Neural Processing Letters (2022) 54:3097–3114
https://doi.org/10.1007/s11063-022-10756-2

Cost-Sensitive Learning based on Performance Metric for Imbalanced Data

Yuri Sousa Aurelio[1] · Gustavo Matheus de Almeida[1] · Cristiano Leite de Castro[1] · Antonio Padua Braga[1]

Accepted: 24 January 2022 / Published online: 9 February 2022

Abstract
Performance metrics are usually evaluated only after the neural network learning process using an error cost function. This procedure can result in suboptimal model selection, particularly for imbalanced classification problems. This work proposes the direct use of these metrics as cost functions, which are often derived from the confusion matrix. Commonly used metrics are covered, namely AUC, G-mean, F1-score and AG-mean. The only implementation change for model training occurs in the backpropagation error term. The results were compared to the standard MLP using the Rprop learning algorithm, SMOTE, SMTTL, WWE and RAMOBoost. Sixteen classical benchmark datasets were used in the experiments. Based on average ranks, the proposed formulation outperformed Rprop and all sampling strategies, namely SMOTE, SMTTL and WWE, for all metrics. These results were statistically confirmed for AUC and G-mean in relation to Rprop. For F1-score and AG-mean, all algorithms were considered statistically equivalent. The proposal was also superior to RAMOBoost for G-mean given average ranks. However, it was statistically faster than RAMOBoost for all metrics. It was also faster than SMTTL and statistically equivalent to Rprop, SMOTE and WWE. More, the solutions obtained are generally non-dominated ones compared to all other techniques, for all metrics. The results showed that the direct use of performance metrics as cost functions for neural network training favors generalization capacity and also computation time in imbalanced classification problems. Its extension to other performance metrics derived directly from the confusion matrix is straightforward.

Keywords Classification · Imbalanced problem · Cost-sensitive function · Multi-Layer perceptron · Back-propagation · Confusion matrix

1 Introduction

Imbalanced classification problems have been a major challenge for neural network learning in recent decades. Since most learning methods are based on global error objective functions,

✉ Gustavo Matheus de Almeida
galmeida@deq.ufmg.br

[1] Federal University of Minas Gerais, Belo Horizonte, MG, Brazil

induced models tend to inherit imbalance that is contained in the data. There are many methods for dealing with this problem, which can be grouped into three categories, namely sampling procedures, ensemble learning and cost-sensitive functions. Reviews on them can be found in [1–10], to mention a few. Sampling methods refer mainly to the under-/over-sampling of the imbalanced data set, having *Synthetic Minority Over-sampling Technique (SMOTE)* [9], *Weighted Wilson's Editing (WWE)* [11] and *Adaptive Synthetic Sampling (ADASYN)* [12] as their most popular representatives. Ensemble learning consists of a combination of learning algorithms [13, 14], which also appear in other contexts, but in this case aim to compensate performance on individual classes by aggregating classifiers. The work presented in this paper refer to a cost-sensitive method [1, 2, 5, 15, 16], since it is based on new cost functions that aim to compensate for the imbalance in the data.

Classifier evaluation, after learning with error functions, is given by additional performance metrics that estimate the response among all classes [17, 18]. Perhaps the greatest difficulty with the class imbalance problem is that such metrics are only evaluated after training, despite being the ultimate learning goal. Learned models that do not satisfy the performance metric for the imbalanced class are often retrained. This situation is due to the fact that most learning algorithms are based on the inductive principle of global error minimization, which are easier to implement and manipulate analytically. This situation prints an empirical component in the entire learning process, since the learned models can not be changed by the performance metric after obtaining its parameters. It seems reasonable, therefore, that metrics should also be considered to induce the set of parameters in the learning phase, aiming for models closer to the final performance goal without the need for retraining.

The following are examples of works related to strategies for dealing with the imbalanced classification problem. Durden et al. [19] investigated the impact of the sizes of the training and validation datasets on the performance of a convolutional neural network classifier given the imbalance problem. This application refereed to the classification of marine fauna images. Using a residual neural network, Langenkämper et al. [20] showed that the problem of concept drift in seafloor fauna images was less important than the amount of training data in the context of imbalance. Slightly shifted visual characteristics in images of the same class occur, for example, due to the use of different imaging systems. In another work, Langenkämper et al. [21] investigated the use of over-/under-sampling methods combined with data augmentation for the class imbalance problem in marine image data using convolutional neural networks. Mellor et al. [22] analyzed the effect of data imbalance on classification accuracy for land cover classification. The authors employed an ensemble learning classifier based on random forests using a margin criteria in the confusion matrix.

The main contribution of this work is to shed light on how performance metrics can be used not only in the evaluation of already trained classifiers, but also as loss functions for training [2]. In other words, instead of considering the metric just as a post-training validation criterion, it is considered as the objective function and is effectively used in the training phase. As performance metrics are often drawn from the confusion matrix, we show how to consider them as loss functions and also how to implement them with gradient descent learning. Instead of using accuracy as a performance metric, which may favor the majority class, other metrics, also often used as post-learning metrics, were adopted in this work [23–26]. Namely, AUC (area under the ROC curve) [27], Kubat's G-mean (Geometric mean) [28], F1-score [29] and AG-mean (Adjusted Geometric-mean) [30]. This approach is capable of compensating for performance and computation time when compared to classical algorithms. The following algorithms, which are representative of the most common strategies to deal with the imbalanced classification problem, were considered for comparison purposes: SMOTE (Synthetic Minority Oversampling Technique) [9], an oversampling method; SMTTL (SMOTE

Table 1 Confusion matrix

	Predicted Positive	Predicted Negative
Positive label	True Positive (TP)	False Negative (FN)
Negative label	False Positive (FP)	True Negative (TN)

Table 2 Approximate confusion matrix

	Predicted Positive	Predicted Negative
Positive label	$\sum_{i=1}^{n} y_i \cdot \hat{y}_i$	$\sum_{i=1}^{n} y_i \cdot (1 - \hat{y}_i)$
Negative label	$\sum_{i=1}^{n} (1 - y_i) \cdot \hat{y}_i$	$\sum_{i=1}^{n} (1 - y_i) \cdot (1 - \hat{y}_i)$

$y = 0$: Positive class label
$y = 1$: Negative class label

+ Tomek Links) [31], an oversampling method with pruning; WWE (Weighted Wilson's Editing) [11], an undersampling method; RAMOBoost (Rated Minority Oversampling in Boosting) [32], an ensemble method; and the Rprop [33], an error minimization learning algorithm for training MLPs, with the cross-entropy loss function.

Section 2 describes the formulation of objective functions based on performance metrics, which are derived from the confusion matrix. Section 3 shows the corresponding derivation of the backpropagation error term to be used during network training. Section 4 presents the results obtained for a series of classical benchmark datasets, which are compared to commonly used methods for imbalanced data. Final considerations are given in Sect. 5.

2 Approximate Performance Metrics

This section presents a general formulation for directly using performance metrics as cost-sensitive objective functions. The starting point for this is the confusion matrix, shown in Table 1, from which most metrics are obtained. The formulation is shown for AUC, G-mean, F1-score and AG-mean, which are widely used to evaluate imbalanced classification problems [23–26]. This formulation can readily be extended to any performance metric based on the confusion matrix.

Considering a binary classification problem, for which targets $y \in \mathcal{Y} = \{0, 1\}$ and model estimates $\hat{y} \in \mathcal{R} = [0, 1]$, the confusion matrix can be approximated in terms of y and $\hat{y}$ according to Table 2, where n is the number of samples.

In sequence, each performance metric is rewritten using the elements from the previous approximate confusion matrix. Table 3 shows the resulting representation for them. Next section shows how to obtain the corresponding backpropagation error terms.

3 Backpropagation Error Term for the Approximate Performance Metrics

Backpropagation equations can be obtained directly from the cost functions presented in Table 3. In fact, the change in the objective function only affects the backpropagation error term (δ^n), since the other weight update terms remain unaltered. The update equation can be obtained by applying the gradient descent algorithm to $\partial J(\theta)/\partial \theta^{n-1}$) and $\partial J(\theta)/\partial z^{n-1}$ as

Table 3 Performance metrics in terms of the approximate confusion matrix

Metrics	Representation	
	Formula	Approximate Confusion Matrix
AUC [34]	$\frac{1}{2} \cdot (1 + TPr - FPr)$	$\frac{1}{2} \cdot \left(1 + \frac{\sum_{i=1}^{n} y_i \cdot \hat{y}_i}{N_p} - \frac{\sum_{i=1}^{n} (1-y_i) \cdot \hat{y}_i}{N_n}\right)$
G-mean	$\sqrt{TPr \cdot TNr}$	$\sqrt{\frac{\sum_{i=1}^{n} y_i \cdot \hat{y}_i}{N_p} \cdot \frac{\sum_{i=1}^{n} (1-y_i) \cdot (1-\hat{y}_i)}{N_n}}$
F1-score	$\frac{2TP}{2TP+FP+FN}$	$\frac{2\sum_{i=1}^{n} y_i \cdot \hat{y}_i}{2\sum_{i=1}^{n} y_i \cdot \hat{y}_i + \sum_{i=1}^{n} (1-y_i) \cdot \hat{y}_i + \sum_{i=1}^{n} y_i \cdot (1-\hat{y}_i)}$
AG-mean	$\frac{(\sqrt{TPr \cdot TNr}) + (TNr \cdot P_n)}{1+P_n}$	$\frac{\left(\sqrt{\frac{\sum_{i=1}^{n} y_i \cdot \hat{y}_i}{N_p} \cdot \frac{\sum_{i=1}^{n} (1-y_i) \cdot (1-\hat{y}_i)}{N_n}}\right) + \left(\frac{\sum_{i=1}^{n} (1-y_i) \cdot (1-\hat{y}_i)}{N_n} \cdot P_n\right)}{1+P_n}$

N_p: Number of positive samples ($\sum_{i=1}^{n} y_i$)
N_n: Number of negative samples ($\sum_{i=1}^{n} (1 - y_i)$)
TPr: True Positive rate (TP/N_p)
TNr: True Negative rate (TN/N_n)

shown in Eqs. 1 and 2, respectively. The resulting backpropagation error term is given in Eq. 3. Rprop [33] was used for implementing gradient descent.

$$\frac{\partial J(\theta)}{\partial \theta^{(n-1)}} = \delta^n a^{(n-1)} \tag{1}$$

$$\frac{\partial J(\theta)}{\partial z^{(n-1)}} = \delta^{(n)} \theta^{(n-1)} g(z^{(n-1)})(1 - g(z^{(n-1)})) = \delta^{(n-1)} \tag{2}$$

$$\delta^n = \frac{\partial J(\theta)}{\partial \hat{y}} = \frac{\partial J(\theta)}{\partial g(z^{(n)})} \tag{3}$$

where:

θ : Network weights
$J(\theta)$: Loss function
n: Layer number
a: Neuron ouput
$g(\cdot)$: Activation function
z: Neuron input

The metrics considered in this work (AUC, G-mean, F1-score and AG-mean) are in the range [0, 1], where 1 represents the best performance, therefore, the derived functions (Table 3) should all be maximized. Also, although they are differentiable, there can be gradient convergence problems, hence, the negative logarithm should be used instead (Eq. 4).

$$\delta^n = \frac{\partial[-log(J(\theta))]}{\partial \hat{y}} \tag{4}$$

Equations 5, 6, 7 and 8 show how to obtain the error term for the loss functions proposed in Table 3. Sigmoid functions were used as activation of output neurons, that is, $\hat{y}_i = g(z_i^{(n)}) = 1/(1 + exp^{-z_i^{(n)}})$. The formulation is general and other activation functions can also be used.

$$\delta^n = \frac{\partial\left[-log\cdot\left(\frac{1}{2}\cdot\left(1+\frac{\sum_{i=1}^{n} y_i\cdot\hat{y}i}{N_p}-\frac{\sum i=1^n(1-y_i)\cdot\hat{y}_i}{N_n}\right)\right)\right]}{\partial\hat{y}_i} \tag{5}$$

$$\delta^n = \frac{\partial\left[-log\cdot\left(\sqrt{\frac{\sum_{i=1}^{n} y_i\cdot\hat{y}i}{N_p}\cdot\frac{\sum i=1^n(1-y_i)\cdot(1-\hat{y}_i)}{N_n}}\right)\right]}{\partial\hat{y}_i} \tag{6}$$

$$\delta^n = \frac{\partial\left[-log\cdot\left(\frac{2\sum_{i=1}^{n} y_i\cdot\hat{y}i}{2\sum i=1^n y_i\cdot\hat{y}i+\sum i=1^n(1-y_i)\cdot\hat{y}i+\sum i=1^n y_i\cdot(1-\hat{y}_i)}\right)\right]}{\partial\hat{y}_i} \tag{7}$$

$$\delta^n = \frac{\partial\left[-log\cdot\left(\frac{\left(\sqrt{\frac{\sum_{i=1}^{n} y_i\cdot\hat{y}i}{N_p}\cdot\frac{\sum i=1^n(1-y_i)\cdot(1-\hat{y}i)}{N_n}}\right)+\left(\frac{\sum i=1^n(1-y_i)\cdot(1-\hat{y}_i)}{N_n}\cdot P_n\right)}{1+P_n}\right)\right]}{\partial\hat{y}_i} \tag{8}$$

Next step refers to obtaining the derivative of the negative logarithm of the loss functions. The resulting backpropagation error terms associated with AUC, G-mean, F1-score and AG-mean are shown in the following sections.

3.1 AUC

One of the most important goals in imbalanced classification problems concerns the correct classification of the minority class without compromising the performance of the majority class. This is the case, for example, with the medical diagnosis of rare events. The AUC (area under the ROC curve) metric is extremely useful for this purpose, as it equates to the probability of labeling a positive instance (rare event) with greater confidence than a negative one [27]. Due to its importance, this metric has been widely used to evaluate classifiers mainly in case of unbalanced data. Differentiable approximations of the Wilcoxon-Mann-Whitney statistic, which is equivalent to AUC, were developed in previous works [35–37]. In this work, the objective function associated with this metric is based on the confusion matrix as shown in Table 3. Equation 9 presents the corresponding backpropagation error term.

$$\delta^n = -\frac{\frac{y_i}{N_p}-\frac{1-y_i}{N_n}}{1+\frac{\sum_{i=1}^{n} y_i\cdot\hat{y}_i}{N_p}-\frac{\sum_{i=1}^{n}(1-y_i)\cdot\hat{y}_i}{N_n}} \tag{9}$$

3.2 G-mean

Obtaining satisfactory scores for minority and majority classes simultaneously is a challenge in binary classification. The geometric mean metric (G-mean) appeared in this context taking into account the accuracy of both classes [28, 38]. This metric has been widely applied to imbalanced classification problems with the main objective of finding a good balance between TP and TN rates [34, 39–43]. The resulting backpropagation error term for the loss function associated with G-mean (Table 3) is given in Eq. 10.

$$\delta^n = -\frac{1}{2}\left(\frac{y_i}{\sum_{i=1}^{n} y_i\cdot\hat{y}_i}+\frac{y_i-1}{\sum_{i=1}^{n}(1-y_i)\cdot(1-\hat{y}_i)}\right) \tag{10}$$

3.3 F-score

When negative cases are not in the center of a classifier's performance, precision and recall are often used measures. The F1-score metric combines the two [39] (Eq. 11). This metric appeared in the context of information retrieval, where samples are positive if they contain attributes of interest [38], in order to compensate for precision and recall [44, 45]. The β parameter controls the balance between them, with $\beta > 1$ favoring recall, and otherwise, precision [46].

$$F\text{-}score = \frac{(1+\beta^2)\cdot \text{recall}\cdot \text{precision}}{\beta^2\cdot \text{precision} + \text{recall}} \tag{11}$$

A commonly used value for β is 1, which results in the harmonic mean between precision and recal [45] (Eq. 12). In this particular case, it is called F1-score or F1-measure. This metric has been applied to many imbalanced classification problems with different classifiers like SVM [47] and logistic regression [48]. The Empirical Utility Maximization (EUM) [49] and the General F-measure Maximize (GFM) [50] algorithms were used in previous works to approximate and maximize F-score. The corresponding backpropagation error term for the loss function associated with F1-score (Table 3) is given in Eq. 13.

$$F1\text{-}score = \frac{2\cdot \text{recall}\cdot \text{precision}}{\text{precision} + \text{recall}} \tag{12}$$

$$\delta^n = -\frac{y_i}{\sum_{i=1}^n y_i\cdot \hat{y}_i} + \frac{1-y_i}{\sum_{i=1}^n (1-y_i)\cdot \hat{y}_i + \sum_{i=1}^n y_i\cdot \hat{y}_i + \sum_{i=1}^n y_i\cdot (1-\hat{y}_i)} \tag{13}$$

3.4 AG-mean

Many practical situations require maximizing the classification of positive samples as much as possible while keeping the majority class rating to a minimum. This is the case with medical diagnosis, where false positives can be very undesirable. However, increasing the scores of both classes simultaneously are conflicting goals [51]. The adjusted G-mean metric (AG mean) was proposed in this context, focusing on the positive class, through the balance between sensitivity (SE) and specificity (SP) [30]. Given its importance for class imbalanced problems, this work proposes its direct use as a loss function for MLP training (Table 3). Equation 14 shows the corresponding backpropagation error term.

$$\delta^n = \frac{\frac{(y_i-1)\cdot P_n}{N_n} + \frac{\frac{y_i}{N_p}\cdot \frac{\sum_{i=1}^n (1-y_i)\cdot(1-\hat{y}_i)}{N_n} + \frac{(y_i-1)}{N_n}\cdot \frac{\sum_{i=1}^n y_i\cdot \hat{y}_i}{N_p}}{2\cdot\sqrt{\frac{\sum_{i=1}^n y_i\cdot \hat{y}_i}{N_p}\cdot \frac{\sum_{i=1}^n (1-y_i)\cdot(1-\hat{y}_i)}{N_n}}}}{\sqrt{\frac{\sum_{i=1}^n y_i\cdot \hat{y}_i}{N_p}\cdot \frac{\sum_{i=1}^n (1-y_i)\cdot(1-\hat{y}_i)}{N_n}} + P_n\cdot \frac{\sum_{i=1}^n (1-y_i)\cdot(1-\hat{y}_i)}{N_n}} \tag{14}$$

4 Results and Discussions

This section presents the results of directly considering AUC, G-gmean, F1-score and AG-mean as cost functions for training MLPs. These metrics were considered given the focus of this work on imbalanced classification problems. To make this analysis broader, sixteen

Table 4 Datasets characteristics

Database	Alias	Number of features	n_1	n_2	$n_1/(n_1+n_2)$
Ionosphere	iono	34	126	225	0.359
Pima Indians Diabetes	pid	08	268	500	0.349
German Credit	gmn	24	300	700	0.300
WP Breast Cancer	wpbc	33	47	151	0.237
Vehicle (4 versus all)	veh	18	199	647	0.235
SPECTF Heart	hrt	44	55	212	0.206
Segmentation (1 versus all)	seg	19	30	180	0.143
Glass (7 versus all)	gls7	10	29	185	0.136
Euthyroid (1 versus all)	euth	24	238	1762	0.119
Satimage (4 versus all)	sat	36	626	5809	0.097
Vowel (1 versus all)	vow	10	90	900	0.091
Abalone (18 versus 9)	a18-9	08	42	689	0.057
Yeast (9 versus 1)	y9-1	08	20	463	0.041
Car (3 versus all)	car	06	69	1659	0.040
Yeast (5 versus all)	y5	08	51	1433	0.034
Abalone (19 versus all)	a19	08	32	4145	0.008

n_1: Number of samples in minority class
n_2: Number of samples in majority class
$n_1/(n_1+n_2)$: Imbalance ratio

classical datasets presenting different imbalance ratios were used [52]. They are summarized in Table 4. All were preprocessed according to [1]. For example, the first dataset, called Ionosphere, has two classes, one with 126 ($n1$) and one with 225 ($n2$) samples, which results in an imbalance ratio ($n_1/(n_1+n_2)$) of 0.359.

Since MLPs are not based on explicit class density estimation for learning, margin information is more relevant than sample size. Considering that network training in this work is based on metrics that are less sensitive to imbalance, the neural network is able to learn even with small sample sizes. This is also due to the properties of the cost functions used (AUC, G-mean, F1-score and AG-mean), that are capable to balance majority and minority classes, regardless of their sizes.

About neural network topology, activation functions for all network units were sigmoidal [1]. Given the binary classification approach using the OvA (one-versus-all) procedure, the cutoff value between classes was equal to 0.5. Rprop was used to implement gradient descent as the learning algorithm in all cases [33]. The number of hidden units was defined by means of a cross validation procedure as follows. Each dataset was initially divided into ten subsets, which were subdivided into ten training (70%) and test (30%) sets. These sets were used for model selection, that is, to define the number of hidden units, through a 5−fold cross-validation procedure. Next, they were then combined for model re-identification, given the selected number of hidden units. Model performance was evaluated with the previous ten test sets.

Our proposal was then compared with commonly used methods for handling imbalanced classes: Rprop [33], SMOTE [9], SMTTL (SMOTE + Tomek Links) [31], WWE (Weighted Wilson's Editing) [11], RAMOBoost (Ranked Minority Oversampling in Boosting) [32].

Table 5 Average AUC values (in %)

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	AUC-MLP
iono	85.66	87.66	88.56	89.15	**89.62**	80.40
pid	66.26	69.01	70.44	71.98	72.38	**74.34**
gmn	60.02	62.40	65.64	63.64	64.99	**69.36**
wpbc	56.25	59.52	56.55	52.48	**60.61**	60.51
veh	95.26	96.16	97.36	96.79	**97.72**	96.43
hrt	61.32	60.53	63.66	**70.08**	65.46	69.86
seg	99.57	99.72	99.69	99.55	**99.80**	98.71
gls7	88.69	91.64	91.28	**92.99**	92.45	91.24
euth	84.80	89.15	86.83	88.23	88.75	**90.87**
sat	76.66	78.16	78.09	**79.80**	77.71	77.21
vow	98.72	97.39	95.88	95.43	**99.39**	96.77
a18-9	71.83	**82.12**	81.69	70.80	78.39	82.03
y9-1	72.32	67.31	73.01	69.88	71.90	**78.07**
car	93.25	90.78	91.71	**97.09**	94.05	95.73
y5	61.21	71.85	71.17	71.35	65.85	**79.34**
a19	53.80	76.91	76.90	57.57	73.73	**83.06**
Average rank	5.06	3.56	3.50	3.50	2.63	2.75

Cross-entropy was used as the loss function in all cases. The comparative analysis employed the classical non-parametric Nemenyi and Bonferroni-Dunn tests, which allows a comparison considering all datasets at once, given a performance metric [53–55].

Tables 5, 7, 9 and 11 show the average performance for AUC, G-mean, F1-score and AG-mean, respectively. The average computation time (in seconds) involving training and testing is presented respectively in Tables 6, 8, 10 and 12. The results for MLP, SMOTE, SMTTL, WWE and RAMOBoost, are also shown for comparison purposes. The results related to the proposed objective functions are referred to as AUC-MLP, G-MLP, F-MLP and AG-MLP, respectively. The best performance in each experiment is highlighted in bold.

4.1 Non-parametric Tests

The Wilcoxon statistical test is applied to compare pairs of classifiers [55, 56], while the Nemenyi post-hoc test (Eq. 15), based on Friedman statistic (Eq. 16) [57], is used in case there are multiple classifiers. The Friedman statistic considers applying L classifiers to M datasets. Its is based on average ranks (R_j), where $R_j = \frac{1}{M}\sum_{i=1}^{M} r_i^j$, $1 \leq j \leq L$, is the average rank of the j^{th} classifier given all datasets. The null hypothesis (H_0) of the Nemenyi test states that all classifiers perform similarly, that is, their average ranks are close. Such values are shown in the last row of Tables 5, 7, 9 and 11, which are relative to the performance metric, and of Tables 6, 8, 10 and 12, for the computation time. When the null hypothesis is rejected, another statistical test should be performed to quantify the difference between the classifiers [55]. The most used test in this case is the Bonferroni-Dunn post-hoc test [58]. Two classifiers are not considered similar if the difference between their average ranks is greater

Table 6 Average computation time (in seconds) for AUC

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	AUC-MLP
iono	1.370	1.608	2.039	**1.334**	36.793	1.376
pid	1.359	1.966	1.984	1.740	37.806	**1.323**
gmn	2.022	2.089	2.380	2.403	50.548	**1.924**
wpbc	1.141	1.150	1.073	**0.980**	24.202	1.141
veh	**1.446**	1.785	3.862	2.103	56.437	1.867
hrt	1.387	2.190	3.197	1.292	73.011	**1.246**
seg	**2.393**	3.817	16.486	5.881	176.462	3.238
gls7	1.023	1.053	**1.007**	1.022	22.875	1.042
euth	**2.662**	2.868	6.353	4.193	101.608	2.895
sat	11.051	12.379	16.490	31.102	531.254	**6.542**
vow	**1.765**	1.771	1.975	1.962	58.863	1.941
a18-9	**1.050**	1.059	1.379	1.257	27.063	1.353
y9-1	**1.218**	1.540	2.104	1.228	32.348	1.402
car	2.297	3.883	7.565	2.974	311.684	**1.872**
y5	2.528	**1.647**	2.306	2.879	57.395	2.213
a19	4.534	**2.767**	18.052	11.981	147.837	4.002
Average rank	2.06	3.00	4.31	3.25	6.00	2.37

Table 7 Average G-mean values (in %)

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	G-MLP
iono	83.86	89.19	88.14	86.14	**90.57**	78.84
pid	72.07	76.17	76.00	**76.45**	74.23	76.37
gmn	67.16	**70.97**	69.37	67.28	67.13	68.77
wpbc	54.10	50.23	**62.26**	56.46	61.35	59.07
veh	95.53	96.72	**97.41**	96.71	97.27	97.07
hrt	55.97	65.57	66.06	**76.21**	66.03	70.31
seg	99.57	99.44	99.54	99.57	**99.95**	98.93
gls7	92.19	90.31	**93.01**	91.39	92.45	91.40
euth	90.67	90.20	90.46	91.08	88.07	**91.43**
sat	73.16	76.36	76.21	79.22	74.27	**85.96**
vow	99.11	98.13	98.81	98.48	**100.00**	99.55
a18-9	71.91	82.54	82.48	80.85	74.01	**82.91**
y9-1	64.78	67.70	54.79	68.08	71.10	**73.86**
car	94.58	**97.62**	92.98	98.08	94.58	97.52
y5	54.44	80.91	**81.11**	70.64	61.96	78.85
a19	10.49	**84.49**	84.46	25.26	34.48	80.32
Average rank	4.88	3.56	3.06	3.25	3.50	2.75

Table 8 Average computation time (in seconds) for G-mean

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	G-MLP
iono	1.427	1.575	1.746	1.281	39.170	**1.192**
pid	**1.106**	1.076	1.437	1.201	31.527	1.147
gmn	**1.270**	1.379	1.888	1.642	31.603	1.872
wpbc	1.054	1.199	**1.036**	1.061	25.348	1.073
veh	**1.273**	1.472	3.595	1.917	53.847	1.658
hrt	**1.200**	2.376	3.153	1.133	33.443	1.214
seg	2.518	4.799	14.498	5.221	172.374	**1.672**
gls7	0.985	**0.842**	1.021	0.910	22.144	0.912
euth	**1.638**	2.301	5.805	3.471	83.363	2.694
sat	**9.541**	11.872	14.135	29.844	428.639	9.858
vow	1.956	**1.840**	2.161	1.964	54.982	1.853
a18-9	1.134	1.104	1.484	1.233	31.856	**1.043**
y9-1	1.226	1.376	1.739	1.241	23.452	**1.049**
car	**1.990**	2.809	5.658	3.431	232.341	2.199
y5	2.237	**1.289**	1.825	2.734	47.847	1.972
a19	3.337	**2.805**	18.315	11.947	522.545	2.863
Average rank	2.19	2.44	4.44	3.50	6.00	2.44

Table 9 Average F1-score values (in %)

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	F-MLP
iono	83.51	**85.95**	83.06	85.20	88.60	77.91
pid	65.11	69.17	**69.65**	68.98	64.11	66.42
gmn	55.58	59.51	**60.07**	57.16	53.19	57.66
wpbc	30.36	36.90	39.74	31.87	**46.49**	42.03
veh	92.09	92.44	92.55	92.02	**94.29**	92.30
hrt	47.33	43.43	42.20	52.40	**54.00**	52.17
seg	97.89	**98.51**	99.40	98.20	99.70	98.47
gls7	**87.67**	76.88	77.63	75.50	79.27	80.13
euth	80.56	76.08	77.97	80.16	79.86	**82.59**
sat	57.93	56.22	56.91	57.48	**64.30**	62.43
vow	95.27	91.94	96.74	93.04	**99.44**	95.01
a18-9	**68.49**	45.24	38.29	68.10	51.80	64.35
y9-1	46.97	37.04	35.62	45.69	53.33	**58.89**
car	**91.61**	86.11	82.00	71.87	89.22	84.47
y5	22.76	34.34	35.82	30.10	**46.16**	40.22
a19	4.17	**10.46**	4.64	9.81	5.02	6.36
Average rank	3.81	3.88	3.81	4.06	2.44	3.00

Table 10 Average computation time (in seconds) for F1-score

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	F-MLP
iono	2.238	2.635	2.896	2.262	63.497	**1.968**
pid	**1.680**	1.764	2.108	1.831	55.017	1.947
gmn	**2.076**	2.301	2.753	2.394	67.442	2.434
wpbc	1.856	1.898	**1.606**	1.691	37.363	1.771
veh	**2.695**	3.546	4.877	2.775	102.566	3.080
hrt	2.172	4.297	5.158	**1.976**	95.999	2.049
seg	3.397	7.898	17.590	6.072	228.785	**2.806**
gls7	1.730	1.514	1.598	1.610	39.222	**1.481**
euth	**2.689**	3.401	7.896	4.341	180.492	3.986
sat	**15.741**	16.405	19.713	34.682	707.848	18.664
vow	3.285	3.211	3.371	3.305	75.572	**3.159**
a18-9	1.988	2.120	2.747	**1.823**	64.545	1.953
y9-1	**1.587**	2.843	2.741	1.582	76.564	1.911
car	3.538	6.751	10.312	5.039	511.339	**3.055**
y5	3.543	2.643	3.442	4.137	88.564	**3.071**
a19	5.857	10.774	24.958	15.174	507.426	**3.810**
Average rank	2.31	3.13	4.38	3.00	6.00	2.19

Table 11 Average AG-mean values (in %)

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	AG-MLP
iono	90.32	90.71	90.35	89.91	**92.38**	86.85
pid	75.10	70.94	73.14	75.53	71.71	**76.92**
gmn	69.03	67.45	68.99	72.90	70.94	**73.12**
wpbc	65.99	66.31	66.14	65.17	**70.28**	69.69
veh	**96.95**	96.60	96.46	96.23	97.48	96.73
hrt	73.42	71.87	70.04	73.17	**74.19**	69.57
seg	99.60	99.56	99.84	99.51	**99.86**	99.50
gls7	92.73	93.06	92.65	92.96	**93.89**	93.73
euth	92.96	92.43	92.35	**93.39**	92.39	93.38
sat	84.89	84.41	85.15	**86.57**	85.46	85.72
vow	98.59	99.00	98.81	98.43	**99.86**	96.22
a18-9	87.50	89.26	89.16	**89.84**	82.88	87.48
y9-1	83.05	76.97	79.02	80.45	82.42	**85.80**
car	94.56	96.65	95.94	96.01	**97.67**	97.52
y5	72.84	**83.94**	84.19	78.14	76.55	83.78
a19	52.55	76.35	76.63	62.39	69.45	**77.41**
Average rank	4.00	3.75	3.94	3.63	2.63	3.06

Table 12 Average computation time (in seconds) for AG-mean

Base	Rprop	SMOTE	SMTTL	WWE	RAMOBoost	AG-MLP
iono	0.954	1.199	1.507	**0.863**	26.710	0.900
pid	**0.843**	0.925	1.347	0.923	25.072	1.131
gmn	**1.075**	1.297	2.112	1.324	34.804	1.223
wpbc	0.973	0.930	0.937	**0.714**	17.089	0.883
veh	1.279	1.696	4.452	1.814	45.801	**1.329**
hrt	0.956	1.710	2.880	**0.810**	53.612	0.918
seg	2.633	4.311	14.145	4.765	126.171	**1.572**
gls7	0.700	0.689	0.731	0.712	16.568	**0.681**
euth	**1.218**	1.890	5.682	2.941	66.049	1.556
sat	9.280	9.431	14.688	29.389	469.620	**7.745**
vow	1.762	2.042	2.083	2.528	40.000	**1.368**
a18-9	**0.724**	0.743	1.057	0.892	27.726	0.926
y9-1	1.023	1.400	2.009	1.141	39.244	**1.022**
car	1.705	2.534	6.423	2.722	220.024	**1.280**
y5	1.648	**1.049**	1.875	2.709	70.839	2.021
a19	2.775	**2.427**	18.402	11.711	250.892	2.455
Average rank	2.13	2.81	4.69	3.38	6.00	2.00

than a critical difference CD (Eq. 17), where q_α is the critical value at the significance level given the Student statistic.

$$F_F = \frac{(M-1)\chi_F^2}{M(L-1)-\chi_F^2} \tag{15}$$

$$\chi_F^2 = \frac{12M}{L(L+1)}\left(\sum_{j=1}^{L} R_j^2 - \frac{L(L+1)^2}{4}\right) \tag{16}$$

$$CD = q_a\sqrt{\frac{L(L+1)}{6M}} \tag{17}$$

Given the average rank values, $M = 16$ (number of datasets) and $L = 6$ (number of classifiers), the Friedman statistic (F_F) is equal to 4.1235, 2.7496, 1.9697 and 1.3823, for Tables 5, 7, 9 and 11 (for performance metric), respectively, and to 22.7953, 26.0758, 22.7104 and 35.3748, for Tables 6, 8, 10 and 12 (for computation time), respectively. The critical value for the Nemenyi statistic is given by $F_F((L-1) = 5; (L-1)(M-1) = 75; \alpha = 0.01) = 1.9256$ [59]. Thus, the null hypothesis was rejected for all metrics except for AG-mean[1], and for all computation times. Next, the Bonferroni-Dunn post-hoc test was applied to verify the performance of the proposed approaches in relation to all other classifiers (strategy one-versus-all). Tables 13, 14, 15 and 16 show the differences between the average rank values for AUC, G-mean, F1-score and AG-mean, respectively. They also show the results for computation time. The critical difference (CD) is equal to 1.7125. Differences beyond this critical value, which point out to different performances, are highlighted in bold.

[1] Still, the post-hoc test was also computed for AG-mean.

Table 13 Classifiers comparison with AUC-MLP (using the Bonferroni-Dunn post-hoc test)

AUC-MLP versus					
	Rprop	SMOTE	SMTTL	WWE	RAMOBoost
AUC	**2.3125**	0.8125	0.7500	0.7500	0.1250
Time	0.3125	0.6250	**1.9375**	0.8750	**3.6250**

Table 14 Classifiers comparison with G-MLP (using the Bonferroni-Dunn post-hoc test)

G-MLP versus					
	Rprop	SMOTE	SMTTL	WWE	RAMOBoost
G-mean	**2.1250**	0.8125	0.3125	0.5000	0.7500
Time	0.2500	0.0000	**2.0000**	1.0625	**3.5625**

Table 15 Classifiers comparison with F-MLP (using the Bonferroni-Dunn post-hoc test)

F-MLP versus					
	Rprop	SMOTE	SMTTL	WWE	RAMOBoost
F1-score	0.8125	0.8750	0.8125	1.0625	0.5625
Time	0.1250	0.9375	**2.1875**	0.8125	**3.8125**

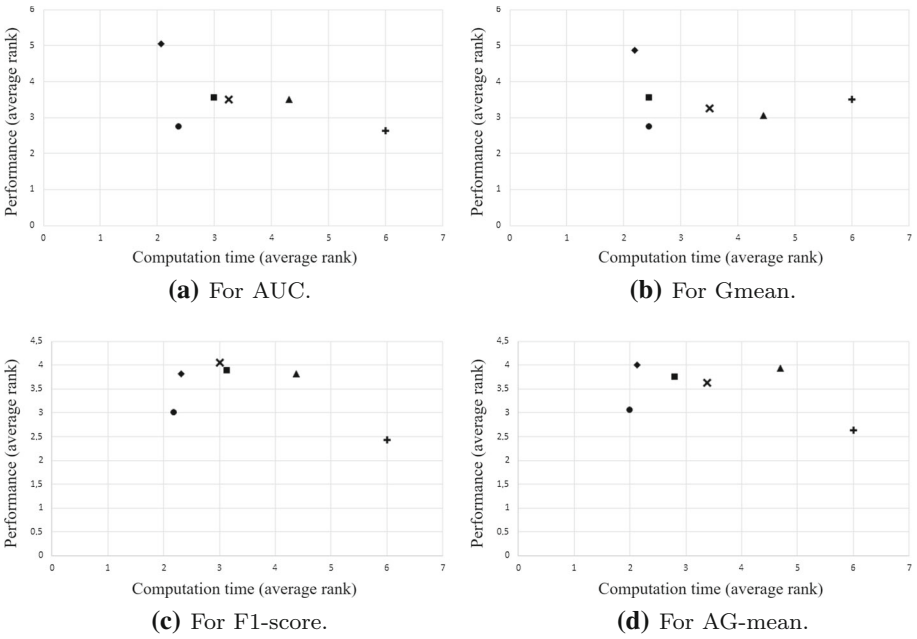
Table 16 Classifiers comparison with AG-MLP (using the Bonferroni-Dunn post-hoc test)

AG-MLP versus					
	Rprop	SMOTE	SMTTL	WWE	RAMOBoost
AG-mean	0.9375	0.6875	0.8750	0.5625	0.4375
Time	0.1250	0.8125	**2.6875**	1.3750	**4.0000**

4.2 Discussions

The results for AUC-MLP are summarized in Tables 5 (metric performance) and 6 (computation time), with average rank values equal to 2.75 and 2.37, respectively. It can be seen that its performance is very close to that of RAMOBoost, which had the best average rank value (2.63), and mainly surpass the standard Rprop (5.06). Regarding computation time, AUC-MLP is comparable to Rprop, which presented the best average rank value (2.06), and much higher than RAMOBoost (6.00). According to the Bonferroni-Dunn post-hoc test (Table 13), its performance is statistically comparable to RAMOBoost and sampling strategies, namely SMOTE, SMTTL and WWE, and superior to Rprop. Furthermore, its computation time is competitive with Rprop, SMOTE and WWE, and statistically better than RAMOBoost and SMTTL. In summary, AUC-MLP is statistically equivalent to SMOTE and WWE, but has higher average ranks for performance and computation time.

The results for G-MLP are presented in Tables 7 and 8. They show that it is best for performance and second best for computation time, with average rank values of 2.75 and 2.44, respectively. The best computation times were obtained by Rprop and SMOTE, with average ranks of 2.19 and 2.44, respectively. The Bonferroni-Dunn post-hoc test (Table 14) indicates that G-MLP outperforms Rprop and is similar to SMOTE, SMTTL, WWE and RAMOBoost. In terms of computation time, it is faster than STMTTL and RAMOBoost, and equivalent to Rprop, SMOTE and WWE. G-MLP is statistically similar to SMOTE and WWE, however when ranks are considered, it performs slightly better than both. Also, its



(a) For AUC. **(b)** For Gmean.

(c) For F1-score. **(d)** For AG-mean.

Fig. 1 Average rank plots considering computation time (x-axis) *versus* performance metric (y-axis) simultaneously. (◊: Rprop, □: SMOTE, △: SMTTL, ×: WWE, +: RAMOBoost, ○: This work)

computation time is equivalent to that of SMOTE, and slightly better than that of WWE, despite being statistically equivalent.

It can be observed that the performance of the F-MLP is competitive with the other classifiers (Table 9), as it presents the second best average rank, equal to 3.00. The best value, for RAMOBoost, is 2.44. Regarding the computation time (Table 10), it proved to be advantageous, as it presented the best average rank value, equal to 2.19. From the Bonferroni-Dunn post hoc test (Table 15), it can be seen that the proposal is statistically equivalent to all other classifiers in terms of model performance. This may be due to the fact that F1-score is a difficult convergence function [48–50, 60]. The number of iterations during model training may also have influence. Regarding computation time, it was similar to Rprop, SMOTE and WWE, and faster than SMTTL and RAMOBoost.

AG-MLP resulted in the second best performance, after RAMOBoost, with average rank values of 3.06 and 2.63, respectively (Table 11). RAMOBoost, however, had the highest computation time than all other classifiers (Table 12), in contrast with AG-MLP which resulted in the shortest computation time with an average rank equal to 2.00. Rprop comes next with 2.13, while RAMOBoost comes with 6.00. According to the Nemenyi post-hoc test, all classifiers performed similarly, as the null hypothesis was not rejected. This result can also be seen in Table 16. Despite the statistically equivalent performance of AG-MLP to Rprop, it can be observed that the proposed approach performs better in 10 out of the 16 datasets (Table 11) and presents a better average rank, 2.63 against 4.00. Regarding computation time, AG-MLP has similar to Rprop, SMOTE and WWE, and is faster than SMTTL and RAMOBoost (Table 12).

To summarize the results previously discussed, Figure 1 presents plots of the average ranks of the computation time against the performance metric, according to Tables 5 to 12.

Each refers to a specific metric. For instance, Figure 1(a) shows the average ranks obtained for the computation time (x axis; Table 12) and performance metric (y axis; Table 5) for AUC metric. Since both time and performance rank values should be minimized, the resulting problem actually involves a trade-off between them, since one objective affects the other. For example, to reduce computation time, performance can be degraded. The optimality of the corresponding bi-objective problem is evaluated according to the non-dominated solutions given the Pareto frontier. The objective in this case, given the formulated optimization problem, is to obtain a good compromise between them. It can be observed that the results of the methods presented in this work are not dominated by the others, as they generally provided lower average rank values in both objectives. For example, considering AUC-MLP (Figure 1a), Rprop has less computation time, however, at the cost of a lower performance (that is, a higher average rank value), and RAMOBoost has slightly higher performance with much higher computation time. This is also the case for G-mean. That is, Rprop had the smallest computation time (lower rank value), but with the worst performance (higher rank value). In contrast, RAMOBoost showed a satisfactory performance, however, at the expense of more computation time. The proposal of this work for G-mean presented a relatively small computation time, with the best performance. For F1-score and AG-mean, RAMOBoost had the best performance, but with the worst computation time. Rprop presented a satisfactory computation time, however, with a worse performance in relation to the best results. The proposals of this work for F1-score and AG-mean presented the best computation time, and the second best performance, only behind RAMOBoost, which, however, presented the highest computation time. These results show that the direct use of performance metrics as loss functions results in well-balanced classifiers between performance and computation time.

5 Conclusions

This work proposes the direct use of performance metrics as cost functions for training MLP neural networks in imbalanced classification problems. Usual performance metrics, namely AUC, G-mean, F1-score and AG-mean, were addressed. The formulation is derived directly from the commonly used confusion matrix and implemented with standard backpropagation, for which the only change is to the error term.

A general experiment, employing average rank values, showed that the use of cost functions based on performance metrics outperformed both the standard RProp and all sampling strategies, namely SMOTE, SMTTL and WWE, for all metrics. These results were statistically corroborated for the AUC and the G-mean in relation to the Rprop. For F1-score and AG-mean, all algorithms were considered statistically equivalent. Also, it was superior to RAMOBoost for G-mean, given the average rank values. However, it was statistically faster than this oversampling procedure for all metrics. The proposal also showed faster computation than SMTTL and was equivalent to Rprop, SMOTE and WWE. Furthermore, it presented non-dominated solutions for the bi-objective problem when performance and computation time are evaluated simultaneously, even considering the standard Rprop. In short, the proposal generally showed higher performance for all metrics under relatively high imbalance ratios. Other methods, such as DEBOHID [61], as well as other performance metrics considering the confusion matrix, can be used. Implementations of more efficient gradient descent learning algorithms are also possible.

In conclusion, this work presented a new perspective for the imbalanced learning problem, which is usually treated as a two-step problem, as it is first trained with an objective function,

such as a global error, and then evaluated with a different metric. Direct use of performance metrics as objective functions allows to approach the problem from a single-step perspective, avoiding retraining. Lastly, the methodology proposed in this work can be extended to multiclass classification problems.

Acknowledgements The authors would like to thank the following Brazilian research funding agencies for their financial support, CNPq (The National Council for Scientic and Technological Development), FAPEMIG (The Minas Gerais Research Foundation) and CAPES (The Coordination for the Improvement of Higher Education Personnel).

Author Contributions (optional: please review the submission guidelines from the journal whether statements are mandatory): The study conception and design were performed by Antonio Padua Braga, Cristiano Leite de Castro, and Yuri Sousa Aurelio. Material preparation, data collection and results generation were carried out by Yuri Sousa Aurelio. Gustavo Matheus de Almeida contributed to the analysis of the results together with the other authors. All contributed and approved the final version of the manuscript.

Funding (information that explains whether and by whom the research was supported): This work was financially supported by the following Brazilian research funding agencies, CNPq (The National Council for Scientific and Technological Development), FAPEMIG (The Minas Gerais Research Foundation) and CAPES (The Coordination for the Improvement of Higher Education Personnel).

Availability of data and material (data transparency): All datasets used in this work are available in the public UCI Machine Learning Repository (http://archive.ics.uci.edu/ml).

Declarations

Conflicts of interest/Competing interests (include appropriate disclosures): There are no conflicts of interest in this work.

References

1. Castro CL, Braga AP (2013) Novel cost-sensitive approach to improve the multilayer perceptron performance on imbalanced data. IEEE Trans Neural Netw Learn Syst 24(6):888
2. Aurelio YS, Almeida GM, Castro CL, Braga AP (2019) Learning from imbalanced data sets with weighted cross-entropy function. Neural Process Lett 50:1937
3. Haixiang G, Yijing L, Shang J, Mingyun G, Yuanyue H, Bing G (2017) Learning from class-imbalanced data: review of methods and applications. Expert Syst Appl 73(1):220
4. Lan J, Hu MY, Patuwo E, Zhang GP (2010) An investigation of neural network classifiers with unequal misclassification costs and group sizes. Decis Support Syst 48(4):582
5. Thai-Nghe N, Gantner Z, Schmidt-Thieme L (2010) Cost-sensitive learning methods for imbalanced data, in *Proc. International Joint Conference on Neural Networks* (IEEE, 2010), pp. 1–8
6. He H, Garcia EA (2009) Learning from imbalanced data. IEEE Trans Know Data Eng 21(9):1263
7. Chawla NV, Japkowicz N, Kotcz A (2004) Editorial: special issue on learning from imbalanced data sets. ACM SIGKDD Explor Newsl 6(1):1
8. Batista GEAPA, Prati RC, Monard MC (2004) A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explor Newslett 6(1):20
9. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: synthetic minority over-sampling technique. J Artif Intell Res 16:321
10. Michie D, Spiegelhalter DJ, Taylor CC (1994) Machine learning, neural and statistical classification. Machine learning neural and statistical classification. Prentice Hall, USA
11. Barandela R, Valdovinos RM, Sánchez JS, Ferri FJ (2004) The imbalanced training sample problem: Under or over sampling?, in *Structural, Syntactic, and Statistical Pattern Recognition*, *LNCS*, vol. 3138, ed. by A. Fred, T.M. Caelli, R.P.W. Duin, A.C. Campilho, D. de Ridder (Springer, 2004), pp. 806–814
12. He H, Bai Y, Garcia EA, Li S (2008) ADASYN: Adaptive synthetic sampling approach for imbalanced learning, in *Proc IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)* (IEEE, 2008), pp. 1322–1328

13. Chen S, He H, Garcia EA (2010) RAMOBoost: ranked minority oversampling in boosting. IEEE Trans Neural Netw 21(10):1624
14. Sun Y, Kamel MS, Wong AKC, Wang Y (2007) Cost-sensitive boosting for classification of imbalanced data. Pattern Recognit 40(12):3358
15. Tao X, Li Q, Guo W, Ren C, Li C, Liu R, Zou J (2019) Self-adaptive cost weights-based support vector machine cost-sensitive ensemble for imbalanced data classification. Inform Sci 487:31
16. Zhang C, Tan KC, Li H, Hong GS (2018) A cost-sensitive deep belief network for imbalanced classification. IEEE Trans Neural Netw Learn Syst 30(1):109
17. Weiss GM (2004) Mining with rarity: a unifying framework. ACM SIGKDD Explor Newsl 6(1):7
18. Caruana R, Niculescu-Mizil A (2004) Data mining in metric space: an empirical analysis of supervised learning performance criteria, in *Proc. 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM, 2004), pp. 69–78
19. Durden JM, Hosking B, Bett BJ, Cline D, Ruhl HA (2021) Automated classification of fauna in seabed photographs: the impact of training and validation dataset size, with considerations for the class imbalance. Prog Oceanogr 196:102612
20. Langenkämper D, van Kevelaer R, Purser A, Nattkemper TW (2020) Gear-induced concept drift in marine images and its effect on deep learning classification, Frontiers in Marine Science (2020)
21. Langenkämper D, van Kevelaer R, Nattkemper TW (2019) Strategies for Tackling the Class Imbalance Problem in Marine Image Classification, in *Pattern Recognition and Information Forensics (ICPR 2018)*, vol. 11188, ed. by Z. Zhang, D. Suter, Y. Tian, A.A. Branzan, N. Sidère, E.H. Jair (Springer, 2019), vol. 11188
22. Mellor A, Boukir S, Haywood A, Jones S (2015) Exploring issues of training data imbalance and mislabelling on random forest performance for large area land cover classification using the ensemble margin. ISPRS J Photogramm Rem Sens 105:155
23. Boughorbel S, Jarray F, El-Anbari M (2017) Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. PloS one 12(6):e0177678
24. Chawla NV (2009) Data mining for imbalanced datasets: an overview, Data mining and knowledge discovery handbook pp. 875–886
25. Gu Q, Zhu L, Cai Z (2009) Evaluation measures of the classification performance of imbalanced data sets, in *International symposium on intelligence computation and applications* (Springer, 2009), pp. 461–471
26. Kuncheva LI, Arnaiz-González Á, Díez-Pastor JF, Gunn IA (2019) Instance selection improves geometric mean accuracy: a study on imbalanced data classification. Prog Artif Intell 8(2):215
27. Fawcett T (2006) An introduction to ROC analysis. Pattern Recognit Lett 27(8):861
28. Kubat M, Matwin S (1997) Addressing the curse of imbalanced training sets: One-sided selection, in *Proc 14th International Conference on Machine Learning*, vol. 97, pp. 179–186
29. Pazzani M, Billsus D (1997) Learning and revising user profiles: the identification of interesting web sites. Mach Learn 27:313
30. Batuwita R, Palade V (2012) Adjusted geometric-mean: a novel performance measure for imbalanced bioinformatics datasets learning. J Bioinform Comput Biol 10(4):1250003
31. Tomek I (1976) Two modifications of CNN IEEE transactions on systems man and cybernetics. SMC 6(11):769
32. Provost F, Fawcett T (2001) Robust classification for imprecise environments. Mach Learn 42:203
33. Riedmiller M, Braun H (1992) RPROP: A fast adaptive learning algorithm, in *Proc. ISCIS VII* (1992)
34. Hong X, Chen S, Harris CJ (2007) A kernel-based two-class classifier for imbalanced data sets. IEEE Trans Neural Netw 18(1):28
35. Castro CL, Braga AP (2008) Optimization of the Area under the ROC Curve, in *Proc. 10th Brazilian Symposium on Neural Networks* (IEEE, 2008), pp. 141–146
36. Rakotomamonjy A (2004) Optimizing area under ROC curves with SVMs, in *Proc. 1st International Workshop on ROC Analysis in Artificial Intelligence* (2004), pp. 71–80
37. Yan L, Dodier RH, Mozer MC, Wolniewicz RH (2003) Optimizing classifier performance via an approximation to the Wilcoxon-Mann-Whitney statistic, in *Proc. 20th International Conference on Machine Learning* (2003), pp. 848–855
38. Kubat M, Holte RC, Matwin S (1998) Machine learning for the detection of oil spills in satellite radar images. Mach Learn 30:195
39. Antanasijević J, Antanasijević D, Pocajt V, Trišović N, Fodor-Csorba K (2016) A QSPR study on the liquid crystallinity of five-ring bent-core molecules using decision trees. MARS Artif Neural Netw, RSC Adv 6(22):18452
40. Kim HJ, Jo NO, Shin KS (2016) Optimization of cluster-based evolutionary undersampling for the artificial neural networks in corporate bankruptcy prediction. Expert Syst Appl 59:226

41. Nguyen GH, Bouzerdoum A, Phung SL (2009) Learning pattern classification tasks with imbalanced data sets, *Learning pattern classification tasks with imbalanced data sets* (2009)
42. Xu L, Chow M, Timmis J, Taylor LS (2007) Power distribution outage cause identification with imbalanced data using artificial immune recognition system (AIRS) algorithm. IEEE Trans Power Syst 22(1):198
43. Xu L, Chow MY (2006) A classification approach for power distribution systems fault cause identification. IEEE Trans Power Syst 21(1):53
44. van Rijsbergen CJ (1979) Information retrieval, Information retrieval. Butterworths, USA
45. Hripcsak G, Rothschild AS (2005) Agreement, the F-measure, and reliability in information retrieval. J Am Med Inform Assoc 12(3):296
46. Sasaki Y (2007) The truth of the F-measure, Teach Tutor Mater (2007)
47. Joachims T (2005) A support vector method for multivariate performance measures, in *Proc. 22nd International Conference on Machine Learning* (ACM, 2005), pp. 377–384
48. Jansche M (2005) Maximum expected F-measure training of logistic regression models, in *Proc. Conference on Human Language Technology and Empirical Methods in Natural Language Processing* (ACL, 2005), pp. 692–699
49. Nan Y, Chai KMA, Lee WS, Chieu HL (2012) Optimizing F-measure: a tale of two approaches, in *Proc. 29th International Conference on Machine Learning*, ed. by J. Langford, J. Pineau (2012), pp. 1555–1562
50. Dembczynski K, Waegeman W, Cheng W, Hüllermeier E (2011) An exact algorithm for F-measure maximization, in *Proc. 24th International Conference on Advances on Neural Information Processing Systems*, ed. by J. Shawe-Taylor, R.S. Zemel, P.L. Bartlett, F. Pereira, K.Q. Weinberger (2011), pp. 1404–1412
51. Batuwita R, Palade V (2009) A new performance measure for class imbalance learning. Application to bioinformatics problems, in *Proc. International Conference on Machine Learning and Applications* (IEEE, 2009), pp. 545–550
52. Dua D, Graff C (2019) UCI Machine Learning Repository Uci machine learning repository (2019). http://archive.ics.uci.edu/ml
53. Trawiński B, Smketek M, Telec Z, Lasota T (2012) Nonparametric statistical analysis for multiple comparison of machine learning regression algorithms. Int J Appl Mathe Comput Sci 22:867
54. Adnan MN, Ip RH, Bewong M, Islam MZ (2021) BDF: a new decision forest algorithm. Inform Sci 569:687
55. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
56. Gibbons JD, Chakraborti S (2011) Nonparametric statistical inference, in *International Encyclopedia of Statistical Science*, ed. by M. Lovric (Springer, 2011), pp. 977–979
57. Friedman M (1937) The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J Am Statist Assoc 32(200):675
58. Dunn OJ (1961) Multiple comparisons among means. J Am Statist Assoc 56(293):52
59. Sheskin DJ (2007) Handbook of parametric and nonparametric statistical procedures, Handbook of parametric and nonparametric statistical procedures
60. Parambath SP, Usunier N, Grandvalet Y (2014) Optimizing F-measures by cost-sensitive classification, in *Proc. 27th International Conference on Neural Information Processing Systems*, vol. 2, ed. by Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (2014), vol. 2, pp. 2123–2131
61. Kaya E, Korkmaz S, Sahman MA, Cinar AC (2021) DEBOHID: a differential evolution based oversampling approach for highly imbalanced datasets. Expert Syst Appl 169:114482