The CURE for Class Imbalance

Colin Bellinger[1(✉)], Paula Branco[2], and Luis Torgo[2]

[1] National Research Council of Canada, Ottawa, Canada
colin.bellinger@nrc-cnrc.gc.ca
[2] Dalhousie University, Halifax, Canada
{pbranco,ltorgo}@dal.ca

Abstract. Addressing the class imbalance problem is critical for several real world applications. The application of pre-processing methods is a popular way of dealing with this problem. These solutions increase the rare class examples and/or decrease the normal class cases. However, these procedures typically only take into account the characteristics of each individual class. This segmented view of the data can have a negative impact. We propose a new method that uses an integrated view of the data classes to generate new examples and remove cases. ClUstered REsampling (CURE) is a method based on a holistic view of the data that uses hierarchical clustering and a new distance measure to guide the sampling procedure. Clusters generated in this way take into account the structure of the data. This enables CURE to avoid common mistakes made by other resampling methods. In particular, CURE prevents the generation of synthetic examples in dangerous regions and undersamples safe, non-borderline, regions of the majority class. We show the effectiveness of CURE in an extensive set of experiments with benchmark domains. We also show that CURE is a user-friendly method that does not require extensive fine-tuning of hyper-parameters.

Keywords: Imbalanced domains · Resampling · Clustering

1 Introduction

Class imbalance is a problem encountered in a wide variety of important classification tasks including oil spill, fraud detection, action recognition, text classification, radiation monitoring and wildfire prediction [4,17,21,22,24,27]. Prior research has shown that class imbalance has a negative impact on the performance of the learned binary classifiers. This problem becomes even more difficult when the underlying distribution is complex and when the minority class is rare [14,26]. Given the frequency of imbalanced learning problems and the possibility for negative impacts on learning, the study of class imbalance and methods for handling it have become important research topics. Indeed, it has been recognised as one of the ten challenging problems in data mining research [29].

The solutions proposed by the research community to solve the class imbalance problem include special-purpose learning methods, pre-processing and post-processing methods. Pre-processing (or resampling) methods transform the original training set making it more suitable for learning the important class(es).

P. Kralj Novak et al. (Eds.): DS 2019, LNAI 11828, pp. 3–17, 2019.
https://doi.org/10.1007/978-3-030-33778-0_1

This is accomplished by following a certain strategy for up- or down-sampling cases. Resampling methods are popular due to their versatility, effectiveness and ease of application. Moreover, they enable the use of any standard learning tool.

In addition to suffering from the class imbalance problem, many real-world domains are also complex by nature. They potentially include noisy instances and sub-concepts that exacerbate the imbalance problem. This complexity dictates that it is important to consider the inherent structure of the data. Failure to do so may negatively impact the effectiveness of the resampling strategy. The problem relates to: (i) the removal of majority class instances from sparse regions of the domain; (ii) the generation of synthetic minority cases between minority class sub-concepts (clusters); (iii) the reinforcement of noisy instances; and/or (iv) the obfuscation of overlapping regions.

To address these issues, we propose the ClUstered REsampling (CURE) method. CURE uses hierarchical clustering with a new class-sensitive distance measure prior to the resampling process. This allows the extraction of essential structural information that is used to guide the resampling. The advantages of this approach are: (i) meaningful clusters of the minority class are emphasised: (ii) the generation of minority class cases is avoided in error-prone regions between sub-concepts; (iii) only "safe" majority class samples are undersampled (*i.e.,* borderline cases are not removed.) In an extensive set of experiments, we show that the CURE algorithm is effective for tackling the class imbalance problem. We also show that CURE does not requires extensive fine-tuning of hyper-parameters to achieve good performance.

This paper is organised as follows. Section 2 provides an overview of the related work. In Sect. 3 the CURE algorithm is described. The results of an extensive experimental evaluation are presented and discussed in Sect. 4. Finally, Sect. 5 presents the main conclusions of the paper.

2 Related Work

Numerous resampling methods have been proposed and applied to address imbalanced classification problems [5]. Random oversampling and random undersampling (e.g. [16]) are the classic approaches to handling imbalance. They are well-known to suffer from the risk of overfitting the minority samples and discarding informative cases, respectively. The SMOTE algorithm [8] incorporates oversampling and undersampling, and was proposed to overcome the issues of over- and under-sampling. It attempts to do so by interpolating new synthetic instances between nearest neighbours rather than replicating instances of the minority class. Two key issues with SMOTE are: (a) it does not account for the structure of the training data when performing the under and oversampling, and (b) it uniformly applies oversampling and undersampling. On complex data domains, this generation process can reinforce noise and increase the class overlap.

Many variations of SMOTE have been proposed to either clean the data after synthetic oversampling or to preemptively avoid generating instances that would negatively impact classifier performance [3,7,11,12]. For instance, Tomek links

(examples from different classes that are each other closest neighbours) can be removed from the training set after the application of SMOTE [3]. ADASYN [13] and Borderline-SMOTE [12] are examples of methods that apply SMOTE only in specific regions of the domain that are considered useful. ADASYN generates more synthetic examples from minority class cases that have more nearest neighbours from the majority class. Borderline-SMOTE generates more examples near the minority class border. However, Borderline-SMOTE applies uniform undersampling and may generate new cases between subconcepts of the minority class while ADASYN uses the local structure disregarding the global structure of the data. Resampling with a neighbourhood bias [6] is an alternative that introduces a bias in the selection of seed cases for both over and undersampling based on the class distribution in the local neighbourhood of each instance. Different biasing modes are proposed allowing to reinforce the most frontier or the most safe cases. Our proposal advances this idea by replacing the need for users to specify the k value necessary for the k-nearest neighbours computation, which is applied homogeneously across all instances and may be difficult to determine *a-priori*. Alternatively, we utilise hierarchical clustering that automatically finds variable sized clusters in the underlying structure of the data for resampling.

Previous research has applied clustering plus random oversampling, clustering plus random undersampling, and clustering plus synthetic oversampling [2,15,18–20,28,30]. Of these methods, our proposal is most closely related to [30]. Whereas the other methods only cluster one class, our method and that of Yen *et* al. [30], clusters the complete training sets, and use the class distribution in each cluster to inform if, and how much, resampling should be applied. By clustering both classes instead of just one, we acquire a more complete view of the data structure. The work of Yen *et* al. [30], uses k-means clustering which has important limitations such as requiring the *a-priori* knowledge of the correct number of clusters. By using hierarchical clustering, we are able to dynamically discover the sub-clusters (clusters at different levels of the hierarchy) that best address our resampling objectives. In addition, our method differs in the fact that it applies both undersampling and synthetic oversampling which inflates the minority class space while smoothing over-represented concepts of the majority class.

3 The CURE Method

3.1 Overview

In this section, we present the ClUstered REsampling (CURE) method. The key feature of CURE is that it utilises the intrinsic structure of the training data from all of the problem classes to decide where and how to apply resampling. This way, CURE avoids resampling mistakes incurred by SMOTE-based methods. In particular, CURE reduces the risk of:

- Synthesising minority class samples deep inside the majority class space; and,
- Naively undersampling informative instances in the majority class.

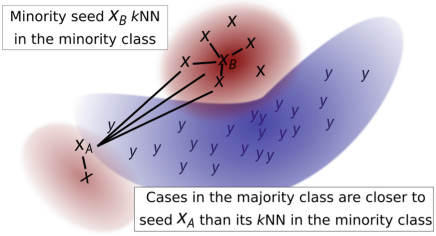


Fig. 1. Illustration of the intrusion into the majority class caused by SMOTE. (Color figure online)

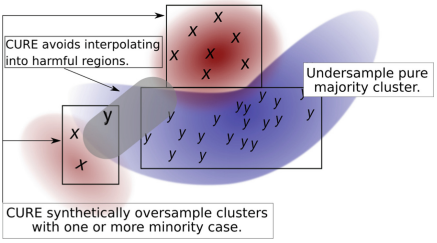


Fig. 2. Illustration of the synthetic oversampling of natural minority class groupings discovery of CURE.

The oversampling issue of SMOTE-based methods is demonstrated in Fig. 1. Here, the nearest neighbours of X_B are all minority class examples and thus interpolating between them is safe. However, between X_A and some of its nearest minority class neighbours, there is an area populated with majority class examples. Interpolating between these neighbours risks generating new synthetic minority class case in the majority class space (the blue region).

The undersampling issue is highlighted in Fig. 3. Here, the resampler naïvely discards some user-specified percentage, p, of the majority class samples (the removed samples are shown as grey y) in order to balance the training set. The random removal process risks the loss of information from the edge of the majority class region, which could have a significant negative impact in the learned decision boundary.

CURE avoids the over/undersampling issues discussed above by ensuring that instances are generated in, and removed from, safe regions of the data-space. This is achieved by applying hierarchical clustering and then resampling each cluster in a manner that is determined by the class makeup of the cluster.

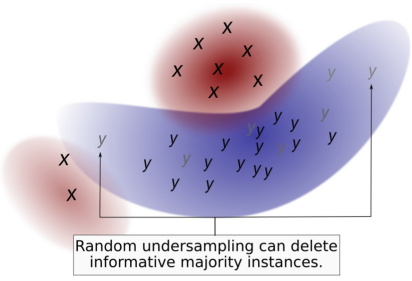


Fig. 3. Illustration of the removal of informative majority samples via random undersampling.

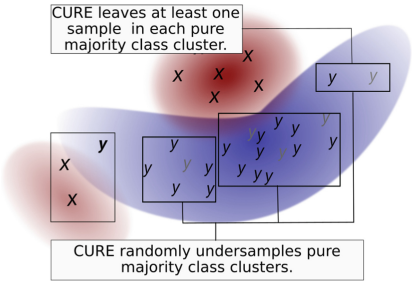


Fig. 4. Illustration of CURE keeping potentially information samples in mixed and small majority class clusters.

Figure 2 illustrates where and how resampling is applied to clusters involving minority class samples. Specifically, interpolation is only applied between minority class instances in the same cluster. This avoids the generation of samples deep inside the majority class (grey zone in figure). Figure 4 demonstrates how CURE randomly undersamples a percentage, p, of instances from each pure majority class cluster, rather than at random from the complete set of majority class instances. After undersampling, CURE will always leave at least one sample in each cluster to avoid wiping out information about edge cases and sub-concepts.

3.2 Hierarchical Clustering

A hierarchical clustering is formed by successively merging instances that are similar to each other. At the bottom of this hierarchy, we have the individual training cases, and at the top node we have a single cluster containing all of the cases. In between these extremes we have different groupings of the training data. Thus, the hierarchy specifies a set of possible clusterings of the data, where the clusters near the bottom of the hierarchy are smaller and more specific, and those nearer the top are larger and more general. It is up to the users to determine which clustering is best for their objectives.

The requirement to identify the "best" clusters from the hierarchy is a limitation in many pure clustering applications. For our purposes, however, it means we do not have to specify the number of clusters a-priori. Rather, we develop a method to automatically discover the clusters in the constructed hierarchy that are appropriate for resampling.

To produce the cluster hierarchy:

1. The pair-wise distance between each sample is calculated; and
2. A hierarchy is constructed by agglomeratively merging similar clusters.

The Ward variance minimisation algorithm [25] is used to construct the linkages in the hierarchy because it minimises the total within-cluster variance. This objective is appropriate for our goal of finding concise sub-concepts in the data to apply informed resampling on.

Given the set of clusters (also known as a forest) C_i at level i in the tree constructed thus far, the Ward variance minimisation algorithm search for clusters s and t in C_i that have the minimum variance according to the Ward metric. The clusters s and t are then merged to form a new cluster $w = \{s \cup t\}$ at level $i - 1$. The linkage process halts when all samples are merged into a single cluster.

3.3 Supervised Distance Measure

Clustering is typically an unsupervised process. We postulate, however, that the discovery of natural groupings in the training data for the purposes of resampling should not be unsupervised. Our hypothesis is that the class labels should have some influence on the cluster formation, but this influence should not be absolute.

Given a seed instance $I_1 = \langle \mathbf{x}_1, A \rangle$ and two query instances $I_2 = \langle \mathbf{x}_2, A \rangle$ and $I_3 = \langle \mathbf{x}_3, B \rangle$, where Euclidean$(\mathbf{x}_1, \mathbf{x}_2)$ is equal to Euclidean$(\mathbf{x}_1, \mathbf{x}_3)$, then I_2 should be considered to be more similarly to I_1 because it is from the same class. Alternatively, if Euclidean$(\mathbf{x}_1, \mathbf{x}_3)$ is significantly less than Euclidean$(\mathbf{x}_1, \mathbf{x}_2)$, then I_3 should be considered to be more similarity regardless of its different class association.

To achieve this, we propose a new supervised measure named Distance with Class Label Integration ($DCLI_\alpha$). The $DCLI_\alpha$ measure is based on a standard user selected distance metric (d) and a parameter α that controls the importance of matching class labels. The $DCLI_\alpha$ measure is defined as,

$$DCLI_\alpha(\langle \mathbf{x}_i, y_i \rangle, \langle \mathbf{x}_j, y_j \rangle) = \begin{cases} m + \alpha(d(\mathbf{x}_i, \mathbf{x}_j) - m) & if\ y_i = y_j \\ d(\mathbf{x}_i, \mathbf{x}_j) & if y_i \neq y_j \end{cases} \tag{1}$$

where $\langle \mathbf{x}_i, y_i \rangle$ represents an example with feature vector $\mathbf{x}_i$ and target class y_i, d is a user selected distance metric, parameter $\alpha \in [0, 1]$ controls the influence of the class labels in the $DCLI_\alpha$ distance measure and m is the minimum distance between instances in the training set measured using metric d.

The parameter α in Eq. 1 has the effect of weighting the significance of the class label agreement, i.e. instances with matching class labels are brought slightly closer together than their respective distances, d. Specifically, the $DCLI_\alpha$ distance between two instances $\mathbf{x}_i$, $\mathbf{x}_j$ with matching class labels is equal to some point, p, between $d(\mathbf{x}_i, \mathbf{x}_j)$ and the minimum distance is the data set $\arg\min_{\mathbf{x}_l, \mathbf{x}_k \in D} m = d(\mathbf{x}_l, \mathbf{x}_k)$. The proximity of p to either extreme is controlled by the α parameter. In this paper, we have used the Euclidean distance for parameter d in $DCLI_\alpha$. Figure 5 shows the effect of the α parameter on the $DCLI_\alpha$ distances for instances with matching class labels ($\mathbf{x}_1$, and $\mathbf{x}_2$), and instances with mismatched class labels ($\mathbf{x}_1$, and $\mathbf{x}_3$).

To summarise, the purpose of the measure is to promote the clustering of sparse groups of minority samples, even when a subset of those samples is slightly closer to the majority class. In Fig. 5, instances $\langle \mathbf{x}_1, A \rangle$ and $\langle \mathbf{x}_2, A \rangle$ will be linked in the hierarchy before $\langle \mathbf{x}_3, B \rangle$ for $\alpha < 0.8$. We discuss the sensitivity of α in Sect. 4.2.

3.4 CURE Algorithm

As previously stated, the CURE method consists of constructing a hierarchy using our proposed $DCLI_\alpha$ measure, and then automatically extracting clusters from the hierarchy for resampling. For clarity, we refer to the sets of instances at each level of the hierarchy as groups or groupings, and we refer to the subset of these groups automatically identified for resampling as clusters.

The groupings for each level of the hierarchy are stored in a data structure along with the corresponding intra-cluster distances, which we use for cluster formation. Each instance in the training set is assigned to a cluster defined by the

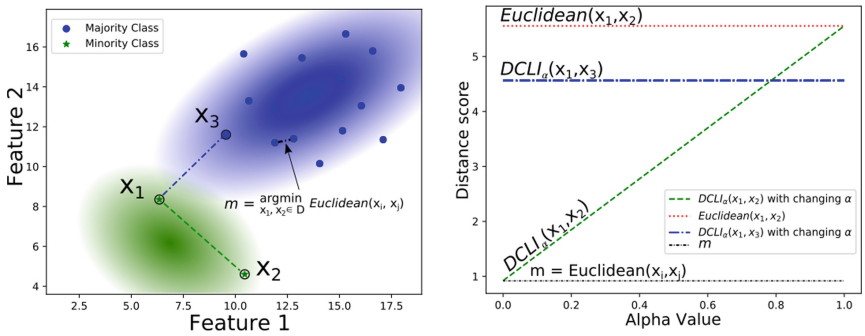


Fig. 5. Illustration of the impact of the α parameter on the $DCLI_\alpha$ score.

largest grouping to which it belongs that has an intra-cluster distance less than the threshold $\tau = \mu + s \times \sigma$, where μ and σ are the mean and standard deviation of the intra-cluster distances, and s is the number of standard deviations above the mean to set the threshold. Empirically, we found the intra-cluster distances to be approximately log-normal, thus it makes statistical sense to set the threshold in this manner.

We view the distribution of intra-cluster distances as a proxy for the overall spread of the data (variance is distances between training samples). As a result, s should be set large enough to represent the variance in single sub-concepts (natural groupings) so as to form clusters around these sub-concepts, but not so large as to join multiple sub-concepts into one cluster. The setting of the s parameter is simplified by the log-normal assumption. We postulate that approximately one standard deviation above the mean should achieve the required balance because it covers most of the variance in the data, whilst excluding exceptional levels of spread. The sensitivity of s is discussed in more detail in Sect. 4.2. The details of the cluster method are shown in Algorithm 1, and Fig. 6 illustrates the process of generating these clusters.

We define three cluster composition for resampling: (i) the cluster includes only majority class cases; (ii) those with exactly one minority class case and zero or more majority class instances; and (iii) those with more than one minority class case and zero or more majority class. If a cluster contains more than one minority class case, we will interpolate between them generating new synthetic cases and will maintain the majority class examples. When the cluster contains exactly one minority class case, synthetic cases are generated by applying Gaussian jitter to it. Finally, if the cluster is formed exclusively by majority class examples, this means we randomly undersample the cluster. Algorithm 2 provides an high level overview of the proposed CURE method.

In summary, the main idea of CURE is to carry out case generation and undersampling inside regions of the input space that are safer. These regions are found by taking into account the intrinsic structure of the training data.

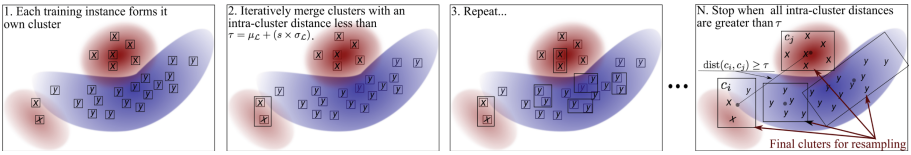


Fig. 6. Illustration of clusters generation using hierarchical clustering (Algorithm 1). The resampling strategy to apply in each cluster is based on the cluster examples.

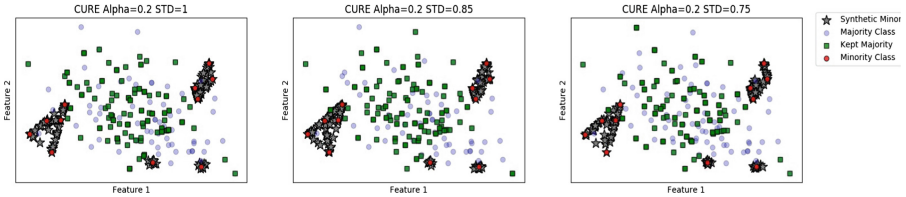


Fig. 7. Impact of changing CURE method hyper parameters in an artificial data set.

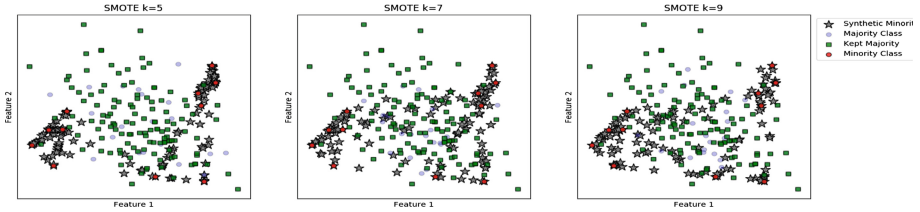


Fig. 8. Impact of changing the number of nearest neighbours considered in SMOTE algorithm in an artificial data set.

We achieve this using a new distance measure in the context of a hierarchical clustering process.

To better understand the way CURE avoids unsafe oversampling when compared to SMOTE, we prepared the 2-dimensional artificial data[1] in Figs. 7 and 8. These figures show the behaviour of each method with respect to their main hyper parameters. These figures illustrate that CURE is capable of detecting safe regions as opposed to SMOTE that generates new cases in regions that belong to the majority class.

[1] This is a hand curated 2-dimensional data set developed to demonstrate the strengths of CURE.

Algorithm 1. Generation of Clusters

Input: $\mathcal{D}$ - a classification data set
$\alpha \in [0, 1]$ - weights the influence of the class labels in DCLI distance
s - threshold on the standard deviation considered during clusters formation
Output: C - the clusters

function GenClusters($\mathcal{D}, \alpha, s$)
  $\mathcal{M}_{\mathcal{D}} \leftarrow$ pairwise distance matrix using $DCLI_\alpha$ measure
  $Z \leftarrow$ agglomerative linkage tree calculated over $\mathcal{M}_{\mathcal{D}}$
  $\mathcal{L} \leftarrow$ log transform of the inter-cluster distances obtained in Z
  $\mu_{\mathcal{L}} \leftarrow$ mean of the inter-cluster distances in $\mathcal{L}$
  $\sigma_{\mathcal{L}} \leftarrow$ standard deviation of the inter-cluster distances in $\mathcal{L}$
  $\tau \leftarrow \mu_{\mathcal{L}} + s \times \sigma_{\mathcal{L}}$ ▷ maximum inter-cluster distance for cluster formation
  $C \leftarrow$ Form clusters using Z s.t. the inter-cluster distances of the new clusters is less or equal to τ
  return C
end function

Algorithm 2. CURE Algorithm

Input: $\mathcal{D}$ - a classification data set
S_{min}, S_{maj} - number of minority and majority class instances to obtain in the new data set
α - class labels weight parameter in $DCLI_\alpha$ distance
s - threshold on the standard deviation considered during clusters formation
Output: $\mathcal{D}'$ - new resampled data set

function CURE($\mathcal{D}, S_{min}, S_{maj}, \alpha, s$)
  $k \leftarrow S_{min}/|$minority class instances in $\mathcal{D}|$▷ minority class instances to generate for each instance
  $q \leftarrow [1 - (S_{maj}/|$majority class instances in $\mathcal{D}|)] \times 100$ ▷ % of majority class examples to remove
  $C \leftarrow$ GenClusters($\mathcal{D}, \alpha, s$)
  $\mathcal{D}' \leftarrow \mathcal{D}$
  for each cluster c_i in C **do**
    if c_i contains only majority class instances **then**
      $\mathcal{D}' \leftarrow \mathcal{D}' \backslash \{$random selection of q% of the instances in $c_i\}$
    else if c_i contains exactly one minority class instance **then**
      new $\leftarrow$ generate k synthetic cases using Gaussian jitter
      $\mathcal{D}' \leftarrow \mathcal{D}' \bigcup new$
    else ▷ several minority class instances in the cluster
      new $\leftarrow$ generate k synthetic cases by interpolating minority cases in c_i
      $\mathcal{D}' \leftarrow \mathcal{D}' \bigcup new$
    end if
  end for
  return $\mathcal{D}'$
end function

4 Experimental Evaluation

4.1 Materials and Methods

We have selected a diverse set of 29 benchmark data sets from the KEEL repository [1]. In order to consider the effectiveness of CURE at different levels of absolute and relative imbalance, we process each original data set into three new versions for the purpose of our experiments. The new versions contain 10, 30 and 50 minority class cases, for which we use the notation of IR10, IR30 and IR50 to refer to these respectively. We conducted our experiments on 87 data sets (29×3). The average imbalance ratios ($|min|/|maj|$) of the three versions range between 0.186 and 0.037. Therefore, they include a wide range of absolute and relative imbalance levels. Table 1 displays the main characteristics of the used data sets.

Table 1. Data sets name, dimensions (Dim), majority class cases ($|maj|$), and imbalance ratios when using 50, 30 and 10 minority class cases (IR50, IR30 and IR10).

Data set	mammographic0vs1	letterBFGMvsOPSTUW	dermatology1-3vs4-6	contraceptive1vs2-3	winequality-red1-5vs6-10	movement-libras1-9vs10-15	optdigits1-3vs5-8	page-blocks1vs2-4	texture2-6vs7-14	marketing1-2vs7-9	shuttle2-3vs5-7	segment1-3vs4-7	satimage1-2vs5-7	letterBFGMvsOPSTUW	penbased3-4vs7-8
Dim	5	16	34	9	11	90	64	10	40	13	9	19	36	16	16
|maj|	214	1560	121	422	428	108	1134	2678	1750	1298	1645	660	1118	1560	1671
IR50	0.234	0.032	0.413	0.118	0.117	0.463	0.044	0.019	0.029	0.039	0.030	0.076	0.045	0.032	0.030
IR30	0.140	0.019	0.248	0.071	0.070	0.278	0.026	0.011	0.017	0.023	0.018	0.045	0.027	0.019	0.018
IR10	0.047	0.006	0.083	0.024	0.023	0.093	0.009	0.004	0.006	0.008	0.006	0.015	0.009	0.006	0.006

Data set	bands1vs0	ionosphere0vs1	vowel1-5vs6-10	wisconsin0vs1	ring0vs1	magic0vs1	cleveland0vs1-4	coil2000vs1	heart0vs1	spambase0vs1	monk-20vs1	wdbc0vs1	vehicle1-2vs3-4	thyroid2vs3	**Average**
Dim	19	33	13	9	20	10	13	85	13	57	6	30	18	21	**25.3**
|maj|	115	112	270	222	1019	2802	80	4618	75	1393	114	179	215	2967	**1053.4**
IR50	0.435	0.446	0.185	0.225	0.049	0.018	0.625	0.011	0.667	0.036	0.439	0.279	0.233	0.017	**0.186**
IR30	0.261	0.268	0.111	0.135	0.029	0.011	0.375	0.006	0.400	0.022	0.263	0.168	0.140	0.010	**0.111**
IR10	0.087	0.089	0.037	0.045	0.010	0.004	0.125	0.002	0.133	0.007	0.088	0.056	0.047	0.003	**0.037**

We compare the performance of CURE to 7 state-of-the-art resampling methods, namely, random undersampling (RUS), random oversampling (ROS), the combined application of RUS and ROS simultaneously (ROS + RUS), adaptive synthetic oversampling (ADASYN), SMOTE algorithm, Borderline-SMOTE

(Borderline), and SMOTE with the removal of Tomek links (SMOTE + TL), and no resampling (None).

Support vector machines (SVM) with the radial basis function (RBF) kernel is selected for classification, because it is an effective non-parametric method that can be trained on small amounts of data relative to deep learning methods. Automatic parameter tuning is conducted after resampling via random search over the $\gamma \in [0.001, 20]$ and $C \in [0.001, 20]$. This promotes the discovery of the best SVM model for the resampled training set.

The evaluation is performed via 5×2-fold cross validation, because it has been observed that it has a lower probability of issuing a Type I error [9]. The performance is reported in terms of the geometric mean (g-mean) [16] and the F_β measure [23]. Given the accuracy on the target class a^+ and the accuracy on the outlier class a^-, the g-mean for a classification model f on test set X is calculated as: $\text{g-mean}_{f(X)} = \sqrt{a^+ \times a^-}$. This metric enables us to evaluate whether the resampling methods are helping to improving the performance on the minority class, whilst having minimal impact on the majority class. The F_β measure expresses the harmonic mean of precision and recall. We used $\beta = 1$ which assigns the same weight to precision and recall measures. The F_β measure is popular in imbalanced domains as it provides a reliable assessment of the models effectiveness on the minority class (e.g. [10]).

Regarding the CURE algorithm, we have set parameters S_{min} and S_{maj} (c.f. Algorithm 2) as follows: $S_{min} = |min| + 0.5 \times |maj|$ and $S_{maj} = 0.5 \times |maj|$, where $|maj|$ and $|min|$ correspond respectively to the number of minority and majority class cases in the original data set. We apply this policy to the alternative resampling methods as well. To ensure an easy replication of our work all code and data sets used in the experiments are available at https://ltorgo.github.io/CURE/.

4.2 Results and Discussion

Aggregated Results: The first set of experiments focuses on the effectiveness of CURE for tackling the class imbalance problem. Figures 9 and 10 show the number of times each resampling method was the best (won) in terms of the average results during cross validation. The results are grouped according to the number of minority cases. Thus, Winner 10 FM in Fig. 9 specifies the number of times each resampling method won on the datasets with 10 minority class cases.

The figures illustrate that CURE has the highest number of wins in comparison to the 8 tested alternatives. Regarding the F_1 measure, CURE achieves 7, 12 and 10 wins for the IR10, IR30 and IR50 data sets respectively. The alternative that shows the most competitive results is Borderline with only 4, 3, and 2 wins for IR10, IR30 and IR50 data sets. Regarding the performance on G-Mean measure we observe that the advantage displayed by CURE method is overwhelming with 8, 14 and 12 wins on IR10, IR30 and IR50 data sets respectively. In this setting, the method showing the second most competitive performance is ADASYN displaying 6, 3 and 4 wins for the IR10, IR30 and IR50 data sets.

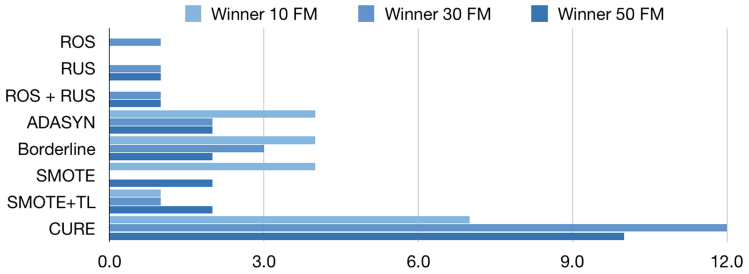


Fig. 9. Number of wins obtained by each tested resampling approach aggregated by data sets with 10, 30 and 50 minority class cases, for the F_1 measure.

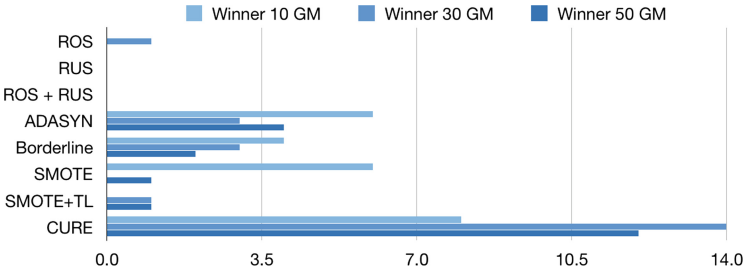
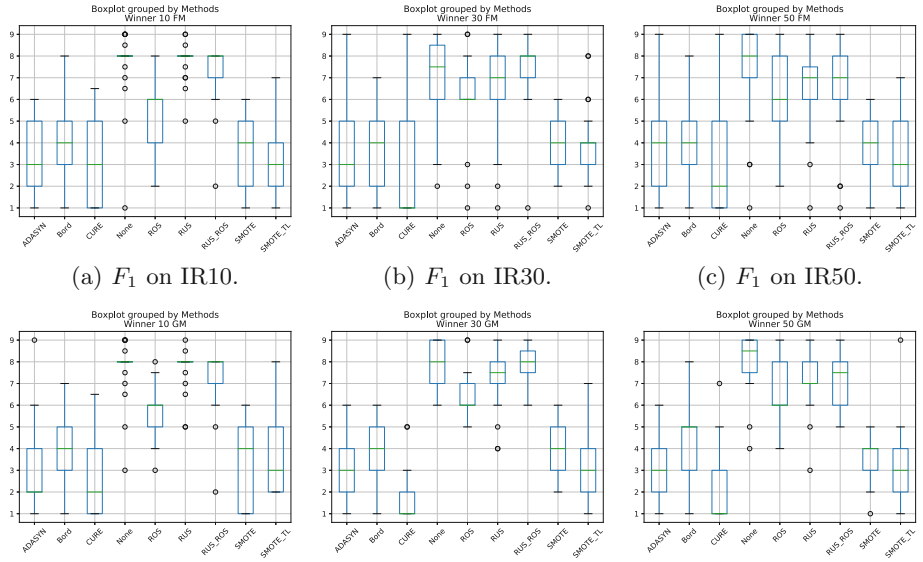


Fig. 10. Number of wins obtained by each tested resampling approach aggregated by data sets with 10, 30 and 50 minority class cases, for the G-Mean metric.



(a) F_1 on IR10. (b) F_1 on IR30. (c) F_1 on IR50.

(d) G-Mean on IR10. (e) G-Mean on IR30. (f) G-Mean on IR50.

Fig. 11. Ranks of each resampling approach on the three data set versions, IR10, IR30 and IR50, for both the F_1 (top row) and G-Mean (bottom row) metrics.

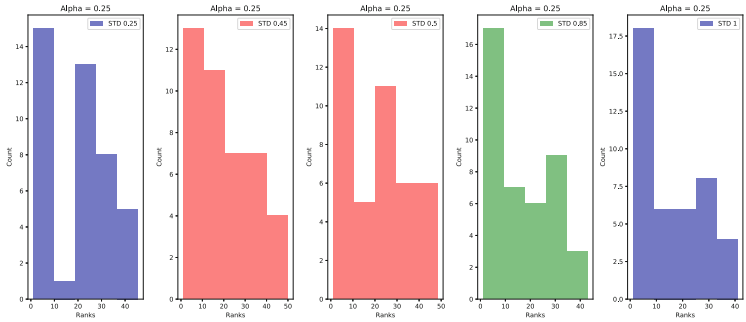


Fig. 12. CURE rankings for IR10 data set versions for: $\alpha = 0.25$ and $0.25 \leq s \leq 1$.

Figure 11 displays the boxplot of the rankings achieved by each resampling method on each performance assessment metric by data set version. The rankings shown were obtained using the average of the cross-validation results. These results clearly show the advantage of using CURE. Overall, the results obtained demonstrate the versatility of our proposed method over different class ratios, and demonstrates the benefit of utilising the inherent structure of data for resampling.

Hyper-parameter Sensitivity: CURE has two parameters: α and s. The α parameter determines the influence of matching class labels on the distance score ($DCLI_\alpha$). The second parameter, s, is number of standard deviations used in threshold for cluster formation.

Figure 12 shows the variation in the rankings of CURE method, on data sets from IR10 version, for parameter α fixed at 0.25 and parameter s ranging between 0.25 and 1. Due to space constraints, we provide more figures that show the results for other parameter variations in: https://ltorgo.github.io/CURE/. The results obtained for $s \approx 1$ are concentrated around the lower (and thus better) rankings. As stated in Sect. 3, setting $s \approx 1$ makes good statistical sense, as well. The α parameter results suggest that values of α between 0.1 and 0.25 provides the best overall results. The good performance of CURE allied to this user-friendly perspective make CURE an excellent approach to tackle the problem of imbalanced domains.

5 Conclusion

We presented CURE, a novel method that uses the inherent structure of data to discover safer regions for resampling. These regions are found using a new class-sensitive distance measure and hierarchical clustering. A suitable resampling strategy is applied inside each cluster based on its characteristics. CURE aims at: (i) avoiding the generation of synthetic cases in unsafe regions of the data space,

and (ii) preventing the removal of informative majority class cases. State-of-the-art resampling methods fail these goals because they only consider a segmented view of the data as opposed to CURE that considers a holistic view of the data.

We demonstrate the effectiveness of CURE on a diverse set of 29 benchmark domains and 87 imbalanced classification datasets. The results show that CURE has an advantage over 7 state-of-the-art alternatives for resampling methods in terms of the g-mean and F_β measures on 5×2-fold cross-validation. In addition, we show that the key parameters of CURE, α and s are easy to set and perform well over a large range of values. Thus, CURE does not require extensive hyper-parameter tuning.

As future work, we plan to demonstrate CURE in multi-class domains, and further improve the method for automatically detect the safe regions. Moreover, we also plan to explore the application of other resampling methods inside each safe region based on the regions characteristics.

References

1. Alcalá-Fdez, J., et al.: Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J. Multiple-Valued Logic Soft Comput. **17**, 255–287 (2011)
2. Barua, S., Islam, M.M., Murase, K.: A novel synthetic minority oversampling technique for imbalanced data set learning. In: Lu, B.-L., Zhang, L., Kwok, J. (eds.) ICONIP 2011. LNCS, vol. 7063, pp. 735–744. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24958-7_85
3. Batista, G.E.A.P.A., Prati, R.C., Monard, M.C.: A study of the behavior of several methods for balancing machine learning training data. SIGKDD Explor. Newsl. **6**(1), 20–29 (2004)
4. Bellinger, C., Drummond, C., Japkowicz, N.: Manifold-based synthetic oversampling with manifold conformance estimation. Mach. Learn. **107**(3), 605–637 (2018)
5. Branco, P., Torgo, L., Ribeiro, R.P.: A survey of predictive modeling on imbalanced domains. ACM Comput. Surv. (CSUR) **49**(2), 31 (2016)
6. Branco, P., Torgo, L., Ribeiro, R.P.: Resampling with neighbourhood bias on imbalanced domains. Expert Syst. **35**(4), e12311 (2018). https://doi.org/10.1111/exsy.12311
7. Bunkhumpornpat, C., Sinapiromsaran, K., Lursinsap, C.: Safe-level-SMOTE: safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. In: Theeramunkong, T., Kijsirikul, B., Cercone, N., Ho, T.-B. (eds.) PAKDD 2009. LNCS (LNAI), vol. 5476, pp. 475–482. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01307-2_43
8. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
9. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput. **10**(7), 1895–1923 (1998)
10. Estabrooks, A., Japkowicz, N.: A mixture-of-experts framework for learning from imbalanced data sets. In: Hoffmann, F., Hand, D.J., Adams, N., Fisher, D., Guimaraes, G. (eds.) IDA 2001. LNCS, vol. 2189, pp. 34–43. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44816-0_4

11. Fernández, A., Garcia, S., Herrera, F., Chawla, N.V.: Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. J. Artif. Intell. Res. **61**, 863–905 (2018)
12. Han, H., Wang, W.-Y., Mao, B.-H.: Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. In: Huang, D.-S., Zhang, X.-P., Huang, G.-B. (eds.) ICIC 2005. LNCS, vol. 3644, pp. 878–887. Springer, Heidelberg (2005). https://doi.org/10.1007/11538059_91
13. He, H., Bai, Y., Garcia, E.A., Li, S.: ADASYN: adaptive synthetic sampling approach for imbalanced learning. In: IJCNN 2008, pp. 1322–1328. IEEE (2008)
14. He, H., Ma, Y.: Imbalanced Learning: Foundations, Algorithms, and Applications. Wiley, Hoboken (2013)
15. Jo, T., Japkowicz, N.: Class imbalances versus small disjuncts. ACM SIGKDD Explor. Newslett. **6**(1), 40–49 (2004). Special issue on learning from imbalanced datasets
16. Kubat, M., Matwin, S., et al.: Addressing the curse of imbalanced training sets: one-sided selection. In: ICML, Nashville, USA, vol. 97, pp. 179–186 (1997)
17. Lewis, D.D., Catlett, J., Hill, M.: Heterogeneous uncertainty sampling for supervised learning. In: International Conference on Machine Learning, pp. 148–156 (1994)
18. Lim, P., Goh, C.K., Tan, K.C.: Evolutionary cluster-based synthetic oversampling ensemble (eco-ensemble) for imbalance learning. IEEE Trans. Cybern. **47**(9), 2850–2861 (2017)
19. Lin, W.C., Tsai, C.F., Hu, Y.H., Jhang, J.S.: Clustering-based undersampling in class-imbalanced data. Inf. Sci. **409–410**, 17–26 (2017)
20. Nickerson, A.S., Japkowicz, N., Milios, E.: Using unsupervised learning to guide resampling in imbalanced data sets. In: Proceedings of the Eighth International Workshop on AI and Statistics, p. 5 (2001)
21. Oliveira, M., Torgo, L., Santos Costa, V.: Predicting wildfires. In: Calders, T., Ceci, M., Malerba, D. (eds.) DS 2016. LNCS (LNAI), vol. 9956, pp. 183–197. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46307-0_12
22. Provost, F., Fawcett, T.: Robust classification for imprecise environments. Mach. Learn. **42**(3), 203–231 (2001)
23. Rijsbergen, C.V.: Information retrieval, 2nd edition. Department of computer science, University of Glasgow (1979)
24. Slama, R., Wannous, H., Daoudi, M., Srivastava, A.: Accurate 3D action recognition using learning on the grassmann manifold. Pattern Recogn. **48**(2), 556–567 (2015)
25. Ward Jr., J.H.: Hierarchical grouping to optimize an objective function. J. Am. Stat. Assoc. **58**(301), 236–244 (1963)
26. Weiss, G.M.: Mining with rarity: a unifying framework. SIGKDD Explor. Newsl. **6**(1), 7–19 (2004). https://doi.org/10.1145/1007730.1007734
27. Williams, D., Myers, V., Silvious, M.: Mine classification with imbalanced data. IEEE Geosci. Remote Sens. Lett. **6**(3), 528–532 (2009)
28. Wu, J., Xiong, H., Chen, J.: COG: local decomposition for rare class analysis. Data Min. Knowl. Discov. **20**(2), 191–220 (2010)
29. Yang, Q., et al.: 10 challenging problems in data mining research. Int. J. Inf. Tech. Decis. **5**(4), 597–604 (2006)
30. Yen, S.J., Lee, Y.S.: Cluster-based under-sampling approaches for imbalanced data distributions. Expert Syst. Appl. **36**(3), 5718–5727 (2009)