbalanced learning. The cost-sensitive methods [18] like WELM [19] and weighted support vector machine (WSVM) [20] assign more penalty for incorrectly classifying the minority class samples with respect to the majority class samples. To find the optimal solution, they minimize the weighted least squares error along with regularization.

Extreme learning machine (ELM) [21,22] has become popular among researchers all over the world. ELM [21] theories show that the hidden layer parameters in single-hidden layer feedforward networks (SLFNs) need not be tuned and can be generated randomly, independent of the training dataset, and the output weights are computed in a single step by employing the least-squares estimate solution. ELM has universal approximation capability provided that the hidden neurons have nonlinear piecewise continuous activation function. Due to the random generation of the hidden layer weights and the biases in ELM, its training speed is much faster compared to the traditional back-propagation (BP) algorithm. Traditional ELM [23,24] has been designed to work on balanced datasets, it does not take into consideration the imbalanced learning problems. Several variants of ELM such as WELM [19], Boosting WELM (BWELM) [25], regularized weighted circular complex valued ELM [26], class-specific cost regulation ELM (CCR-ELM) [27], CS-ELM [28], class-specific kernelized ELM (CSKELM) [29], generalized CSKELM (GCSKELM) [30], UnderBagging based kernelized ELM (UBKELM) [31], UnderBagging based reduced kernelized WELM (UBRKWELM) [32] and class-specific cost-sensitive boosting WELM [33] have been designed to address the imbalanced learning effectively. The kernelized ELM [34] has usually good performance than the sigmoid node based ELM. However, it does not work well for datasets with huge number of instances as it to compute inverse of size $N \times N$, where N represents the number of training samples.

WELM [19] is a cost-sensitive method designed to handles the class imbalance problems. WELM uses $N \times N$ diagonal weight matrix. The weight associated with the samples is set to a larger value if the sample comes from the minority class. It has been shown in [35] that, the Lagrangian multiplier, $\boldsymbol{\alpha}$ corresponding to the minority class sample must be higher in weight than the $\boldsymbol{\alpha}$ of the majority class samples. CS-ELM strengthens the impact of the minority class sample using the aforementioned approach for determining the decision boundary.

This work proposes a novel classifier, SMOTE-CSELM a variant of CS-ELM, which uses minority oversampling to balance the class distribution. The SMOTE-CSELM employs class-specific regularization parameter whose value is decided by employing the class proportion.

The main contributions of this work are highlighted below.

(1) SMOTE-CSELM is a variant of CS-ELM. It has been shown in [28] that CS-ELM performs better than WELM and also has less computational complexity. The proposed work also has comparable computational complexity in contrast with WELM for imbalanced learning.
(2) The proposed SMOTE-CSELM, exploits the benefit of both the minority oversampling and the class-specific regularization. For minority oversampling, this work uses SMOTE. It increases the significance of the minority class samples for determining the decision region of the algorithms.

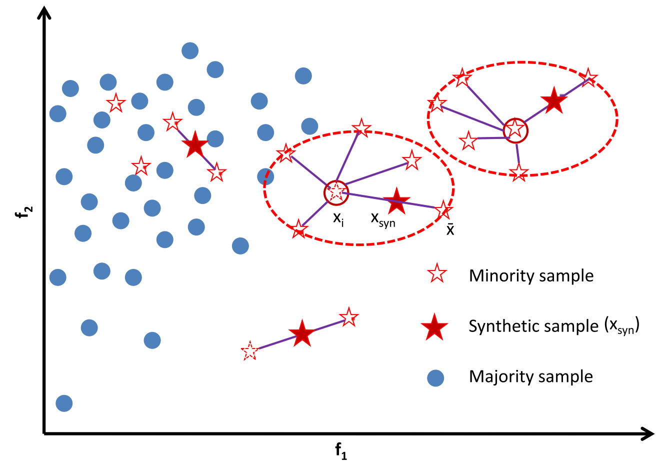


Fig. 1. SMOTE linearly interpolates a randomly chosen minority class sample and one of its k = 5 nearest neighbors.

(3) This work performs extensive experiments to compare SMOTE-CSELM with the other state-of-art methods. The statistical significance test is also performed to check whether the new algorithm significantly outperforms the other algorithms in terms of G-mean, Recall and AUC.

The remaining of the paper is structured as below. The Section 2 reviews the preliminaries in detail. The Section 3 presents the proposed method. The experimental results and comparisons are described in Section 4. The final section concludes the paper along with the future work.

2. Preliminaries

2.1. Synthetic minority oversampling technique (SMOTE)

SMOTE algorithm proposed by Chawla et al. [10] is one of the most widely used oversampling methods. SMOTE algorithm inserts a new synthetic minority class sample on the line that connects a randomly chosen minority class sample and one of its k-nearest neighbors [36] belonging to the minority class samples, as shown in Fig. 1. SMOTE uses an iterative search and selection method. To generate new artificial minority class samples, a threshold number of samples among the k nearest neighbors are selected. The value of the threshold depends on the number of synthetic minority samples to be generated. The process continues until the required number of synthetic minority class samples have been generated. SMOTE method synthetically generates new minority class samples in the feature space rather than in the data space to balance the class distribution. This widens the decision region of the minority class. In SMOTE, a new synthetic minority class sample is generated, which lies on the line segment between x_i and $\bar{x}$ here, $x_i, \bar{x} \in N_{min}$ can be described as

$$x_{syn} = x_i + (\bar{x} - x_i). \times rand(0, 1). \quad (1)$$

Here, x_i is the minority class sample which is to be oversampled; $\bar{x}$ is another minority sample, which is usually selected from the N_{min} samples near x_i; the symbol ". × " represents element-wise multiplication; $rand(0, 1)$ indicates a random number within the interval (0, 1).

2.2. Extreme learning machine

The ELM is a single-hidden layer feedforward network, proposed by [21,22], whose weights from the inputs to the hidden neurons are randomly selected, The weights from the hidden neurons to the output are analytically computed. The ELM offers benefits like fast learning speed, ease of implementation, and less human intervention compared to the standard neural networks.

Given N samples $[\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_N}]$ and their corresponding targets $[\mathbf{t_1}, \mathbf{t_2}, \ldots, \mathbf{t_N}]$. Here, the input vector, $\mathbf{x_i} = [x_{i1}, x_{i2}, \ldots, x_{in}]^T \in R^n$ and its desired output, $\mathbf{t_i} = [t_{i1}, t_{i2}, \ldots, t_{im}]^T \in R^m$, here, T represents the transpose. Here, n is the input feature dimensional space, L and m are the dimensions of the hidden layer and the output layer respectively. The dimension of output layer is same as the dimension of classes. The input weight matrix is represented by $\mathbf{U} = [\mathbf{u_1}, \mathbf{u_2}, ..\mathbf{u_j}, ..\mathbf{u_L}]^T \in R^{L\times n}$ and the bias of the hidden neurons, $\mathbf{b} = \left[b_1, b_2, ..b_j, ..b_L\right]^T \in R^L$, here, $\mathbf{u_j} = [u_{j1}, u_{j2}, ..u_{jn}]$ are the weights connecting the jth hidden neuron to the input neurons, b_j is the bias of the jth hidden neuron. These weights remain unaltered amid the training phase. The output of the hidden layer for the ith sample is given as

$$\phi(\mathbf{x_i}) = G(\mathbf{U}\mathbf{x_i} + b). \tag{2}$$

Here, $G(.)$ is the hidden layer activation function. The output of the hidden layer for all the training samples is given by $\mathbf{\Phi}$ can be defined as

$$\mathbf{\Phi} = \begin{bmatrix} \phi_1(x_1) & \phi_2(x_1) & . & \phi_L(x_1) \\ \phi_1(x_2) & \phi_2(x_2) & . & \phi_L(x_2) \\ . & . & . & . \\ \phi_1(x_N) & \phi_2(x_N) & . & \phi_L(x_N) \end{bmatrix}_{N\times L.} \tag{3}$$

The output weights, $\boldsymbol{\beta}$ are learned from the training samples by solving the following objective function.

$$\text{Minimize:} \quad \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\frac{1}{2}\sum_{i=1}^{N}\left\|\xi_i\right\|^2 \tag{4}$$

Subject to: $\phi(\mathbf{x_i})\boldsymbol{\beta} = \mathbf{t_i^T} - \xi_{\mathbf{i}}^{\mathbf{T}}, \quad i = 1, \ldots, N.$

Here, $\boldsymbol{\xi}_\mathbf{i}$ is the training error vector with respect to the ith training sample, C is the regularization factor, $\|\boldsymbol{\beta}\|^2$ is the parameter of the separating hyperplane and also known as the structural risk, $\|\boldsymbol{\xi}\|^2$ is the sum error square, also known as the empirical risk. The structural risk increases the margin for separating the classes [37]. Here, $\boldsymbol{\beta}$ is the output weight vector between the hidden and the output layer. The objective function of ELM defined in (4) is obtained by [22], which can be expressed as follows.

$$\boldsymbol{\beta} = \begin{cases} \mathbf{\Phi^T}\left(\dfrac{I}{C} + \mathbf{\Phi}\mathbf{\Phi^T}\right)^{-1}\mathbf{T}, & \text{if } N < L \\ \left(\dfrac{I}{C} + \mathbf{\Phi^T}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi^T}\mathbf{T}, & \text{if } N > L \end{cases} \tag{5}$$

Here, I is the identity matrix of appropriate dimension. The ELM output is determined as follows.

$$\mathbf{f}(\mathbf{x}) = \begin{cases} sign\ \phi(\mathbf{x})\mathbf{\Phi^T}\left(\dfrac{I}{C} + \mathbf{\Phi}\mathbf{\Phi^T}\right)^{-1}\mathbf{T}, & \text{if } N < L \\ sign\ \phi(\mathbf{x})\left(\dfrac{I}{C} + \mathbf{\Phi^T}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi^T}\mathbf{T}, & \text{if } N > L \end{cases} \tag{6}$$

where, $\mathbf{f}(\mathbf{x}) = [f_k(\mathbf{x}), \ldots, f_m(\mathbf{x})]$ is the output vector. The predicted label of $\mathbf{x}$ is defined as

$$label(\mathbf{x}) = \underset{k}{\operatorname{argmax}} f_k(\mathbf{x}), \quad k = 1, \ldots, m. \tag{7}$$

2.3. Weighted extreme learning machine

WELM [19] was proposed for tackling the imbalanced classification problems efficiently. In WELM, the weights associated with the minority class samples are relatively larger compared to the majority class samples. Thus, the impact of the minority class is strengthened whereas, the relative impact of the majority class is diminished. The two weighting schemes empirically employed to determine the weight matrix, $\mathbf{W}$ which utilize the class distribution are given below.

First:

$$W_{ii} = \frac{1}{s_k}. \tag{8}$$

Here, s_k represents the total number of samples belonging to the kth class.

Second:

$$W_{ii} = \begin{cases} \frac{0.618}{s_k}, & if\ (s_k > s_{avg}) \\ \frac{1}{s_k}, & if\ (s_k <= s_{avg}). \end{cases} \tag{9}$$

Here, s_{avg} is the average number of samples per class. It has been stated [19] that the value 0.618 is the golden standard that represents perfection in nature.

The optimization problem of WELM [19] is formulated as follows.

$$\text{Minimize:} \quad \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C\sum_{i=1}^{N}W_{ii}\left\|\xi_i\right\|^2 \tag{10}$$

Subject to: $\phi(\mathbf{x_i})\boldsymbol{\beta} = \mathbf{t_i^T} - \boldsymbol{\xi}_\mathbf{i}^\mathbf{T}, \quad i = 1, \ldots, N.$

Here, a diagonal matrix $\mathbf{W} = diag\ (W_{ii})$ is associated to allocate weight to each training sample $\mathbf{x_i}$. Referring to the Karush–Kuhn–Tucker (KKT) conditions, the solution of (10) is delineated below.

$$\boldsymbol{\beta} = \begin{cases} \mathbf{\Phi^T}\left(\dfrac{I}{C} + \mathbf{W}\mathbf{\Phi}\mathbf{\Phi^T}\right)^{-1}\mathbf{W}\mathbf{T}, & \text{if } N < L \\ \left(\dfrac{I}{C} + \mathbf{\Phi^T}\mathbf{W}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi^T}\mathbf{W}\mathbf{T}, & \text{if } N > L. \end{cases} \tag{11}$$

2.4. Class-specific extreme learning machine

We have recently proposed class-specific extreme learning machine [28] for addressing the imbalanced classification problems more effectively. CS-ELM employs class-specific regularization parameters, which are determined by using class proportion and the regularization parameter, C. The optimization problem of CS-ELM is given below.

$$\begin{aligned} \text{Minimize:} \quad & \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}\frac{N_{min}\times C}{N}\left\|\boldsymbol{\xi}_\mathbf{maj}\right\|^2 \\ & + \frac{1}{2}\frac{N_{maj}\times C}{N}\left\|\boldsymbol{\xi}_\mathbf{min}\right\|^2 \\ \text{Subject to:} \quad & \phi(\mathbf{x_i^{maj}})\boldsymbol{\beta} = \mathbf{t_i^{maj}} - \boldsymbol{\xi}_\mathbf{i}^\mathbf{maj}, \quad i = 1, \ldots, N_{maj} \\ & \phi(\mathbf{x_i^{min}})\boldsymbol{\beta} = \mathbf{t_i^{min}} - \boldsymbol{\xi}_\mathbf{i}^\mathbf{min}, \ i = 1, \ldots, N_{min}. \end{aligned} \tag{12}$$

Here,

$$\left\|\boldsymbol{\xi}_\mathbf{min}\right\|^2 = \sum_{i=1|\mathbf{t_i}=+1}^{N_{min}}\left\|\boldsymbol{\xi}_\mathbf{i}\right\|^2 \ \& \ \left\|\boldsymbol{\xi}_\mathbf{maj}\right\|^2 = \sum_{i=1|\mathbf{t_i}=-1}^{N_{maj}}\left\|\boldsymbol{\xi}_\mathbf{i}\right\|^2. \tag{13}$$

Here, N_{min} and N_{maj} represent the number of samples belonging to the minority and the majority class respectively. The regularization parameter, C of CS-ELM is selected over a range of $(2^{-18}, 2^{-16}...2^{48}, 2^{50})$ to get the best results. Based on the KKT theorem, training CS-ELM is equivalent to solving the following Lagrangian function.

$$\begin{aligned} \mathcal{L}_{D_{\text{CS-ELM}}} &= \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C^{maj}\left\|\boldsymbol{\xi}_\mathbf{maj}\right\|^2 + \frac{1}{2}C^{min}\left\|\boldsymbol{\xi}_\mathbf{min}\right\|^2 \\ &- \begin{bmatrix}\alpha^{maj} & \alpha^{min}\end{bmatrix}\left(\begin{bmatrix}\mathbf{\Phi_{maj}} \\ \mathbf{\Phi_{min}}\end{bmatrix}\boldsymbol{\beta} - \begin{bmatrix}\mathbf{T_{maj}} \\ \mathbf{T_{min}}\end{bmatrix} + \begin{bmatrix}\boldsymbol{\xi}_\mathbf{maj} \\ \boldsymbol{\xi}_\mathbf{min}\end{bmatrix}\right). \end{aligned} \tag{14}$$

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

Here, $\mathcal{L}_{D_{\text{CS-ELM}}}$ represents the Lagrangian function. Here, C^{maj} and C^{min} are the class-specific regularization parameters which are defined as follows.

$$C^{maj} = \frac{N_{min} \times C}{(N_{maj} + N_{min})}, \quad C^{min} = \frac{N_{maj} \times C}{(N_{maj} + N_{min})}, \tag{15}$$

$$\mathbf{t_i} = [t_{i,1}, t_{i,2}, \ldots, t_{i,m}]^T, \quad \mathbf{T} = [\mathbf{t_1}, \ldots, \mathbf{t_i}, \ldots, \mathbf{t_N}]^T, \tag{16}$$

$$\begin{aligned}\phi(\mathbf{x_i}) &= [\phi_1(x_i), \phi_2(x_i), \ldots, \phi_L(x_i)]^T, \\ \boldsymbol{\Phi} &= [\phi(\mathbf{x_1}), \ldots, \phi(\mathbf{x_i}), \ldots, \phi(\mathbf{x_N})]^T,\end{aligned} \tag{17}$$

$$\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{\xi}_{\mathbf{maj}} \\ \boldsymbol{\xi}_{\mathbf{min}} \end{bmatrix}, \boldsymbol{\Phi} = \begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{maj}} \\ \boldsymbol{\Phi}_{\mathbf{min}} \end{bmatrix}, \boldsymbol{\alpha} = \begin{bmatrix} \boldsymbol{\alpha}^{\boldsymbol{maj}} \\ \boldsymbol{\alpha}^{\boldsymbol{min}} \end{bmatrix} \text{ and } \mathbf{T} = \begin{bmatrix} \mathbf{T_{maj}} \\ \mathbf{T_{min}} \end{bmatrix}, \tag{18}$$

$$\begin{aligned}\boldsymbol{\Phi}_{\mathbf{maj}} &= \left[\phi(\mathbf{x_1^{maj}}), \ldots, \phi(\mathbf{x^{maj}_{N_{maj}}})\right]^T, \\ \boldsymbol{\Phi}_{\mathbf{min}} &= \left[\phi(\mathbf{x_1^{min}}), \ldots, \phi(\mathbf{x^{min}_{N_{min}}})\right]^T,\end{aligned} \tag{19}$$

$$\boldsymbol{\alpha}^{\boldsymbol{maj}} = \left[\boldsymbol{\alpha}_{\mathbf{1}}^{\boldsymbol{maj}}, \ldots, \boldsymbol{\alpha}_{\boldsymbol{N_{maj}}}^{\boldsymbol{maj}}\right]^T, \quad \boldsymbol{\alpha}^{\boldsymbol{min}} = \left[\boldsymbol{\alpha}_{\mathbf{1}}^{\boldsymbol{min}}, \ldots, \boldsymbol{\alpha}_{\boldsymbol{N_{min}}}^{\boldsymbol{min}}\right]^T, \tag{20}$$

$$\mathbf{T_{maj}} = \left[\mathbf{t_1^{maj}}, \ldots, \mathbf{t^{maj}_{N_{maj}}}\right]^T, \quad \mathbf{T_{min}} = \left[\mathbf{t_1^{min}}, \ldots, \mathbf{t^{min}_{N_{min}}}\right]^T. \tag{21}$$

Here, $\boldsymbol{\Phi}_{\mathbf{maj}}$ and $\boldsymbol{\Phi}_{\mathbf{min}}$ represent the output vector of the hidden layer for the majority and the minority class samples respectively. The vectors, $\boldsymbol{\alpha}^{\boldsymbol{maj}}$ and $\boldsymbol{\alpha}^{\boldsymbol{min}}$ represent the Lagrangian coefficient for the equality constraints corresponding to the majority and the minority class samples respectively. The vectors, $\mathbf{T_{maj}}$ and $\mathbf{T_{min}}$ represent the target of the majority and the minority class samples respectively.

$$\boldsymbol{\alpha} = [\boldsymbol{\alpha_1}, \ldots, \boldsymbol{\alpha_i}, \ldots, \boldsymbol{\alpha_N}]^T, \quad \boldsymbol{\alpha_i} = [\alpha_{i,1}, \alpha_{i,2}, \ldots, \alpha_{i,m}]^T. \tag{22}$$

The parameter, $\boldsymbol{\alpha_i}$ is the Lagrangian coefficient for the equality constraint corresponding to the sample $\mathbf{x_i}$.

The solution of CS-ELM given by Eq. (10) is determined in [28], which is reproduced in Box I.

3. Proposed method

3.1. SMOTE based class-specific extreme learning machine (SMOTE-CSELM)

This work proposes synthetic minority over-sampling techniques based class-specific extreme learning machine to tackle the imbalanced classification problems more effectively. Using SMOTE method, the number of samples corresponding to the minority class are increased by creating new synthetic samples. This oversampled dataset is used to train the proposed SMOTE-CSELM classifier. It increases the significance of the minority class samples for determining the decision region. The proposed method does not require assignment of weights to the training sample. The SMOTE-CSELM employs class-specific regularization parameter whose value is decided by the class proportion and the regularization parameter, C. The Algorithm 1 shows the pseudo-code of the proposed SMOTE-CSELM algorithm. The optimization function of SMOTE-CSELM is formulated as follows:

$$\text{Minimize: } \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C^{maj}\left\|\boldsymbol{\xi}_{\mathbf{maj}}\right\|^2 + \frac{1}{2}C^{min}\left\|\boldsymbol{\xi}_{\mathbf{min}}\right\|^2 + \frac{1}{2}C^{syn}\left\|\boldsymbol{\xi}_{\mathbf{syn}}\right\|^2$$

$$\begin{aligned}\text{Subject to: } \phi(\mathbf{x_i^{maj}})\boldsymbol{\beta} &= \mathbf{t_i^{maj}}^T - \boldsymbol{\xi}_{\mathbf{i}}^{\mathbf{maj}T}, \quad i = 1, \ldots, N_{maj} \\ \phi(\mathbf{x_i^{min}})\boldsymbol{\beta} &= \mathbf{t_i^{min}}^T - \boldsymbol{\xi}_{\mathbf{i}}^{\mathbf{min}T}, \quad i = 1, \ldots, N_{min} \\ \phi(\mathbf{x_i^{syn}})\boldsymbol{\beta} &= \mathbf{t_i^{syn}}^T - \boldsymbol{\xi}_{\mathbf{i}}^{\mathbf{syn}T}, \quad i = 1, \ldots, N_{syn}.\end{aligned} \tag{24}$$

Here,

$$\left\|\boldsymbol{\xi}_{\mathbf{min}}\right\|^2 = \sum_{i=1|\mathbf{t_i}=+1}^{N_{min}} \left\|\boldsymbol{\xi}_{\mathbf{i}}\right\|^2, \ \left\|\boldsymbol{\xi}_{\mathbf{syn}}\right\|^2 = \sum_{i=1|\mathbf{t_i}=+1}^{N_{syn}} \left\|\boldsymbol{\xi}_{\mathbf{i}}\right\|^2 \ \& \ \left\|\boldsymbol{\xi}_{\mathbf{maj}}\right\|^2 = \sum_{i=1|\mathbf{t_i}=-1}^{N_{maj}} \left\|\boldsymbol{\xi}_{\mathbf{i}}\right\|^2. \tag{25}$$

Here, N_{maj}, N_{min} and N_{syn} represent the number of the samples belonging to the majority, minority and the synthetic dataset respectively. The parameters, C^{maj}, C^{maj} and C^{syn} represent the class-specific regularization parameters corresponding to the majority, minority and the synthetic dataset respectively, which representing the trade-off between the minimization of training errors and the maximization of generality ability. Note that all the data points in N_{syn} are assigned the class label of the minority samples. Class-specific regularization parameter setting is delineated below.

$$\begin{aligned}C^{maj} &= \frac{\left(\widehat{N} - N_{maj}\right) \times C}{\widehat{N}}, \quad C^{min} = \frac{\left(\widehat{N} - N_{min}\right) \times C}{\widehat{N}}, \\ C^{syn} &= \frac{\left(\widehat{N} - (N_{min} + N_{syn})\right) \times C}{\widehat{N}}.\end{aligned} \tag{26}$$

Here, $\widehat{N} = N + N_{syn}$, $N = N_{min} + N_{maj}$. Here, the regularization parameter, C of SMOTE-CSELM is selected over the range $(2^{-18}, 2^{-16} \ldots 2^{48}, 2^{50})$ to obtain the optimal performance. The Lagrangian function of (24) can be written as follows:

$$\begin{aligned}\mathcal{L}_{D_{\text{SMOTE-CSELM}}} = {} & \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \frac{1}{2}C^{maj}\left\|\boldsymbol{\xi}_{\mathbf{maj}}\right\|^2 + \frac{1}{2}C^{min}\left\|\boldsymbol{\xi}_{\mathbf{min}}\right\|^2 \\ & + \frac{1}{2}C^{syn}\left\|\boldsymbol{\xi}_{\mathbf{syn}}\right\|^2 \\ & - \left[\boldsymbol{\alpha}^{\mathbf{maj}} \quad \boldsymbol{\alpha}^{\boldsymbol{min}} \quad \boldsymbol{\alpha}^{\mathbf{syn}}\right] \\ & \left(\begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{maj}} \\ \boldsymbol{\Phi}_{\mathbf{min}} \\ \boldsymbol{\Phi}_{\mathbf{syn}} \end{bmatrix}\boldsymbol{\beta} - \begin{bmatrix} \mathbf{T_{maj}} \\ \mathbf{T_{min}} \\ \mathbf{T_{syn}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\xi}_{\mathbf{maj}} \\ \boldsymbol{\xi}_{\mathbf{min}} \\ \boldsymbol{\xi}_{\mathbf{syn}} \end{bmatrix}\right).\end{aligned} \tag{27}$$

Here, $\mathcal{L}_{D_{\text{SMOTE-CSELM}}}$ is the Lagrangian function. Here, $\boldsymbol{\Phi}_{\mathbf{maj}}$, $\boldsymbol{\Phi}_{\mathbf{min}}$ and $\boldsymbol{\Phi}_{\mathbf{syn}}$ represent the output vector of the hidden layer for the majority, minority and synthetic samples respectively. The Lagrangian coefficient, $\boldsymbol{\alpha}^{\boldsymbol{maj}}$, $\boldsymbol{\alpha}^{\boldsymbol{min}}$ and $\boldsymbol{\alpha}^{\boldsymbol{syn}}$ are the equality constraints corresponding to the majority, minority and synthetic samples respectively. The vectors, $\mathbf{T_{maj}}$, $\mathbf{T_{min}}$ and $\mathbf{T_{syn}}$ are the output vector of the majority, minority and synthetic samples respectively. Based on the KKT theorem, by taking the partial derivatives of the aforementioned equation with respect to variables $(\boldsymbol{\beta}, \boldsymbol{\xi}_{\mathbf{maj}}, \boldsymbol{\xi}_{\mathbf{min}}, \boldsymbol{\xi}_{\mathbf{syn}}, \boldsymbol{\alpha}^{\boldsymbol{maj}}, \boldsymbol{\alpha}^{\boldsymbol{min}}, \boldsymbol{\alpha}^{\boldsymbol{syn}})$ and equating them to zero, the following conditions are obtained.

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \boldsymbol{\beta}} = 0 \Rightarrow \boldsymbol{\beta} = \left(\boldsymbol{\Phi}_{\mathbf{maj}}{}^T\boldsymbol{\alpha}^{\boldsymbol{maj}} + \boldsymbol{\Phi}_{\mathbf{min}}{}^T\boldsymbol{\alpha}^{\boldsymbol{min}} + \boldsymbol{\Phi}_{\mathbf{syn}}{}^T\boldsymbol{\alpha}^{\boldsymbol{syn}}\right) \tag{28}$$

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \boldsymbol{\xi}_{\mathbf{maj}}} = 0 \Rightarrow \left(C^{maj}\boldsymbol{\xi}_{\mathbf{maj}} - \boldsymbol{\alpha}^{\boldsymbol{maj}}\right) = 0 \tag{29}$$

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \boldsymbol{\xi}_{\mathbf{min}}} = 0 \Rightarrow \left(C^{min}\boldsymbol{\xi}_{\mathbf{min}} - \boldsymbol{\alpha}^{\boldsymbol{min}}\right) = 0 \tag{30}$$

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \boldsymbol{\xi}_{\boldsymbol{syn}}} = 0 \Rightarrow \left(C^{syn}\boldsymbol{\xi}_{\mathbf{syn}} - \boldsymbol{\alpha}^{\boldsymbol{syn}}\right) = 0 \tag{31}$$

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \boldsymbol{\alpha}^{\boldsymbol{maj}}} = 0 \Rightarrow \left(\boldsymbol{\Phi}_{\mathbf{maj}}\boldsymbol{\beta} - \mathbf{T_{maj}} + \boldsymbol{\xi}_{\mathbf{maj}}\right) = 0 \tag{32}$$

ARTICLE IN PRESS

$$\beta=\begin{cases}\begin{bmatrix}\mathbf{\Phi_{maj}^T} & \mathbf{\Phi_{min}^T}\end{bmatrix}\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\begin{bmatrix}\left(1+\frac{C^{maj}}{C^{min}}\right)\mathbf{\Phi_{maj}}\\ \left(1+\frac{C^{min}}{C^{maj}}\right)\mathbf{\Phi_{min}}\end{bmatrix}\begin{bmatrix}\mathbf{\Phi_{maj}^T} & \mathbf{\Phi_{min}^T}\end{bmatrix}\right)^{-1} \\ \begin{bmatrix}\left(1+\frac{C^{maj}}{C^{min}}\right)\mathbf{T_{maj}}\\ \left(1+\frac{C^{min}}{C^{maj}}\right)\mathbf{T_{min}}\end{bmatrix}, \quad if\ N<L \\ \left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\left(1+\frac{C^{maj}}{C^{min}}\right)\mathbf{\Phi_{maj}}^{\mathbf{T}}\mathbf{\Phi_{maj}}+\left(1+\frac{C^{min}}{C^{maj}}\right)\mathbf{\Phi_{min}}^{\mathbf{T}}\mathbf{\Phi_{min}}\right)^{-1} \\ \left(\left(1+\frac{C^{maj}}{C^{min}}\right)\mathbf{\Phi_{maj}}^{\mathbf{T}}\mathbf{T}_{maj}+\left(1+\frac{C^{min}}{C^{maj}}\right)\mathbf{\Phi_{min}}^{\mathbf{T}}\mathbf{T_{min}}\right), \quad if\ N>L.\end{cases} \tag{23}$$

Box I.

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \alpha^{min}}=0 \Rightarrow \left(\mathbf{\Phi_{min}}\beta-\mathbf{T_{min}}+\xi_{\mathbf{min}}\right)=0 \tag{33}$$

$$\frac{\partial \mathcal{L}_{D_{\text{SMOTE-CSELM}}}}{\partial \alpha^{syn}}=0 \Rightarrow \left(\mathbf{\Phi_{syn}}\beta-\mathbf{T_{syn}}+\xi_{\mathbf{syn}}\right)=0. \tag{34}$$

The aforementioned Eqs. (28)–(34) can be equivalently written as follows.

$$\beta=\left(\mathbf{\Phi_{maj}}^T\alpha^{\mathbf{maj}}+\mathbf{\Phi_{min}}^T\alpha^{\mathbf{min}}+\mathbf{\Phi_{syn}}^T\alpha^{\mathbf{syn}}\right) \tag{35}$$

$$\frac{\alpha^{maj}}{C^{maj}}=\xi_{\mathbf{maj}} \tag{36}$$

$$\frac{\alpha^{min}}{C^{min}}=\xi_{\mathbf{min}} \tag{37}$$

$$\frac{\alpha^{syn}}{C^{syn}}=\xi_{\mathbf{syn}} \tag{38}$$

$$\left(\mathbf{\Phi_{maj}}\beta+\xi_{\mathbf{maj}}\right)=\mathbf{T_{maj}} \tag{39}$$

$$\left(\mathbf{\Phi_{min}}\beta+\xi_{\mathbf{min}}\right)=\mathbf{T_{min}} \tag{40}$$

$$\left(\mathbf{\Phi_{syn}}\beta+\xi_{\mathbf{syn}}\right)=\mathbf{T_{syn}}. \tag{41}$$

A positive value $\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)$ can be added to the diagonal of the output weight, β for the stable and better generalization performance [38]. Then, multiplying $\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)$ to both the sides of (35), we have

$$\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)\beta = \left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)\left(\mathbf{\Phi^T_{maj}}\alpha^{\mathbf{maj}}+\mathbf{\Phi^T_{min}}\alpha^{\mathbf{min}}+\mathbf{\Phi^T_{syn}}\alpha^{\mathbf{syn}}\right). \tag{42}$$

Using (36)–(41), Eq. (42) can be determined as follows.

$$\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)\beta=\mathbf{\Phi^T_{maj}}\xi_{\mathbf{maj}}+\frac{\mathbf{\Phi_{min}}^TC^{maj}\xi_{\mathbf{maj}}}{C^{min}}+\frac{\mathbf{\Phi_{min}}^{\mathbf{T}}C^{maj}\xi_{\mathbf{maj}}}{C^{syn}}+\frac{\mathbf{\Phi_{min}}^TC^{min}\xi_{\mathbf{min}}}{C^{maj}}+\mathbf{\Phi_{min}}^T\xi_{\mathbf{min}}+\frac{\mathbf{\Phi_{min}}^TC^{min}\xi_{\mathbf{min}}}{C^{syn}}+\frac{\mathbf{\Phi_{syn}}^TC^{syn}\xi_{\mathbf{syn}}}{C^{maj}}+\frac{\mathbf{\Phi_{syn}}^TC^{syn}\xi_{\mathbf{syn}}}{C^{min}}+\mathbf{\Phi^T_{syn}}\xi_{syn}. \tag{43}$$

By substituting the values of $\xi_{\mathbf{maj}}$, $\xi_{\mathbf{min}}$ and $\xi_{\mathbf{syn}}$ using Eqs. (39)–(41) into Eq. (43), we have

$$\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}\right)\beta=\mathbf{\Phi^T_{maj}}\left(\mathbf{T_{maj}}-\mathbf{\Phi_{maj}}\beta\right)+\frac{\mathbf{\Phi_{maj}}^TC^{maj}\left(\mathbf{T_{maj}}-\mathbf{\Phi_{maj}}\beta\right)}{C^{min}}+\frac{\mathbf{\Phi_{maj}}^{\mathbf{T}}C^{maj}\left(\mathbf{T_{maj}}-\mathbf{\Phi_{maj}}\beta\right)}{C^{syn}}+\frac{\mathbf{\Phi_{min}}^TC^{min}\left(\mathbf{T_{min}}-\mathbf{\Phi_{min}}\beta\right)}{C^{maj}}+\mathbf{\Phi_{min}}^T\left(\mathbf{T_{min}}-\mathbf{\Phi}_{min}\beta\right)+\frac{\mathbf{\Phi_{min}}^TC^{min}\left(\mathbf{T_{min}}-\mathbf{\Phi_{min}}\beta\right)}{C^{syn}}+\frac{\mathbf{\Phi_{syn}}^TC^{syn}\left(\mathbf{T_{syn}}-\mathbf{\Phi_{syn}}\beta\right)}{C^{syn}}+\frac{\mathbf{\Phi_{syn}}^TC^{syn}\left(\mathbf{T_{syn}}-\mathbf{\Phi_{syn}}\beta\right)}{C^{min}}+\mathbf{\Phi^T_{syn}}\left(\mathbf{T_{syn}}-\mathbf{\Phi_{syn}}\beta\right). \tag{44}$$

Eq. (44) can be rewritten as follows.

$$\beta=\begin{cases}\left(\frac{I}{C^{maj}}+\frac{I}{C^{min}}+\frac{I}{C^{syn}}+\left(1+\frac{C^{maj}}{C^{min}}+\frac{C^{maj}}{C^{syn}}\right)\mathbf{\Phi_{maj}}^T\mathbf{\Phi_{maj}}+\left(1+\frac{C^{min}}{C^{maj}}+\frac{C^{min}}{C^{syn}}\right)\mathbf{\Phi_{min}}^T\mathbf{\Phi_{min}}+\left(1+\frac{C^{syn}}{C^{maj}}+\frac{C^{syn}}{C^{min}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{\Phi_{syn}}\right)^{-1} \\ \left(\left(1+\frac{C^{maj}}{C^{min}}+\frac{C^{maj}}{C^{syn}}\right)\mathbf{\Phi_{maj}}^T\mathbf{T_{maj}}+\left(1+\frac{C^{min}}{C^{maj}}+\frac{C^{min}}{C^{syn}}\right)\mathbf{\Phi_{min}}^{\mathbf{T}}\mathbf{T_{min}}+\left(1+\frac{C^{syn}}{C^{maj}}+\frac{C^{syn}}{C^{min}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{T_{syn}}\right), \quad if\ \widehat{N}>L.\end{cases} \tag{45}$$

Based on Eq. (45), the output weights, β can be computed for the case where the number of training samples is large, $N>L$. The output weight, β for both the cases $N>L$ and for the case where the number of training samples is not large i.e. $N<L$ can be computed using Eq. (46) (see Box II).

The predicted output of SMOTE-CSELM corresponding to sample, $\boldsymbol{x}$ can be obtained as in Box III:

3.2. SMOTE-CSELM for multiclass imbalance problem

Similar to ELM [22] and WELM [19], the proposed SMOTE-CSELM is also capable of solving multiclass problems. For the multiclass imbalanced classification problems, Eq. (45) can be extended as in Box IV:

Based on Eq. (48), the output weights, β can be computed for the case where the number of training samples is large, $N>L$.

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

$$
\boldsymbol{\beta} = \begin{cases}
\begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \end{bmatrix} \left(\dfrac{I}{C^{maj}} + \dfrac{I}{C^{min}} + \dfrac{I}{C^{syn}} \right. \\
\qquad \left. + \begin{bmatrix}
\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \\
\left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \\
\left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}
\end{bmatrix} \right)^{-1} \\
\begin{bmatrix}
\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \mathbf{T}_{\mathbf{maj}} \\
\left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \mathbf{T}_{\mathbf{min}} \\
\left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \mathbf{T}_{\mathbf{syn}}
\end{bmatrix}, \qquad if\ \widehat{N} < L \\
\left(\dfrac{I}{C^{maj}} + \dfrac{I}{C^{min}} + \dfrac{I}{C^{syn}} + \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}}{}^{T} \boldsymbol{\Phi}_{\mathbf{maj}} + \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}}{}^{T} \boldsymbol{\Phi}_{\mathbf{min}} \right. \\
\qquad \left. + \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}} \boldsymbol{\Phi}_{\mathbf{syn}} \right)^{-1} \\
\left(\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}}{}^{T} \mathbf{T}_{\mathbf{maj}} + \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}}{}^{\mathbf{T}} \mathbf{T}_{\mathbf{min}} \right. \\
\qquad \left. + \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}} \mathbf{T}_{\mathbf{syn}} \right), \qquad if\ \widehat{N} > L.
\end{cases} \tag{46}
$$

Box II.

$$
\mathbf{f}(\mathbf{x}) = \begin{cases}
sign\ h(\mathbf{x}) \begin{bmatrix} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \end{bmatrix} \left(\dfrac{I}{C^{maj}} + \dfrac{I}{C^{min}} + \dfrac{I}{C^{syn}} \right. \\
\qquad \left. + \begin{bmatrix}
\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \\
\left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}} \\
\left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}} & \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}} & \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}} \boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}
\end{bmatrix} \right)^{-1} \\
\begin{bmatrix}
\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \mathbf{T}_{\mathbf{maj}} \\
\left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \mathbf{T}_{\mathbf{min}} \\
\left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \mathbf{T}_{\mathbf{syn}}
\end{bmatrix}, \qquad if\ \widehat{N} < L \\
sign\ h(\mathbf{x}) \left(\dfrac{I}{C^{maj}} + \dfrac{I}{C^{min}} + \dfrac{I}{C^{syn}} + \left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}}{}^{T} \boldsymbol{\Phi}_{\mathbf{maj}} \right. \\
\qquad \left. + \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}}{}^{T} \boldsymbol{\Phi}_{\mathbf{min}} + \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}} \boldsymbol{\Phi}_{\mathbf{syn}} \right)^{-1} \\
\left(\left(1 + \frac{C^{maj}}{C^{min}} + \frac{C^{maj}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{maj}}{}^{T} \mathbf{T}_{\mathbf{maj}} + \left(1 + \frac{C^{min}}{C^{maj}} + \frac{C^{min}}{C^{syn}}\right) \boldsymbol{\Phi}_{\mathbf{min}}{}^{\mathbf{T}} \mathbf{T}_{\mathbf{min}} \right. \\
\qquad \left. + \left(1 + \frac{C^{syn}}{C^{maj}} + \frac{C^{syn}}{C^{min}}\right) \boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}} \mathbf{T}_{\mathbf{syn}} \right), \qquad if\ \widehat{N} > L.
\end{cases} \tag{47}
$$

Box III.

The output weight, $\boldsymbol{\beta}$ for both the cases $N > L$ and for the case where the number of training samples is not large i.e. $N < L$ can be computed as in Box V:

The predicted output of SMOTE-CSELM corresponding to sample, $\boldsymbol{x}$ can be determined using Eq. (50) given in Box VI.

3.3. Estimation of computational cost

The following subsection evaluates the computational cost of SMOTE-CSELM, ELM and WELM.

3.3.1. Computational cost of SMOTE-CSELM

Eq. (46) can be employed to determine the output weight, $\boldsymbol{\beta}$ of SMOTE-CSELM. Primarily the hidden layer output matrix, $\boldsymbol{\Phi}_{\mathbf{maj}}$, $\boldsymbol{\Phi}_{\mathbf{min}}$ and $\boldsymbol{\Phi}_{\mathbf{syn}}$ are computed, which have the computational complexities of $O(nLN_{maj})$, $O(nLN_{min})$ and $O(nLN_{syn})$ respectively. The computational complexities of matrix multiplication $\boldsymbol{\Phi}_{\mathbf{maj}}\boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}}$, $\boldsymbol{\Phi}_{\mathbf{min}}\boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}}$, $\boldsymbol{\Phi}_{\mathbf{syn}}\boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}$, $\boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}}\boldsymbol{\Phi}_{\mathbf{maj}}$, $\boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}}\boldsymbol{\Phi}_{\mathbf{min}}$, $\boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}\boldsymbol{\Phi}_{\mathbf{syn}}$, $\boldsymbol{\Phi}_{\mathbf{maj}}^{\mathbf{T}}\mathbf{T}$, $\boldsymbol{\Phi}_{\mathbf{min}}^{\mathbf{T}}\mathbf{T}$ and $\boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}\mathbf{T}$ are equal to $O(L^2N_{maj})$, $O(L^2N_{min})$, $O(L^2N_{syn})$, $O(LN_{maj}^2)$, $O(LN_{min}^2)$, $O(LN_{syn}^2)$, $O(mLN_{maj})$, $O(mLN_{min})$ and $O(mLN_{syn})$ respectively. The computational cost of the output weight, $\boldsymbol{\beta}$ defined in

$$\beta=\begin{cases}\left(\sum_{k=1}^{m}\frac{I}{C^k}+\frac{I}{C^{syn}}+\sum_{k=1}^{m}\left(\left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{\Phi_k}^T\mathbf{\Phi_k}+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{\Phi_{syn}}\right)\right)^{-1}\\ \sum_{k=1}^{m}\left(\left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{\Phi_k}^T\mathbf{T_k}+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{T_{syn}}\right), \qquad if\ \widehat{N}>L.\end{cases} \tag{48}$$

Box IV.

$$\beta=\begin{cases}\left[\mathbf{\Phi_1^T}\ldots\mathbf{\Phi_m^T}\quad\mathbf{\Phi_{syn}^T}\right]\left(\sum_{k=1}^{m}\frac{I}{C^k}+\frac{I}{C^{syn}}+\begin{bmatrix}\left(\frac{C^1}{C^1}+\cdots+\frac{C^1}{C^m}+\frac{C^1}{C^{syn}}\right)\mathbf{\Phi_1}\\ .\\ \left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{\Phi_k}\\ .\\ \left(\frac{C^m}{C^1}+\cdots+\frac{C^m}{C^m}+\frac{C^m}{C^{syn}}\right)\mathbf{\Phi_m}\\ \left(\frac{C^{syn}}{C^1}+..+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{\Phi_{syn}}\end{bmatrix}\left[\mathbf{\Phi_1^T}\ldots\mathbf{\Phi_m^T}\quad\mathbf{\Phi_{syn}^T}\right]\right)^{-1}\\ \begin{bmatrix}\left(\frac{C^1}{C^1}+\cdots+\frac{C^1}{C^m}+\frac{C^1}{C^{syn}}\right)\mathbf{T_1}\\ .\\ \left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{T_k}\\ .\\ \left(\frac{C^m}{C^1}+\cdots+\frac{C^m}{C^m}+\frac{C^m}{C^{syn}}\right)\mathbf{T_m}\\ \left(\frac{C^{syn}}{C^1}+..+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{T_{syn}}\end{bmatrix}, \qquad if\ \widehat{N}<L\\ \left(\sum_{k=1}^{m}\frac{I}{C^k}+\frac{I}{C^{syn}}+\sum_{k=1}^{m}\left(\left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{\Phi_k}^T\mathbf{\Phi_k}+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{\Phi_{syn}}\right)\right)^{-1}\\ \sum_{k=1}^{m}\left(\left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{\Phi_k}^T\mathbf{T_k}+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{\Phi_{syn}}^{\mathbf{T}}\mathbf{T_{syn}}\right), \qquad if\ \widehat{N}>L.\end{cases} \tag{49}$$

Box V.

(46) is given as follows:

$$O((\widehat{N})^3+2LN_{maj}^2+2LN_{min}^2+2LN_{syn}^2+mL(N_{maj}+N_{min}+N_{syn})),$$
$$O((\widehat{N})^3+2LN_{maj}^2+2LN_{min}^2+2LN_{syn}^2+mL(\widehat{N})), \quad if\ \widehat{N}<L \tag{51}$$
$$O(L^3+L^2N_{maj}+L^2N_{min}+L^2N_{syn}+mL(N_{maj}+N_{min}+N_{syn})),$$
$$O(L^3+L^2(\widehat{N})+mL(\widehat{N})), \quad if\ \widehat{N}>L. \tag{52}$$

3.3.2. Comparisons with ELM and WELM

The computational cost of the output weight, β for ELM as determined in [28] is reproduced below.

$$O(N^3+2LN^2+mLN), \quad if\ N<L \tag{53}$$
$$O(L^3+L^2N+mLN), \quad if\ N>L. \tag{54}$$

The computational cost of the output weight, β for WELM as determined in [28] is reproduced below.

$$O(2N^3+3LN^2+mLN), \quad if\ N<L \tag{55}$$
$$O(L^3+2LN^2+L^2N+mLN), \quad if\ N>L. \tag{56}$$

The computational cost for developing the SMOTE-CSELM method is significantly lower than the WELM, for the large dimension datasets such as pageblocks0, spambase and abalone19. For larger value of N, the time taken for designing WELM method significantly increases due to the size of weight matrix, $\mathbf{W_{N\times N}}$.

4. Experimental setup and result analysis

4.1. Dataset description

In this subsection, we present the setup of the experimental framework used for the performance evaluation. The experiments have been carried out using 50 datasets (41 binary and 9 multi-class datasets) that were obtained from online repositories, which are UCI Machine Learning Repository [39] and the KEEL data repository [40]. All the datasets were partitioned using 5-fold cross-validation. All partitions of these datasets are available for

$$
\mathbf{f}(\mathbf{x})=\begin{cases}
h(\mathbf{x})\left[\boldsymbol{\Phi}_{\mathbf{1}}^{\mathbf{T}}\ldots\boldsymbol{\Phi}_{\mathbf{m}}^{\mathbf{T}}\quad\boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}\right]\left(\sum_{k=1}^{m}\dfrac{I}{C^{k}}+\dfrac{I}{C^{syn}}+\right.\\
\qquad\left.+\begin{bmatrix}\left(\frac{C^1}{C^1}+\cdots+\frac{C^1}{C^m}+\frac{C^1}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{1}}\\ \cdot\\ \left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{k}}\\ \cdot\\ \left(\frac{C^m}{C^1}+\cdots+\frac{C^m}{C^m}+\frac{C^m}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{m}}\\ \left(\frac{C^{syn}}{C^1}+..+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{syn}}\end{bmatrix}\left[\boldsymbol{\Phi}_{\mathbf{1}}^{\mathbf{T}}\ldots.\boldsymbol{\Phi}_{\mathbf{m}}^{\mathbf{T}}\quad\boldsymbol{\Phi}_{\mathbf{syn}}^{\mathbf{T}}\right]\right)^{-1}\\
\begin{bmatrix}\left(\frac{C^1}{C^1}+\cdots+\frac{C^1}{C^m}+\frac{C^1}{C^{syn}}\right)\mathbf{T}_{\mathbf{1}}\\ \cdot\\ \left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\mathbf{T}_{\mathbf{k}}\\ \cdot\\ \left(\frac{C^m}{C^1}+\cdots+\frac{C^m}{C^m}+\frac{C^m}{C^{syn}}\right)\mathbf{T}_{\mathbf{m}}\\ \left(\frac{C^{syn}}{C^1}+..+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\mathbf{T}_{\mathbf{syn}}\end{bmatrix}, & if\ \widehat{N}<L\\
h(\mathbf{x})\left(\sum_{k=1}^{m}\dfrac{I}{C^{k}}+\dfrac{I}{C^{syn}}+\sum_{k=1}^{m}\right.\\
\qquad\left.\left(\left(\frac{C^k}{C^1}+\cdots+\frac{C^k}{C^m}+\frac{C^k}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{k}}{}^{T}\boldsymbol{\Phi}_{\mathbf{k}}+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}}\boldsymbol{\Phi}_{\mathbf{syn}}\right)\right)^{-1}\\
\sum_{k=1}^{m}\left(\left(\dfrac{C^k}{C^1}+\cdots+\dfrac{C^k}{C^m}+\dfrac{C^k}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{k}}{}^{T}\mathbf{T}_{\mathbf{k}}\right.\\
\qquad\left.+\left(\frac{C^{syn}}{C^1}+\cdots+\frac{C^{syn}}{C^m}+\frac{C^{syn}}{C^{syn}}\right)\boldsymbol{\Phi}_{\mathbf{syn}}{}^{\mathbf{T}}\mathbf{T}_{\mathbf{syn}}\right), & if\ \widehat{N}>L.
\end{cases}
\tag{50}
$$

Box VI.

Algorithm 1: Proposed SMOTE-CSELM

Input: The training dataset:
$\left\{(\mathbf{x_i},\mathbf{t_i})|\ \mathbf{x_i}\in R^n,\mathbf{t_i}\in R^m, i=1,2,...,\widehat{N}\right\}$,
Number of synthetic samples N_{syn}, Number of nearest neighbors, k; Regularization parameter, C; Number of hidden neurons, L

Output: SMOTE-CSELM model for classification.

procedure SMOTE-CSELM
Initialization: seed sample set $S_{seed}=\{\}$, nearest neighbors sample set $S_{neighbor}=\{\}$
for $i=1$ **to** N_{syn} **do**
Randomly select x_i from N_{min} minority samples
Compute k nearest neighbors of x_i belonging to the minority class
Randomly select one of these k nearest neighbors, $\bar{x}$
Compute x_{syn} by using Eq. (1)
Add x_i and x_{syn} to S_{seed} and $S_{neighbor}$ respectively
end for
Initialize the weights between the hidden and the input layer, **U** size ($n\times L$) randomly.
Compute the hidden layer output matrix, $\boldsymbol{\Phi}_{\mathbf{maj}}$, $\boldsymbol{\Phi}_{\mathbf{min}}$ and $\boldsymbol{\Phi}_{\mathbf{syn}}$ for the majority class, the minority class and the synthetic samples respectively using Eq. (2).
Compute the class specific regularization parameter using Eq. (26)
Compute the output layer weights, $\boldsymbol{\beta}$ by employing Eq. (46)
return $\boldsymbol{\beta}$
end procedure

---

downloading at the KEEL dataset repository. The imbalance ratio (IR) is computed as the ratio between the number of samples belonging to the minority class and the number of samples belonging to the majority class. In this way, the smaller the IR value, the higher the imbalance of the problem. IR is computed as follows:

$$Binary:(IR)=\frac{\#minority\ \ samples}{\#majority\ \ samples} \tag{57}$$

$$Multiclass:(IR)=\frac{min(\#t_k)}{max(\#t_k)},\quad k=1,2,3,\ldots,m. \tag{58}$$

Here, # is "the number of". Tables 1 and 2 give the details for each dataset. Some of these datasets are almost balanced like spambase, sonar, penbased, led7digit and wine datasets and have higher imbalance ratio. It may be noted that high imbalance ratio corresponds to the less skewed class proportion. The attribute values are normalized in the range [-1 1], by utilizing the following equation:

$$x'=\left(\frac{x-min_n}{max_n-min_n}\right)\times 2-1. \tag{59}$$

Here, x represents the original attribute value and x' represents the normalized attribute value, max_n represents the maximum value of the attribute n and min_n represents the minimum value of the attribute n.

SMOTE based minority oversampling method needs to select a specified amount of neighbors, k for performing local searches to identify the nearby minority samples for oversampling. In this work, the number of nearest neighbors, k is set equal to 5 as recommended in the original implementation of SMOTE [10]. The same value of k is also used for WKSMOTE [14].

ARTICLE IN PRESS

Table 1
Description of binary imbalanced datasets.

Dataset description	Dataset	Number of samples	Number of features	Minority (%)	Majority (%)	Imbalance ratio
Imbalanced version of the abalone dataset	abalone19	4174	8	0.77	99.23	0.0078
	abalone9vs18	731	8	5.65	94.25	0.0599
Imbalanced version of the yeast dataset	yeast6	1484	8	2.36	97.64	0.0242
for localization site of protein	yeast1289vs7	947	8	3.17	96.83	0.0327
	yeast2vs8	482	8	4.15	95.85	0.0433
	yeast1458vs7	693	8	4.33	95.67	0.0453
	yeast1vs7	459	7	6.72	93.28	0.0720
	yeast05679vs4	528	8	9.66	90.34	0.1069
	yeast2vs4	514	8	9.92	90.08	0.1101
	yeast3	1484	8	10.98	89.02	0.1233
	yeast1	1484	8	28.91	71.09	0.4066
Imbalanced version of the ecoli dataset	ecoli0137vs26	281	7	2.49	97.51	0.0255
which contains protein localization sites	ecoli4	336	7	6.85	93.15	0.0735
	ecoli01vs5	240	6	8.33	91.67	0.0909
	ecoli3	336	7	10.71	89.26	0.1200
	ecoli2	336	7	15.48	84.52	0.1831
Statlog (shuttle)	shuttleC2vsC4	129	9	4.65	95.35	0.0488
	shuttleC0vsC4	1829	9	6.72	93.28	0.0720
Imbalanced version of the blocks dataset	pageblocks13vs4	472	10	5.93	94.07	0.0631
predict the page layout of the document	pageblocks0	5472	10	10.23	89.77	0.1140
Imbalanced version of the glass identification	glass4	214	9	6.07	93.93	0.0646
dataset from USA Forensic Science Service	glass2	214	9	8.78	91.22	0.0962
It has 6 types of glass defined in terms	glass016vs2	192	9	8.85	91.15	0.0971
of their oxide content	glass04vs5	92	9	9.78	90.22	0.1085
	glass015vs2	172	9	9.88	90.12	0.1096
	glass6	214	9	13.55	86.45	0.1567
	glass0123vs456	214	9	23.84	76.17	0.3130
	glass0	214	9	32.71	67.29	0.4861
	glass1	214	9	35.51	64.49	0.5506
Deterding vowel recognition	vowel0	988	13	9.01	90.99	0.0990
Image segmentation	segment0	2308	19	14.26	85.74	0.1663
New thyroid dataset with data of five lab tests	newthyroid1	215	5	16.28	83.72	0.1945
used to predict a patient's thyroid class	newthyroid2	215	5	16.28	83.72	0.1945
Spectfheart, diagnosing of cardiac SPECT images	spectfheart	267	44	20.60	79.40	0.2594
Imbalanced version of the vehicle silhouettes dataset	vehicle0	846	18	23.52	76.48	0.3076
which has a set of features extracted from	vehicle1	846	18	28.37	71.63	0.3960
the silhouette of four type of vehicles	vehicle2	846	18	28.37	71.63	0.3960
Haberman's survival	haberman	306	3	27.42	73.58	0.3727
Pima Indians diabetes dataset	pima	768	8	34.84	66.16	0.5266
Breast cancer wisconsin (diagnostic) dataset	wisconsin	683	9	35.00	65.00	0.5385
Spambase contains information about 4597 e-mail messages	spambase	4597	57	39.40	60.60	0.6503
Sonar mines vs. rocks dataset	sonar	208	60	46.63	53.37	0.8739

Table 2
Description of multiclass imbalanced datasets.

Dataset description	Dataset	Number of samples	Number of features	Minority (%)	Majority (%)	Imbalance ratio
Thyroid disease (thyroid0387)	tyroid	720	21	3	17/37/666	0.0271
Balance scale dataset	balance	625	4	3	49/288/288	0.1701
Imbalanced version of the dermatology	dermatology	358	34	6	20/48/48/60/71/111	0.1802
Imbalanced version of the thyroid disease	newthyroid	215	5	3	30/35/150	0.2000
Artificial dataset	hayesroth	160	4	3	31/64/65	0.4762
Penbased recognition of handwritten digits	penbased	1100	16	10	105/105/106/106/106/114/114/114/115/115	0.5130
LED display domain	led7digit	500	7	10	37/45/47/49/51/52/52/53/57/57	0.6494
Wine recognition dataset	wine	178	13	3	48/59/71	0.6757

4.2. Parameter settings

The proposed SMOTE based class-specific extreme learning machine utilizes the sigmoid activation function. The averaged performances obtained for the datasets downloaded in 5-fold cross validation format are reported in this paper. The experimental setting is same as that of WELM [19] for a fair comparison. As the ELM, WELM, CCR-ELM, CS-ELM and the proposed algorithms utilize random weights between the input and the hidden layer. This work reports the averaged performance of these algorithms for 10 independent trials. This work shows the optimal results computed by grid search of the regularization parameter, C on $\left\{2^{-18}, 2^{-16}, \ldots, 2^{48}, 2^{50}\right\}$ and the number of hidden neurons L on $\{10, 20, \ldots.990, 1000\}$. Class-specific regularization parameters,

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

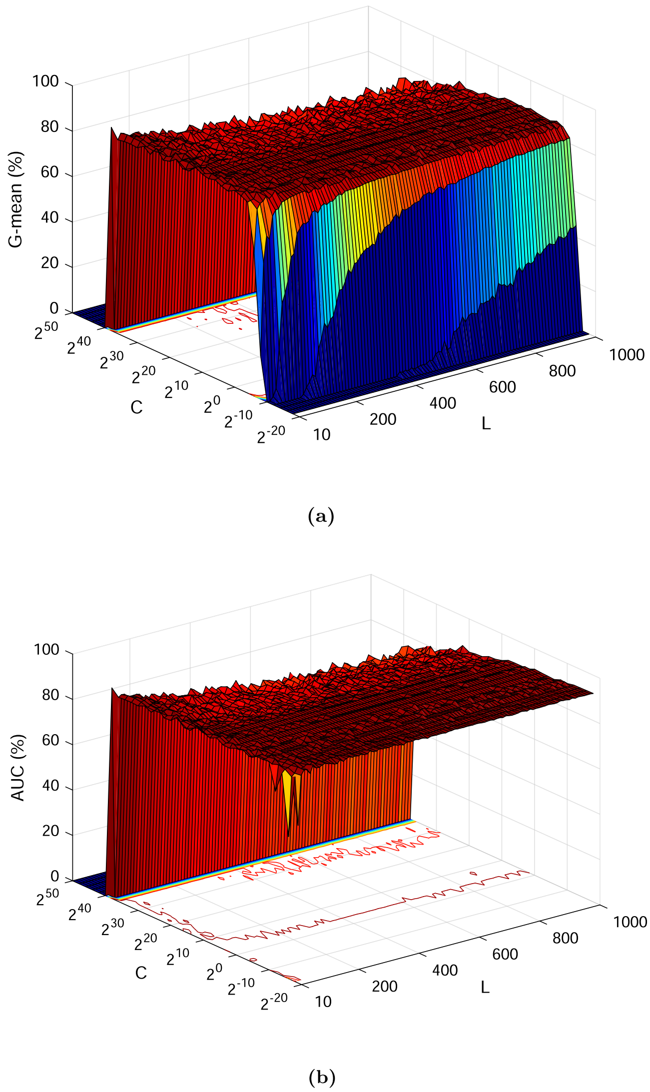


Fig. 2. Performance variation of SMOTE-CSELM for yeast6 dataset when C and L vary: (a) G-mean (b) AUC.

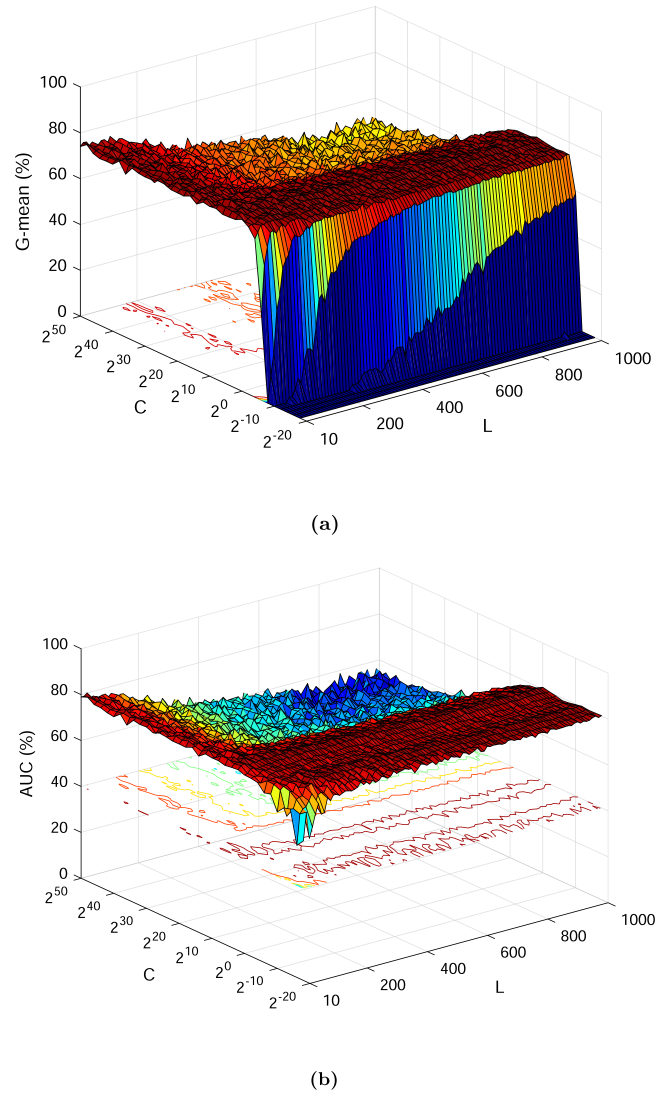


Fig. 3. Performance variation of SMOTE-CSELM for pima dataset when C and L vary: (a) G-mean (b) AUC.

C^{maj}, C^{min} and C^{syn} are set by employing the Eq. (26). The effect of the regularization parameter, C and the number of hidden neurons, L on the testing performance of SMOTE-CSELM is shown in Figs. 2 and 3. The impact of varying the number of hidden neurons L on the performance with the optimal regularization value is shown in Fig. 4. The mean training time is computed by setting the number of the hidden neurons equal to 1000 and the regularization parameter, C is set equal to 1. The mean of training time required by the 10 independent trials of all the five folds is reported as the mean training time.

4.3. Evaluation metrics for classification

Following are the evaluation metrics conventionally employed to quantify the performance of the algorithms. It can be noted that not all of these metrics are appropriate when the datasets have imbalanced class distribution.

$$\text{Overall accuracy} = \frac{TP + TN}{TP + FP + TN + FN}. \tag{60}$$

Here, TP stands for the number of correctly classified positive samples, TN represents the number of correctly classified negative samples, FP stands for the number of incorrectly classified

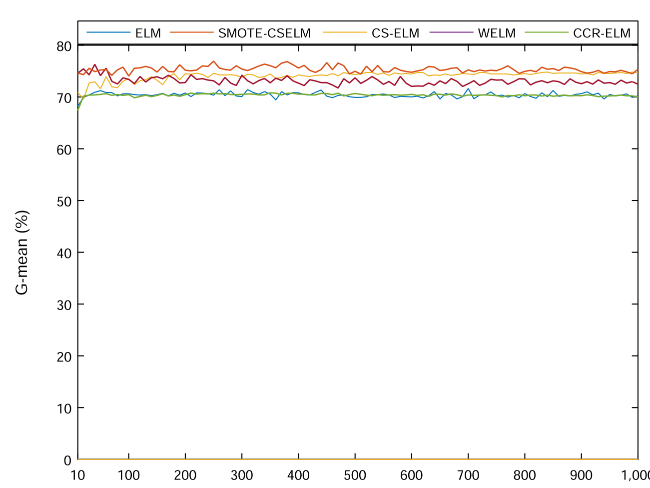
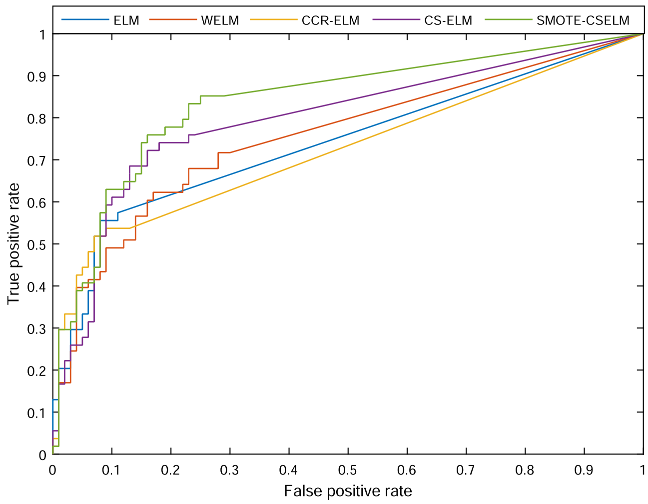
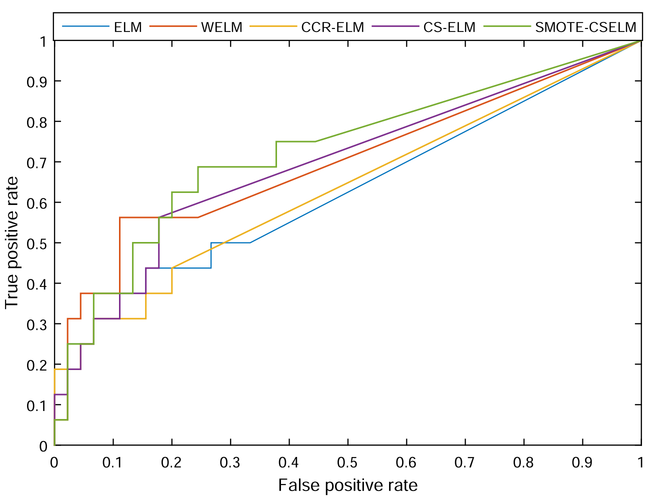


Fig. 4. G-mean of SMOTE-CSELM for pima dataset when C is optimal and L is varied.

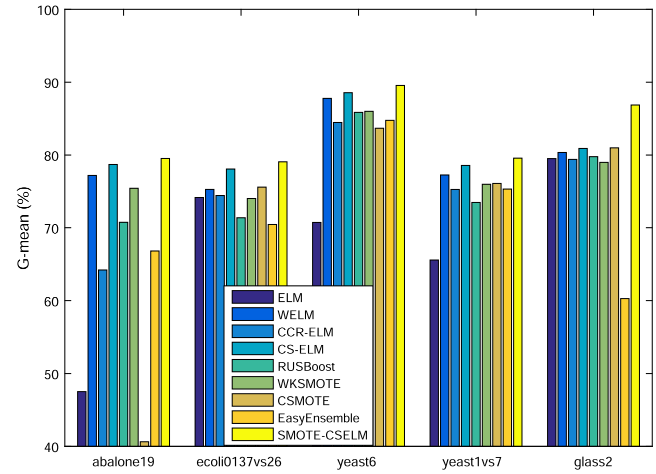


(a)

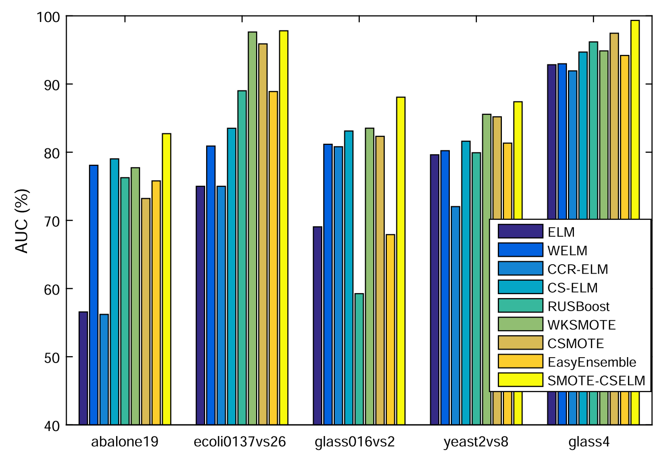


(b)

Fig. 5. ROC curves for different methods: (a) G-mean (b) AUC.



(a)



(b)

Fig. 6. Effectiveness comparison for different methods: (a) G-mean (b) AUC.

negative samples and FN represents the number of incorrectly classified positive samples.

$$TP_{rate} = \frac{TP}{TP + FN} \quad and \quad FP_{rate} = \frac{FP}{TN + FP} \tag{61}$$

where, TP_{rate} stands for the true positive rate and FP_{rate} represents the false positive rate.

$$\text{Precision} = \frac{TP}{TP + FP} \quad and \quad \text{Recall} = \frac{TP}{TP + FN} \tag{62}$$

$$\text{G-mean} = \sqrt{TP_{rate} \times (1 - FP_{rate})}. \tag{63}$$

G-mean for the multiclass problem is defined as follows:

$$\text{G-mean} = \left(\prod_{k=1}^{m} Recall_k \right)^{\frac{1}{m}}. \tag{64}$$

Here, m is the number of classes. Overall prediction accuracy is not a suitable metric for imbalanced classification problem. For example, consider a problem, which has 92 samples belonging to the majority class and 08 samples belonging to the minority class. A method, which predicts all the samples as the majority class samples will have 92% prediction accuracy. G-mean metric is better than overall prediction accuracy metric when the class distribution is not uniform. For the aforementioned example, G-mean will be equal to zero. The receiver operating characteristic (ROC) graph [41] calculates the algorithm performance by changing the confidence level for the algorithm score to get distinct values of TP_{rate} and FP_{rate} as shown in Fig. 5. In the ROC graph, the X-axis is the FP_{rate} and Y-axis is the TP_{rate}. The area under the curve (AUC) [41,42] can be employed to compute the performance of the method. The ideal point on the ROC graph would be (0, 1), for which all the minority samples are classified correctly and no majority samples are misclassified as the minority. The bigger the AUC indicates the better the generalization of the method. If the method is of hard type i.e. its predicted outcomes are discrete class label. Here, AUC can be determined as follows:

$$\text{AUC} = \frac{\left(1 + \frac{TP}{TP + FN} - \frac{FP}{FP + TN}\right)}{2} \tag{65}$$

If the method is of soft type, its predicted outcome is the score or continuous numeric value, which is the degree of confidence with which the sample belongs to the minority class. AUC for soft type method is determined by changing the confidence level. This work uses the extension of AUC for multiclass as stated in [43]. It is the average AUC of all pairs of the classes.

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

Table 3
Performance evaluation in term of G-mean for binary imbalance problems (The best result of each dataset is emphasized in bold).

Dataset	ELM		WELM		CCR-ELM		CS-ELM	RUSBoost	WKSMOTE	CSMOTE	EasyEnsemble	SMOTE-CSELM	
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^{+}, C^{-}, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
abalone19	$(2^{42}, 990)$	47.52	$(2^{6}, 150)$	77.19	$(2^{-10}, 2^{4}, 400)$	64.21	78.68	70.78	75.45	40.63	66.82	$(2^{2}, 20)$	**79.51**
abalone9vs18	$(2^{40}, 150)$	75.29	$(2^{16}, 70)$	87.99	$(2^{24}, 2^{36}, 180)$	76.22	91.99	86.40	91.94	**92.96**	80.24	$(2^{4}, 520)$	90.61
yeast3	$(2^{40}, 100)$	80.75	$(2^{16}, 700)$	93.25	$(2^{-8}, 2^{6}, 100)$	91.11	**93.57**	89.22	91.00	83.32	92.92	$(2^{4}, 230)$	93.54
ecoli0137vs26	$(2^{2}, 600)$	74.14	$(2^{4}, 400)$	75.29	$(2^{6}, 2^{4}, 450)$	74.41	78.08	71.37	74.00	75.60	70.46	$(2^{50}, 880)$	**79.06**
pageblock0	$(2^{34}, 830)$	89.92	$(2^{24}, 820)$	93.40	$(2^{16}, 2^{24}, 800)$	90.89	93.38	90.27	91.14	86.09	91.80	$(2^{18}, 820)$	**93.97**
pageblock13vs4	$(2^{8}, 660)$	97.60	$(2^{14}, 420)$	98.16	$(2^{12}, 2^{12}, 300)$	97.33	98.10	97.96	97.38	**100**	96.48	$(2^{16}, 720)$	99.77
yeast6	$(2^{44}, 350)$	70.77	$(2^{34}, 20)$	87.77	$(2^{8}, 2^{-14}, 40)$	84.45	88.55	85.85	86.00	83.69	84.76	$(2^{38}, 10)$	**89.54**
ecoli01vs5	$(2^{10}, 100)$	91.04	$(2^{14}, 20)$	94.47	$(2^{6}, 2^{10}, 90)$	92.36	94.17	88.92	88.00	75.48	90.07	$(2^{44}, 20)$	**95.55**
glass016vs2	$(2^{34}, 150)$	67.78	$(2^{14}, 380)$	83.77	$(2^{34}, 2^{30}, 240)$	76.44	85.13	52.46	79.00	69.13	58.23	$(2^{6}, 700)$	**85.68**
glass015vs2	$(2^{24}, 160)$	72.70	$(2^{16}, 180)$	**85.91**	$(2^{8}, 2^{8}, 40)$	76.93	82.33	48.94	65.00	80.13	65.92	$(2^{4}, 590)$	84.75
glass04vs5	$(2^{14}, 60)$	99.36	$(2^{6}, 610)$	**100**	$(2^{2}, 2^{8}, 450)$	99.17	99.60	96.20	92.00	99.88	78.37	$(2^{2}, 480)$	**100**
shuttleC0vsC4	$(2^{14}, 10)$	**100**	$(2^{38}, 10)$	**100**	$(2^{6}, 2^{6}, 20)$	**100**	**100**	60.00	**100**	**100**	**100**	$(2^{2}, 160)$	**100**
shuttleC2vsC4	$(2^{40}, 20)$	93.54	$(2^{28}, 10)$	**100**	$(2^{-12}, 2^{-12}, 10)$	**100**	**100**	68.50	**100**	**100**	**100**	$(2^{2}, 360)$	**100**
yeast05679vs4	$(2^{32}, 390)$	64.49	$(2^{-2}, 150)$	81.05	$(2^{-8}, 2^{-2}, 200)$	80.02	81.24	77.55	81.00	67.20	77.69	$(2^{4}, 600)$	**83.18**
yeast1vs7	$(2^{40}, 960)$	65.58	$(2^{16}, 550)$	77.26	$(2^{18}, 2^{-2}, 40)$	75.27	78.56	73.49	76.00	76.10	75.33	$(2^{0}, 740)$	**79.58**
yeast1458vs7	$(2^{44}, 970)$	61.07	$(2^{10}, 120)$	67.10	$(2^{46}, 2^{8}, 800)$	66.24	64.94	59.59	67.00	58.75	60.80	$(2^{24}, 920)$	**68.80**
yeast1289vs7	$(2^{42}, 880)$	59.23	$(2^{42}, 20)$	**75.83**	$(2^{32}, 2^{20}, 920)$	59.28	72.20	67.83	69.83	31.55	69.80	$(2^{4}, 130)$	74.57
yeast2vs8	$(2^{0}, 290)$	72.83	$(2^{8}, 60)$	76.01	$(2^{8}, 2^{-2}, 200)$	73.02	78.11	72.16	78.57	**95.32**	71.40	$(2^{-10}, 820)$	80.12
yeast2vs4	$(2^{36}, 280)$	86.25	$(2^{26}, 940)$	91.56	$(2^{-8}, 2^{24}, 400)$	90.02	92.42	84.60	80.00	**95.95**	89.89	$(2^{30}, 40)$	94.73
ecoli3	$(2^{44}, 70)$	77.38	$(2^{46}, 10)$	90.17	$(2^{-12}, 2^{-18}, 10)$	91.45	89.86	80.90	88.00	78.35	87.23	$(2^{14}, 40)$	**91.52**
ecoli4	$(2^{22}, 60)$	91.96	$(2^{6}, 180)$	97.83	$(2^{-8}, 2^{-12}, 10)$	98.43	**98.56**	89.30	92.00	88.71	91.85	$(2^{2}, 60)$	98.40
glass2	$(2^{28}, 110)$	79.49	$(2^{22}, 140)$	80.33	$(2^{-4}, 2^{-12}, 10)$	79.40	80.89	79.76	79.00	80.98	60.28	$(2^{14}, 910)$	**86.87**
glass4	$(2^{34}, 30)$	85.72	$(2^{12}, 120)$	91.34	$(2^{-8}, 2^{-2}, 40)$	96.18	96.21	87.31	89.00	83.81	86.89	$(2^{10}, 230)$	**98.22**
vowel0	$(2^{28}, 110)$	**100**	$(2^{50}, 120)$	**100**	$(2^{-8}, 2^{-2}, 400)$	**100**	**100**	**100**	**100**	**100**	98.61	$(2^{6}, 500)$	**100**
haberman	$(2^{44}, 910)$	49.16	$(2^{34}, 10)$	65.11	$(2^{36}, 2^{34}, 20)$	49.81	65.71	53.36	65.21	40.31	61.42	$(2^{12}, 10)$	**65.92**
pima	$(2^{32}, 30)$	70.10	$(2^{14}, 20)$	74.74	$(2^{2}, 2^{48}, 280)$	70.99	75.73	70.34	74.00	**77.19**	73.92	$(2^{2}, 320)$	76.65
wisconsin	$(2^{34}, 50)$	96.32	$(2^{34}, 60)$	97.07	$(2^{2}, 2^{2}, 420)$	96.94	97.37	95.46	96.33	96.86	96.57	$(2^{-14}, 280)$	**97.99**
vehicle0	$(2^{10}, 460)$	98.57	$(2^{16}, 850)$	99.32	$(2^{20}, 2^{8}, 600)$	97.81	98.32	95.82	75.52	**100**	99.45	$(2^{8}, 980)$	99.46
vehicle1	$(2^{8}, 570)$	79.29	$(2^{14}, 450)$	85.30	$(2^{8}, 2^{16}, 500)$	79.60	86.12	76.05	81.23	**91.26**	86.40	$(2^{4}, 840)$	86.17
vehicle2	$(2^{12}, 600)$	98.43	$(2^{16}, 800)$	99.12	$(2^{8}, 2^{-2}, 900)$	98.63	99.37	99.92	99.24	**100**	99.23	$(2^{10}, 800)$	99.29
newthyroid1	$(2^{18}, 180)$	98.24	$(2^{18}, 30)$	**99.44**	$(2^{-14}, 2^{-18}, 10)$	99.24	**99.44**	98.05	88.69	92.72	96.67	$(2^{8}, 640)$	99.16
newthyroid2	$(2^{18}, 40)$	95.82	$(2^{16}, 290)$	**99.72**	$(2^{22}, 2^{2}, 60)$	98.44	99.44	96.94	90.72	89.64	98.48	$(2^{12}, 280)$	99.16
glass0123vs456	$(2^{8}, 200)$	92.42	$(2^{8}, 420)$	**96.02**	$(2^{2}, 2^{10}, 30)$	93.26	95.80	93.74	94.19	85.05	92.89	$(2^{2}, 680)$	**96.02**
glass0	$(2^{14}, 950)$	79.61	$(2^{22}, 800)$	81.17	$(2^{8}, 2^{-2}, 880)$	**88.56**	80.70	82.71	78.00	78.81	79.03	$(2^{14}, 450)$	82.01
glass1	$(2^{16}, 440)$	78.36	$(2^{22}, 900)$	78.31	$(2^{-10}, 2^{12}, 70)$	76.07	**79.64**	73.02	73.00	60.98	72.41	$(2^{20}, 180)$	78.66
ecoli2	$(2^{36}, 60)$	91.17	$(2^{28}, 40)$	93.91	$(2^{-4}, 2^{-4}, 20)$	92.80	94.26	93.45	**95.00**	87.88	87.23	$(2^{18}, 320)$	93.64
glass6	$(2^{46}, 450)$	94.96	$(2^{44}, 30)$	95.72	$(2^{-12}, 2^{-4}, 20)$	91.29	95.78	88.70	90.00	72.06	91.05	$(2^{16}, 400)$	**95.79**
segment0	$(2^{8}, 720)$	99.24	$(2^{18}, 30)$	99.75	$(2^{8}, 2^{8}, 800)$	99.18	99.87	99.99	**100**	99.08	99.37	$(2^{8}, 480)$	99.67
sonar	$(2^{-2}, 710)$	**88.58**	$(2^{10}, 890)$	87.63	$(2^{-16}, 2^{2}, 880)$	88.41	88.00	32.08	86.46	85.17	69.08	$(2^{14}, 600)$	86.79
spectfheart	$(2^{40}, 320)$	62.02	$(2^{-2}, 400)$	69.95	$(2^{-16}, 2^{16}, 400)$	60.36	77.56	72.15	74.30	63.18	**77.78**	$(2^{12}, 40)$	68.72
spambase	$(2^{-4}, 660)$	88.06	$(2^{16}, 820)$	92.24	$(2^{4}, 2^{-4}, 660)$	89.52	92.42	91.32	89.12	63.30	83.51	$(2^{4}, 1000)$	**92.46**
Avg. G-mean		82.06		88.30		85.46	88.86	80.30	84.93	81.22	83.23		**89.40**
Mean-ranks		6.83		3.35		5.41	2.84	6.55	5.49	5.95	6.28		**2.29**

Table 4
Performance evaluation in term of Recall for binary imbalance problems (The best result of each dataset is emphasized in bold).

Dataset	ELM		WELM		CCR-ELM		CS-ELM	RUSBoost	WKSMOTE	CSMOTE	EasyEnsemble	SMOTE-CSELM	
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^{+}, C^{-}, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
abalone19	$(2^{44}, 950)$	27.62	$(2^{8}, 30)$	78.10	$(2^{4}, 2^{2}, 470)$	42.12	90.48	55.24	93.00	81.25	74.29	$(2^{-2}, 530)$	**96.00**
abalone9vs18	$(2^{40}, 130)$	65.28	$(2^{-18}, 50)$	92.78	$(2^{4}, 2^{20}, 100)$	58.20	87.50	78.61	89.20	75.76	82.78	$(2^{8}, 30)$	**93.06**
yeast3	$(2^{2}, 370)$	57.56	$(2^{-18}, 210)$	**100**	$(2^{6}, 2^{2}, 700)$	70.10	**100**	86.48	**100**	80.40	90.17	$(2^{-18}, 10)$	**100**
ecoli0137vs26	$(2^{-6}, 670)$	70.00	$(2^{-16}, 70)$	90.00	$(2^{4}, 2^{-12}, 500)$	75.20	**100**	87.50	**100**	**100**	90.00	$(2^{-8}, 40)$	**100**
pageblock13vs4	$(2^{10}, 270)$	**100**	$(2^{4}, 550)$	**100**	$(2^{10}, 2^{2}, 110)$	**100**	**100**	96.00	**100**	96.00	**100**	$(2^{-8}, 20)$	**100**
yeast6	$(2^{10}, 990)$	42.86	$(2^{-16}, 210)$	**100**	$(2^{8}, 2^{12}, 350)$	80.00	**100**	80.00	**100**	57.14	85.71	$(2^{-18}, 10)$	**100**
yeast05679vs4	$(2^{10}, 470)$	43.64	$(2^{-16}, 50)$	94.00	$(2^{4}, 2^{22}, 60)$	49.42	**96.00**	68.36	37.82	57.09	78.36	$(2^{6}, 450)$	86.00
yeast1vs7	$(2^{10}, 410)$	43.33	$(2^{-18}, 210)$	93.33	$(2^{-4}, 2^{-12}, 10)$	79.40	93.33	63.33	23.33	33.33	76.67	$(2^{-18}, 10)$	**100**
yeast2vs8	$(2^{16}, 770)$	65.00	$(2^{-18}, 10)$	85.00	$(2^{14}, 2^{0}, 520)$	60.34	80.00	65.00	92.00	75.00	65.00	$(2^{-18}, 10)$	**100**
yeast2vs4	$(2^{14}, 90)$	72.36	$(2^{22}, 140)$	92.00	$(2^{14}, 2^{2}, 100)$	74.12	94.00	86.18	**100**	74.36	88.00	$(2^{-18}, 10)$	**100**
ecoli3	$(2^{2}, 950)$	62.86	$(2^{-18}, 210)$	**100**	$(2^{6}, 2^{16}, 670)$	67.50	**100**	85.71	**100**	74.29	88.57	$(2^{-2}, 10)$	94.29
ecoli4	$(2^{8}, 930)$	85.00	$(2^{2}, 30)$	**100**	$(2^{2}, 2^{6}, 310)$	82.40	**100**	80.00	**100**	80.00	85.00	$(2^{-18}, 10)$	**100**
glass2	$(2^{44}, 310)$	81.67	$(2^{-18}, 70)$	**100**	$(2^{4}, 2^{14}, 870)$	85.50	**100**	75.00	**100**	**100**	88.33	$(2^{14}, 910)$	**100**
glass4	$(2^{30}, 30)$	76.67	$(2^{-12}, 630)$	**100**	$(2^{44}, 2^{2}, 10)$	82.09	**100**	76.67	80.00	70.00	93.33	$(2^{-8}, 10)$	**100**

(*continued on next page*)

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

Table 4 (*continued*).

Dataset		ELM		WELM		CCR-ELM	CS-ELM	RUSBoost	WKSMOTE	CSMOTE	EasyEnsemble		SMOTE-CSELM
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^{+}, C^{-}, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
haberman	$(2^{4}, 90)$	36.99	$(2^{34}, 10)$	65.11	$(2^{36}, 2^{34}, 20)$	49.81	62.35	71.54	**97.33**	40.31	61.62	$(2^{6}, 10)$	77.72
pima	$(2^{8}, 750)$	58.54	$(2^{-16}, 150)$	64.19	$(2^{2}, 2^{8}, 480)$	60.90	75.73	83.56	**87.21**	74.97	74.58	$(2^{4}, 10)$	80.96
wisconsin	$(2^{-2}, 890)$	97.07	$(2^{-6}, 810)$	92.06	$(2^{-4}, 2^{-12}, 700)$	97.40	97.91	97.91	60.15	93.72	97.92	$(2^{-8}, 10)$	**98.74**
vehicle0	$(2^{4}, 750)$	99.00	$(2^{-18}, 150)$	**100**	$(2^{4}, 2^{12}, 410)$	99.45	**100**	97.49	**100**	97.00	98.50	$(2^{-18}, 10)$	**100**
vehicle1	$(2^{6}, 570)$	72.30	$(2^{-18}, 50)$	78.76	$(2^{8}, 2^{2}, 720)$	76.25	92.63	73.71	94.80	79.28	77.40	$(2^{14}, 910)$	**98.00**
vehicle2	$(2^{8}, 370)$	98.17	$(2^{2}, 710)$	94.96	$(2^{8}, 2^{0}, 200)$	98.50	98.62	96.79	**100**	96.77	96.80	$(2^{-18}, 10)$	**100**
newthyroid1	$(2^{12}, 730)$	97.14	$(2^{-16}, 150)$	**100**	$(2^{8}, 2^{12}, 660)$	98.30	**100**	**100**	**100**	97.14	97.14	$(2^{-18}, 10)$	**100**
glass0123vs456	$(2^{6}, 30)$	86.36	$(2^{8}, 410)$	96.00	$(2^{4}, 2^{2}, 700)$	91.40	96.00	92.18	**100**	84.36	92.00	$(2^{0}, 780)$	96.00
glass0	$(2^{14}, 950)$	79.61	$(2^{22}, 800)$	81.17	$(2^{8}, 2^{-2}, 880)$	**88.56**	80.70	84.29	**100**	87.14	84.29	$(2^{-8}, 10)$	**100**
glass1	$(2^{20}, 290)$	79.08	$(2^{-8}, 50)$	**96.00**	$(2^{44}, 2^{2}, 60)$	85.20	94.67	75.17	67.83	69.83	71.17	$(2^{38}, 60)$	95.21
ecoli2	$(2^{4}, 350)$	80.73	$(2^{-16}, 190)$	**100**	$(2^{4}, 2^{8}, 240)$	92.21	80.89	92.55	**100**	92.55	90.73	$(2^{-8}, 10)$	**100**
glass6	$(2^{26}, 90)$	96.67	$(2^{-18}, 10)$	**100**	$(2^{30}, 2^{2}, 800)$	96.80	93.33	78.67	**100**	78.67	93.33	$(2^{-8}, 20)$	**100**
segment0	$(2^{-4}, 890)$	98.18	$(2^{-18}, 30)$	**100**	$(2^{-18}, 2^{-2}, 10)$	**100**	80.89	99.09	**100**	99.09	98.78	$(2^{-18}, 10)$	**100**
sonar	$(2^{-2}, 370)$	84.53	$(2^{-18}, 190)$	97.95	$(2^{-4}, 2^{12}, 510)$	89.42	97.95	**100**	85.53	85.00	77.42	$(2^{-2}, 230)$	85.37
spectfheart	$(2^{6}, 310)$	54.55	$(2^{-18}, 70)$	**100**	$(2^{16}, 2^{2}, 110)$	61.40	**100**	**100**	**100**	60.00	34.55	$(2^{-18}, 10)$	**100**
spambase	$(2^{2}, 130)$	80.96	$(2^{2}, 230)$	83.00	$(2^{4}, 2^{2}, 150)$	80.56	89.96	94.49	82.10	78.00	72.73	$(2^{-2}, 320)$	**98.10**
Mean-ranks		7.57		3.46		6.13	3.65	5.72	3.45	6.73	5.97		**2.32**

Table 5
Performance evaluation in term of AUC for binary imbalance problems (the best result on each dataset is emphasized in bold).

Dataset		ELM		WELM		CCR-ELM	CS-ELM	RUSBoost	WKSMOTE	CSMOTE	EasyEnsemble		SMOTE-CSELM
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^{+}, C^{-}, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
abalone19	$(2^{44}, 990)$	56.57	$(2^{4}, 620)$	78.08	$(2^{40}, 2^{26}, 680)$	56.21	79.02	76.26	77.72	73.21	75.79	$(2^{2}, 20)$	**82.72**
abalone9vs18	$(2^{30}, 420)$	75.93	$(2^{12}, 160)$	94.91	$(2^{4}, 2^{22}, 300)$	78.22	94.26	93.67	90.91	**97.32**	88.61	$(2^{4}, 260)$	94.45
yeast3	$(2^{38}, 380)$	82.83	$(2^{-8}, 310)$	92.99	$(2^{48}, 2^{6}, 60)$	85.41	94.71	94.21	95.24	**97.39**	96.61	$(2^{-14}, 380)$	96.85
ecoli0137vs26	$(2^{50}, 880)$	75.00	$(2^{-10}, 60)$	80.90	$(2^{40}, 2^{18}, 720)$	75.00	83.51	89.01	97.63	95.89	88.90	$(2^{-18}, 240)$	**97.81**
page-block0	$(2^{34}, 830)$	92.86	$(2^{16}, 820)$	94.47	$(2^{16}, 2^{4}, 920)$	93.89	94.71	96.50	96.35	**98.08**	97.42	$(2^{16}, 390)$	96.02
page-block13vs4	$(2^{12}, 100)$	97.63	$(2^{12}, 610)$	99.54	$(2^{8}, 2^{8}, 270)$	98.67	99.54	99.86	99.96	99.97	99.58	$(2^{4}, 280)$	**100**
yeast6	$(2^{44}, 335)$	70.77	$(2^{14}, 900)$	91.37	$(2^{8}, 2^{-14}, 40)$	84.45	91.02	90.20	**95.22**	94.06	92.17	$(2^{-18}, 70)$	93.46
ecoli01vs5	$(2^{46}, 40)$	92.39	$(2^{-18}, 80)$	97.50	$(2^{44}, 2^{2}, 220)$	93.50	98.27	93.94	96.22	95.00	95.89	$(2^{-4}, 20)$	**98.86**
glass016vs2	$(2^{30}, 70)$	69.05	$(2^{46}, 20)$	81.16	$(2^{6}, 2^{10}, 60)$	80.80	83.11	59.25	83.52	82.33	67.91	$(2^{2}, 800)$	**88.07**
glass015vs2	$(2^{24}, 160)$	73.76	$(2^{14}, 160)$	82.99	$(2^{12}, 2^{4}, 40)$	79.16	83.09	58.70	81.80	83.92	73.98	$(2^{4}, 460)$	**87.74**
glass04vs5	$(2^{14}, 60)$	99.38	$(2^{6}, 330)$	**100**	$(2^{12}, 2^{-4}, 90)$	99.45	**100**	80.82	99.77	**100**	96.37	$(2^{0}, 780)$	**100**
shuttleC0vsC4	$(2^{4}, 80)$	99.68	$(2^{-14}, 120)$	**100**	$(2^{-18}, 2^{0}, 20)$	99.47	**100**	80.00	**100**	**100**	**100**	$(2^{-14}, 300)$	**100**
shuttleC2vsC4	$(2^{16}, 160)$	99.20	$(2^{-18}, 240)$	**100**	$(2^{22}, 2^{-8}, 200)$	99.15	99.00	81.91	**100**	**100**	**100**	$(2^{-18}, 20)$	**100**
yeast05679vs4	$(2^{16}, 330)$	73.94	$(2^{-10}, 260)$	84.74	$(2^{12}, 2^{8}, 60)$	71.02	84.26	87.97	78.80	87.28	**88.31**	$(2^{6}, 320)$	87.32
yeast1vs7	$(2^{10}, 500)$	71.28	$(2^{-16}, 540)$	79.43	$(2^{8}, 2^{16}, 200)$	75.27	79.59	**86.53**	82.89	85.42	82.72	$(2^{-18}, 100)$	86.24
yeast1458vs7	$(2^{38}, 500)$	65.68	$(2^{14}, 10)$	70.98	$(2^{26}, 2^{8}, 10)$	69.76	74.67	65.94	74.91	69.74	69.43	$(2^{-14}, 60)$	**75.89**
yeast1289vs7	$(2^{48}, 850)$	68.06	$(2^{8}, 150)$	79.83	$(2^{42}, 2^{6}, 900)$	72.45	80.28	74.91	77.51	**81.25**	77.46	$(2^{6}, 30)$	80.60
yeast2vs8	$(2^{18}, 360)$	79.62	$(2^{18}, 60)$	80.22	$(2^{22}, 2^{26}, , 200)$	72.02	81.61	79.92	85.56	85.19	81.33	$(2^{-14}, 320)$	**87.40**
yeast2vs4	$(2^{4}, 280)$	89.31	$(2^{16}, 940)$	93.76	$(2^{-8}, 2^{6}, , 60)$	90.02	93.48	88.96	92.48	**98.08**	97.21	$(2^{30}, 40)$	96.65
ecoli3	$(2^{42}, 50)$	80.35	$(2^{6}, 10)$	92.79	$(2^{2}, 2^{-2}, 20)$	90.18	92.85	**96.22**	92.24	94.25	93.38	$(2^{14}, 40)$	94.53
ecoli4	$(2^{10}, 40)$	92.50	$(2^{24}, 10)$	99.12	$(2^{8}, 2^{-6}, 40)$	95.59	99.26	97.88	96.45	98.85	96.67	$(2^{-8}, 780)$	**99.68**
glass2	$(2^{44}, 100)$	68.75	$(2^{10}, 100)$	83.51	$(2^{48}, 2^{8}, 90)$	82.72	83.86	88.09	**88.80**	86.87	68.18	$(2^{2}, 440)$	85.44
glass4	$(2^{40}, 140)$	92.83	$(2^{16}, 20)$	92.97	$(2^{20}, 2^{8}, 560)$	91.93	94.69	96.18	94.86	97.45	94.19	$(2^{14}, 210)$	**99.33**
vowel0	$(2^{-18}, 100)$	**100**	$(2^{-18}, 40)$	**100**	$(2^{-16}, 2^{-6}, 20)$	**100**	**100**	**100**	57.38	**100**	**100**	$(2^{-2}, 580)$	**100**
haberman	$(2^{48}, 850)$	65.19	$(2^{10}, 160)$	67.63	$(2^{48}, 2^{50}, 640)$	64.83	67.63	**70.38**	67.34	67.48	67.72	$(2^{12}, 10)$	70.27
pima	$(2^{20}, 50)$	75.08	$(2^{10}, 240)$	78.72	$(2^{40}, 2^{8}, 180)$	73.87	79.02	79.91	79.18	**83.94**	80.83	$(2^{-10}, 240)$	81.39
wisconsin	$(2^{0}, 450)$	98.16	$(2^{6}, 870)$	98.44	$(2^{36}, 2^{12}, 360)$	98.88	98.92	98.37	98.80	**99.58**	98.95	$(2^{-16}, 160)$	99.57
vehicle0	$(2^{6}, 360)$	99.66	$(2^{16}, 960)$	**99.99**	$(2^{34}, 2^{20}, 620)$	99.72	99.86	99.34	84.29	99.90	99.98	$(2^{8}, 600)$	99.96
vehicle1	$(2^{8}, 510)$	83.05	$(2^{14}, 380)$	89.42	$(2^{-14}, 2^{12}, 30)$	85.78	89.51	84.75	89.45	89.93	**90.28**	$(2^{4}, 840)$	89.94
vehicle2	$(2^{12}, 870)$	99.69	$(2^{22}, 400)$	99.54	$(2^{40}, 2^{20}, 630)$	99.29	99.54	99.63	99.80	**99.84**	99.59	$(2^{10}, 800)$	99.71
new-thyroid1	$(2^{12}, 240)$	98.57	$(2^{8}, 320)$	**100**	$(2^{14}, 2^{-8}, 430)$	98.68	**100**	99.70	99.71	**100**	99.69	$(2^{-18}, 200)$	**100**
new-thyroid2	$(2^{36}, 30)$	96.06	$(2^{-16}, 50)$	**100**	$(2^{24}, 2^{0}, 70)$	99.60	**100**	99.60	99.92	**100**	99.71	$(2^{-18}, 100)$	**100**
glass0123vs456	$(2^{8}, 200)$	92.81	$(2^{50}, 20)$	97.34	$(2^{2}, 2^{8}, 20)$	95.84	97.18	97.48	**98.88**	98.86	97.85	$(2^{-16}, 290)$	98.33
glass0	$(2^{14}, 940)$	80.61	$(2^{4}, 310)$	79.45	$(2^{34}, 2^{6}, 260)$	81.88	82.35	86.44	**89.80**	86.34	86.86	$(2^{14}, 60)$	84.58
glass1	$(2^{16}, 230)$	77.46	$(2^{22}, 260)$	78.99	$(2^{18}, 2^{16}, 230)$	76.27	80.64	**80.70**	80.15	78.87	79.40	$(2^{2}, 400)$	79.86
ecoli2	$(2^{12}, 720)$	92.34	$(2^{18}, 20)$	95.42	$(2^{44}, 2^{2}, 30)$	91.71	95.11	94.24	93.70	**96.51**	96.14	$(2^{8}, 40)$	95.93
glass6	$(2^{44}, 80)$	96.25	$(2^{22}, 750)$	92.49	$(2^{48}, 2^{-8}, 120)$	95.38	92.75	92.01	90.36	95.71	96.60	$(2^{-16}, 140)$	**97.48**
segment0	$(2^{6}, 730)$	99.24	$(2^{18}, 920)$	99.83	$(2^{6}, 2^{16}, 980)$	99.69	99.80	**100**	99.91	**100**	**100**	$(2^{8}, 480)$	99.85
sonar	$(2^{2}, 340)$	90.16	$(2^{10}, 890)$	89.88	$(2^{-16}, 2^{2}, 880)$	**90.62**	89.474	59.92	85.22	85.28	88.88	$(2^{14}, 600)$	89.12
spectfheart	$(2^{10}, 410)$	62.79	$(2^{-10}, 960)$	76.62	$(2^{-16}, 2^{44}, 240)$	62.96	85.60	81.82	82.94	78.40	**85.91**	$(2^{-12}, 440)$	77.18
spambase	$(2^{-4}, 660)$	89.07	$(2^{16}, 820)$	94.02	$(2^{8}, 2^{-2}, 20)$	89.70	94.50	**96.99**	90.28	73.34	85.19	$(2^{6}, 860)$	94.33
Avg. AUC		84.48		89.98		86.32	90.65	86.23	90.45	91.11	89.65		**92.36**
Mean-ranks		7.83		5.00		7.38	4.46	5.52	4.43	3.27	4.51		**2.59**

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.

Table 6
Performance evaluation in term of G-mean for multiclass datasets (the best result obtained for each dataset is highlighted in bold).

Dataset		ELM		WELM		CCR-ELM	RUSBoost	EasyEnsemble	WKSMOTE		SMOTE-CSELM
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^+, C^-, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
thyroid	$(2^{26}, 770)$	46.49	$(2^{14}, 450)$	67.57	$(2^{8}, 2^{14}, 880)$	48.45	63.08	70.60	39.00	$(2^{44}, 140)$	**73.14**
glass	$(2^{14}, 890)$	54.61	$(2^{6}, 840)$	66.15	$(2^{10}, 2^{4}, 800)$	55.21	63.29	**83.93**	77.57	$(2^{2}, 590)$	67.82
balance	$(2^{40}, 440)$	44.18	$(2^{6}, 960)$	55.70	$(2^{8}, 2^{24}, 200)$	47.70	58.47	70.46	46.92	$(2^{6}, 980)$	**81.99**
dermatology	$(2^{0}, 600)$	97.44	$(2^{0}, 700)$	97.41	$(2^{8}, 2^{-2}, 10)$	95.43	06.90	81.28	**100**	$(2^{0}, 490)$	98.27
newthyroid	$(2^{8}, 210)$	93.50	$(2^{12}, 30)$	**99.16**	$(2^{8}, 2^{6}, 40)$	92.28	97.46	98.00	92.58	$(2^{18}, 550)$	98.61
hayesroth	$(2^{16}, 40)$	77.34	$(2^{8}, 50)$	82.12	$(2^{-2}, 2^{8}, 50)$	78.21	92.98	**93.34**	92.29	$(2^{6}, 140)$	78.15
penbased	$(2^{40}, 20)$	93.54	$(2^{14}, 710)$	98.00	$(2^{2}, 2^{-4}, 80)$	92.96	33.80	96.56	94.70	$(2^{2}, 590)$	**98.61**
led7digit	$(2^{0}, 910)$	63.29	$(2^{8}, 200)$	70.73	$(2^{0}, 2^{2}, 40)$	65.35	61.77	74.61	**86.38**	$(2^{-8}, 50)$	73.17
wine	$(2^{0}, 290)$	**100**	$(2^{6}, 420)$	**100**	$(2^{0}, 2^{2}, 240)$	**100**	98.44	86.77(7)	97.42	$(2^{2}, 370)$	**100**
Avg. G-mean		71.71		81.76		75.07	64.02	83.95	80.76		**85.20**
Mean-ranks		5.39		3.17		5.17	4.88	3.00	4.00		**2.38**

Table 7
Performance evaluation in term of AUC for multiclass datasets (The best result obtained for each dataset is highlighted in bold).

Dataset		ELM		WELM		CCR-ELM	RUSBoost	EasyEnsem-ble	WKSMOTE		SMOTE-CSELM
	(C, L)	Testing result (%)	(C, L)	Testing result (%)	(C^+, C^-, L)	Testing result (%)	Testing result (%)	Testing result (%)	Testing result (%)	(C, L)	Testing result (%)
thyroid	$(2^{8}, 1000)$	66.15	$(2^{14}, 900)$	84.47	$(2^{8}, 2^{-14}, 40)$	84.45	69.27	81.60	56.20	$(2^{6}, 40)$	**92.36**
glass	$(2^{26}, 230)$	68.16	$(2^{8}, 730)$	84.26	$(2^{-10}, 2^{4}, 400)$	66.21	88.67	94.72	61.50	$(2^{22}, 390)$	**97.82**
balance	$(2^{48}, 990)$	59.96	$(2^{6}, 960)$	99.70	$(2^{8}, 2^{24}, 200)$	77.70	74.90	74.69	63.58	$(2^{30}, 600)$	**100**
dermatology	$(2^{2}, 300)$	99.25	$(2^{6}, 900)$	98.75	$(2^{-8}, 2^{-12}, 10)$	98.43	55.70	95.77	52.99	$(2^{-8}, 380)$	**100**
newthyroid	$(2^{34}, 30)$	95.72	$(2^{12}, 120)$	**100**	$(2^{-8}, 2^{-2}, 40)$	96.18	99.80	97.85	98.42	$(2^{6}, 870)$	98.33
hayesroth	$(2^{14}, 10)$	**100**	$(2^{8}, 980)$	**100**	$(2^{0}, 2^{6}, 20)$	**100**	99.78	99.43	95.00	$(2^{10}, 300)$	**100**
penbased	$(2^{40}, 20)$	93.54	$(2^{14}, 910)$	99.52	$(2^{2}, 2^{12}, 300)$	95.22	66.86	98.96	57.70	$(2^{30}, 460)$	**100**
led7digit	$(2^{0}, 930)$	69.70	$(2^{-8}, 110)$	**96.12**	$(2^{0}, 2^{2}, 780)$	73.57	87.82	93.20	93.80	$(2^{2}, 750)$	91.22
wine	$(2^{-4}, 40)$	**100**	$(2^{0}, 210)$	**100**	$(2^{-4}, 2^{2}, 240)$	**100**	99.63	93.43	99.52	$(2^{-4}, 360)$	**100**
Avg. AUC		83.61		95.87		87.97	82.49	92.18	75.41		**97.75**
Mean-ranks		4.89		2.33		4.11	4.56	4.33	5.78		**2.00**

Table 8
The Wilcoxon signed-rank test results based on the G-mean. here, R^+ is the sum of ranks for the datasets in which the first method outperforms the second and R^- is the sum of ranks for the opposite.

Comparison	R^+	R^-	p-value	Hypothesis (0.05)
SMOTE-CSELM vs. ELM	771.0	9.0	$1.0558e^{-07}$	**Rejected**
SMOTE-CSELM vs. WELM	565.5	100.5	$2.5926e^{-04}$	**Rejected**
SMOTE-CSELM vs. CCR-ELM	699.0	42.0	$1.8979e^{-06}$	**Rejected**
SMOTE-CSELM vs. CS-ELM	575.0	166.0	0.0030	**Rejected**
SMOTE-CSELM vs. RUSBoost	800.0	20.0	$1.5875e^{-07}$	**Rejected**
SMOTE-CSELM vs. WKSMOTE	702.5	38.5	$1.4733e^{-06}$	**Rejected**
SMOTE-CSELM vs. CSMOTE	663.0	78.0	$2.2156e^{-05}$	**Rejected**
SMOTE-CSELM vs. EasyEnsemble	746.0	34.0	$6.7633e^{-07}$	**Rejected**

Table 9
The Wilcoxon signed-rank test results based on the Recall. here, R^+ is the sum of ranks for the datasets in which the first method outperforms the second and R^- is the sum of ranks for the opposite.

Comparison	R^+	R^-	p-value	Hypothesis (0.05)
SMOTE-CSELM vs. ELM	435.0	0.0	$2.5631e^{-06}$	**Rejected**
SMOTE-CSELM vs. WELM	112.5	40.5	0.0883	Not rejected
SMOTE-CSELM vs. CCR-ELM	400.0	6.0	$7.2583e^{-06}$	**Rejected**
SMOTE-CSELM vs. CS-ELM	155.0	35.0	0.0157	**Rejected**
SMOTE-CSELM vs. RUSBoost	386.0	20.0	$3.0822e^{-05}$	**Rejected**
SMOTE-CSELM vs. WKSMOTE	91.0	29.0	0.0833	Not rejected
SMOTE-CSELM vs. CSMOTE	406.0	0.0	$3.7869e^{-06}$	**Rejected**
SMOTE-CSELM vs. EasyEnsemble	435.0	0.0	$2.5614e^{-06}$	**Rejected**

4.4. Experimental results

The efficacy of the proposed SMOTE-CSELM method was reported in terms of G-mean, Recall and AUC as shown in Tables 3–7. The results of ELM, WELM, CCR-ELM, CS-ELM, RUSBoost [44],

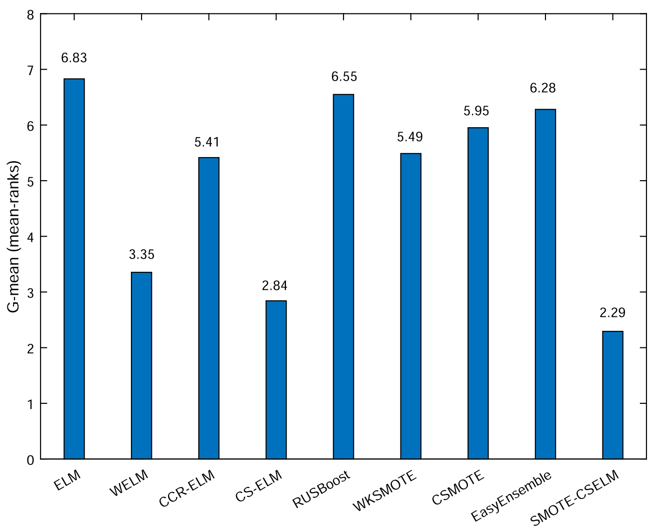
Table 10
The Wilcoxon signed-rank test results based on the AUC. here, R^+ is the sum of ranks for the datasets in which the first method outperforms the second and R^- is the sum of ranks for the opposite.

Comparison	R^+	R^-	p-value	Hypothesis (0.05)
SMOTE-CSELM vs. ELM	813.0	7.0	$6.0661e^{-08}$	**Rejected**
SMOTE-CSELM vs. WELM	607.0	23.0	0.0032	**Rejected**
SMOTE-CSELM vs. CCR-ELM	809.0	11.0	$8.1816e^{-08}$	**Rejected**
SMOTE-CSELM vs. CS-ELM	606.5	59.5	$1.7320e^{-05}$	**Rejected**
SMOTE-CSELM vs. RUSBoost	713.5	147.5	$2.4517e^{-04}$	**Rejected**
SMOTE-CSELM vs. WKSMOTE	600.5	140.5	$8.5112e^{-04}$	**Rejected**
SMOTE-CSELM vs. CSMOTE	403.5	226.5	0.1472	Not Rejected
SMOTE-CSELM vs. EasyEnsemble	616.0	125.0	$3.7042e^{-04}$	**Rejected**

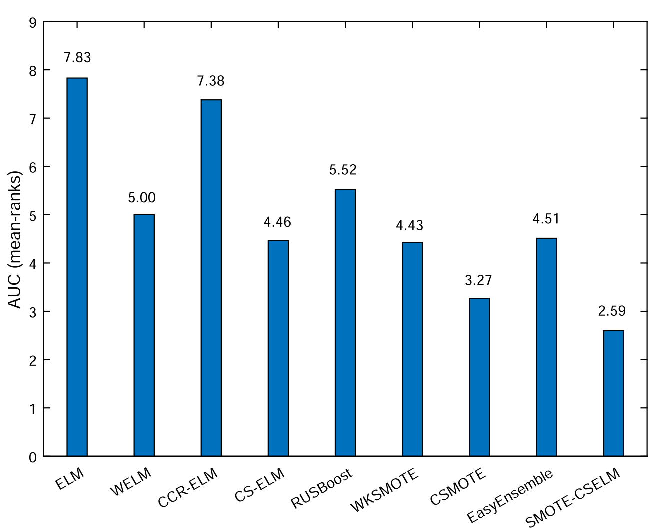
CSMOTE [45], EasyEnsemble [46] and WKSMOTE [14] are obtained by the experimentation. This work uses the MATLAB code available online [45] for evaluating the results of EasyEnsemble, RUSBoost and CSMOTE. It can be observed from Tables 3–7 that SMOTE-CSELM outperforms ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, CSMOTE, EasyEnsemble and WKSMOTE for most of the datasets. It can also be observed in Fig. 6 that SMOTE-CSELM performs better than ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, CSMOTE, EasyEnsemble and WKSMOTE for most of the problems. It may be noted that generation of synthetic minority class and employing different regularization parameters for the minority, the majority and the synthetic minority class samples leads to improved classification accuracy as shown in Table 4, for the minority classes which has not been taken care of by other classifiers. For further comparison of their performance, this work uses the Wilcoxon signed-rank test. The results of the test are shown in Tables 8–10. The confidence level for this test is set to 0.05. If the p-value is lower than 0.05, then there is

Table 11
The mean training time (MTT) (in seconds) for the different methods.

Dataset	ELM	WELM	CCR-ELM	CS-ELM	RUSBoost	WKSMOTE	CSMOTE	EasyEnsemble	SMOTE-CSELM
pageblocks0	3.6028	22.7756	3.8258	**0.7656**	36.0030	9.8182	8.5448	56.3403	1.6213
spambase	1.9685	12.2626	1.9943	**0.7585**	148.4519	8.1654	7.4270	37.6080	1.1417
abalone19	1.6035	8.9437	1.8243	**0.5962**	3.8221	6.0858	5.0417	3.0860	1.4452
segment0	0.4950	1.6617	0.4986	**0.3510**	4.5844	2.5651	2.2625	7.8398	0.7608
shuttleC0vsC4	**0.2052**	1.1988	0.2363	0.2709	0.7969	1.2916	1.2968	3.5022	0.6572
yeast6	**0.1228**	0.4934	0.1426	0.2456	2.5535	1.1720	0.8668	2.5961	0.5770
yeast3	**0.1293**	0.4633	0.1409	0.2461	0.5787	1.1686	0.8894	2.1830	0.6333
abalone9vs18	**0.0290**	0.0784	0.0294	0.1666	1.8014	0.1833	0.3369	2.2280	0.3489
pima	**0.0296**	0.0774	0.0314	0.1523	1.6307	0.3317	0.3592	1.8347	0.3338

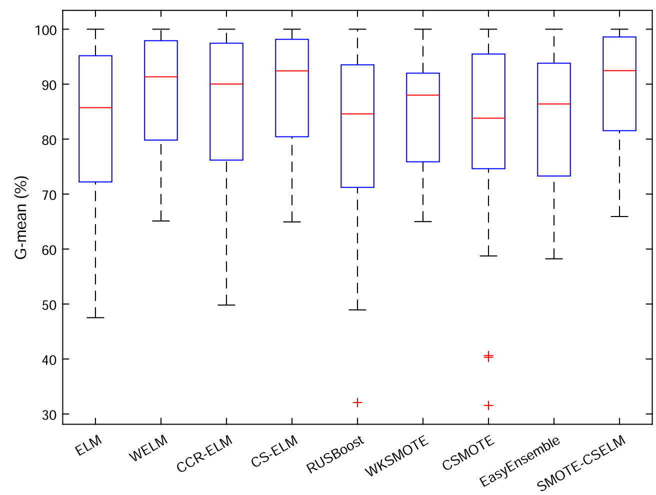


(a)

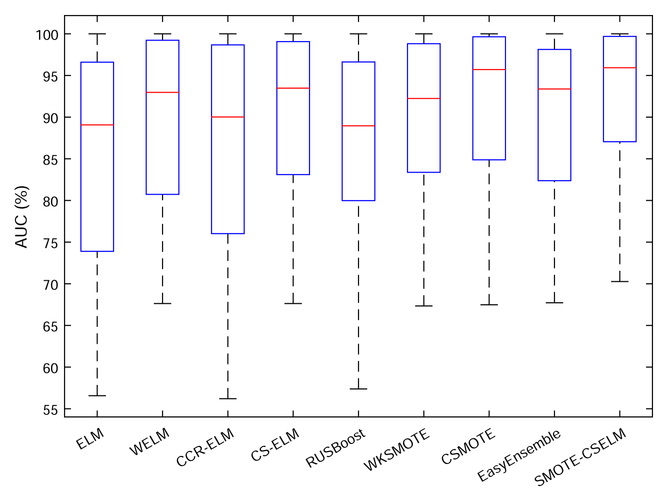


(b)

Fig. 7. The Friedman aligned-ranks test for different methods: (a) G-mean (b) AUC.



(a)



(b)

Fig. 8. Display of the Box-plot for different methods: (a) G-mean (b) AUC.

a substantial difference between the two methods. The lower the p-value, the difference is more statistically significant. The Wilcoxon signed-rank test results in terms of G-mean, Recall, and AUC are given in Tables 8–10 validate the proposed SMOTE-CSELM outperforms ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, CSMOTE, EasyEnsemble and WKSMOTE. The average ranking of each classifier is additionally, determined by employing the Friedman aligned-ranks test [47] for the G-mean and AUC scores of the classifiers in consideration. The proposed SMOTE-CSELM is selected as the control method, as it gets the least mean-ranks in the Friedman aligned-ranks test, which is shown in Fig. 7. The Friedman test with the corresponding post-hoc test [47] for the 9 methods by utilizing 41 binary class datasets. Let the average rank of the lth algorithms among a set of l algorithms is R_j, then

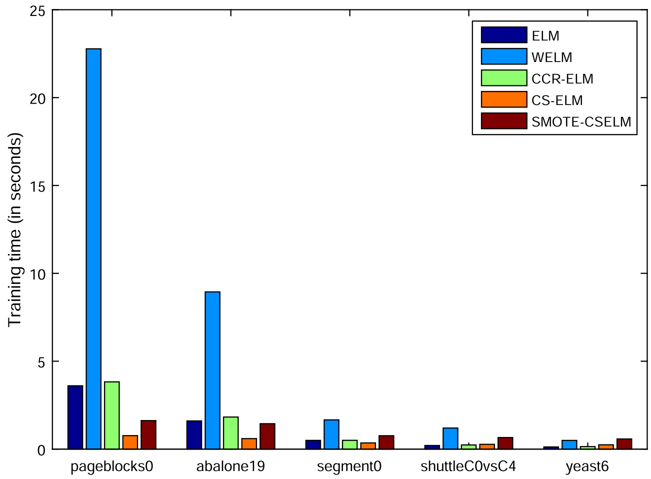


Fig. 9. Display of the training time for different methods.

the null hypothesis that all l algorithms are equal can be rejected. The Friedman statistic is determined for the AUC values given in Table 5 as follows:

$$\chi_F^2 = \frac{12 \times N}{l(l+1)}\left[\sum_{j=1}^{k} R_j^2 - \frac{l(l+1)^2}{4}\right] \tag{66}$$

where, l is the number of algorithms and N is the number of datasets.

The Friedman statistic is computed for the AUC as follows:

$$\chi_F^2 = \frac{12 \times 41}{9(9+1)}\left[(7.83^2 + 5.00^2 + 7.38^2 + 4.46^2 + 5.52^2 \right. \tag{67}$$

$$\left. +4.43^2 + 3.27^2 + 4.51^2 + 2.59^2) - \frac{9(9+1)^2}{4}\right]$$

$$\cong 155.9021 \tag{68}$$

$$F_F = \frac{(41-1) \times 155.9021}{41 \times (9-1) - 155.9021} \cong 36.2357. \tag{69}$$

Here, F_F is distribution according to the F-distribution with $(9-1, (9-1) \times (41-1)) = (8, 320)$ degrees of freedom with 9 algorithms and 41 datasets. The critical difference value of $F(8, 320)$ is 1.9771 for the predefined confidence level, $\alpha = 0.05$. The value of $F_F = 36.2357 > 1.9771$. So, we reject the null hypothesis. In addition, the Nemenyi post-hoc test is also performed for pair-wise comparison of algorithms.

$$\text{Critical difference (CD)} = q_{0.10}\sqrt{\frac{l(l+1)}{6 \times N}}$$

$$= 2.855\sqrt{\frac{9(9+1)}{6 \times 41}} \cong 1.7227. \tag{70}$$

The differences between the mean ranks of ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, WKSMOTE and EasyEnsemble with respect to SMOTE-CSELM are $(7.83 - 2.59 = 5.24)$, $(5.00 - 2.59 = 2.41)$, $(7.38 - 2.59 = 4.79)$, $(4.46 - 2.59 = 1.87)$, $(5.52 - 2.59 = 2.93)$, $(4.43 - 2.59 = 1.84)$ and $(4.51 - 2.59 = 1.92)$ respectively, which is larger than 1.7227. So, this work concludes that the SMOTE-CSELM is significantly better than ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, WKSMOTE and EasyEnsemble for imbalanced learning. Additionally, the box plot generated by the proposed method in term of G-mean and AUC shows that the dispersion degree of SMOTE-CSELM is relatively lower than the other methods, which can be observed from Fig. 8. The box plots shown in Fig. 8 confirm the superiority of SMOTE-CSELM over the rest of the methods. The mean training time of the ELM, WELM, CCR-ELM, CS-ELM, RUSBoost, WKSMOTE, CSMOTE, EasyEnsemble and the proposed SMOTE-CSELM are reported in Table 11. For the large dimension datasets such as pageblocks0, spambase, abalone19, segment0 and shuttleC0vsC4, the proposed SMOTE-CSELM takes lower training time than WELM, which can be observed from Table 11 and Fig. 9. However, when the dataset is small such as pima, abalone9vs18 and yeast3 datasets SMOTE-CSELM takes more training time compared to the WELM.

5. Conclusion

Class imbalance is one of the foremost data challenges in classification problems. In order to tackle the imbalanced classification problems with skewed class distribution, this work proposes and assesses a SMOTE based class-specific extreme learning machine. SMOTE increases the significance of the minority class samples for determining the decision region of the classifiers by creating synthetic samples belonging to the minority class. This work extends CS-ELM to create classification model for the oversampled data obtained using SMOTE. The proposed algorithm does not assign any weight to the training samples. The proposed method has comparable computational cost in contrast with WELM. The proposed algorithm is assessed by utilizing the real-world imbalanced datasets. The results demonstrate the high efficacy of the proposed method than the other tested algorithms. In addition, the efficacy of SMOTE-CSELM is also demonstrated by the statistical test analysis. Our future work will include developing novel regression algorithm by extending SMOTE-CSELM. The future work also incorporates the extension of SMOTE-CSELM as online sequential SMOTE-CSELM to address the large size imbalanced classification problems more effectively.

References

[1] H. Parvin, B. Minaei-Bidgoli, H. Alizadeh, Detection of cancer patients using an innovative method for learning at imbalanced datasets, in: J. Yao, S. Ramanna, G. Wang, Z. Suraj (Eds.), Rough Sets and Knowledge Technology, Springer Berlin Heidelberg, 2011, pp. 376–381.
[2] M. Kubat, R.C. Holte, S. Matwin, Machine learning for the detection of oil spills in satellite radar images, Mach. Learn. 30 (2) (1998) 195–215.
[3] S. Wang, X. Yao, Using class imbalance learning for software defect prediction, IEEE Trans. Reliab. 62 (2) (2013) 434–443.
[4] B. Krawczyk, M. Galar, L. Jele, F. Herrera, Evolutionary undersampling boosting for imbalanced classification of breast cancer malignancy, Appl. Soft Comput. 38 (C) (2016) 714–726.
[5] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[6] A. Sarmanova, S. Albayrak, Alleviating class imbalance problem in data mining, in: 2013 21st Signal Processing and Communications Applications Conference (SIU), 2013, pp. 1–4.
[7] G. Haixiang, L. Yijing, J. Shang, G. Mingyun, H. Yuanyue, G. Bing, Learning from class-imbalanced data: Review of methods and applications, Expert Syst. Appl. 73 (2017) 220–239.
[8] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, F. Herrera, A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches, IEEE Trans. Syst. Man Cybern. C 42 (4) (2012) 463–484.
[9] X.Y. Liu, J. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern. B 39 (2) (2009) 539–550.
[10] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: Synthetic minority over-sampling technique, J. Artificial Intelligence Res. 16 (1) (2002) 321–357.
[11] H. He, Y. Bai, E.A. Garcia, S. Li, ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in: 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322–1328.
[12] H. Han, W.-Y. Wang, B.-H. Mao, Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning, in: D.-S. Huang, X.-P. Zhang, G.-B. Huang (Eds.), Advances in Intelligent Computing, Springer Berlin Heidelberg, ISBN: 978-3-540-31902-3, 2005, pp. 878–887.
[13] S. Barua, M.M. Islam, X. Yao, K. Murase, MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning, IEEE Trans. Knowl. Data Eng. 26 (2) (2014) 405–425.

ARTICLE IN PRESS

[14] J. Mathew, C.K. Pang, M. Luo, W.H. Leong, Classification of imbalanced data by oversampling in Kernel space of support vector machines, IEEE Trans. Neural Netw. Learn. Syst. (2018) 1–12.

[15] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (3) (2011) 27:1–27:27.

[16] D.A. Cieslak, T.R. Hoens, N.V. Chawla, W.P. Kegelmeyer, Hellinger distance decision trees are robust and skew-insensitive, Data Min. Knowl. Discov. 24 (1) (2012) 136–158.

[17] Y. Tang, Y. Zhang, N.V. Chawla, S. Krasser, SVMs modeling for highly imbalanced classification, IEEE Trans. Syst. Man Cybern. B 39 (1) (2009) 281–288.

[18] Z.-H. Zhou, X.-Y. Liu, On multi-class cost-sensitive learning, in: Proceedings of the 21st National Conference on Artificial Intelligence, Vol. 1, AAAI'06, AAAI Press, ISBN: 978-1-57735-281-5, 2006, pp. 567–572.

[19] W. Zong, G.-B. Huang, Y. Chen, Weighted extreme learning machine for imbalance learning, Neurocomputing 101 (2013) 229–242.

[20] X. Yang, Q. Song, Y. Wang, A weighted support vector machine for data classification, Int. J. Pattern Recognit. Artif. Intell. 21 (05) (2007) 961–976.

[21] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (1–3) (2006) 489–501.

[22] G.B. Huang, H. Zhou, X. Ding, R. Zhang, Extreme learning machine for regression and multiclass classification, IEEE Trans. Syst. Man Cybern. B 42 (2) (2012) 513–529.

[23] V.M. Janakiraman, X. Nguyen, J. Sterniak, D. Assanis, Identification of the dynamic operating envelope of HCCI engines using class imbalance learning, IEEE Trans. Neural Netw. Learn. Syst. 26 (1) (2015) 98–112.

[24] V.M. Janakiraman, X. Nguyen, D. Assanis, Stochastic gradient based extreme learning machines for stable online learning of advanced combustion engines, Neurocomputing 177 (2016) 304–316.

[25] K. Li, X. Kong, Z. Lu, L. Wenyin, J. Yin, Boosting weighted ELM for imbalanced learning, Neurocomputing 128 (2014) 15–21.

[26] S. Shukla, R.N. Yadav, Regularized weighted circular complex-valued extreme learning machine for imbalanced learning, IEEE Access 3 (2015) 3048–3057.

[27] W. Xiao, J. Zhang, Y. Li, S. Zhang, W. Yang, Class-specific cost regulation extreme learning machine for imbalanced classification, Neurocomputing 261 (2017) 70–82.

[28] B.S. Raghuwanshi, S. Shukla, Class-specific extreme learning machine for handling binary class imbalance problem, Neural Netw. 105 (2018) 206–217.

[29] B.S. Raghuwanshi, S. Shukla, Class-specific kernelized extreme learning machine for binary class imbalance learning, Appl. Soft Comput. 73 (2018) 1026–1038.

[30] B.S. Raghuwanshi, S. Shukla, Generalized class-specific kernelized extreme learning machine for multiclass imbalanced learning, Expert Syst. Appl. 121 (2019) 244–255.

[31] B.S. Raghuwanshi, S. Shukla, Class imbalance learning using underbagging based kernelized extreme learning machine, Neurocomputing 329 (2019) 172–187.

[32] B.S. Raghuwanshi, S. Shukla, Underbagging based reduced kernelized weighted extreme learning machine for class imbalance learning, Eng. Appl. Artif. Intell. 74 (2018) 252–270.

[33] B.S. Raghuwanshi, S. Shukla, Class-specific cost-sensitive boosting weighted ELM for class imbalance learning, Memetic Comput. (2018).

[34] A. Iosifidis, A. Tefas, I. Pitas, Approximate kernel extreme learning machine for large scale data classification, Neurocomputing 219 (2017) 210–220.

[35] H. He, Y. Ma, Class imbalance learning methods for support vector machines, in: Imbalanced Learning: Foundations, Algorithms, and Applications, Wiley-IEEE Press, ISBN: 9781118646106, 2013, p. 216.

[36] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Trans. Inform. Theory 13 (1) (1967) 21–27.

[37] W. Deng, Q. Zheng, L. Chen, regularized extreme learning machine, in: IEEE Symposium on Computational Intelligence and Data Mining, 2009, pp. 389–395.

[38] A.E. Hoerl, R.W. Kennard, Ridge regression: Biased estimation for nonorthogonal problems, Technometrics 42 (1) (2000) 80–86.

[39] D. Dheeru, E. Karra Taniskidou, UCI machine learning repository, 2017, URL http://archive.ics.uci.edu/ml.

[40] J. Alcalá, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework, J. Mult.-Valued Logic Soft Comput. 17 (2–3) (2011) 255–287.

[41] T. Fawcett, ROC Graphs: Notes and Practical Considerations for Researchers, Tech. Rep., HP Labs, Tech. Rep. HPL-2003-4, 2003.

[42] J. Huang, C.X. Ling, Using AUC and accuracy in evaluating learning algorithms, IEEE Trans. Knowl. Data Eng. 17 (3) (2005) 299–310.

[43] D.J. Hand, R.J. Till, A simple generalisation of the area under the ROC curve for multiple class classification problems, Mach. Learn. 45 (2) (2001) 171–186.

[44] C. Seiffert, T.M. Khoshgoftaar, J.V. Hulse, A. Napolitano, RUSBoost: A hybrid approach to alleviating class imbalance, IEEE Trans. Syst. Man Cybern. A 40 (1) (2010) 185–197.

[45] L. Nanni, C. Fantozzi, N. Lazzarini, Coupling different methods for overcoming the class imbalance problem, Neurocomputing 158 (C) (2015) 48–61.

[46] X.Y. Liu, J. Wu, Z.H. Zhou, Exploratory undersampling for class-imbalance learning, IEEE Trans. Syst. Man Cybern. B 39 (2) (2009) 539–550.

[47] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.

Please cite this article as: B.S. Raghuwanshi and S. Shukla, SMOTE based class-specific extreme learning machine for imbalanced learning, Knowledge-Based Systems (2019), https://doi.org/10.1016/j.knosys.2019.06.022.