Machine Learning (2023) 112:2083–2104
https://doi.org/10.1007/s10994-022-06282-w

Automated imbalanced classification via layered learning

Vitor Cerqueira[1] · **Luis Torgo**[1] · **Paula Branco**[2] · **Colin Bellinger**[3]

Received: 5 May 2022 / Revised: 8 September 2022 / Accepted: 9 November 2022 /
Published online: 28 December 2022

Abstract
In this paper we address imbalanced binary classification (IBC) tasks. Applying resampling strategies to balance the class distribution of training instances is a common approach to tackle these problems. Many state-of-the-art methods find instances of interest close to the decision boundary to drive the resampling process. However, under-sampling the majority class may potentially lead to important information loss. Over-sampling also may increase the chance of overfitting by propagating the information contained in instances from the minority class. The main contribution of our work is a new method called `ICLL` for tackling IBC tasks which is not based on resampling training observations. Instead, `ICLL` follows a layered learning paradigm to model the data in two stages. In the first layer, `ICLL` learns to distinguish cases close to the decision boundary from cases which are clearly from the majority class, where this dichotomy is defined using a hierarchical clustering analysis. In the subsequent layer, we use instances close to the decision boundary and instances from the minority class to solve the original predictive task. A second contribution of our work is the automatic definition of the layers which comprise the layered learning strategy using a hierarchical clustering model. This is a relevant discovery as this process is usually performed manually according to domain knowledge. We carried out extensive experiments using 100 benchmark data sets. The results show that the proposed method leads to a better performance relatively to several state-of-the-art methods for IBC.

Keywords Imbalanced classification · Layered learning · Hierarchical clustering

Editors: Nuno Moniz, Nathalie Japkowicz, Michal Wozniak, Shuo Wang.

✉ Vitor Cerqueira
vitor.cerqueira@dal.ca

[1] Dalhousie University, Halifax, Canada

[2] University of Ottawa, Ottawa, Canada

[3] National Research Council of Canada, Ottawa, Canada

1 Introduction

In supervised learning, we face an imbalanced problem when the distribution of classes is significantly biased towards a particular value and at the same time, the least frequent class is also the most relevant one (Branco et al., 2016). An archetype of imbalanced classification is the detection of fraudulent cases, which are rare occurrences among a large proportion of normal activity. The accurate detection of these rare but important instances is fundamental across many domains of application. In effect, learning from imbalanced domains is one of the most active research topics in the machine learning literature.

The skewed class distribution hinders the learning process of algorithms, and several strategies have been devised to overcome this problem. The large majority of approaches designed to tackle the class imbalance issue are based on resampling methods. These transform the training data to improve the relevance of the minority class. Examples include SMOTE (Chawla et al., 2002), ADASYN (He et al., 2008), among many others (Branco et al., 2016). These methods are versatile and easy to couple with any learning system. However, under-sampling the majority class may potentially lead to important information loss. Over-sampling may increase the chance of overfitting by propagating the information contained in instances from the minority class.

Many of the existing resampling approaches work by finding instances close the decision boundary and using those instances to drive the resampling process. For example, ADASYN (He et al., 2008) is a popular method which synthesizes instances from the minority class whose neighborhood belong to the majority class. In this paper we propose a novel approach for imbalanced binary classification problems, which is called `ICLL` (Imbalanced Classification via Layered Learning). Specifically, we propose a method that, unlike resampling approaches, attempts to capture and improve the modelling of instances close to the decision boundary without synthesizing new instances or removing information from the training set.

First, we consider that an instance is arbitrarily close to the decision boundary according to a hierarchical clustering analysis. After applying the clustering model and cutting the hierarchy using an automatic heuristic (Bellinger et al., 2019), instances are assigned to one of three groups: the pure majority group, if that instance is clustered with only observations of the majority class; the pure minority group, if the respective cluster contains only observations of the minority class; or the mixed group, if the respective cluster contains observations from both classes. Accordingly, an observation is considered borderline (i.e., close to the decision boundary) if it belongs to the mixed group.

After assigning the training instances to one of these three groups based on the class distribution of the respective cluster, we model the data set using a layered learning approach (Stone & Veloso, 2000). Layered learning represents a hierarchical learning paradigm in which a predictive task is split into multiple, expectedly simpler, sub-tasks. In this work, we adopt a two-layer strategy following the work by Cerqueira et al. (2020). The first layer denotes a binary task designed to distinguish instances of the pure majority group from instances of either the mixed or pure minority group. In other words, we attempt to separate those instances which are clearly from the majority class (i.e. belong to the pure majority group). The second layer represents the original predictive task, where the objective is to distinguish instances from the majority class from instances of the minority class. The major difference is that only observations from mixed or pure minority groups are considered; instances from the pure majority group are discarded as the system models them in the first layer. Inference is performed according to the product of the output of each

layer. The main motivation for this layered approach is the assumption that by proceeding this way we obtain two learning tasks that are simpler to solve than the original imbalanced task. The main reason for this simplification lies on the fact that the imbalance is strongly decreased on each of the tasks, thus making the modelling task simpler for most learning algorithms. Finally, we also remark that the proposed method automates the problem of defining the layers within the layered learning approach using a hierarchical clustering analysis. In previous works (e.g. Cerqueira et al., 2020) this was carried out manually.

We carried out experiments using 100 benchmark binary classification data sets, and compared the proposed approach with several state-of-the-art methods. These include several resampling approaches, such as SMOTE (Chawla et al., 2002) and ADASYN (He et al., 2008), and a special-purpose algorithm designed for imbalanced problems, namely the balanced random forest (Chen et al., 2004). The results suggest that `ICLL` outperforms other approaches significantly according to the area under the ROC curve (AUC) metric.

In summary, the main contributions of this paper are two-fold:

- A novel method called `ICLL` for tackling binary imbalanced classification problems based on layered learning which does not require any parameters;
- An automatic framework for defining the layers in a layered learning methodology designed for classification.

The experiments and proposed method are publicly available.[1] The paper is organized as follows. In the next section we overview the literature related to our work. We focus on two specific topics: imbalanced classification and layered learning. In Sect. 3, we define the predictive task, and formalize the proposed approach which is named `ICLL`. We provide empirical evidence for the predictive performance of our method in Sect. 4, which includes a significance analysis based on the Bayes signed-rank test (Benavoli et al., 2017). The results from our work are discussed in Sect. 5, where we highlight the advantages of our approach and list its known limitations. Finally, we conclude the paper in Sect. 6.

2 Related work

In this section we overview the literature related to our work. Section 2.1 lists the main approaches used in the literature that deal with imbalanced classification problems. Section 2.2 describes layered learning approaches.

2.1 Imbalanced classification

Some of the most popular solutions used to tackle class imbalance in classification problems are resampling methods. These approaches transform the training set to enhance the prevalence of the minority class. This involves a strategy based on under-sampling the majority class, over-sampling the minority class, or both. Since the resampling occurs before model fitting, these methods are agnostic to the learning algorithm.

The simplest resampling methods are random under-sampling (`RU`) and random over-sampling (`RO`). `RU` randomly selects instances from the majority class, and removes those

[1] https://github.com/vcerqueira/icll.

instances from the training set. RO also selects instances at random, but from the minority class. The selected instances are replicated in the training data.

SMOTE (Synthetic Minority Over-sampling Technique) is a widely used over-sampling approach. It works by synthesising new examples based on existing ones. This is accomplished by interpolating between instances from the minority class within their neighbourhood. More precisely, an observation from the minority class is selected at random, along with its k nearest neighbours. Then, a new synthetic instance is created by interpolating between one of the k neighbours and the selected observation. This process can be carried out several times, for example until the distribution of classes is balanced.

After the initial publication (Chawla et al., 2002) several extensions of this method have been published, for example Borderline-SMOTE (Han et al., 2005) (Borderline). A comprehensive list of SMOTE variants can be found in the survey by Fernández et al. (2018).

ADASYN (Adaptive Synthetic) (He et al., 2008) is another over-sampling method which follows a similar approach to SMOTE and creates new synthetic instances. The key difference is that ADASYN focuses on instances which are difficult to learn, i.e., close to the decision boundary. The information about which instances are hard to classify is also explored in the work by Smith et al. (2014). They proposed an under-sampling method that discards instances based on their hardness (Hardness).

Informed under-sampling of the majority class is also a common approach to deal with the imbalance problem that embeds some domain information in the selection of the majority class examples to be removed. One example of this approach is the One-Sided Selection (OSS) method (Kubat et al., 1997). OSS identifies and removes instances close to the decision boundary using Tomek links (Tomek et al., 1976). Let x_1 and x_2 denote two instances from different classes, and $d(x_1, x_2)$ the distance between these examples. The pair (x_1,x_2) is considered a Tomek link if there exists no other instance with a smaller distance d to either x_1 or x_2. Typically, these instances are considered to be noise or close to the decision boundary. Tomek links have also been explored along SMOTE with the work by Batista et al. (2003) (SMOTETomek).

Near-Miss is another informed under-sampling method (Mani & Zhang, 2003). The main idea behind it is, contrary to the approach taken by OSS, it tries to retain only the instances from the majority class which are close to the decision boundary, instead of the other way around.

Resampling methods can be embedded in some learning algorithms. For example, the balanced random forest (Chen et al., 2004) extends the original method (Breiman, 2001) by applying random under-sampling in each bootstrap sample. Several ensemble methods have been modified through the integration of resampling to tackle the class imbalance problem (Galar et al., 2011). For instance, SMOTEBoost (Chawla et al., 2003) and SMOTE-Bagging (Wang & Yao, 2009) integrate SMOTE into a boosting and bagging ensemble, respectively, There are also hybrid ensembles that combine both bagging and boosting with resampling methods (e.g. EasyEnsemble and BalanceCascade (Liu et al., 2008)).

Previous works in the literature tried to couple clustering with resampling methods, e.g. Wu et al. (2007), Nickerson et al. (2001) and Bellinger et al. (2019). For example, CURE (Clustered Resampling) is a resampling approach which applies clustering analysis to the data and can perform both over-sampling and under-sampling (Bellinger et al., 2019). First, the method learns the structure of the data using hierarchical clustering analysis. Their approach for hierarchical clustering includes a novel semi-supervised metric used to compute the distance between each instance, and a novel heuristic for cutting the hierarchy and forming the clusters. Under-sampling of instances from the majority class is only performed in cases not

close to the decision boundary. On the other hand, over-sampling of minority instances is carried out with instances from the same concept or cluster. Our proposed method leverages ideas from CURE. In particular, as we will detail in the next section, we cluster the observations using a hierarchical method. We also apply the heuristic developed by the authors of CURE to cut the hierarchy and form the clusters. However, while CURE applies a semi-supervised mechanism for clustering our approach is purely unsupervised and based on the Euclidean distance. Finally, the purpose of our clustering analysis is to automatically create the layers for the layered learning methodology rather than to resample the training data.

The proposed approach is automated insofar as it does not require any parameters. In this context, it can be regarded as an automated machine learning (AutoML) approach for imbalanced problems, specifically binary classification. AutoML is becoming increasingly relevant in the machine learning literature. In the case of imbalanced problems, the literature is scarce. Moniz and Cerqueira (2021) presented a recent work which leverages meta-learning to select the best resampling strategy to apply in a given data set. Li et al. (2021) developed an automatic loss function for training deep neural networks.

2.2 Layered learning

Layered learning denotes a hierarchical procedure in which a predictive task is decomposed into simpler sub-tasks or layers, and each layer influences the learning process of the subsequent ones. For example, this may occur by influencing which training instances or predictor variables are used.

Layered learning was originally introduced by Stone and Veloso (2000) for robotic soccer, where they split the task of *passing a ball* into three sub-tasks: (i) intercepting the ball; (ii) evaluation of passing possibilities; and (iii) sending the ball. More generally, splitting a task into different parts is a common approach in the hierarchical reinforcement learning literature, such as the options framework devised by Sutton et al. (1999).

Cerqueira et al. (2020) have applied a layered learning process to tackle classification problems, specifically the detection of impending critical health episodes in the intensive care unit of hospitals. They, and subsequent related works (Ribeiro et al., 2021), show the advantage of this approach relative to a standard classification strategy coupled with resampling pre-processing methods. The crucial difference to our work is that they manually define the layers according to domain expertise or by optimization. Conversely, we provide a novel approach which accomplishes this task automatically using hierarchical clustering analysis. Moreover, we also systematise layered learning approaches for imbalanced binary classification problems.

There are similar hierarchical procedures to layered learning in the literature. For example, cascade classifiers (Sharma et al., 2012) denote a stepwise methodology in which a meta-classifier is built to predict which concept an instance refers to. Then that instance is passed to the appropriate base model. The critical difference of this approach to layered learning is that the learning of each layer affects the learning of subsequent ones. Besides, each instance traverses all layers instead of being routed to a specific model.

3 Methodology

In this section we start by defining the predictive task addressed in this work (Sect. 3.1). Next we formalize the method we propose to tackle this task (Sect. 3.2).

3.1 Problem definition

Let $\mathcal{D}$ denote a data set, which is defined as $\mathcal{D} = \{\langle x_i, y_i \rangle\}_{i=1}^{n}$, where $x_i \in X$ represents the feature vector for the i-th instance, and $y_i \in Y$ represents the respective target value. The target variable Y is discrete and can take two values. The objective is to carry out supervised learning, which, given the domain of Y, amounts to a binary classification task. Moreover, let $\mathcal{D}_0$ denote a subset of $\mathcal{D}$ in which all $y \in \mathcal{D}_0$ belong to the class 0; $\mathcal{D}_1$ represents the subset of observations for the class 1. An imbalanced classification problem arises because the number of observations belonging to one class is significantly larger than the number of observations of the other class: $|\mathcal{D}_0| \gg |\mathcal{D}_1|$. We will refer to 0 as the majority class, and 1 to the minority class.

3.2 Imbalanced classification layered learning

In this sub-section we formalize the proposed method which is named `ICLL` for Imbalanced Classification Layered Learning. The training of the `ICLL` method is based on three main steps:

1. Clustering the training instances using a hierarchical clustering method (Murtagh & Contreras, 2011);
2. Automatic creation of the layers for the layered learning strategy based on the output of the clustering model;
3. Training a predictive model in each layer.

In the rest of this section we will describe each step in detail.

3.2.1 Step 1: clustering analysis

In the first stage of the methodology a hierarchical clustering algorithm is applied to cluster the training instances. The main motivation for clustering is to extract structural information from the data, which will be used to automatically define the layers. Hierarchical clustering methods group data into a tree structure, which provides different levels of abstraction of that data (Murtagh & Contreras, 2011). We adopt an agglomerative approach for the clustering algorithm, which is a common strategy that can be described as follows. Each instance is first assigned as their own cluster. Then, pairs of clusters are successively merged until there is only a single cluster that contains all observations. One of the advantages of using a hierarchical algorithm for clustering is that it does not require the number of clusters as an input parameter.

The hierarchical clustering process is detailed in Algorithm 1. In order to group the data we start by measuring the dissimilarity between each pair of instances according to the Euclidean distance (line 1). As the linkage criterion, we apply the Ward method, which minimizes the total within-cluster variance (line 2). We use the Ward method to bias the clusters to contain instances that are highly concentrated. In other words, the goal is to obtain clusters which represent sub-concepts or groups of highly similar instances. These groups will then be sorted according to those that are easy and hard to classify (in the next step of the methodology).

Algorithm 1: Hierarchical clustering analysis

Input : X - Predictor variables for classification training data set
Output: C - Set of clusters

1 $M_X \leftarrow$ PairwiseDistance(X, method = Euclidean) `// Pairwise distance matrix of` X `using the Euclidean distance`
2 $Z \leftarrow$ Linkage(M_X, method=Ward) `// Agglomerative linkage tree using` M_X `and the Ward method`
3 InterClusterDistance(Z) $\leftarrow$ log(InterClusterDistance(Z)) `// Log transformation of the inter-cluster distances obtained in the tree` Z
4 $\mu \leftarrow$ Mean(InterClusterDistance(Z)) `// Mean of the inter-cluster distances`
5 $\sigma \leftarrow$ StandardDeviation(InterClusterDistance(Z)) `// Standard deviation of the inter-cluster distances`
6 $\tau \leftarrow \mu + \sigma$ `// Maximum inter-cluster distance for cluster formation`
7 $C \leftarrow$ FormClusters(Z, threshold=τ) `// Form clusters using the linkage tree, subject to the threshold` τ
8 Return C

Once the cluster hierarchy is formed, we must extract a specific clustering of the data from the hierarchy. Bellinger et al. (2019) proposed an automatic strategy by which to extract this clustering such that it accounts for the natural spread in the data. The objective is to define clusters that are large enough to capture entire sub-concepts without including multiple sub-concepts. In this method (lines 3–7), each instance is assigned to the cluster with the largest cardinality that it belongs to and has an intra-cluster distance less than the threshold $\tau = \mu + \sigma$; μ and σ denote the mean and standard deviation of the log transformed intra-cluster distances. Intuitively, utilising $\mu + \sigma$ as the threshold caps distance between the samples in the clusters at a level that is natural according the target dataset. The log transformation is applied because Bellinger et al. (2019) discovered that the intra-cluster distances approximately followed a lognormal distribution.

3.2.2 Step 2: automatic layer definition

At this point, we have the cluster allocation for each training observation $(x, y) \in \mathcal{D}$. In the second stage of the methodology we create the layers which comprise the layered learning strategy. We follow the layer architecture developed by Cerqueira et al. (2020) for early event detection and construct a workflow with two layers.

The two layers are automatically defined according to the clustering composition obtained in the previous step. Specifically, we assign each instance to one of three possible groups:

- Pure majority group: if the corresponding cluster comprises only instances of the majority class;
- Pure minority group: if the corresponding cluster comprises only instances of the minority class;
- Mixed group: if the corresponding cluster contains observations from both majority and minority classes.

This idea is depicted in Fig. 1, where part of a dendrogram is shown. In this synthetic example there are six clusters, two of each comprise only majority instances (Pure

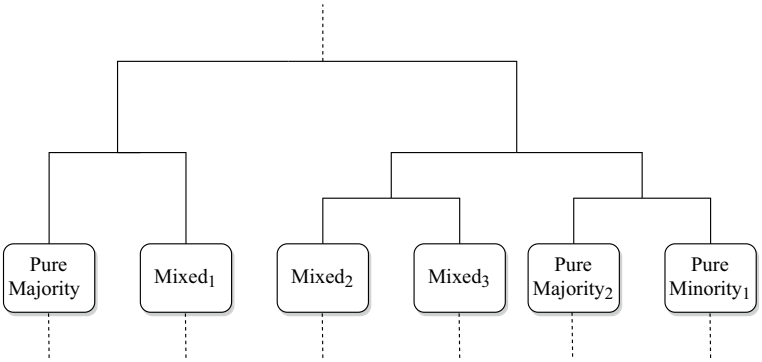

Fig. 1 An example of a partial dendrogram resulting from the hierarchical clustering. The instances are assigned to three possible groups according to the class distribution of the respective cluster after the cut-off

Majority 1 and 2), one contains only instances of the minority class (Pure Minority 1), and the remaining ones are Mixed. The process of assigning the observations into the three possible groups is formalized in Algorithm 2.

Algorithm 2: Group assignment based on cluster class distribution

Input : $\mathcal{D}$ - Binary classification data set
: C - Set of clusters
Output: $C_{maj}, C_{min}, C_{mix}$

$C_{maj} \leftarrow [\,]$ // Initialize empty group of pure majority instances
$C_{min} \leftarrow [\,]$ // Initialize empty group of pure minority instances
$C_{mix} \leftarrow [\,]$ // Initialize empty group of borderline instances
foreach *instance* (x_i, y_i) *in* $\mathcal{D}$ **do**
  $c_i \leftarrow$ Cluster of (x_i, y_i)
  if c_i *contains only instances of the majority class* **then**
    $C_{maj}.add((x_i, y_i))$;
    // Add instance to C_{maj}
  else if c_i *contains only instances of the minority class* **then**
    $C_{min}.add((x_i, y_i))$;
    // Add instance to C_{min}
  else
    $C_{mix}.add((x_i, y_i))$;
    // c_i contains instances from both classes -- add instance to C_{mix}
  end
end
Return $C_{maj}, C_{min}, C_{mix}$

This process enables the understanding of the position of each training instance with respect to the decision boundary. If an instance belongs to the pure majority group it can be said that it is clear that this observation belongs to the majority class. In other words, the instance is arbitrarily far from the decision boundary on the side of the majority class. The same can be argued for instances belonging to the pure minority group. In principle, these observations are far from the decision boundary but on the side of the

minority class. Finally, we have instances from the mixed group whose cluster comprises instances from both minority and majority classes. We regard these observations as borderline instances, i.e. arbitrarily close to the decision boundary.

In order to leverage the information regarding the group of each training instance we adopt a stepwise methodology based on layered learning. As we described in Sect. 2.2, layered learning denotes a learning approach in which a predictive task is split into multiple ones which are, in principle, easier to solve. We construct two layers, Layer 1 and Layer 2, according to the definitions below.

Layer 1 The two layers denote two sub-tasks of the original problem. These sub-tasks comprise a more balanced class distribution. We hypothesise that this will lead to easier classification tasks and, consequently, better performance. Let L1 denote the event "The observation belongs to the Mixed or Pure Minority group". For the first layer, the target value for a given instance is defined as:

$$y_i^{L1} = \begin{cases} 1 \text{ if L1 happens,} \\ 0 \text{ otherwise.} \end{cases} \tag{1}$$

Essentially, we attempt to model whether or not each observation belongs to a purely majority group, in which case $y^{L1} = 0$, or not ($y^{L1} = 1$). In practice, the original predictor variables remain the same, but the target values $y \in Y$ are replaced with $y^{L1} \in Y^{L1}$, where Y^{L1} denotes the target variable for the layer L1.

This first layer can be regarded as an approach designed to distinguish instances from the majority class which are *easy* to learn (belong to the pure majority group) from the others, which are either borderline (mixed group) or scarce (pure minority group).

Layer 2 Assuming that the first layer is modelled successfully, the second layer attempts to solve the remaining problem: Given that L1 occurs, i.e., an observations belongs to the mixed or pure minority group, we want to find whether it belongs to the minority class (i.e. $y_i = 1$). The target variable for this sub-task (y^{L2}) is formalised in Eq. (2).

$$\text{Given } y^{L1} = 1, y_i^{L2} = \begin{cases} 1 \text{ if } y_i = 1, \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

In the second layer we discard all observations from the pure majority group, and only fit a model with those from the mixed and pure minority group ($y^{L1} = 1 \forall y^{L1} \in Y^{L1}$).

3.2.3 Step 3: model fitting and inference

In the training stage, a predictive model is fit in each one of the two layers. We note that, since the two layers are independent, the training can occur in parallel.

Let f^{L1} and f^{L2} denote the models trained in the layers L1 and L2, respectively. We combine the output of these models according to a function g:

$$g(x_i) = f^{L1}(x_i) \cdot f^{L2}(x_i) \tag{3}$$

The final decision is made according to the multiplication of the individual predictions. Therefore, our approach postulates that the probability that a given instance belong to the

minority class is estimated according to the probability that it belongs to either the mixed or pure minority group times the probability that, given that the observations it belongs to either the mixed or pure minority group, it belongs to the minority class. This process is also applicable to non-probabilistic classifiers. That is, our approach predict that a given instance x_i belongs to the minority class if both $f^{L1}(x_i)$ and $f^{L2}(x_i)$ predict 1.

3.3 Methodological limitations

The layer definition stage of the methodology strongly depends on the outcome of the hierarchical clustering model. We formalized our method assuming that all three groups C_{maj}, C_{min}, and C_{mix} are non-empty, but this might not be the case depending on the input data.

The methodology works normally in the case that C_{min} is empty, i.e. there are no instances in the pure minority group. By definition, the minority class comprises a relatively low number of observations. In effect, it is expected that, even if the minority class is more prevalent in a given cluster, all clusters contain some instances from the majority class.

If C_{mix} is empty, this means that the hierarchical clustering is able to perfectly split the two classes in the chosen cut-off. In this case the proposed methodology becomes redundant. However, this also means that the predictive task is solved.

The scenario in which C_{maj} is empty is more problematic as the first layer cannot be defined properly. Our approach to solve this problem would be to change the threshold τ and move the formation of the clusters up in the hierarchy. Notwithstanding we remark that, given the relative high prevalence of instances of the majority class, this represents a highly unlikely scenario.

4 Experiments

This section describes the experiments carried out to validate `ICLL`. These were designed to address the following research questions:

1. **RQ1** How does `ICLL` perform relatively to state-of-the-art methods for IBC problems?;
2. **RQ2** Does applying resampling to the proposed method improve its performance? Since resampling methods are agnostic to the learning algorithm, we attempt to couple these approaches with `ICLL` and assess whether performance improvements are obtained;
3. **RQ3** Is `ICLL` a model-based strategy for under-sampling? The second layer of `ICLL` is carried out after discarding observations which are arbitrarily far from the decision boundary on the side of the majority class. Thus, it can be argued that the first layer is performing under-sampling and the discrimination between classes occurs in the second layer. We will test this hypothesis in the experiments;
4. **RQ4** Are the conclusions consistent when taking into account only *difficult* problems? We examine the results of the experiments with all available data sets (presented in Sect. 4.1) and using only a subset were a baseline performs poorly.

4.1 Case study and experimental design

The experiments were carried out using 100 data sets. These were retrieved from the KEEL repository (Keel data, 2022; Fernández et al., 2008, 2009), which provides benchmark data

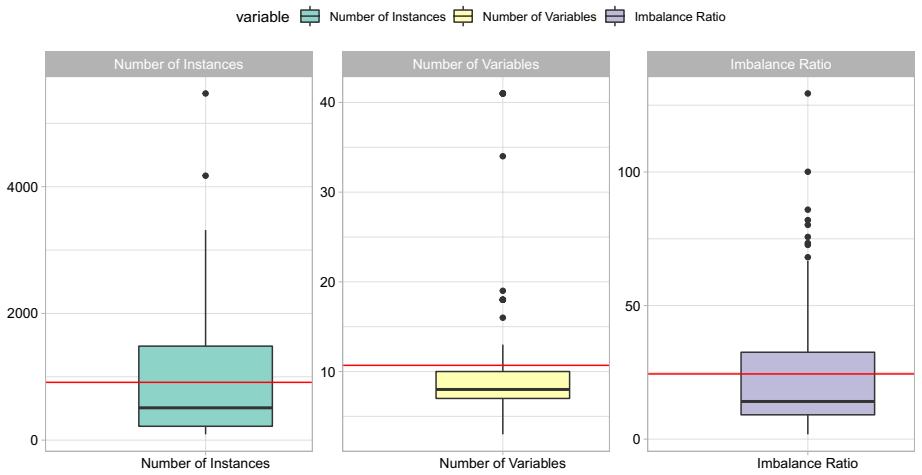

Fig. 2 Boxplots showing the distribution of number of instances, number of explanatory variables, and imbalance ratio of the data sets used in the experiments. The horizontal red line denotes the mean value for each characteristic (Color figure online)

sets for imbalanced domain learning. From this repository, we collected all binary imbalanced classification data sets. The distribution of basic characteristics of these 100 data sets are shown in Fig. 2, namely the number of observations, number of explanatory variables, and imbalance ratio. The number of instances ranges from 92 to 5472 with an average of 934 data points, while the number of variables range from 3 to 100 with an average of 11.6. The minimum, maximum, and average imbalance ratio is 1.8, 129.4, and 25.7 respectively. In effect, the case study covers data sets with different imbalance ratio profiles.

We applied a 2×5-fold stratified cross-validation procedure for estimating the predictive performance of each approach. We take a probabilistic perspective to the imbalanced classification task. Therefore, performance was measured according to the area under the ROC curve (AUC). In terms of learning algorithms, we tested a Random Forest (RF), a Support Vector Machine (SVM), and a Logistic Regression (LR). We resorted to the implementation from scikit-learn (Pedregosa et al., 2011) to apply these algorithms with their default parameters. The RF provided the overall best results. In effect, we will show the complete results only for this method. Notwithstanding, we include a variant of the remaining approaches in the interest of completeness. All conclusions also hold for these methods. We also remark that the Random Forest is used in both layers of `ICLL`, but this is not a requirement of the method. Different learning algorithms could be applied in each layer.

4.2 Methods

Besides the proposed approach `ICLL`, we include the following methods in the experiments:

- `NoResample-RF`, `NoResample-SVM`, `NoResample-LR`: A standard binary classifier which is fit in the training data set without any specific mechanism for dealing with the imbalanced class distribution. As explained above we test three learning algorithms for this approach;

- `SMOTE`, `CURE`, `ADASYN`, `NearMiss`, `Borderline`, `SMOTETomek`, `Hardness`, `OSS`, `RO`, `RU`: A Random Forest classifier is fit after the training data is pre-processed with the respective resampling method for balancing the class distribution. These approaches were described in Sect. 2.1;
- `BalancedRF`: A variant of the Random Forest algorithm in which random undersampling is carried out for each bootstrap sample (Chen et al., 2004);

We remark that all resampling approaches were applied using their default parameter setting according to the implementation provided by the *imblearn* python library.[2] Besides these state-of-the-art approaches, we also include the following five variants of `ICLL`:

- `ICLL+SMOTE`: The proposed method described in Sect. 3.2 works without any resampling approach. In this variant we couple our approach with a SMOTE (Chawla et al., 2002) resampling method. Essentially, we apply SMOTE in each of the two layers to balance the distribution of the classes;
- `ICLL+SMOTE(L1)`: This approach is similar to `ICLL+SMOTE`, but `SMOTE` is applied only in the first layer;
- `ICLL+SMOTE(L2)`: Another approach similar to `ICLL+SMOTE`, in which `SMOTE` is applied only in the second layer;
- `ICLL(L2)`: A variant which only uses the output from the second layer. It can be argued that the proposed layered learning method is performing a model-based undersampling in the first layer. Thus, the performance advantage is obtained only in the second layer, making the first layer unnecessary in the inference stage. We test this hypothesis using this variant.
- `ICLL(L1)`: In the interest of completeness, we also include a variant which uses only the output from the first layer.

4.3 Results

We present the results of the experiments in this section. First (Sect. 4.3.1), we perform a preliminary analysis on the applicability of the proposed method. Then (Sect. 4.3.2), we compare all methods according to the average rank and measure the percentage difference in predictive performance. Finally, we repeat this analysis but only taking into account the datasets where the approach `NoResample-RF` does not provide a good predictive performance (Sect. 4.3.3).

4.3.1 Preliminary analysis

In this section we present the results of the experiments. As we explained in Sect. 3.3 there are scenarios in which the stepwise approach followed by `ICLL` becomes redundant, specifically when the mixed group (C_{mix}) is empty. As we mentioned, this means that the hierarchical clustering method was able to perfectly split the majority instances from the minority instances to different clusters. This case occurred in at least one of the ten iterations of 11 (out of 100) data sets. In principle, if the clustering model is able to split the classes perfectly then a standard classifier should be able to do so as well. Indeed, in all 11

[2] https://pypi.org/project/imblearn/.

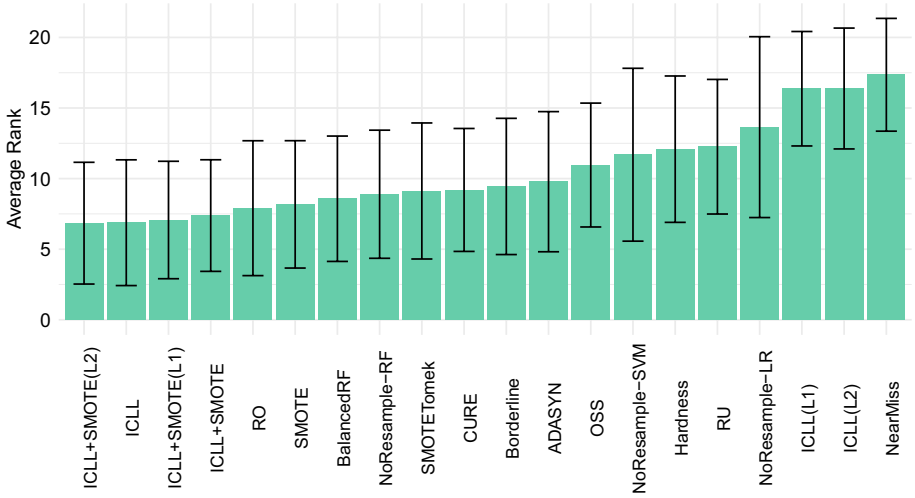

Fig. 3 Average rank of each method. Lower values denote better performance

datasets where this issue occurred, the final AUC score of the `NoResample-RF` approach was 1 (i.e. a perfect score). We continue the analysis without these 11 datasets.

4.3.2 Average rank, magnitude of differences, and significance analysis

Regarding the analysis of results we start by observing the average rank of each method across the remaining 89 problems. Afterwards, we analyse the magnitude in the differences in performance to assess their significance. In Fig. 3 we show the average rank of each method across the 89 data sets. A given method has a rank of 1 in a given problem if it shows the best performance (AUC) in that problem. Effectively, the average rank represents the average position of each approach relative to the remaining ones.

Four of the variants of `ICLL` are the best four methods in average rank. The best one is `ICLL+SMOTE(L2)`, which applies the SMOTE resampling method in the second layer. The version of the proposed method which does not apply any resampling approach (`ICLL`) shows a better score than any state-of-the-art resampling approach. Notwithstanding, its performance improves when coupled with SMOTE. The variants of the proposed method which only use the output of one of the layers (`ICLL(L1)` and `ICLL(L2)`) show one of the worse average ranks. This suggests that the output of both layers is critical for the predictive performance of `ICLL`. This outcome contradict the hypothesis that `ICLL` is a model-based under-sampling approach (RQ3). `ICLL` performs considerably better than `ICLL(L2)`, which means that both layers are important during the inference stage. Regarding state-of-the-art resampling methods, `RO` shows the best average rank, followed by `SMOTE`. Finally, the standard classifier trained with a Random Forest (`NoResampler-RF`) shows a better score relative to the same approach trained with either a Logistic Regression (`NoResampler-LR`) or a SVM (`NoResampler-SVM`).

As mentioned before, the average rank measures the average position of each method relative to the remaining ones. However, it does not take into account the magnitude of differences of predictive performance. Therefore, we complement the average rank by analysing the percentage difference in performance between each method and the variant of the

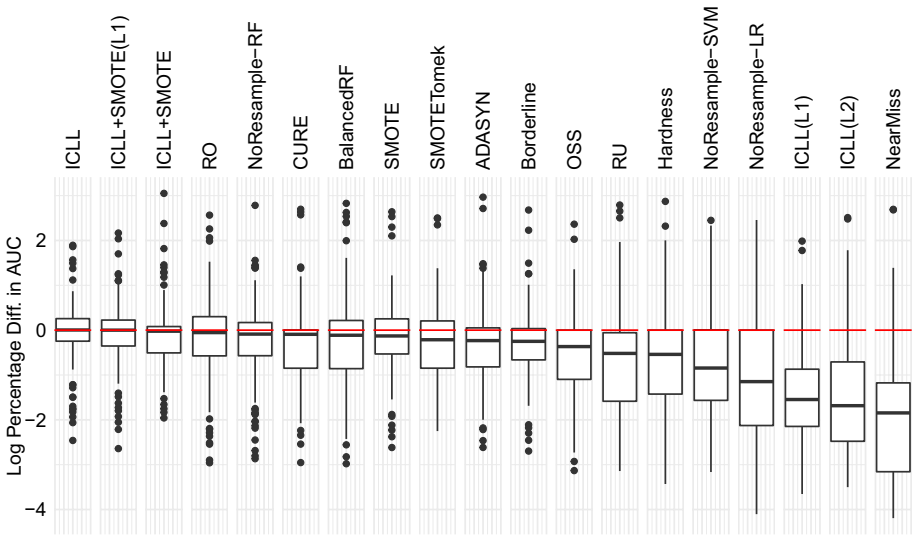

Fig. 4 Boxplots showing the distribution of the log percentage difference between each method and `ICLL+SMOTE(L2)`. Negative values denote better performance of `ICLL+SMOTE(L2)`

proposed approach with best average rank (`ICLL+SMOTE(L2)`). This can be formalized as follows for a given method `m`:

$$100 \times \frac{\text{AUC}_{\texttt{m}} - \text{AUC}_{\texttt{ICLL+SMOTE(L2)}}}{\text{AUC}_{\texttt{ICLL+SMOTE(L2)}}}$$

where $\text{AUC}_{\texttt{m}}$ and $\text{AUC}_{\texttt{ICLL+SMOTE(L2)}}$ represent the AUC of method `m` and `ICLL+SMOTE(L2)`, respectively. Since AUC should be maximized, negative values in percentage difference denote better performance of `ICLL+SMOTE(L2)`.

We show this analysis in Fig. 4. This figure depicts several boxplots showing the distribution of the log percentage difference between each method and `ICLL+SMOTE(L2)`, where negative values denote better performance for the proposed method. Moreover, the methods are ordered by decreasing median percentage difference in AUC. Thus, more competitive methods appear first (from left to right). In terms of ranking, the order of the methods is similar to that obtained according to the average rank analysis. The main take away is that, for all methods, most of the distribution lies below the zero line. This shows that `ICLL+SMOTE(L2)` outperforms the other methods more times than not.

While it is clear that `ICLL+SMOTE(L2)` shows a better performance, Fig. 4 also shows that the percentage difference is close to zero in many cases. In this context, we perform a new analysis which considers small differences in performance to be negligible and the pair of models practically equivalent.

We define this interval to be $[-1\%, 1\%]$. This means that the performance of two methods under comparison is considered equivalent if their percentage difference is within this interval.

Figure 5 shows the probability of `ICLL+SMOTE(L2)` winning in blue (percentage difference below -1%), drawing in light grey (results within $[-1\%, 1\%]$), or losing in red (percentage difference above 1%) against each remaining method. For example, relative

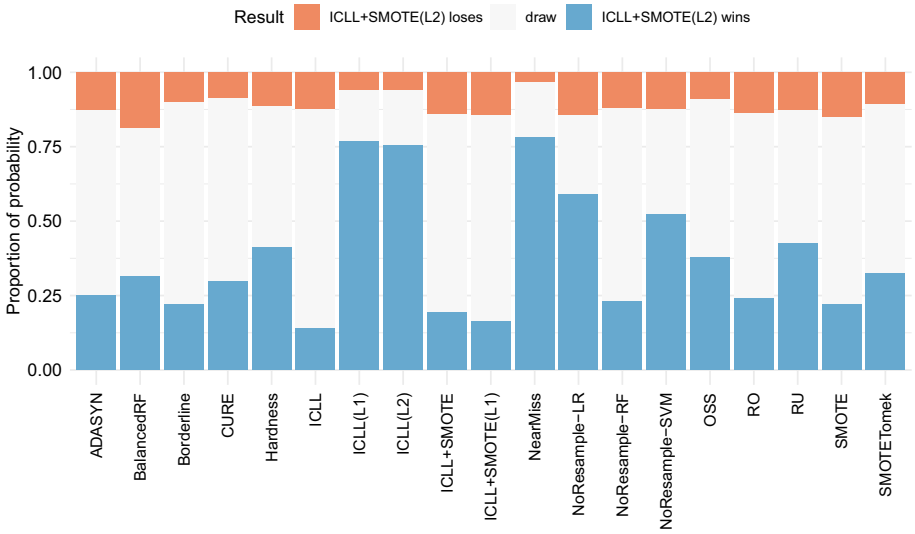

Fig. 5 Paired comparisons between `ICLL+SMOTE(L2)` and each remaining approach. Each stacked barplot shows the probability of `ICLL+SMOTE(L2)` winning in blue (result below − 1%), drawing in light grey (result withing [− 1%, 1%]), or losing in red (result above 1%)

to `ADASYN`, `ICLL+SMOTE(L2)` has a probability of winning of around 25%, a probability of losing of about 12.5%, and a probability of drawing of about 62.5%. Analysing the scores, it is clear that `ICLL+SMOTE(L2)` outperforms the other methods. That is, the probability of winning is larger than the losing. Nonetheless, there is a considerable probability that the results end up being comparable (a percentage difference in AUC below 1%).

Finally, we carried out a bayesian analysis to assess the significance of the results using the Bayes signed-rank test (Benavoli et al., 2017). This test is used to compare pairs of predictive models across multiple data sets. In this case, we compare `ICLL+SMOTE(L2)` with all remaining methods. We also define the region of practical equivalence (ROPE) for the Bayes signed-rank test to be the interval [− 1%, 1%]. The results are shown in Fig. 6, which follow a similar structure as Fig. 5. The results of the test show that `ICLL+SMOTE(L2)` either wins significantly with high probability or draws when compared with other methods and considering a 1% ROPE level. While `ICLL+SMOTE(L2)` tends to outperform state-of-the-art approaches, the differences are often small and not statistically significant above 1% when compared with some approaches, e.g. `ICLL+SMOTE`, `RO`, or `NoResample-RF`.

4.3.3 Repeating the study for difficult problems

In many of the data sets, specifically in 71 out of 100, the `NoResample-RF` approach is able to achieve at least 0.9 AUC. This means that, even without any special mechanism for dealing with the imbalanced distribution, the predictive model is able to distinguish between both classes with a good performance. In this context, we decided to repeat the result analysis but only considering the data sets in which `NoResample-RF` has an AUC score lower than 0.9. Overall, there are 29 data sets where this occurs. For simplicity, we will refer to these data sets as the difficult problems within our case study.

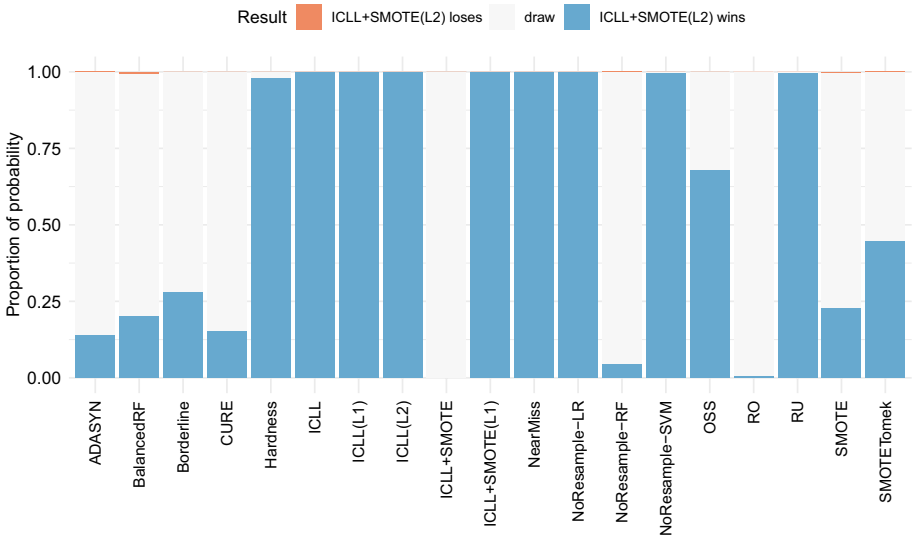

Fig. 6 Paired comparisons between `ICLL+SMOTE(L2)` and each remaining approach using the Bayesian signed-rank test. Each bar represents the proportion of probability of each outcome: the blue part denotes the probability that `ICLL+SMOTE(L2)` wins significantly; the red part represents the proportion of probability in which `ICLL+SMOTE(L2)` loses significantly; the grey area represents the probability of a draw (Color figure online)

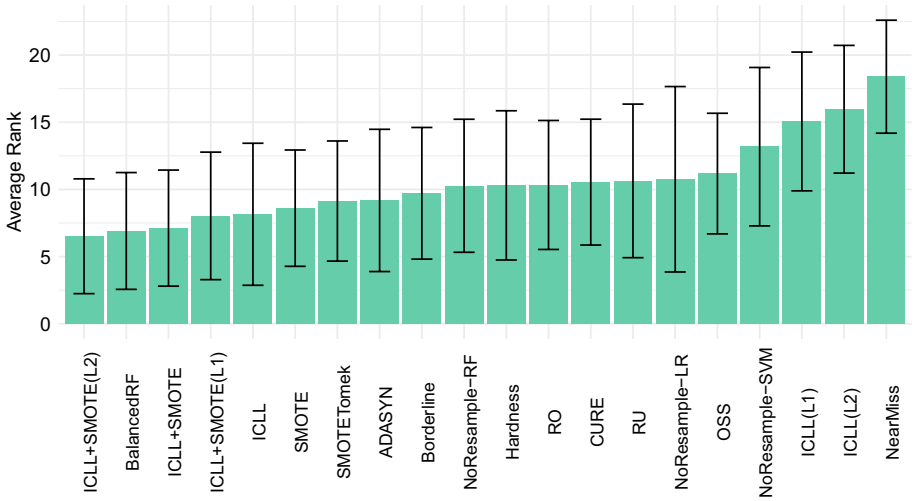

Fig. 7 Average rank of each method using the difficult problems. Lower values denote better performance

Figure 7 shows the average rank of each method across the difficult problems. The relative position of each method is similar. However, the variant of the proposed method without resampling (`ICLL`) shows a slightly worse average rank relative to `BalancedRF`. Notwithstanding, `ICLL+SMOTE(L2)` shows the best score overall.

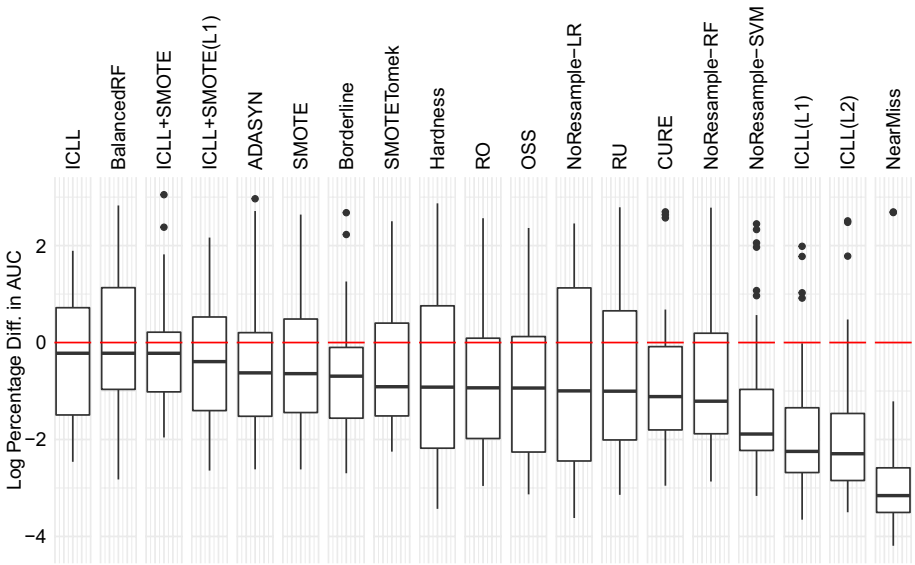

Fig. 8 Boxplots showing the distribution of the log percentage difference between each method and `ICLL+SMOTE(L2)` using only the difficult problems

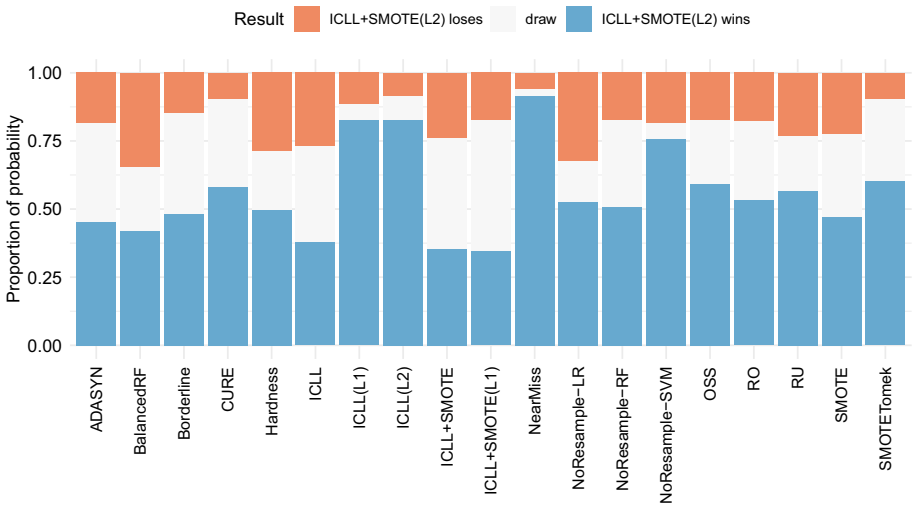

Fig. 9 Paired comparisons between `ICLL+SMOTE(L2)` and each remaining approach. Each stacked barplot shows the probability of `ICLL+SMOTE(L2)` winning in blue (result below − 1%), drawing in light grey (result withing [− 1%, 1%]), or losing in red (result above 1%). This analysis considers only difficult problems (Color figure online)

The distribution of the percentage difference in AUC is presented in Fig. 8. Overall, for difficult problems the distribution of the percentage differences becomes even more favourable towards `ICLL+SMOTE(L2)`. These conclusion can also be drawn from Fig. 9, which shows the probability of `ICLL+SMOTE(L2)` winning, losing, or drawing (percentage

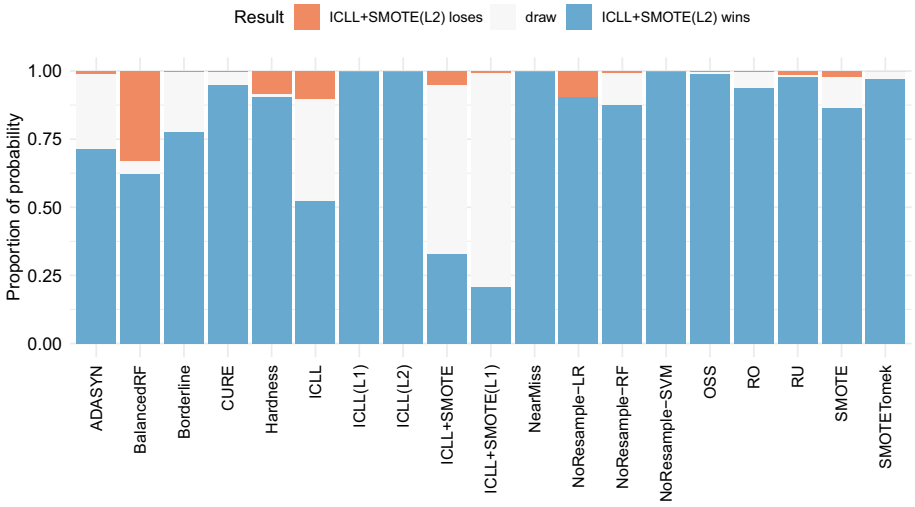

Fig. 10 Paired comparisons between `ICLL+SMOTE(L2)` and each remaining approach using the Bayesian sign test with 1% ROPE level. Each bar represents the proportion of probability of each outcome: the blue part denotes the probability that `ICLL+SMOTE(L2)` wins significantly; the red part represents the proportion of probability in which `ICLL+SMOTE(L2)` loses significantly; the grey area represents the probability that the difference in AUC is below 1% (Color figure online)

difference below 1%). The results of the Bayesian signed-rank test are shown in Fig. 10. In this subset of problems, `ICLL+SMOTE(L2)` shows statistically significant better performance when compared with all other methods, except for the variants `ICLL+SMOTE(L1)` and `ICLL+SMOTE`. Indeed, when considering this subset of more difficult problems the advantage of the proposed method is enhanced and the probability of outperforming other methods increases considerably.

Finally, we studied in which conditions the proposed method led to better performance. We analysed the characteristics of the data sets based on three complexity measures: k-Disagreeing Neighbors (kDN) (Smith et al., 2014), number of borderline examples (Borderline) (Napierała et al., 2010), and Degree of Overlap (Mercier et al., 2018). We analysed the distribution of these metrics while controlling for whether `ICLL+SMOTE(L2)` outperformed `NoResample-RF`. The results are shown in Fig. 11. Higher values denotes greater problem complexity. These values are presented in a log scale for visualization purposes. Besides the boxplots, the figure includes horizontal red lines which represent the median value of each metric. The distributions are similar in both conditions, though the average value is slightly higher for when `ICLL+SMOTE(L2)` performs better. This indicates that the proposed method is more appropriate for more complex problems with respect to `NoResample-RF`.

5 Discussion

In the previous section we provided compelling evidence for the advantage of applying `ICLL` when tackling binary classification tasks with an imbalanced class distribution. We showed that `ICLL` performs better than several state-of-the-art methods for IBC problems (RQ1). The improvements are enhanced when coupled with the `SMOTE` resampling

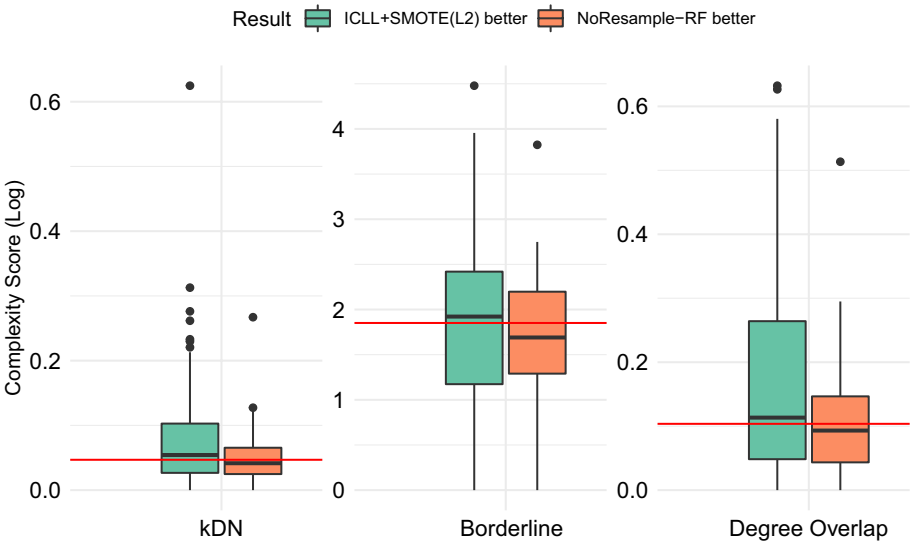

Fig. 11 Distribution of three complexity measures and controlling whether `ICLL+SMOTE(L2)` outperforms `NoResample-RF`

method in the second layer (RQ2). Regarding the behaviour of `ICLL`, it is clear that both layers are important during the inference stage (RQ3). Finally, the performance advantage of `ICLL` is greater when considering difficult problems where a baseline method performs poorly. Besides the gains in predictive performance, it is worth mentioning that the proposed method does not require any parameters besides the base learning algorithm, which in our case is a Random Forest. This is an important advantage of the proposed `ICLL` solution as it does not require the end-user to carry out any additional hyper-parameter tuning for using `ICLL`. However, the best results were achieve when `ICLL` was coupled with `SMOTE`, which is not automated. Notwithstanding, in this work we applied `SMOTE` with its default configuration. The hierarchical clustering procedure is automated. Specifically, the process of cutting the hierarchy and obtaining the cluster compositions is carried out using the heuristic described by Bellinger et al. (2019).

One interesting thing we noted in the experiments is that `ICLL+SMOTE(L2)` shows a greater performance advantage in difficult data sets. These are problems in which the decision boundary is, in principle, more difficult to model. We believe that the concept of mixed group we introduced, and the subsequent stepwise approach based on layered learning, can be beneficial for these cases.

Layered learning approaches have been used for tackling classification problems. Notwithstanding, the layer definition is usually, to our knowledge, carried out manually – either treating these as parameters to optimize or defined by domain experts. Therefore, automating the process of defining the layers within these approaches is a valuable contribution.

Our work is limited by the successful application of the hierarchical clustering procedure. To be more precise, the layers require the existence of both pure majority and mixed groups. Otherwise, these cannot be defined properly. Notwithstanding, during our experiments we found that majority groups were common because of the high prevalence of majority class instances. In 11 out of the 100 data sets, mixed groups could not be found. These represented easy data sets where the hierarchical clustering was

able to perfectly split the two classes. Indeed, the classifier without any resampling method, or any other mechanism for dealing with class imbalance, achieved a perfect AUC score (1) in all those 11 data sets. In such cases, we do not need to go beyond the clustering analysis as this process indicated that an advanced classification strategy is not necessary.

Our work is focused on tabular problems. Therefore, the experiments did not include any deep learning approach. Recent developments in imbalanced deep learning are primarily fixated on images, text and graphs. This type of data sets are not considered. Notwithstanding, relying on clustering and a Random Forest model means our solution is more interpretable relative to deep learning.

In terms of future developments, we outline two potential research directions: the first is a better exploitation of the hierarchy output by the clustering model. Specifically, it may be possible to devise different layer architectures depending on the result of the clustering analysis. Second, we will attempt to apply `ICLL` to other predictive tasks, namely multi-class problems, one-class classification, or imbalanced regression tasks.

6 Conclusions

We proposed a new approach for IBC problems, which is one of the most active research topics in machine learning. The proposed approach models the data in a two-stage fashion according to a layered learning methodology (Stone & Veloso, 2000). The layers are automatically defined using hierarchical clustering analysis, and the class distribution of the resulting clusters. We provided extensive empirical evidence which shows that the proposed approach leads to a better performance relatively to several state-of-the-art methods for IBC tasks. Contrary to the state of the art approaches to IBC, our proposal does not require tuning of any parameter—it is essentially parameter-less.

We believe that our method represents a promising direction towards modelling approaches which are not based on resampling the training data.

Author contributions All authors contributed to writing and research.

Funding The work of L. Torgo was undertaken, in part, thanks to funding from the Canada Research Chairs program;

Availability of data and materials All experiments and data are publicly available (c.f. footnote 1).

Code availability Publicly available (c.f. footnote 1).

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

Consent to participate Not applicable.

Consent for publication Not applicable.

Ethics approval Not applicable.

References

Batista, G. E., Bazzan, A. L., & Monard, M. C., et al. (2003). Balancing training data for automated annotation of keywords: a case study. In: WOB, pp. 10–18.

Bellinger, C., Branco, P., & Torgo, L. (2019). The cure for class imbalance. In *International conference on discovery science* (pp. 3–17). Springer.

Benavoli, A., Corani, G., Demšar, J., & Zaffalon, M. (2017). Time for a change: A tutorial for comparing multiple classifiers through Bayesian analysis. *The Journal of Machine Learning Research, 18*(1), 2653–2688.

Branco, P., Torgo, L., & Ribeiro, R. P. (2016). A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR), 49*(2), 1–50.

Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.

Cerqueira, V., Torgo, L., & Soares, C. (2020). Early anomaly detection in time series: A hierarchical approach for predicting critical health episodes. arXiv preprint arXiv:2010.11595

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority oversampling technique. *Journal of Artificial Intelligence Research, 16,* 321–357.

Chawla, N. V., Lazarevic, A., Hall, L. O., & Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In *European conference on principles of data mining and knowledge discovery* (pp. 107–119). Springer.

Chen, C., Liaw, A., Breiman, L., et al. (2004). Using random forest to learn imbalanced data. *University of California, Berkeley, 110*(1–12), 24.

Fernández, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *Journal of Artificial Intelligence Research, 61,* 863–905.

Fernández, A., García, S., del Jesus, M. J., & Herrera, F. (2008). A study of the behaviour of linguistic fuzzy rule based classification systems in the framework of imbalanced data-sets. *Fuzzy Sets and Systems, 159*(18), 2378–2398.

Fernández, A., del Jesus, M. J., & Herrera, F. (2009). Hierarchical fuzzy rule based classification systems with genetic rule selection for imbalanced data-sets. *International Journal of Approximate Reasoning, 50*(3), 561–577.

Galar, M., Fernandez, A., Barrenechea, E., Bustince, H., & Herrera, F. (2011). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42*(4), 463–484.

Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing* (pp. 878–887). Springer.

He, H., Bai, Y., Garcia, E. A., & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE international joint conference on neural networks (IEEE world congress on computational intelligence)* (pp. 1322–1328). IEEE.

Keel data set repository. https://sci2s.ugr.es/keel/imbalanced.php#subA. Accessed 28 January 2022.

Kubat, M., & Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Icml* (Vol. 97, pp. 179–186). Citeseer.

Li, M., Zhang, X., Thrampoulidis, C., Chen, J., & Oymak, S. (2021). Autobalance: Optimized loss functions for imbalanced data. *Advances in Neural Information Processing Systems, 34,* 3163–3177.

Liu, X. Y., Wu, J., & Zhou, Z. H. (2008). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39*(2), 539–550.

Mani, I., & Zhang, I. (2003). kNN approach to unbalanced data distributions: A case study involving information extraction. In: Proceedings of workshop on learning from imbalanced datasets (Vol. 126). ICML United States.

Mercier, M., Santos, M. S., Abreu, P. H., Soares, C., Soares, J. P., & Santos, J. (2018). Analysing the footprint of classifiers in overlapped and imbalanced contexts. In *International symposium on intelligent data analysis* (pp. 200–212). Springer.

Moniz, N., & Cerqueira, V. (2021). Automated imbalanced classification via meta-learning. *Expert Systems with Applications, 178,* 115011.

Murtagh, F., & Contreras, P. (2011). Methods of hierarchical clustering. arXiv preprint arXiv:1105.0121

Napierała, K., Stefanowski, J., & Wilk, S. (2010). Learning from imbalanced data in presence of noisy and borderline examples. In *International conference on rough sets and current trends in computing* (pp. 158–167). Springer.

Nickerson, A., Japkowicz, N., & Milios, E. E. (2001). Using unsupervised learning to guide resampling in imbalanced data sets. In *International workshop on artificial intelligence and statistics* (pp. 224–228). PMLR.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., et al. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research, 12,* 2825–2830.

Ribeiro, B., Cerqueira, V., Santos, R., & Gamboa, H. (2021). Layered learning for acute hypotensive episode prediction in the ICU: An alternative approach. In *2021 International Conference on e-Health and Bioengineering (EHB)* (pp. 1–4). IEEE.

Sharma, S., Bellinger, C., Japkowicz, N., Berg, R., & Ungar, K. (2012). Anomaly detection in gamma ray spectra: A machine learning perspective. In *2012 IEEE symposium on computational intelligence for security and defence applications* (pp. 1–8). IEEE.

Smith, M. R., Martinez, T., & Giraud-Carrier, C. (2014). An instance level analysis of data complexity. *Machine Learning, 95*(2), 225–256.

Stone, P., & Veloso, M. (2000). Layered learning. In *European conference on machine learning* (pp. 369–381). Springer.

Sutton, R. S., Precup, D., & Singh, S. (1999). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence, 112*(1–2), 181–211.

Tomek, I., et al. (1976). Two modifications of CNN.

Wang, S., & Yao, X. (2009). Diversity analysis on imbalanced data sets by using ensemble models. In *2009 IEEE symposium on computational intelligence and data mining* (pp. 324–331). IEEE.

Wu, J., Xiong, H., Wu, P., & Chen, J. (2007). Local decomposition for rare class analysis. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 814–823).