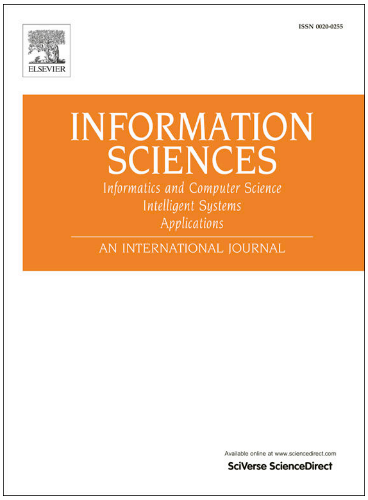

Assessing the Data Complexity of Imbalanced Datasets

Victor H. Barella, Luís P. F. Garcia, Marcilio C. P. de Souto, Ana C. Lorena, André C. P. L. F. de Carvalho

PII: S0020-0255(20)31171-3
DOI: https://doi.org/10.1016/j.ins.2020.12.006
Reference: INS 16042

To appear in: *Information Sciences*

Received Date: 26 March 2020
Accepted Date: 7 December 2020

Please cite this article as: V.H. Barella, L.P. F. Garcia, M.C. P. de Souto, A.C. Lorena, A.C. P. L. F. de Carvalho, Assessing the Data Complexity of Imbalanced Datasets, *Information Sciences* (2020), doi: https://doi.org/10.1016/j.ins.2020.12.006

Assessing the Data Complexity of Imbalanced Datasets

Victor H. Barella[a], Luís P. F. Garcia[b], Marcilio C. P. de Souto[c], Ana C. Lorena[d], André C. P. L. F. de Carvalho[a]

[a]*Institute of Mathematics and Computer Sciences, University of São Paulo,*
Trabalhador São-carlense Av. 400, São Carlos, São Paulo 13560-970, Brazil
[b]*Computer Science Department, University of Brasília,*
Brasília, Distrito Federal ,70910-900, Brazil
[c]*Fundamental Computer Science Laboratory, University of Orleans*
Léonard de Vinci, B.P. 6759 F-45067 Orleans Cedex 2, France
[d]*Aeronautics Institute of Technology,*
Praça Marechal Eduardo Gomes, 50, São José dos Campos, São Paulo 12228-900, Brazil

Abstract

Imbalanced datasets are an important challenge in supervised Machine Learning (ML). According to the literature, class imbalance does not necessarily impose difficulties for ML algorithms. Difficulties mainly arise from other characteristics, such as overlapping between classes and complex decision boundaries. For binary classification tasks, calculating imbalance is straightforward, e.g., the ratio between class sizes. However, measuring more relevant characteristics, such as class overlapping, is not trivial. In the past years, complexity measures able to assess more relevant dataset characteristics have been proposed. In this paper, we investigate their effectiveness on real imbalanced datasets and how they are affected by applying different data imbalance treatments (DIT). For such, we perform two data-driven experiments: (1) We adapt the complexity measures to the context of imbalanced datasets. The experimental results show that our proposed measures assess the difficulty of imbalanced problems better than the original ones. We also compare the results with the state-of-art on data complexity measures for imbalanced datasets. (2) We analyze the behavior of complexity measures before and after applying DITs. According to the results, the difference in data complexity, in general, correlates to the predictive performance improvement obtained by applying DITs to the original datasets.

Keywords: Classification, Imbalanced dataset, Complexity Measures, Pre-processing techniques.

1. Introduction

In classification tasks, class imbalance is a disproportion of the number of instances from each class in the dataset. Although several articles report poor predictive performances of traditional Machine

Email addresses: victorhb@icmc.usp.br (Victor H. Barella), luis.garcia@unb.br (Luís P. F. Garcia), marcilio.desouto@univ-orleans.fr (Marcilio C. P. de Souto), aclorena@ita.br (Ana C. Lorena), andre@icmc.usp.br (André C. P. L. F. de Carvalho)

Learning (ML) algorithms when applied to these datasets [20, 13, 7, 25, 4, 1], Batista et al. [5] showed that imbalance is not a problem per se. In fact, it increases the adverse effect of other data intrinsic characteristics, such as class overlapping. Data topology characteristics, such as overlapping, linear separability, among others, are not easily measured. They have many more complex concepts and aggregate more information about the data than a simple class imbalance ratio.

Topological characteristics can be estimated by using data complexity measures, which were initially proposed by Ho and Basu [21] and extended by many other authors [22, 34, 26, 27]. Several studies investigate the use of these measures in classification tasks [31, 14, 30, 16], some of them for imbalanced datasets [29, 10, 12]. Although their use in imbalanced classification tasks seems straightforward, Barella et al. [3] showed that for artificial datasets, the complexity measures do not adequately represent the difficulties found in imbalanced datasets. To deal with this deficiency, the authors proposed modifications to these measures. The modifications consist of decomposing the data complexity for each class in the dataset. We investigate, using real imbalanced datasets, the effectiveness of those measures and the original ones. Additionally, we define them formally, make a package publicly available, and compare them with the state-of-the-art complexity measures for imbalanced datasets.

Data Imbalance Treatments (DITs) have been proposed to balance the number of instances between the dataset classes [13, 7, 25, 4, 1]. Furthermore, they can modify other characteristics of the datasets, which can affect their predictive performance. In this paper, we also investigate the relation between data complexity measures and predictive performance before and after applying DITs. Figure 1 illustrates DITs modifying characteristics of a dataset and affecting the predictive performance. In this figure, the solid box represents what the literature usually discusses, which is the improvement of predictive performance based on balancing the classes. In this study, we investigate the improvement of predictive performance based on the decrease of data complexity, represented by the dashed box in the figure.

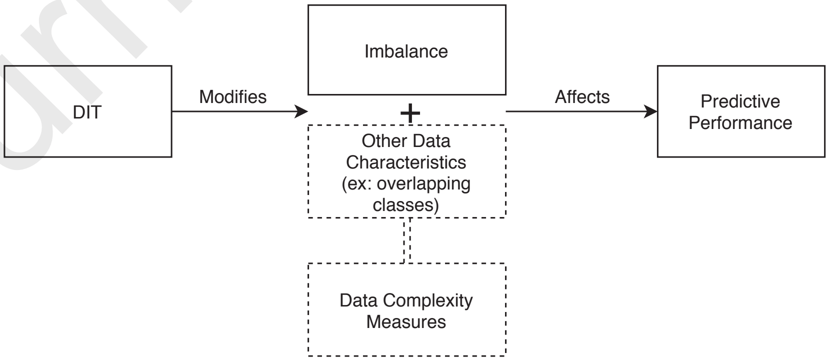

Figure 1: Diagram of DIT techniques modifying data characteristics and affecting predictive performance

We believe this paper will help to understand the difficulty that an imbalanced dataset may pose to any classification algorithm and how DITs can deal with this problem. Thus, the main contributions of this paper are:

1. Formally define the adapted complexity measures to imbalanced domains;
2. Show that the adapted data complexity measures assess the difficulty on real imbalanced datasets;
3. Show that the adapted data complexity measures assess the difficulty of a dataset before and after applying DITs.

This paper is organized in five sections. Section 2 describes the original complexity measures, how they are adapted to estimate the difficulty of each individual class, and the state-of-art on complexity measures for imbalanced datasets. Moreover, it presents the main DITs considered in this paper, as well as related works. Next, Section 3 presents the experimental designs for this study. We show and discuss the experimental results in Section 4. In Section 5, we stress the main contributions and limitations, and indicate future work directions.

2. Background

In this section, we describe the main data complexity measures and our proposed adaptations of them for DIT. We also describe the main pre-processing techniques found in the literature for imbalanced classification.

2.1. Data Complexity Measures and Adaptations

The original data complexity measures were proposed by Ho and Basu [21] and extended by many studies [22, 34, 26, 27]. Orriols-Puig et al. [34] implemented a package called `DCoL` (Data Complexity Library) and proposed generalizations of complexity measures for multiclass problems. Moreover, limitations remained and some were solved later by Lorena et al. [27], who surveyed, standardized and implemented the cutting edge measures in a revised R package called `ECoL` (Extended Complexity Library) [15]. In order to adapt them for the imbalance problem, Barella et al. [3] decomposed the measures to assess the complexity of each class separately. This subsection describes the original data complexity measures used in this paper, as defined by Lorena et al. [27], and their adaptations for estimating the difficulty of each class in an imbalanced dataset, proposed by Barella et al. [3] and formalized here. The aim is to decompose the measures per class, enabling us to assess classification difficulties from the perspective of the minority class.

To describe the measures, we consider a training set T with n instances, in which each instance is a pair $(\mathbf{x}_i, y_i)$, where $\mathbf{x}_i$ is a vector of characteristics (which we will call features) $\mathbf{x}_i = (x_{i1}, \dots, x_{im})$, m is the number of features and $y_i \in \{0, 1\}$. Consider also a function $c(T)$ whose output is the value of complexity measure c applied in dataset T, with $c(T) \in [0, 1]$. According to Lorena et al. [27], the higher the $c(T)$ value, the more complex the dataset. The complexity measures are organized into three main groups: feature overlapping, neighborhood information, and linear separability.

To illustrate the differences between the original and the adapted measures in balanced and imbalanced datasets, we use two artificial datasets. They are shown in Figure 2, where the classes were sampled from multivariate normal distributions. The class 0 represents the negative class, and the class 1 represents the positive class. Both classes have 1000 instances in the balanced datasets, and the class distribution in the imbalanced dataset is 1000 and 100 examples for class 0 and class 1, respectively. We show the values of the data complexity measures for the two datasets in addition to their detailed description.

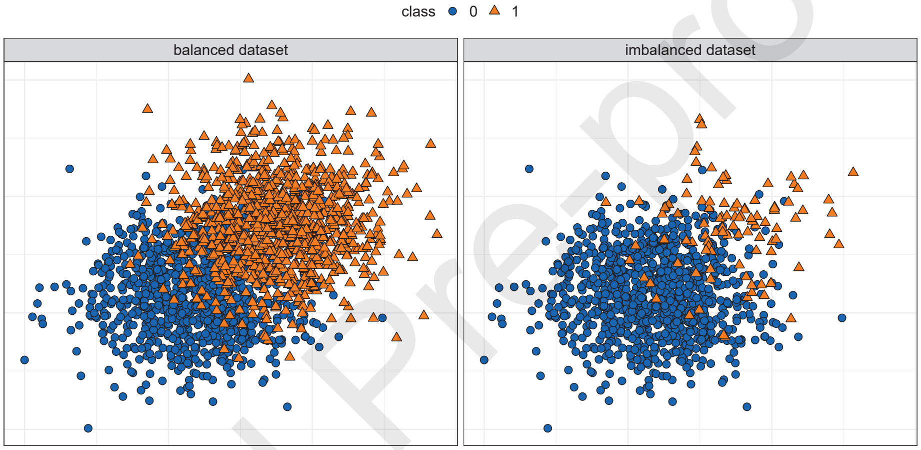

Figure 2: Example datasets to illustrate the differences between the measures

2.1.1. Feature overlapping measures

The feature overlapping measures assess the discrimination power of the predictive attributes. Most of them evaluate the features individually and the most discriminate feature is selected, while others use a combination of the individual feature assessments. The feature overlapping measures considered in this article are F1, F2, F3, and F4. The feature overlapping complexity measures are detailed next, as well as a description of the proposed adaptation.

- **F1: Maximum Fisher's discriminant ratio.**

  F1 computes the Fisher's discriminant ratio for each attribute. The aim is to assess how close the classes, for each of the features in the feature space. To do this, the measure considers the mean and variance values of each feature in each class.

$$F1(T) = argmax_{j=1}^{m}(f_j) \tag{1}$$

$$f_j = \frac{(\mu_{fc_0} - \mu_{fc_1})^2}{\sigma_{fc_0}^2 + \sigma_{fc_1}^2} \tag{2}$$

F1 is defined by Equation 1 for a problem with two classes, where μ_{jc_y} and σ_{jc_y} are, respectively, the mean and the variance of the values of the feature j in the objects from class y. F1 outputs the maximum f among all features. This measure has an unbounded limit interval, since the values are in the interval $[0, \infty[$. For normalization matters, Lorena et al. (2018) [27] applied Equation 3, where M is the value of the measure. The implementation in the package ECoL [15] also uses this value. Thus, we will also use it. This equation guarantees that the measured value is in the interval $]0, 1]$, whereby the more complex the dataset is, the higher the value.

$$M_{norm} = \frac{1}{M+1} \tag{3}$$

Since F1 relates two means and variances, it was not possible to adapt it and obtain similar information per class. Therefore, F1 is maintained in the experiments as initially proposed. We opted to include F1 in the analysis because its use is reported in previous papers dealing with imbalanced datasets, e.g., [29, 12, 10].

The F1 values for the datasets in Figure 2 are shown in Table 1. F1 assessed that the imbalanced dataset is more difficult than the balanced one.

Table 1: F1 values for the datasets in Figure 2

Dataset	**F1**
Balanced Dataset	0.58
Imbalanced Dataset	0.82

- **F2: Volume of overlap region**

F2 computes the volume of class overlapping regions, using the minimum and maximum values of each feature per class. It considers, for each feature, the range of possible values in which instances belonging to both classes can be found. It is calculated using Equation 4,

$$F2(T) = \prod_{i=1}^{l} \frac{max\{0, minmax(f_i) - maxmin(f_i)\}}{maxmax(f_i) - minmin(f_i)} \tag{4}$$

where:

$$minmax(f_i) = min(max(f_i^{c_0}), max(f_i^{c_1})) \tag{5}$$

$$maxmin(f_i) = max(min(f_i^{c_0}), min(f_i^{c_1})) \tag{6}$$

$$maxmax(f_i) = max(max(f_i^{c_0}), max(f_i^{c_1})) \tag{7}$$

$$minmin(f_i) = min(min(f_i^{c_0}), min(f_i^{c_1})) \tag{8}$$

The values $max(f_i^{c_j})$ and $min(f_i^{c_j})$ are the maximum and minimum values of feature f_i in a class c_j, respectively.

Thus, if the attribute ranges overlap in a region, this region is considered ambiguous regarding the attribute. Next, a product of the normalized size of the ambiguous regions for all attributes is output. As an example, suppose an attribute whose values for class 0 range between 0 and 1, and values for class 1 range between 0.75 and 1.25. The overlapping region for this attribute has size 0.25. Taking the full range of values for normalization, the final overlapping for this attribute is $\frac{0.25}{1.25} = 0.2$. F2 is zero if at least one of the attributes does not have any overlapping region and is equal to 1 when the classes are entirely overlapped for all attributes.

Classes may have different overlapping regions, and a single measure value may not represent the real complexity of the dataset, especially when the classes are imbalanced. Considering the previous example, although half of the class 1 range is inside the ambiguous region, F2 evaluates that only 20% of the attribute's range represents the ambiguous region. F2 tends to underestimate the complexity of the dataset with the smallest range, which can undermine the proper assessment of the minority class complexity. The proposed adaptation considers the impact of the overlapping volume per class. The difference between the original F2 and the adaptation is the division of the size of the ambiguous region of each attribute by the range of value for the class of interest, instead of the range of all values of the attribute. This is illustrated by Equation 9 for class c_1. Considering the previous example, F2 for class 0 would be $\frac{0.25}{1} = 0.25$ and F2 for class 1 would be $\frac{0.25}{0.5} = 0.5$.

$$F2_{c_1}(T) = \prod_{i=1}^{l} \frac{max(0, minmax(f_i) - maxmin(f_i))}{max(f_i^{c_1}) - min(f_i^{c_1})} \tag{9}$$

The F2 values for the datasets in Figure 2 are shown in Table 2. The original F2 assessed that the balanced and the imbalanced dataset with similar complexity. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 2: F2 values for the datasets in Figure 2

Dataset	**Original**	**Negative class**	**Positive class**
	F2	**F2**	**F2**
Balanced Dataset	0.33	0.52	0.56
Imbalanced Dataset	0.34	0.43	0.74

- **F3: Feature efficiency**

In F3, the number of instances inside the ambiguous region defines the inefficiency of a feature. The greater the amount of instances inside the ambiguous region, the more inefficient this feature is in separating the classes. Equation 10 illustrates how F3 is calculated.

$$F3(T) = \underset{i=1}{\overset{m}{min}} \frac{n_o(f_i)}{n} \tag{10}$$

In this equation, $n_o(f_i)$ returns the number of instances in the overlapping region for f_i, whose value is defined by:

$$n_o(f_i) = \sum_{j=1}^{n} I(x_{ji} > maxmin(f_i) \wedge x_{ji} < minmax(f_i)) \tag{11}$$

where I is the indicator function that returns 1 if its argument is true and 0, otherwise.

Similar to F2, F3 has a bias towards the majority class, since the whole minority class can be inside the ambiguous region and F3 can still be close to 1. The adaptation of F3 considers one class at a time. The F3 per class divides the number of instances from that class of interest inside the ambiguous region by the number of instances from the class only. In our adaptation, Equations 10 and 11 are changed to Equations 12 and 13 respectively, where n_{c_1} is the number of instances from class c_1 and $x_{ji}^{c_1}$ is the value of the j-th attribute from the i-th instance of class c_1.

$$F3_{c_1}(T) = \underset{i=1}{\overset{m}{min}} \frac{n_o^{c_1}(f_i)}{n_{c_1}} \tag{12}$$

$$n_o^{c_1}(f_i) = \sum_{j=1}^{n_{c_1}} I(x_{ji}^{c_1} > maxmin(f_i) \wedge x_{ji}^{c_1} < minmax(f_i)) \tag{13}$$

The F3 values for the datasets in Figure 2 are shown in Table 3. The original F3 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 3: F3 values for the datasets in Figure 2

Dataset	Original	Negative class	Positive class
	F3	**F3**	**F3**
Balanced Dataset	0.92	0.89	0.90
Imbalanced Dataset	0.83	0.81	0.94

- **F4: Collective feature efficiency**

F4 is similar to F3, but instead of using the minimum value from all attributes, it combines their discrimination power. The proportion of instances remaining, after using all features to discriminate them, is the outcome of F4. For this purpose, first, it finds the most discriminative attribute according to $\arg\min_{i=1}^{m} \frac{n_o(f_i)}{n}$; next, it removes the instances correctly discriminated by this attribute. It repeats the previous steps until all instances are correctly discriminated or until all attributes are removed. F4 is equal to the proportion of instances not discriminated at the end of the process. Equation 14 illustrates how F4 is calculated, where T_l is the dataset of the l-th iteration (with l in interval $[1, m]$) and $n_o(f_{min}(T_l))$ measures the number of instances in the overlapping region of attribute f_{min} from dataset T_l.

$$F4(T) = \frac{n_o(f_{min}(T_l))}{n} \tag{14}$$

Considering any i-th iteration of F4, the most discriminative attribute (f_{max}) of dataset T_i can be found using Equation 15, where $n_o(f_j)$ is computed according to Equation 11. The dataset of each iteration can be defined by Equations 16 and 17. Thus, the dataset at the i-th iteration is a subset of the previous dataset (T_{i-1}), considering only the instances inside the overlapping region of f_{min}.

$$f_{min}(T_i) = \{f_j | \overset{m}{\underset{j=1}{min}}(n_o(f_j))\}_{T_i} \tag{15}$$

$$T_1 = T \tag{16}$$

$$T_i = T_{i-1} - \{\mathbf{x}_j | x_{ji} < maxmin(f_{min}(T_{i-1})) \vee x_{ji} > minmax(f_{min}(T_{i-1}))\} \tag{17}$$

Our adaptation of F4 computes the number of misclassified instances in each class divided by the number of instances from that class, i.e., we adapted F4 to calculate the complexity of a class c_1. For such, Equations 14, 15 and 17 must be substituted, respectively, by Equations 18, 19 and 20, where $n_o^{c_1}$ is defined in Equation 13.

$$F4_{c_1}(T) = \frac{n_o^{c_1}(f_{min}^{c_1}(T_l))}{n_{c_1}} \tag{18}$$

$$f_{min}^{c_1}(T_i) = \{f_j | \overset{m}{\underset{j=1}{min}}(n_o^{c_1}(f_j))\}_{T_i} \tag{19}$$

$$T_i = T_{i-1} - \{\mathbf{x}_j | x_{ji} < maxmin(f_{min}^{c_1}(T_{i-1})) \vee x_{ji} > minmax(f_{min}^{c_1}(T_{i-1}))\} \tag{20}$$

The F4 values for the datasets in Figure 2 are shown in Table 4. The original F4 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 4: F4 values for the datasets in Figure 2

Dataset	Original	Negative class	Positive class
	F4	F4	F4
Balanced Dataset	0.87	0.89	0.90
Imbalanced Dataset	0.71	0.81	0.94

2.1.2. Neighborhood measures

The neighborhood measures use the concept of Nearest Neighbor (NN) to assess classification difficulties. They use the distance between instances to assess, for example, the shape of decision boundaries and class distributions. In this paper, we considered the measures N1, N2, N3, N4, and T1. A description of the original data complexity measures and an explanation of our adaptations are presented next.

- **N1: Fraction of points on the class boundary**

  N1 builds a minimum spanning tree (MST) that connects all instances in a dataset based on their pairwise distances, despite their classes. Next, it counts the number of instances connected to at least one instance from another class. These instances are possibly borderline and the ratio between their number and the total number of instances is the final N1 measure. N1 is bounded between 0 and 1, the closer to 0, the lower the complexity. Equation 21 expresses N1, where $(\mathbf{x}_i, \mathbf{x}_j)$ represents a connection between instances $\mathbf{x}_i$ and $\mathbf{x}_j$ and MST represents the set of all connections in the tree.

$$N1(T) = \frac{1}{n}\sum_{i=1}^{n} I((\mathbf{x}_i, \mathbf{x}_j) \in MST \wedge y_i \neq y_j) \tag{21}$$

N1 has a bias towards the majority class since the use of the normalization factor n leads to underestimation of the minority class complexity as the imbalance aggravates. Our adaptation considers each class separately. For such, considering one class at a time, we calculate the proportion of instances from that class that connects with an instance from a different class. With this adaptation, we can measure how complex a class is considering the concept of the N1. Equation 22 shows the adaptation, where $\mathbf{x}_i^{c_1}$ denotes an example of class c_1.

$$N1_{c_1}(T) = \frac{1}{n_{c_1}} \sum_{i=1}^{n_{c_1}} I((\mathbf{x}_i^{c_1}, \mathbf{x}_j) \in MST \wedge y_j \neq c_1) \tag{22}$$

The N1 values for the datasets in Figure 2 are shown in Table 5. The original N1 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 5: N1 values for the datasets in Figure 2

Dataset	**Original**	**Negative class**	**Positive class**
	N1	**N1**	**N1**
Balanced Dataset	0.25	0.26	0.24
Imbalanced Dataset	0.13	0.08	0.64

- **N2: Ratio of average intra/inter class NN distance**

N2 compares the intraclass and interclass dispersions of the classes. For each instance, its distance to the NN of the same class (intraclass) and its distance to the NN of a different class (interclass) are computed. N2 is the ratio of the intraclass distance average and the interclass distance average. Higher values represent problems of higher complexity. Equation 23 shows how N2 is calculated. In this equation, $d(\mathbf{x}_i, \mathbf{x}_j)$ is the distance function between $\mathbf{x}_i$ and $\mathbf{x}_j$, $NN(\mathbf{x}_i) \in \{T|y = y_i\}$ is the nearest neighbor of $\mathbf{x}_i$ from the same class and $NN(\mathbf{x}_i) \in \{T|y \neq y_i\}$ is the nearest neighbor of $\mathbf{x}_i$ from a different class.

$$N2(T) = \frac{\sum_{i=1}^{n} d(\mathbf{x}_i, NN(\mathbf{x}_i) \in \{T|y = y_i\})}{\sum_{i=1}^{n} d(\mathbf{x}_i, NN(\mathbf{x}_i) \in \{T|y \neq y_i\})} \tag{23}$$

By taking the averages of all instances, N2 values are biased towards the majority class. Our adaptation takes the averages for one class at a time. Therefore, the N2 value for a specific class, for example a class 1, will be the ratio of two averages: the average of intraclass distances for class 1 (i.e., the distance between each instance from class 1 and its NN from also class 1) and the average of the interclass distances for class 1 (i.e., the distance between each instance of class 1 with its NN from a different class). Equation 24 shows the modified N2, where $\mathbf{x}_i^{c_1}$ denotes an example of class c_1.

$$N2_{c_1}(T) = \frac{\sum_{i=1}^{n_{c_1}} d(\mathbf{x}_i^{c_1}, NN(\mathbf{x}_i^{c_1}) \in \{T|y = c_1\})}{\sum_{i=1}^{n_{c_1}} d(\mathbf{x}_i^{c_1}, NN(\mathbf{x}_i^{c_1}) \in \{T|y \neq c_1\})} \tag{24}$$

The N2 values for the datasets in Figure 2 are shown in Table 6. The original N2 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 6: N2 values for the datasets in Figure 2

Dataset	**Original**	**Negative class**	**Positive class**
	N2	**N2**	**N2**
Balanced Dataset	0.18	0.18	0.19
Imbalanced Dataset	0.13	0.11	0.44

- **N3: Leave-one-out error rate of the NN classifier**

N3 is the ratio between the number of examples whose NN are from a different class and the total number of examples from T. It is the same concept of the leave-one-out error of a NN classifier, which is easy to calculate and is a good indicator of the separability of classes. The following equation expresses how N3 is defined:

$$N3(T) = \frac{\sum_{i=1}^{n} I(NN(\mathbf{x}_i) \neq y_i)}{n} \tag{25}$$

When a dataset is highly imbalanced, N3 tends to be closer to the majority class error, which can be inadequate. To overcome this problem, we adapted N3 to take into account the error per class, i.e., the ratio between the number of examples from the class of interest whose NN are from a different class and the number of examples from that class. Equation 26 represents our adaptation in which c_1 represents the class of interest.

$$N3_{c_1}(T) = \frac{\sum_{i=1}^{n_{c_1}} I(NN(\mathbf{x}_i^{c_1}) \neq c_1)}{n_{c_1}} \tag{26}$$

The N3 values for the datasets in Figure 2 are shown in Table 7. The original N3 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 7: N3 values for the datasets in Figure 2

Dataset	**Original** **N3**	**Negative class** **N3**	**Positive class** **N3**
Balanced Dataset	0.17	0.18	0.17
Imbalanced Dataset	0.10	0.05	0.52

- **N4: Nonlinearity of a 1-NN classifier**

N4 uses a method that creates a new test set by interpolating randomly selected instances from the same class. Next, an NN classifier uses training set T to predict the labels of the instances in the interpolated test set. N4 returns the error rate obtained. A value closer to 1 may indicate either overlapped classes or that the classes do not form convex sets. Equation 27 shows how to calculate F4, where l is the number of interpolated instances, $\mathbf{x}'_i$ is an interpolated instance, $NN_T(\mathbf{x}'_i)$ is the NN from T to $\mathbf{x}'_i$ and y'_i is the class of $\mathbf{x}'_i$.

$$N4(T) = \frac{1}{l} \sum_{i=1}^{l} I(NN_T(\mathbf{x}'_i) \neq y'_i) \tag{27}$$

Using the same criterion as in N3, N4 was adapted to return the error rate per class. Thus, an NN classifier using the dataset T labels each interpolated instance $\mathbf{x}'_i$ from the class of interest c_1. The error rate is used as a measure. Considering a c_1 as the class of interest, Equation 28 represents our adaptation for F4. In this adaptation, l_{c_1} is the number of interpolated instances from class c_1 and ${\mathbf{x}_i^{c_1}}'$ is an interpolated example from class c_1.

$$N4_{c_1}(T) = \frac{1}{l_{c_1}} \sum_{i=1}^{l_{c_1}} I(NN_T({\mathbf{x}_i^{c_1}}') \neq c_1) \tag{28}$$

The N4 values for the datasets in Figure 2 are shown in Table 8. The original N4 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 8: N4 values for the datasets in Figure 2

Dataset	**Original** **N4**	**Negative class** **N4**	**Positive class** **N4**
Balanced Dataset	0.13	0.13	0.13
Imbalanced Dataset	0.06	0.03	0.42

- **T1: Fraction of maximum covering spheres**

  T1 looks for an interpretation of a training set using hyper-spheres. To explain how it works, suppose that every instance in the training set has a hypersphere with radius zero. If we gradually increase the radius of all hyperspheres, some of them will touch a hypersphere from a different class. When this occurs, both hyperspheres stop expanding. The method finishes when there is no more expanding hypersphere, discarding the hyperspheres contained in another hypersphere. T1 is the ratio between the number of remaining hyperspheres and the number of instances in the dataset. A number closer to 0 indicates that there is no need for many hyperspheres to describe the training set. A number closer to 1 indicates a higher complexity and that as many hyperspheres as number of instances are needed to describe the training set. Equation 29 represents T1, where $Hyperspheres(T)$ calculates the number of hyperspheres needed to cover the dataset.

$$T1(T) = \frac{Hyperspheres(T)}{n} \tag{29}$$

  Consider a binary training set entirely overlapped and highly imbalanced, T1 may be low for this training set, since a small number of hyperspheres is needed to describe the data compared to the number of instances. However, to describe the minority class we need almost the same number of minority class instances as hyperspheres. Therefore, our adaptation of T1 takes the ratio between the hyperspheres necessary to describe each class and the number of instances in the class. Equation 30 substitutes Equation 29 in our definition, when $Hyperspheres(T, c_1)$ calculates the number of hyperspheres needed to cover the examples of class c_1.

$$T1_{c_1}(T) = \frac{Hyperspheres(T, c_1)}{n_{c_1}} \tag{30}$$

  The T1 values for the datasets in Figure 2 are shown in Table 9. The original T1 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 9: T1 values for the datasets in Figure 2

Dataset	Original T1	Negative class T1	Positive class T1
Balanced Dataset	0.26	0.27	0.26
Imbalanced Dataset	0.13	0.08	0.64

2.1.3. Linear Separability Measures

These measures assess whether the classes can be linearly separable in the attribute space. They assume that a classification problem solved with a hyperplane is simpler than another with a non-linear boundary. The measures from this category considered in this article are L1, L2, and L3.

To build the linear classifier for the complexity measures, Ho and Basu (2002) [21] suggest solving the optimization problem proposed by Smith (1968) [37]. Recent studies propose the using a Support Vector Machine (SVM) with a linear kernel [34, 27]. SVM obtains the hyperplane by solving the following optimization problem:

$$\underset{w,b,\epsilon}{Minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\left(\sum_{i=1}^{n} \epsilon_i\right) \tag{31}$$

$$Subject\,to: \begin{cases} y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1 - \epsilon_i, \\ \epsilon_i \geq 0, i = 1, \dots, n \end{cases} \tag{32}$$

where C is the trade-off between the margin maximization, achieved by minimizing the norm of $\mathbf{w}$, and the minimization of the training errors, modeled by ϵ. The hyperplane is given by $\mathbf{w}\cdot\mathbf{x} + b = 0$, where $\mathbf{w}$ is a weight vector and b is an offset value. All the linearity measures described in this article will adopt this notation. Next, we describe the measures investigated in this study.

- **L1: Minimized sum of error distance of a linear classifier**

  L1 uses a linear model (e.g., a linear SVM) induced by a training set and the distances between misclassified instances and a hyperplane representing the model. L1 returns the average of these distances, which is equal to 0 for linearly separable problems.

  Considering the SVM hyperplane, L1 can be calculated using all ϵ_i, as shown in Equation 33. We normalize L1 to the interval $[0, 1]$, whereby the larger the value, the more complex the dataset, using $1 - \frac{1}{L1+1}$.

  $$L1(T) = \frac{1}{n}\sum_{i=1}^{n} \epsilon_i | h(\mathbf{x}_i) \neq y_i \tag{33}$$

  where $h(\mathbf{x}_i)$ represents the SVM prediction for the i-th training example.

  As L1 has a bias towards the majority class, we adapt it so that only the distances of misclassified instances from each specific class are summed up, as shown in Equation 34:

  $$L1_{c_1}(T) = \frac{1}{n_{c_1}}\sum_{i=1}^{n_{c_1}} \epsilon_i^{c_1} | \mathbf{h}(\mathbf{x}_i) \neq c_1 \tag{34}$$

The L1 values for the datasets in Figure 2 are shown in Table 10. The original L1 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 10: L1 values for the datasets in Figure 2

Dataset	**Original**	**Negative class**	**Positive class**
	L1	**L1**	**L1**
Balanced Dataset	0.08	0.08	0.09
Imbalanced Dataset	0.05	0.00	0.33

- **L2: Training error of a linear classifier**

  L2 is the training error of a linear classifier. For its calculation, we induce a linear classifier from the training set and use its classification error rate. The higher the values the less linear is the classification boundary. Equation 35 shows how L2 is calculated. In this equation, $h(\mathbf{x}_i)$ is the predicted class for the instance $\mathbf{x}_i$.

$$L2(T) = \frac{\sum_{i=1}^{n} I(h(\mathbf{x}_i) \neq y_i)}{n} \tag{35}$$

  Our adaptation returns the error rate per class, using Equation 36:

$$L2_{c_1}(T) = \frac{\sum_{i=1}^{n_{c_1}} I(h(\mathbf{x}_i^{c_1}) \neq c_1)}{n_{c_1}} \tag{36}$$

  The L2 values for the datasets in Figure 2 are shown in Table 11. The original L2 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 11: L2 values for the datasets in Figure 2

Dataset	**Original**	**Negative class**	**Positive class**
	L2	**L2**	**L2**
Balanced Dataset	0.11	0.11	0.12
Imbalanced Dataset	0.05	0.01	0.44

- **L3: Nonlinearity of the linear classifier**

  Similar to N4, L3 interpolates a test set and, instead of a KNN classifier, uses a linear classifier to classify instances from the test set. Equation 37 shows how L3 is calculated. In this equation,

$h_T(\mathbf{x}'_i)$ is the prediction of the linear model induced using training set T for the interpolated instance $\mathbf{x}'_i$.

$$L3(T) = \frac{1}{l}\sum_{i=1}^{l} I(h_T(\mathbf{x}'_i) \neq y'_i) \tag{37}$$

Our adaptation returns the error rate per class, using Equation 38:

$$L3_{c_1}(T) = \frac{1}{l_{c_1}}\sum_{i=1}^{l_{c_1}} I(h_T({\mathbf{x}_i^{c_1}}') \neq c_1) \tag{38}$$

The L3 values for the datasets in Figure 2 are shown in Table 12. The original L3 assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 12: L3 values for the datasets in Figure 2

Dataset	Original L3	Negative class L3	Positive class L3
Balanced Dataset	0.08	0.06	0.07
Imbalanced Dataset	0.04	0.00	0.46

2.1.4. *Other Complexity Measures for Imbalanced Datasets*

Recently, four other data complexity measures were proposed specifically for imbalanced datasets [2, 36, 28]. They are CM, wCM, dwCM, and BI[3]. All of them use a kNN classifier in their calculation. In the experimental analysis, we compare them with our adaptations on the original data complexity measures. Next, we describe these four measures.

- **CM: Complexity measure for imbalanced datasets**

  CM considers the k nearest neighbors of each minority class instance [2]. If the majority of the k nearest neighbors does not belong to the minority class, this instance is considered difficult. CM is the percentage of difficult minority class instances. Equation 39 shows how CM is calculated, considering c_1 as the minority class, k as a parameter defined by the user, and $NN_j(\mathbf{x}_i^{c_1})$ as the j-th nearest neighbor of instance $\mathbf{x}_i^{c_1}$.

$$CM(T,k) = \frac{1}{n_{c_1}}\sum_{i=1}^{n_{c_i}} I\left(\frac{\sum_{j=1}^{k} I(NN_j(\mathbf{x}_i^{c_1}) \neq c_1)}{k} > 0.5\right) \tag{39}$$

The CM values for the datasets in Figure 2 are shown in Table 13. We used the CM for the whole dataset and a k optimization defined by Anwar et al. [2]. The CM for the whole dataset assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 13: CM values for the datasets in Figure 2

Dataset	Dataset CM	Negative class CM	Positive class CM
Balanced Dataset	0.14	0.13	0.15
Imbalanced Dataset	0.07	0.03	0.49

- **wCM: Weighted complexity metric**

wCM extends CM using a distance weighted kNN classifier instead of a kNN [36]. On this measure, each neighbor j of each instance i from the minority class has a weight defined by their distance. The weights are normalized using the distances of the closest neighbor and the farthest neighbor. The calculation of W_{ij}, which is the weight of the j-th neighbor of the i-th minority instance is defined by Equation 40. wCM then uses the weights on its calculation, as defined by Equation 41.

$$W_{ij} = \begin{cases} \dfrac{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) - d(\mathbf{x}_i, NN_j(\mathbf{x}_i))}{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) - d(\mathbf{x}_i, NN_1(\mathbf{x}_i))}, \text{if } d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) \neq d(\mathbf{x}_i, NN_1(\mathbf{x}_i)) \\ 1, \text{if } d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) = d(\mathbf{x}_i, NN_1(\mathbf{x}_i)) \end{cases} \tag{40}$$

$$wCM(T,k) = \frac{1}{n_{c_1}} \sum_{i=1}^{n_{c_i}} I\left(\frac{\sum_{j=1}^{k} W_{ij} I(NN_j(\mathbf{x}_i^{c_1}) \neq c_1)}{\sum_{j=1}^{k} W_{ij}} > 0.5 \right) \tag{41}$$

The wCM values for the datasets in Figure 2 are shown in Table 14. We used the wCM for the whole dataset and $k = 11$ as suggested in Singh et al. [36]. The CM for the whole dataset assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 14: wCM values for the datasets in Figure 2

Dataset	Dataset wCM	Negative class wCM	Positive class wCM
Balanced Dataset	0.13	0.12	0.14
Imbalanced Dataset	0.06	0.02	0.5

- **dwCM: Dual weighted complexity metric**

  According to the authors, wCM may not be robust enough depending on the value of k, and therefore they also propose a dual weighted complexity metric, the dwCM [36]. The difference between wCM and dwCM are the weights. In dwCM, the weights are calculated according to the Equation 42.

$$W_{ij} = \begin{cases} \dfrac{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) - d(\mathbf{x}_i, NN_j(\mathbf{x}_i))}{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) - d(\mathbf{x}_i, NN_1(\mathbf{x}_i))} \times \dfrac{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) + d(\mathbf{x}_i, NN_1(\mathbf{x}_i))}{d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) + d(\mathbf{x}_i, NN_j(\mathbf{x}_i))}, \\ \qquad \text{if } d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) \neq d(\mathbf{x}_i, NN_1(\mathbf{x}_i)) \\ 1, \text{if } d(\mathbf{x}_i, NN_k(\mathbf{x}_i)) = d(\mathbf{x}_i, NN_1(\mathbf{x}_i)) \end{cases} \tag{42}$$

  The dwCM values for the datasets in Figure 2 are shown in Table 15. We used the dwCM for the whole dataset and $k = 11$ as suggested in Singh et al. [36]. The CM for the whole dataset assessed that the imbalanced dataset is easier than the balanced one. The decomposed measures assessed that the imbalanced dataset is easier for the majority class and more difficult for the minority class.

Table 15: wCM values for the datasets in Figure 2

Dataset	Dataset	Negative class	Positive class
	dwCM	dwCM	dwCM
Balanced Dataset	0.14	0.12	0.15
Imbalanced Dataset	0.06	0.02	0.48

- **BI^3: Bayes imbalance impact index**

  Inspired by the Bayes optimal classifier, Lu et al. [28] proposes a measure called Bayes Imbalance Impact Index (BI^3). It is calculated according to Equation 43, where $f_n(\mathbf{x}_i, k) = \frac{\sum_{j=1}^{k} I(NN_j(\mathbf{x}_i) \neq c_1)}{k}$, $f_p(\mathbf{x}_i, k) = \frac{\sum_{j=1}^{k} I(NN_j(\mathbf{x}_i) = c_1)}{k}$, and $f'_p(\mathbf{x}_i, k) = \frac{n_{c_0}}{n_{c_1}} \times f_p(\mathbf{x}_i, k)$.

$$BI^3(T, k) = \frac{1}{n_{c_1}} \sum_{i=1}^{n_{c_i}} \frac{f'_p(\mathbf{x}_i, k)}{f_n(\mathbf{x}_i, k) + f'_p(\mathbf{x}_i, k)} - \frac{f_p(\mathbf{x}_i, k)}{f_n(\mathbf{x}_i, k) + f_p(\mathbf{x}_i, k)} \tag{43}$$

  The BI^3 values for the datasets in Figure 2 are shown in Table 16. We used $k = 5$ as suggested by Lu et al. [28]. BI^3 assessed that the imbalanced dataset is more difficult than the balanced one.

All four measures described above are parameter dependent. The user must set the parameter k, and its choice may change the outcome of the measure. For example, Singh et al. [36] reported that CM and wCM may be sensitive to the parameter choice. CM proposes a strategy to choose k. The other three

Table 16: BI^3 values for the datasets in Figure 2

Dataset	BI^3
Balanced Dataset	0.00
Imbalanced Dataset	0.29

fix a value for the parameter. BI^3 presents a strategy of flexible k to avoid 0 values on its calculation. They also do not compare their results with N3 - which has a similar concept in terms of assessing the data complexity. In this paper, we evaluate the related work aforementioned not only with N3, but all our proposed adaptations on the data complexity measures.

In this work, we consider only these data complexity measures because they are the most used, studied and have different biases. We also consider measures proposed specifically for imbalanced datasets. Nevertheless, there are other complexity measures that were not described [24, 38]. For example, measures extracted from a structural representation of the dataset using graphs, which take into account the relationship between instances [24]. In Smith et al. [38], a subset of measures that extract instance hardness is proposed, i.e. considering an instance as hard if it is misclassified by a diverse set of simple classification algorithms.

2.2. Pre-processing techniques for imbalanced classification tasks

There are two main approaches to deal with imbalanced data classification tasks: (1) pre-processing the data to make it more balanced [7, 18, 19, 23, 25]; (2) developing classification algorithms which are more robust to imbalanced data [17, 8, 6, 9]. Pre-processing techniques are usually independent from classification algorithms. However, they may modify the original data distribution, removing important instances or adding noise [20]. Adapted classification algorithms reduce Data Mining pipelines, but do not improve data quality. In this paper, we focus on the former, which is more often adopted.

Pre-processing techniques are used based on data undersampling and/or oversampling [13]. To balance the data, undersampling techniques remove instances from the majority class and oversampling techniques insert instances in the minority class [13]. Both undersampling and oversampling can occur randomly or based on some criteria. Next, we discuss some of the main pre-processing strategies for balancing datasets in ML.

2.2.1. Random sampling

Random undersampling (RU) removes instances from the majority class at random until obtaining a data distribution considered balanced [20]. Random oversampling (RO) replicates instances from the minority class at random until obtaining a data distribution considered balanced [20]. In the literature, they are usually used until classes are equally represented in the number of instances.

2.2.2. *Synthetic minority oversampling techniques*

SMOTE (*Synthetic Minority Oversampling Technique*) [7] is an oversampling technique that creates artificial data by interpolation, as follows. At each iteration, SMOTE selects at random an instance $\mathbf{x}$ from the minority class. Next, it uses KNN to find the k closest instances to $\mathbf{x}$ in the minority class. It selects one of the neighbors $\mathbf{z}$ at random and creates a new instance that is a combination of $\mathbf{x}$ and $\mathbf{z}$. The combination is an interpolation that randomly creates any possible point between $\mathbf{x}$ and $\mathbf{z}$. This step is repeated until a distribution of instances considered balanced is obtained.

BorderlineSMOTE is a version of SMOTE that searches for minority class instances close to decision boundaries to interpolate [18]. Instead of selecting minority class instances from all training sets, it selects minority class instances close to the decision boundary. The procedure that BorderlineSMOTE uses to select them is: (1) find the k NN for a minority class instance $\mathbf{x}$; (2) count the number N_{maj} of neighbors that belongs to the majority class; (3) if $\frac{k}{2} \leq N_{maj} < k$ then $\mathbf{x}$ is put in a set called DANGER; (4) repeat the steps for all minority class instances. Afterwards SMOTE is run to balance the dataset but it selects only instances from the DANGER subset.

ADASYN, also based on SMOTE, addresses the number of instances to be interpolated by each minority class instance [19]. For such, it follows three steps: (1) it defines G, which indicates how many instances should be interpolated for the entire minority class; (2) for each instance in the minority class, it calculates the percentage of majority class instances in the k nearest neighbors; (3) it normalizes the set of all percentages (Γ_i, where i is the minority class instance), so that $\sum \Gamma_i = 1$; finally, $\Gamma_i \times G$ gives the number of instances to be interpolated using SMOTE for each minority class instance i.

2.2.3. *Cluster based oversampling*

CBO (*Cluster-Based Oversampling*) [23] is an oversampling technique that takes into account both inter and intraclass imbalance. Differently from the inter and intraclass distance defined in Section 2.1, inter and intraclass imbalance considers the disproportion between classes and inside a class, respectively. Interclass imbalance is the concept commonly used to describe a disproportion between classes in number of instances. The intraclass imbalance describes the disproportion inside a class, i.e when the subconcepts of the same class have a disproportion between them.

For such, CBO first applies, separately, a clustering algorithm to the instances from the majority class and to the instances from the minority class, generating two sets of clusters - one for each class. Next, *CBO* oversamples all clusters belonging to the majority class, except the largest cluster. In the end, each cluster of the majority class should have the same number of instances as the largest cluster. Finally, oversampling is applied to all clusters belonging to the minority class, making (1) the number of instances in the minority class equal to the number of instances in the majority class after oversampling, and (2) each cluster in the minority class equally balanced.

2.2.4. One-sided selection

OSS (*One-sided Selection*) [25] is an undersampling technique that keeps only the most representative instances of the majority class. For such, *OSS* initially chooses one instance $\mathbf{x}$ of the majority class at random. Next, using the instances of the minority class and $\mathbf{x}$ as training data, *OSS* applies the k-Nearest Neighbors (*KNN*) algorithm with $k = 1$ to classify the remaining instances of the majority class. The correctly classified instances are excluded from the majority class, as they are considered redundant. Thus, after the undersampling, the majority class will have only the instances that were incorrectly classified by $k-$NN and $\mathbf{x}$. Finally, *OSS* uses a data cleaning technique to remove borderline and noisy instances, originally, *Tomek Links* [39].

All techniques modify the values of the data complexity measures described in Section 2.1. For example, SMOTE modifies the neighborhood measures by generating new instances near existing ones. More specifically, the N3 measure may be reduced by generating samples near overlapping decision borders; and OSS may modify overlapping measures, such as F2, when it reduces the range of the values considered by the measures.

In the same way that, according to their bias, pre-processing techniques modify the complexity measures, these techniques can artificially modify the complexity measure values. For example, since N3 is based on NN, the duplication of instances in the training set by RO decreases the N3 value. In an extreme case, when all minority class instances are duplicated, the N3 value for this class would become 0, but the predictive performance of a classifier using the new dataset would not improve.

New pre-processing techniques for imbalanced classification have been recently proposed, including other SMOTE adaptations [4, 1], undersampling based on clustering [33], and sampling based on evolutionary algorithms [41]. According to Barua et al. [4], SMOTE adaptations favored noisy instances. To overcome this problem, the authors proposed a new approach to select minority class instances that discard those with no minority neighbor. This SMOTE adaptation, called MWMOTE (Majority Weighted Minority Oversampling Technique), weights the minority class instances and generates new instances within a minority cluster. MDO (Mahalanobis Distance-based Over-sampling technique) [1], another SMOTE adaptation, generates synthetic minority class instances that have the same Mahalanobis distance to the class mean as other existing minority class instances. In this article, we use the standard pre-processing techniques described in Section 2.2 because they are the most used and studied.

Data complexity measures have also been used to tackle the imbalance problem. Luengo et al. [29] used complexity measures to predict whether a DIT technique would be useful. They found intervals of values for some complexity measures in which the techniques were useful. Complexity measures have also been used to analyze the suitability of using a specific DIT technique. Díez-Pastor et al. [10] used complexity measures to predict data complexity intervals in which some diversity-enhancing techniques may improve the results of an ensemble of classifiers. Fernández et al. [12] used one complexity measure combined with other characteristics (such as imbalance) in a multi-objective approach to select attributes and instances from an imbalanced dataset. Fernandes and de Carvalho [11] adapted the N1 measure to the context of imbalanced multi-class classification and used it in a multi-objective approach as undersampling.

To the best of our knowledge, no work in the literature has analyzed the data complexity measures regarding the imbalance problem on real datasets by decomposing the original measures per class. All the works previously mentioned use the original data complexity measures. Next, we show experimentally that the traditional complexity measures do not capture complexity in imbalanced datasets properly. Therefore, the contributions of the aforementioned studies can be improved by using our adaptations.

3. Experimental Settings

The contributions of this study are guided by the following research questions: Are the original data complexity measures suitable for imbalanced datasets? Does a decomposition by class improve their performance on imbalanced datasets? Is there a correlation between the difference in data complexity and the difference in predictive performance after applying DITs?

To answer these questions, we performed an extensive empirical analysis, using 203 datasets, which were randomly divided into two groups. We use the first group of datasets to evaluate the performance of the data complexity measures on assessing imbalanced datasets. From these results, we select the most relevant complexity measures for the studied cases. Next, we use the selected measures to analyze the complexity after applying DITs. We collected the datasets from OpenML [40] and made them available, together with the experiment results[1]. We also implemented a package for the adapted data complexity measures, called ImbCoL [2].

3.1. Data Complexity Measures Experiments

In these experiments, we used a group of 102 datasets to investigate whether the original data complexity measures can assess how difficult an imbalanced classification dataset is. The predictive performance of a classification model was used to estimate the difficulty of a dataset using a grid search approach. Table 17 shows a summary of the 102 datasets used in this experiment. Minimum, maximum and mean values for the number of instances, number of features and percentage of the minority class are shown. For more details, please see Table 20 in the Appendices or the GitHub link[3]. 33 out of the

[1] https://github.com/victorhb/IS2020_results

[2] https://github.com/victorhb/ImbCoL

[3] https://github.com/victorhb/IS2020_results

102 datasets have less than 25% of minority class instances. We call them the high imbalanced datasets. The remaining 69 ones are called the low imbalanced datasets.

Table 17: Summary of the 102 datasets used on the experiment to evaluate the data complexity measures.

Dataset Characteristic	Min Value	Max Value	Mean Value
Number of Instances	36	2,534	486
Number of Features	3	95	16
% Minority Class	2.15	49.70	32.27

To reduce the influence of the bias of the ML algorithm, we used a pool of six algorithms, which were tuned using grid search. The hyperparameters and their possible values are listed in Table 18, in which m is the number of attributes and $a = \frac{(m+2)}{2}$. We considered all data complexity measures described in Section 2.1.

Table 18: Classification algorithms used and their possible hyperparameter values

Classification Algorithms	Hyperparameters	Values
Support Vector Machines (SVM)	kernel	linear, radial, polynomial, sigmoidal
	cost	2^{-10}, 2^{-9}, ..., 2^{10}
	gamma	2^{-10}, 2^{-9}, ..., 2^{10}
	degree	2, 3, 4, 5
Random Forest (RF)	number of trees	100, 200, ..., 1000
	number of variables	$\frac{\sqrt{m}}{2}$, $\sqrt{m}$, $\sqrt{m} \times 2$
K-Nearest Neighbours (KNN)	k	1, 3, 5, ..., 31
Naive Bayes (NB)	None	None
C4.5	threshold for pruning	0.1, 0.2, ..., 0.5
	min instances per leaf	2, 3, ..., 10
Multi-Layer Perceptron Neural Networks (MLP)	learning rate	0.1, 0.2, ..., 1
	number of neurons in hidden layer	$a-3$, $a-2$, ..., $a+3$

Using 30 repetitions of stratified 5-fold cross-validation, we extracted, for each dataset, 150 sets of data complexity measures from the training subsets and 150 sets of predictive performances from the validation subsets. Next, we assigned, to each dataset, the mean of the 150 values of each complexity measure and the predictive performance. The best model was chosen according to the highest predictive performance on average for each dataset. Figure 3 illustrates the steps followed in these experiments. Afterwards, we correlated, for each dataset, the mean data complexity measures with the mean predictive performance.

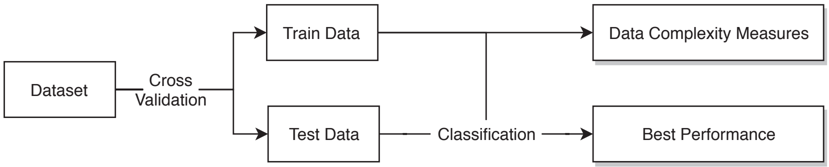

Figure 3: Diagram illustrating the steps of the first experiment

We measured the predictive performance using gmean, which is widely used in the imbalanced data literature. Gmean is the geometric mean between the true positive rate (TPR) and the true negative rate (TNR), defined by $gmean = \sqrt{\text{TPR} \times \text{TNR}}$. We used the Pearson correlation to correlate the data complexity measures and the Gmean. For the complexity measures dependent of k, we set the parameter according to their original publications: for CM, we estimated a different k for each dataset, for wCM and dwCM we set $k = 9, 11$, and for BI[3], we set $k = 5$.

3.2. Experiments with Data Imbalance Treatment Techniques

The second group of experiments, with 101 datasets, is carried out to assess the effectiveness of complexity measures when DIT techniques are applied to the training dataset. Table 19 shows a summary of the 102 datasets used in this experiment. Minimum, maximum and mean values for the number of instances, number of features and percentage of the minority class are shown. For more details, please see Table 21 in the Appendices or the GitHub link[4]. 29 out of the 101 datasets have less than 25% of minority class instances. We call them the high imbalanced datasets. The remaining 72 ones are called the low imbalanced datasets. The two sets of datasets used in both experiments, the one described in this section and the one described on Section 3.2, share similar characteristics regarding the number of instances, number of features and imbalance. For further details about the similarities between the two sets, please see Figure 12 in the Appendices.

Table 19: Summary of the 101 datasets used in the experiment using data imbalance treatment techniques.

Dataset Characteristic	Min Value	Max Value	Mean Value
Number of Instances	34	2,372	342
Number of Features	3	71	17
% Minority Class	2.33	49.80	32.57

Previous studies have shown that the application of DIT techniques to imbalanced datasets can improve the predictive performance obtained by ML algorithms [7, 18, 19, 23, 25]. However, there are

[4] https://github.com/victorhb/IS2020_results

situations in which their use either reduces or does not affect the predictive performance, and increases the overall computational cost. Additionally, as when using ML algorithms, each DIT technique has a bias, thus some techniques are better than others for particular data conformations [7, 18, 19, 23, 25]. In these experiments, we investigate how the DIT techniques change the data complexity and whether the changes correlate with the predictive performance of ML algorithms.

Thus, for each dataset, we extracted the data complexity measures and predictive performance before and after applying the DIT techniques. In these experiments, we used the same ML algorithms previously mentioned, with default hyperparameter values. We used a different experimental design from the previous experiment because, in the second experiment, we apply DITs to the datasets. In the literature, when DITs are applied, no hyperparameter tuning is performed in the classification algorithms [35, 1, 7, 18, 23]. This decision is motivated by the fact that tuning would interfere with the DIT analysis, once it would not be possible to track if the observed behavior is due to tuning or the DIT application. The classification algorithms used, and their default hyperparameter values were: SVM with radial kernel, $cost = 1$, and $gamma = \frac{1}{m}$; Random Forest with 500 trees and $\sqrt{m}$ variables; k-NN with $k = 3$; Naive Bayes; C4.5 with 0.2 of threshold for pruning and 2 instances per leaf at minimum; MLP with 0.3 of learning rate and a neurons in the hidden layer; in which m is the number of attributes and $a = \frac{(m+2)}{2}$.

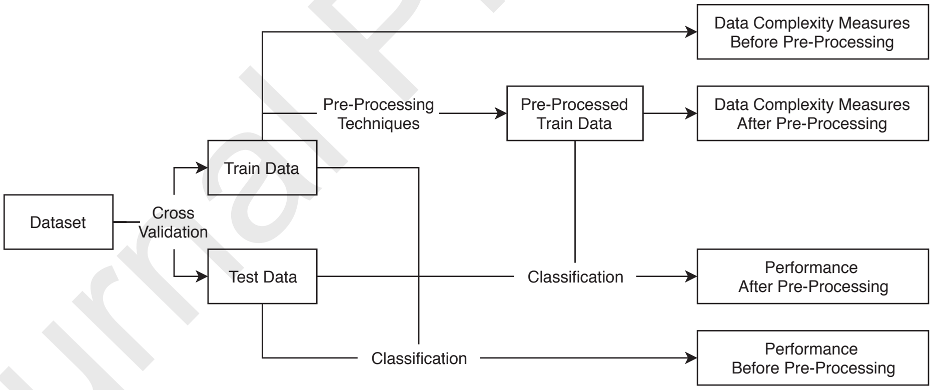

Figure 4: Diagram illustrating the steps of the second experiment

To assess the effect of applying DIT techniques, we measured the gmean of ML algorithms before and after the application. For each dataset, we used 5-fold cross-validation 30 times to compute the mean values for the data complexity measures and the predictive performance. Figure 4 illustrates the followed steps.

The final ratio hyperparameter of the DIT techniques was set to make the two classes completely balanced, except for OSS, which does not have this hyperparameter. For SMOTE, BorderlineSMOTE and ADASYN, the interpolation used the 3 nearest neighbors of each instance. For CBO, we used the k-means clustering algorithm with 4 groups and 100 iterations. We combined two oversampling strategies with CBO: random oversampling (CBO+RO) and SMOTE (CBO+SMOTE).

4. Experimental Results and Discussion

Next, we present and discuss the main results obtained in the evaluation of the original and modified complexity measures for the artificial and real datasets and the effect of DIT techniques on these measures.

4.1. Data Complexity Measures and Real Datasets

In Barella et al. [3], the authors evaluated how data complexity measures performed on artificial imbalanced datasets. There, the authors generated artificial datasets in which the instances were sampled from multivariated normal distributions, and they varied the number of features, class density and imbalance ratio of the datasets. Their experimental results showed that, for the datasets used, they were not suitable for imbalanced data. In the same work, the authors proposed adaptations to these measures, which improved their adequacy to assess the difficulty of the artificial imbalanced datasets considered. In this paper, we expand this analysis by deepening the previous analysis, but this time on real datasets and performing additional evaluations.

Figure 5 compares the Pearson correlations of the gmean performance with the data complexity measures, both the original measures and their adaptations assessing the majority and the minority class. The figure shows the results for the artificial datasets described in Barella et al. [3] and the real datasets described in Section 3.

Regarding the complexity measures, a value close to 1 should be read as describing a very difficult dataset, while, regarding the gmean performance, a value close to 1 means that a classifier achieved the perfect performance. Difficult datasets are expected to have high values of complexity measures and low values in gmean performance. Thus, in order for the complexity measures to adequately assess the difficulty of a dataset, negative correlations between the measures and the gmean performance are expected. Indeed, all observed correlations were negative, with the exception of the linearity measures for the adapted ones assessing the majority class.

The results show a low correlation between the original data complexity measures and gmean in both artificial and real datasets. The average of the absolute values of the correlations for these measures are 0.42 and 0.49 for artificial and real datasets, respectively. The F1 measure has the strongest correlation among all original complexity measures. However, it has a low value, smaller than 0.75 in the absolute value. It is important to point out that in previous work [3] F1 had an even lower correlation with the gmean, probably because it is not standardized as described in Section 2.1.

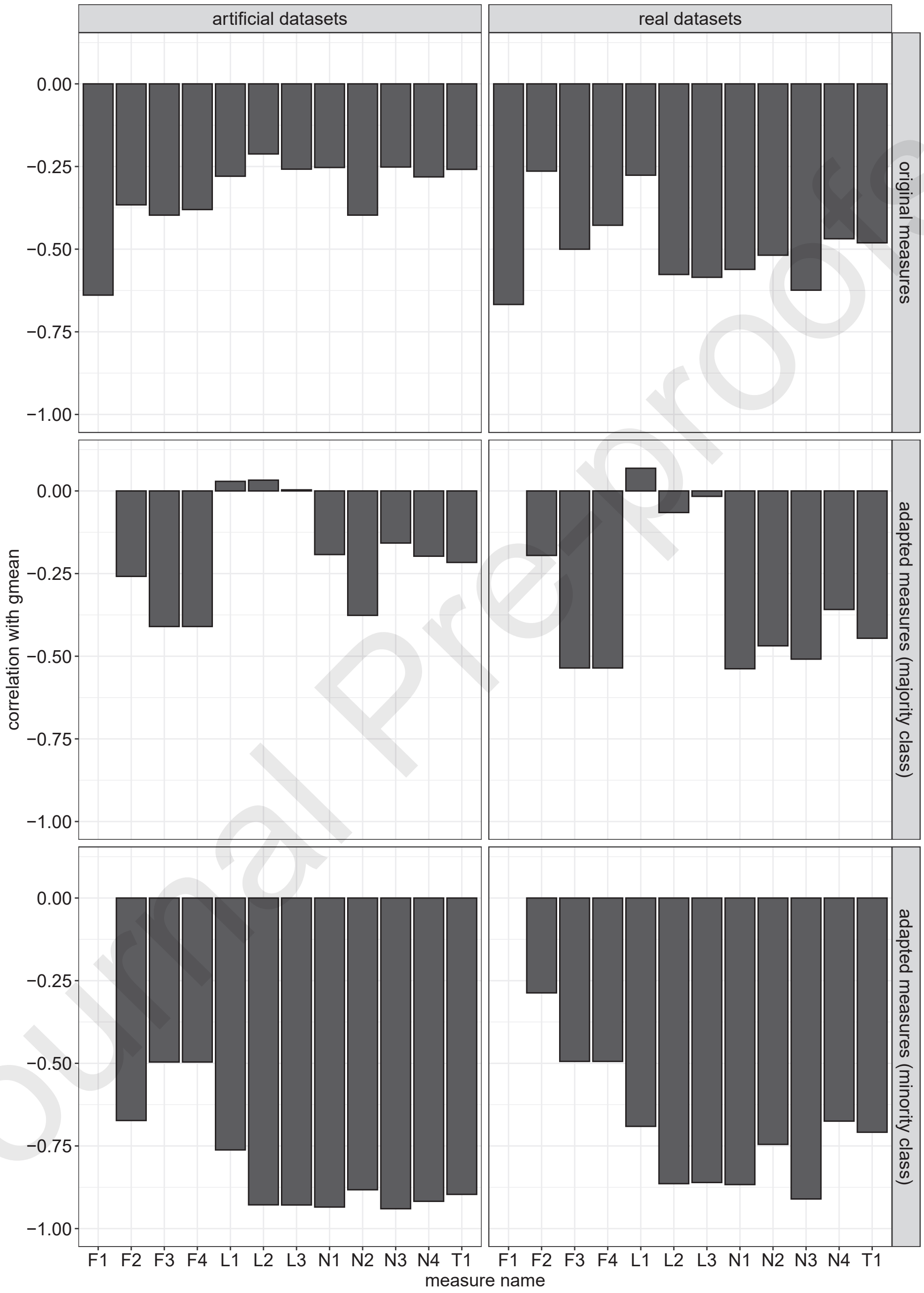

Figure 5: Correlation between the data complexity (original and adapted) and gmean measures, for the artificial and real datasets.

Regarding the experimental results using data complexity measures adapted for the minority class, the correlations are improved. On average, the absolute correlation values are increased to 0.4 for the synthetic datasets and to 0.2 for the real datasets. The main difference when compared with the results with the original complexity measures is that the correlation is now higher. As an example, for the real datasets, the correlation of the original N3 is -0.63 and the correlation of N3 adapted for the minority class is -0.91.

Overall, the results for the artificial datasets are similar to those obtained using real datasets. Therefore, the benefits of the data complexity measures also apply to the real datasets.

Regarding the results for the adapted measures in the majority class, they showed lower correlations, since the gmean performance is more affected by the performance in the minority class. The correlation between gmean and TPR was 0.94 and between gmean and TNR was 0.62. We expected gmean to be more correlated with TPR than TNR because the minority class is usually more difficult to learn. For this reason, from now on, we only consider the complexity of the minority class for the adapted measures in this paper. Gmean will continue to be calculated the same way, considering both classes.

In the imbalanced data literature, the imbalance ratio is mainly used to show the difficulty a dataset may impose on a classification task. In our experiment, the correlation between the predictive performance and the imbalance ratio was just 0.26 while the correlation between the N3 for the minority class and the predictive performance is 0.91, both in absolute values. These results show that our adaptations can provide relevant information for future studies in imbalanced data classification.

Next, we detail our analysis of the behavior of the most correlated measure, N3. Figure 6 illustrates the behavior of N3. In this figure, each triangle/circle is one dataset, with the shape and color representing different imbalance levels, high and low. The x-axis is the value of N3 measure and the y-axis is the gmean performance. We discretized imbalance into two categories: low imbalance (more than 25% of examples from the minority class in the dataset) and high imbalance (less or equal to 25% of examples from the minority class in the dataset).

It can be observed in Figure 6 that there is a high correlation between the original N3 measure and gmean when the datasets have low imbalance levels. When the datasets have a high imbalance level, the original N3 loses its ability to correlate with gmean, corroborating the fact that they do not correctly capture the difficulty in imbalanced scenarios. The Pearson correlations for the slightly imbalanced and the highly imbalanced scenarios are -0.92 and -0.41, respectively.

Regarding the adapted N3 for the minority class, we can see that the correlations are strong for both highly imbalanced and slightly imbalanced datasets. There is only one dataset in the figure with a high divergence between the difficulty assessed by N3 for the minority class and the predictive performance obtained. Apart from this dataset, all others compose a strong linear correlation. The Pearson correlations for the low imbalanced and the high imbalanced scenarios are -0.89 and -0.93, respectively.

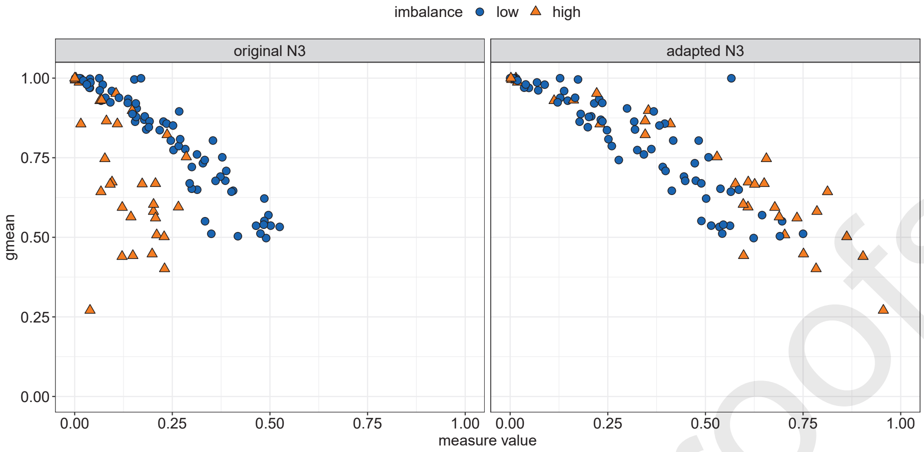

Figure 6: Relation between N3 and gmean.

These results also show that the adapted N3 leads to a smaller difference between these two correlations than the original N3.

To check if the relation found in N3 can be generalized to the other complex measures, we plot their values in Figure 7. In this figure, we also separate the correlations into low and high imbalance. It can be seen that what was seen for N3 also holds for the other neighborhood and the linearity measures, N1, N2, N4, T1, L1, L2, and L3. Thus, again, while for the original measures there is a strong correlation for low imbalance and a weak correlation for high imbalance, for the adapted complexity measures, the correlations are strong for both slightly and highly imbalanced datasets.

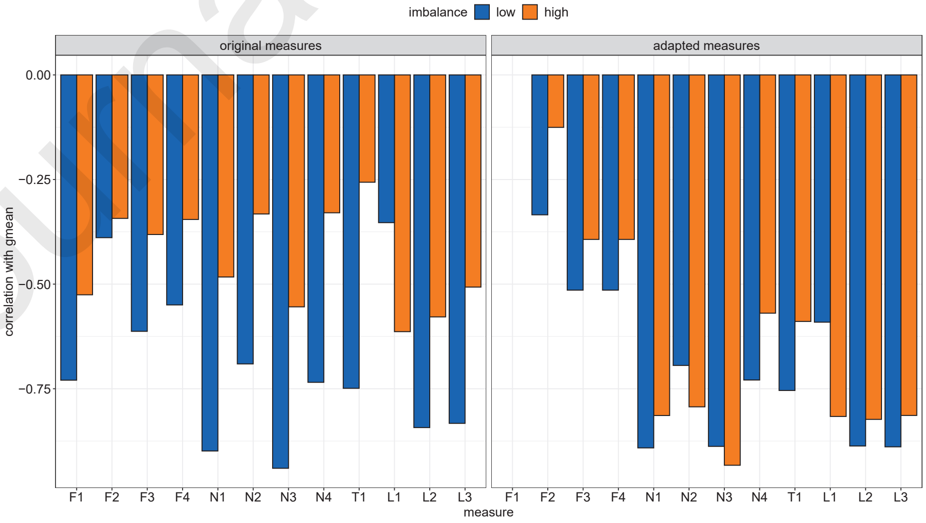

Figure 7: Correlation between the data complexity measures and gmean, separated by imbalance degree.

These results indicate that the original data complexity measures work correctly only for datasets with low imbalance levels. When the datasets have high imbalance levels, they lose their ability to assess their difficulty. They also show that most of our adapted complexity measures work well, regardless of the dataset imbalance level.

4.1.1. Comparison with related work

Figure 8 shows the correlation between the data complexity measures in the minority class and F1 over the 102 datasets. We also included the measures described in the related work considering only the minority class. The most correlated measures with gmean are N1, N3, L2, L3, CM, wCM9, wCM11, dwCM9, dwCM11. The correlation between the percentage of instances belonging to the minority class (% min class) and the number of instances in the minority class (# min class) are the least correlated with the gmean performance. BI^3 is highly correlated with % min class, but presented a low correlation with the gmean. L2 and L3 measures are highly correlated with each other, with a correlation of 0.99, indicating they are capturing similar characteristics from the datasets. Both measures assess the linear separability of the class, but L3 also considers the convexity of the class border. Moreover, N1, N3, CM, wCM9, wCM11, dwCM9, dwCM11 are correlated with each other with correlations above 0.9. All of these measures use the concept of nearest neighbors to be calculated. These correlations are stronger between CM, wCM9, wCM11, dwCM9, dwCM11, varying from 0.97 to 1, indicating they are capturing very similar characteristics. All CMs measures use a kNN classifier to calculate the data complexity.

We selected the measures with a correlation above 0.8 with gmean to understand how the main DIT techniques modify the data complexity. They are N1, N3, L2, L3, CM, wCM9, wCM11, dwCM9, dwCM11. In the next section, we show the results only for N1, N3, L2, and wCM11 to avoid redundancy. The complete table of results for the next section, with all 9 measures, can be accessed from the GitHub link[5].

4.2. DIT Techniques Discussion

In this section, we show that the DIT techniques modify the data complexity of the datasets and that there is a correlation between the difference in data complexity and the improvement of predictive performance for most DIT techniques.

4.2.1. The reduction of data complexity caused by DIT techniques

To see how DIT techniques affect the complexity of imbalanced datasets, we investigated their effect on the 4 previously selected data complexity measures before and after their application to 101 datasets not used in the experiments reported in the previous section.

[5] `https://github.com/victorhb/IS2020_results`

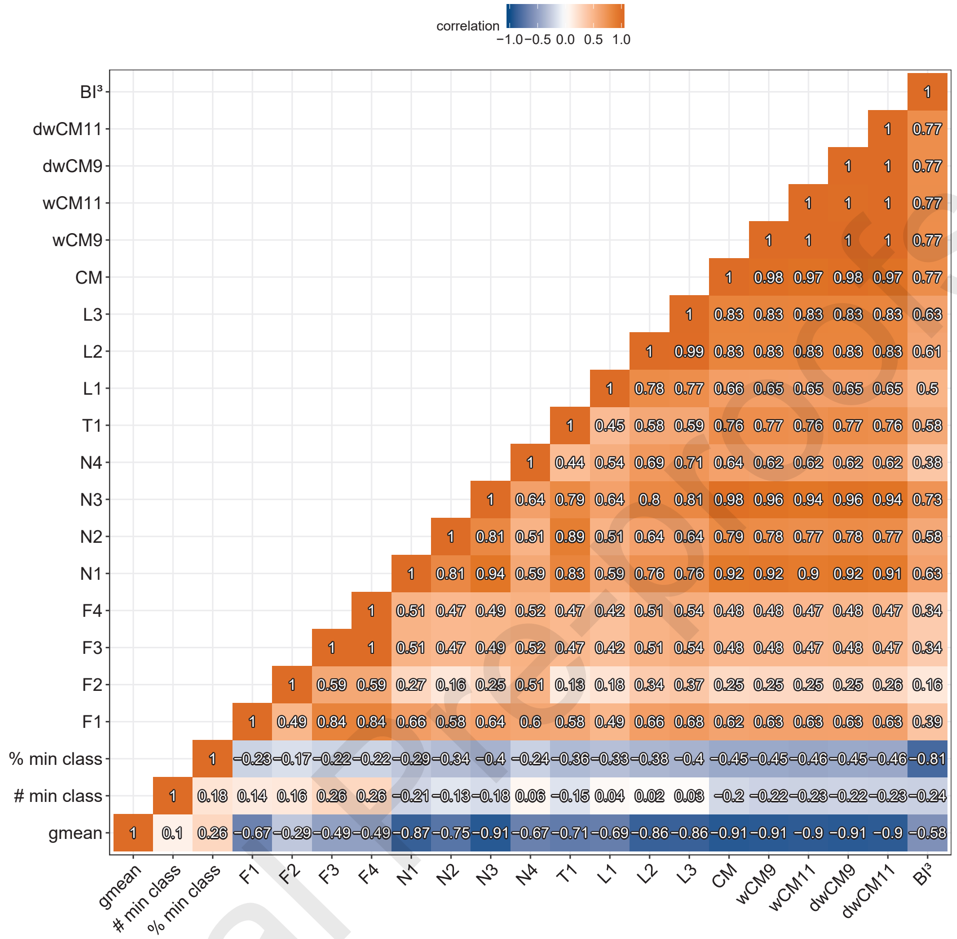

Figure 8: Correlations between gmean performance, data characteristics, original F1, and complexity measures for the minority class.

Figure 9 shows four boxes, one for each data complexity measure considered. The boxplots represent the difference in data complexity between after and before the application of each DIT technique, represented on the x-axis. The orange boxplots consider the datasets whose minority class have less than 25% of representation (high imbalance). The blue boxplots show these results for the datasets with more than 25% of minority class representation (low imbalance).

On the low imbalanced datasets, the application of DIT techniques had, in general, a small impact on their complexity, for the four data complexity measures. For N1 and N3 measures, the CBO-based techniques presented the highest reduction, with a statistical difference between them and the other techniques, but no statistical difference between the two CBO techniques considering a Friedman-Nemenyi

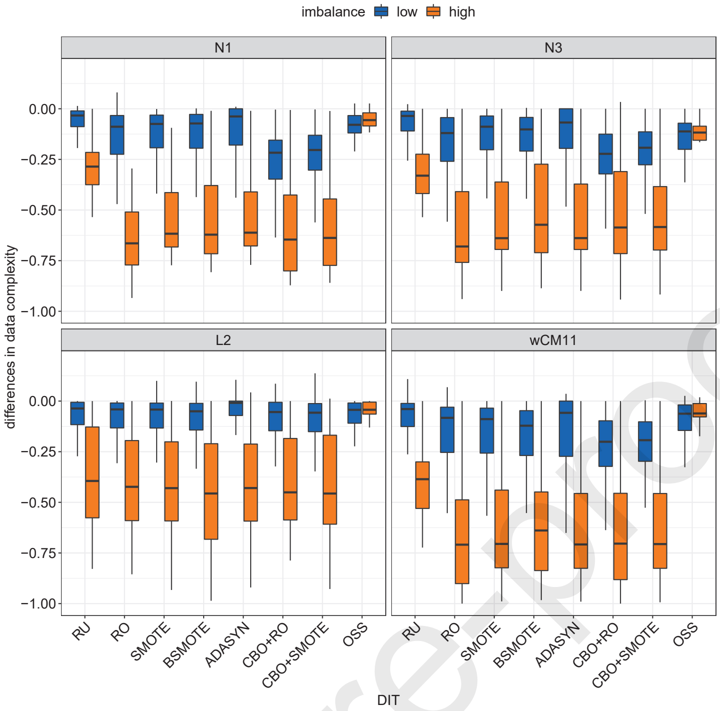

Figure 9: Data complexity measures differences before and after using the DIT techniques.

test with a confidence level of 95%. For the N1 measure, also the CBO-based techniques showed the largest decrease, which is statistically different from all other techniques, besides BSMOTE. For the L2 measure, most of the DIT techniques were similar in median, with the exception of ADASYN, that was statistically different compared to CBO+RO, CBO+SMOTE, SMOTE, and BSMOTE.

Overall, the DIT techniques did not obtain a large complexity reduction when applied to the low imbalance datasets. One reason may be that the techniques completely balance the training set and, because those datasets have a low imbalance ratio, they modified them modestly.

The highly imbalanced datasets, on the other hand, were strongly modified by most of the DIT techniques, for the four data complexity measures. RO, e.g., obtained an average difference of 0.45 for N3 and it was statistically different to RU, SMOTE, BSMOTE, ADASYN, and OSS according to a Friedman-Nemenyi test with a confidence level of 95%. Considering N1, RO also obtained the highest difference in median, and was statistically different from all other techniques, except for BSMOTE and the CBO-based techniques. For wCM11, RO was statistically different from RU, OSS, SMOTE, and BSMOTE. Considering L2, BSMOTE was the best in median, but statistically different only to RU, and OSS.

In general, N1, N3 and wCM11 behaved similarly among all the DIT techniques. They had lower differences for RU and OSS, in both high and low imbalance; CBO-based techniques had a larger difference in median for the low imbalanced datasets; and RO had a slightly larger difference for the high imbalanced datasets. All three measures use the concept of NN in their calculation and were highly correlated to each other in the previous experiment, as shown in Figure 8.

4.2.2. The performance gain caused by the DIT techniques

To investigate whether there is a relation between the values returned by the complexity measures and the gains obtained by the DIT techniques, we first investigated the effect of the DIT techniques on the predictive performance of ML algorithms for the 101 datasets used. For such, we assessed the gmean performance of six classification algorithms, using their default hyperparameter values. Figure 10 shows six boxes, one for each ML algorithm, with the differences in the predictive performance of each classifier before and after applying each DIT technique. The blue boxplots show the results for the datasets whose minority class has more than 25% of representation (low imbalance) and the orange boxplots show the results for the datasets with less than 25% of minority class representation (high imbalance).

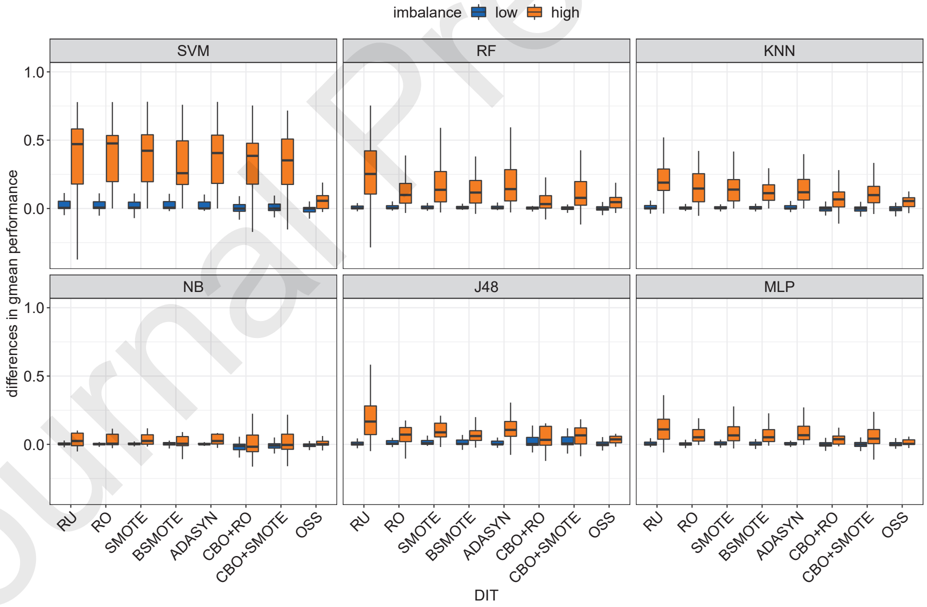

Figure 10: Differences in gmean performance before and after applying DIT

Figure 10 shows an improvement for all DIT techniques for most classification algorithms, with the exception of OSS. The improvements are larger for the highly imbalanced datasets. In general, ML algorithms have more difficulty in learning good models from these datasets.

The SVM classifier was the algorithm with the largest improvement in the predictive performance after applying the DIT techniques. However, it was the algorithm with the lowest predictive performance on average before applying the DIT techniques. Besides, SVM predictive performance is highly affected by the hyperparameter values [32], which were not tuned in these experiments.

As discussed previously, RO was the DIT technique that reduced the data complexity the most regarding the N3 measure. However, RO did not outperform the other DIT techniques in predictive performance. Actually, in some cases, it performed worse than other techniques, for example RU and SMOTE, when inducing RF and J48. To balance the training set, RO duplicates the number of instances in the minority class. Since it was applied to highly imbalanced datasets, probably RO duplicated most, if not all, minority class examples from the datasets. As N1, N3, and wCM11 are based on the nearest neighbors, the duplication of the instances in the minority class affects their values. However, the improvement on the predictive performance was not better than the other techniques. We believe that RO artificially over reduces the values of those measures causing an underestimation of the data complexity after its application.

4.2.3. The relation between data complexity modification and performance gain

In order to verify if the reduction in data complexity and the improvement in predictive performance are correlated, we used the Pearson correlation. The results are shown in Figure 11, where the x-axis represents the data complexity measures considered, the y-axis represents the method used (combination of DIT technique and classification algorithm), and each cell of the heatmap is the value of the Pearson correlation between the differences in data complexity and differences in predictive performance when a method is applied to the 101 datasets considered. Values followed by "*" mean that the p-value for that correlation was above 0.05.

The DIT technique that, in general, presented the lowest correlations in magnitude was OSS. Moreover, NB and J48 performances are usually not well correlated with the reduction in data complexity of any of the measures considered compared to the other classification algorithms when the same DIT technique is applied. N1, N3 and wCM11 do not correlate well when CBO-based techniques were applied.

Despite some low correlations in magnitude pointed out previously, most of the differences in data complexity are highly correlated with the predictive performance of the methods considered. They corroborate with the evidence discussed in Section 4.1 that the adapted data complexity measures considered are suitable tools to assess the data complexity of imbalanced datasets, now considering when DIT techniques are applied.

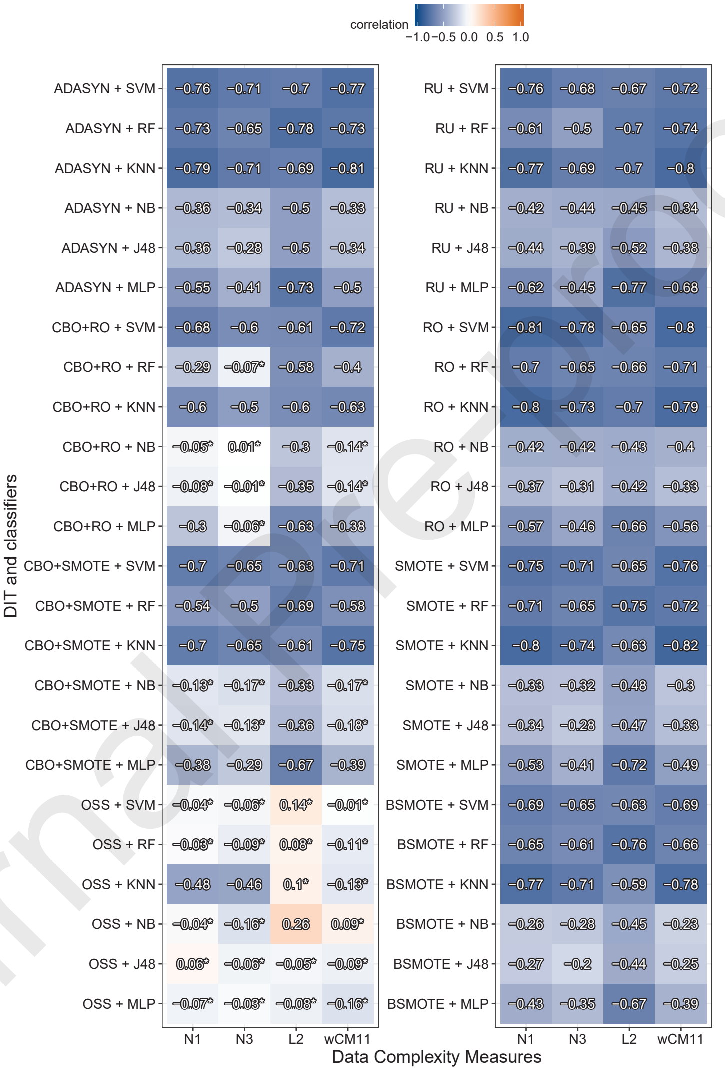

Figure 11: Correlation between the difference in data complexity and difference in predictive performance for all DIT techniques and classification algorithms considered

5. Final Remarks

As confirmed in this paper, dataset imbalance is not a problem in itself. But it increases the chances of the adverse effects of other characteristics, such as overlapping and difficult border decisions. To investigate these effects, we used data complexity measures. The original data complexity measures have been used in the literature to assess these characteristics, including their occurrence in imbalanced tasks. We show in this paper that the original data complexity measures do not work well with imbalance in real datasets. Therefore we strongly discourage their use in these scenarios. However, we also show that simple adaptations of these measures can make them useful to assess the difficulty of ML classification algorithms to deal with imbalanced datasets.

According to our experimental results, most of the adapted data complexity measures correlated better with the difficulty in imbalanced tasks than the imbalance ratio itself. Thus, the adapted measures can assess the difficulty of inducing a good model from a dataset better than the imbalance ratios used in the literature. Thus, the adapted measures can provide meaningful insights for data science researchers and practitioners. They can improve the understanding of the difficulty of the datasets used and guide the application of ML algorithms to these datasets. Another contribution from this study is to show the importance of selecting DIT techniques that can effectively reduce the data complexity, instead of only balancing the training set.

The experimental results show that the reduction of data complexity obtained by using DIT techniques occur mainly for highly imbalanced datasets. They also show that, for most of the DIT investigated, there is a correlation between the reduction in data complexity and gain in predictive performance.

Our adaptations of complexity measures were designed and tested only on binary datasets. For use in multiclass datasets, some of the data complexity measures may be dependent on the class decomposition strategy used (one versus all; one versus one). We believe that this is a good direction for future work. Moreover, we considered only the gmean performance for the experiments, because it is the most popular in recent works. Although it is a widely used performance measure for imbalanced data classification tasks, it would be interesting to study the behavior of the data complexity measures with other metrics used in imbalanced dataset classification tasks, such as AUC, F-measure, and kappa.

Future work shall consider the use of the adapted complexity measures in the proposal of meta-learning systems for the recommendation of suitable DIT techniques for a new dataset. The measures values can also be explored in the proposal of new data balancing strategies. For instance, one may guide the generation of new instances in the minority class in order to optimize a given measure value.

Acknowledgment

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES). The authors would like to thank CNPq, FAPESP (grants 2012/22608-8 and 2015/01382-0), CAPES (grant 88887.162551/2018-00), and CAPES/COFECUB (grant 99999.000555/2016-09) for their financial support. The authors would like to thank CeMEAI-FAPESP (grant 2013/07375-0) for the computational resources and Intel for the hardware and software server used in part of the experiments.

References

[1] Abdi, L. and Hashemi, S. (2016). To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge & Data Engineering*, 28(1):238–251.

[2] Anwar, N., Jones, G., and Ganesh, S. (2014). Measurement of data complexity for classification problems with unbalanced data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 7(3):194–211.

[3] Barella, V. H., Garcia, L. P. F., de Souto, M. P., Lorena, A. C., and de Carvalho, A. (2018). Data complexity measures for imbalanced classification tasks. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

[4] Barua, S., Islam, M. M., Yao, X., and Murase, K. (2014). MWMOTE–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):405–425.

[5] Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29.

[6] Cano, A., Zafra, A., and Ventura, S. (2013). Weighted data gravitation classification for standard and imbalanced data. *IEEE transactions on cybernetics*, 43(6):1672–1687.

[7] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

[8] Cieslak, D. A., Hoens, T. R., Chawla, N. V., and Kegelmeyer, W. P. (2012). Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158.

[9] Diamantini, C. and Potena, D. (2009). Bayes vector quantizer for class-imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 21(5):638–651.

[10] Díez-Pastor, J. F., Rodríguez, J. J., García-Osorio, C. I., and Kuncheva, L. I. (2015). Diversity techniques improve the performance of the best imbalance learning ensembles. *Information Sciences*, 325:98–117.

[11] Fernandes, E. R. and de Carvalho, A. C. (2019). Evolutionary inversion of class distribution in overlapping areas for multi-class imbalanced learning. *Information Sciences*, 494:141–154.

[12] Fernández, A., del Jesus, M. J., and Herrera, F. (2015). Addressing overlapping in classification with imbalanced datasets: A first multi-objective approach for feature and instance selection. In *International Conference on Intelligent Data Engineering and Automated Learning (IDEAL)*, pages 36–44.

[13] Fernndez, A., Garca, S., Galar, M., Prati, R., Krawczyk, B., and Herrera, F. (2018). *Learning from Imbalanced Data Sets*. Springer International Publishing.

[14] Garcia, L. P. F., de Carvalho, A. C. P. L. F., and Lorena, A. C. (2015). Effect of label noise in the complexity of classification problems. *Neurocomputing*, 160:108–119.

[15] Garcia, L. P. F. and Lorena, A. C. (2018). ECoL: Complexity measures for classification problems. https://CRAN.R-project.org/package=ECoL.

[16] Garcia, L. P. F., Lorena, A. C., de Souto, M. P., and Ho, T. K. (2018). Classifier recommendation using data complexity measures. In *24th International Conference on Pattern Recognition (ICPR)*, volume 1, pages 874–879.

[17] Gonzalez-Abril, L., Nuñez, H., Angulo, C., and Velasco, F. (2014). GSVM: An SVM for handling imbalanced accuracy between classes inbi-classification problems. *Applied Soft Computing*, 17:23–31.

[18] Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *International Conference on Intelligent Computing (ICIC)*, pages 878–887.

[19] He, H., Bai, Y., Garcia, E. A., and Li, S. (2008). ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1322–1328.

[20] He, H. and Garcia, E. A. (2008). Learning from imbalanced data. *IEEE Transactions on Knowledge & Data Engineering*, 21(9):1263–1284.

[21] Ho, T. K. and Basu, M. (2002). Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300.

[22] Ho, T. K., Basu, M., and Law, M. H. C. (2006). Measures of geometrical complexity in classification problems. In *Data Complexity in Pattern Recognition*, pages 1–23.

[23] Jo, T. and Japkowicz, N. (2004). Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1):40–49.

[24] Kolaczyk, E. D. (2009). *Statistical Analysis of Network Data: Methods and Models*. Springer Publishing Company, Incorporated.

[25] Kubat, M., Matwin, S., et al. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *14th International Conference on Machine Learning (ICML)*, volume 97, pages 179–186.

[26] Lorena, A. C. and de Souto, M. C. P. (2015). On measuring the complexity of classification problems. In *International Conference on Neural Information Processing*, pages 158–167.

[27] Lorena, A. C., Garcia, L. P. F., Lehmann, J., de Souto, M. C. P., and Ho, T. K. (2019). How complex is your classification problem? A survey on measuring classification complexity. *ACM Computing Surveys (CSUR)*, 52(5).

[28] Lu, Y., Cheung, Y.-m., and Tang, Y. Y. (2019). Bayes imbalance impact index: A measure of class imbalanced data set for classification problem. *IEEE transactions on neural networks and learning systems*.

[29] Luengo, J., Fernández, A., García, S., and Herrera, F. (2011). Addressing data complexity for imbalanced data sets: Analysis of SMOTE-based oversampling and evolutionary undersampling. *Soft Computing*, 15(10):1909–1936.

[30] Luengo, J. and Herrera, F. (2015). An automatic extraction method of the domains of competence for learning classifiers using data complexity measures. *Knowledge and Information Systems*, 42(1):147–180.

[31] Macià, N. and Bernadó-Mansilla, E. (2014). Towards UCI+: A mindful repository design. *Information Sciences*, 261:237–262.

[32] Mantovani, R. G., Rossi, A. L., Vanschoren, J., Bischl, B., and Carvalho, A. C. (2015). To tune or not to tune: Recommending when to adjust SVM hyper-parameters via meta-learning. In *International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

[33] Ng, W. W., Hu, J., Yeung, D. S., Yin, S., and Roli, F. (2015). Diversified sensitivity-based undersampling for imbalance classification problems. *IEEE transactions on cybernetics*, 45(11):2402–2412.

[34] Orriols-Puig, A., Maciá, N., and Ho, T. K. (2010). Documentation for the data complexity library in C++. Technical report, La Salle - Universitat Ramon Llull.

[35] Sáez, J. A., Luengo, J., Stefanowski, J., and Herrera, F. (2015). SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering. *Information Sciences*, 291:184–203.

[36] Singh, D., Gosain, A., and Saha, A. (2020). Weighted k-nearest neighbor based data complexity metrics for imbalanced datasets. *Statistical Analysis and Data Mining: The ASA Data Science Journal*.

[37] Smith, F. W. (1968). Pattern classifier design by linear programming. *IEEE transactions on computers*, 100(4):367–372.

[38] Smith, M. R., Martinez, T., and Giraud-Carrier, C. (2014). An instance level analysis of data complexity. *Machine learning*, 95(2):225–256.

[39] Tomek, I. (1976). Two modifications of CNN. *IEEE Trans. Systems, Man and Cybernetics*, 6:769–772.

[40] Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. (2013). OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60.

[41] Yu, H., Ni, J., and Zhao, J. (2013). ACOSampling: An ant colony optimization-based undersampling method for classifying imbalanced DNA microarray data. *Neurocomputing*, 101:309–318.

Appendices

Table 20: Information about the 102 datasets used in the experiment to evaluate the data complexity measures

OpenML ID	Number of Instances	Number of Features	% Minority Class	OpenML ID	Number of Instances	Number of Features	% Minority Class
31	1000	21	30	**927**	42	17	40.48
43	306	4	26.47	**928**	46	5	45.65
444	132	4	46.21	**931**	662	4	47.43
463	180	33	13.89	**934**	1156	6	22.15
467	52	10	48.08	**938**	42	11	45.24
472	87	4	40.23	**945**	76	7	47.37
714	125	5	39.2	**949**	559	5	14.31
717	508	11	43.7	**950**	559	5	3.4
724	468	4	44.44	**958**	2310	20	14.29
729	44	4	38.64	**962**	2000	7	10
733	209	7	26.79	**964**	36	23	33.33
736	111	4	47.75	**983**	1473	10	42.7
747	167	5	22.75	**987**	500	24	16
748	163	6	28.83	**988**	67	16	38.81
753	194	33	46.39	**991**	1728	7	29.98
758	67	16	26.87	**994**	846	19	25.77
764	450	4	12.22	**1009**	63	32	39.68
767	475	4	12.84	**1014**	797	5	19.45
770	625	7	49.6	**1016**	990	14	9.09
772	2178	4	44.49	**1020**	2000	65	10
777	47	8	42.55	**1025**	400	6	22.5
778	252	15	49.21	**1026**	155	9	31.61
780	51	7	41.18	**1045**	145	95	5.52
782	120	3	47.5	**1050**	1563	38	10.24
785	45	47	48.89	**1055**	89	9	22.47
787	50	6	48	**1061**	107	30	18.69
790	55	3	43.64	**1064**	101	30	14.85
791	43	3	39.53	**1066**	145	95	41.38
800	74	28	41.89	**1067**	2109	22	15.46
801	185	4	47.03	**1073**	274	9	48.91
811	264	3	38.26	**1075**	130	9	8.46
818	310	9	46.77	**1443**	661	38	7.87
825	506	21	44.07	**1444**	1043	38	12.18
826	576	12	41.49	**1446**	296	38	12.84
827	662	4	49.7	**1450**	125	40	35.2
841	950	10	48.63	**1451**	705	38	8.65
848	38	6	26.32	**1452**	745	37	2.15
859	74	10	41.89	**1462**	1372	5	44.46
860	380	3	48.68	**1464**	748	5	23.8
875	100	4	19	**1473**	100	10	12
882	60	16	48.33	**1487**	2534	73	6.31
885	131	4	36.64	**1488**	195	23	24.62
890	108	8	29.63	**1495**	250	7	42.8
891	93	7	38.71	**1504**	1941	34	34.67
893	73	6	45.21	**1506**	470	17	14.89
895	222	3	39.64	**1600**	267	45	20.6
900	400	7	41.25	**23499**	277	10	29.24
907	400	8	48.5	**40669**	160	7	43.75
915	315	14	42.22	**40705**	959	45	36.08
921	132	4	34.85	**40710**	303	14	45.54
925	323	5	45.82	**40981**	690	15	44.49

Table 21: Information about the 102 datasets used in the experiment to evaluate the data complexity measures

OpenML ID	Number of Instances	Number of Features	% Minority Class	OpenML ID	Number of Instances	Number of Features	% Minority Class
37	768	9	34.9	**946**	88	3	48.86
40	208	61	46.63	**947**	559	5	4.29
50	958	10	34.66	**951**	559	5	2.33
53	270	14	44.44	**955**	151	6	34.44
59	351	35	35.9	**965**	101	18	40.59
311	937	50	4.38	**969**	150	5	33.33
336	267	23	20.6	**970**	841	71	37.69
448	120	4	35	**973**	178	14	39.89
450	264	5	7.2	**974**	132	5	38.64
459	83	4	44.58	**996**	214	10	35.51
461	100	7	27	**997**	625	5	46.08
465	97	11	24.74	**1004**	600	62	16.67
479	92	11	20.65	**1006**	148	19	45.27
713	52	4	46.15	**1011**	336	8	42.56
719	137	8	31.39	**1012**	194	30	35.57
721	200	11	48.5	**1013**	138	3	6.52
731	96	5	48.96	**1015**	72	4	16.67
741	1024	3	49.71	**1048**	369	9	44.72
745	159	16	33.96	**1054**	161	40	32.3
750	500	8	49.2	**1059**	121	30	7.44
765	475	4	13.47	**1060**	63	30	12.7
771	108	5	44.44	**1062**	36	30	22.22
774	662	4	47.89	**1063**	522	22	20.5
788	186	61	41.4	**1065**	458	40	9.39
795	662	4	49.4	**1071**	403	38	7.69
796	209	8	25.36	**1121**	294	12	28.57
798	106	58	22.64	**1167**	320	9	33.44
804	70	8	48.57	**1412**	226	24	15.49
814	468	3	45.3	**1441**	123	40	13.01
815	52	10	46.15	**1442**	253	38	10.67
817	48	5	47.92	**1447**	327	38	12.84
820	235	13	39.57	**1448**	194	40	18.56
835	48	5	43.75	**1449**	253	38	10.67
836	34	9	44.12	**1463**	100	6	32
853	506	14	41.3	**1467**	540	21	8.52
857	40	8	35	**1480**	583	11	28.64
862	87	11	48.28	**1490**	182	13	28.57
864	60	8	45	**1494**	1055	42	33.74
865	100	4	7	**1498**	462	10	34.63
867	130	3	19.23	**1510**	569	31	37.26
874	50	6	42	**1511**	440	9	32.27
886	500	8	49.8	**1524**	310	7	32.26
887	61	3	47.54	**1556**	120	7	49.17
892	50	8	48	**4329**	470	17	14.89
902	147	7	46.94	**40660**	42	12	30.95
905	39	4	30.77	**40680**	1324	11	22.05
906	400	8	48.25	**40683**	88	9	27.27
908	400	8	48	**40702**	1066	11	17.07
909	400	8	49.25	**40999**	2351	47	44.02
941	189	10	47.62	**41007**	2352	47	40.35
942	50	5	48				

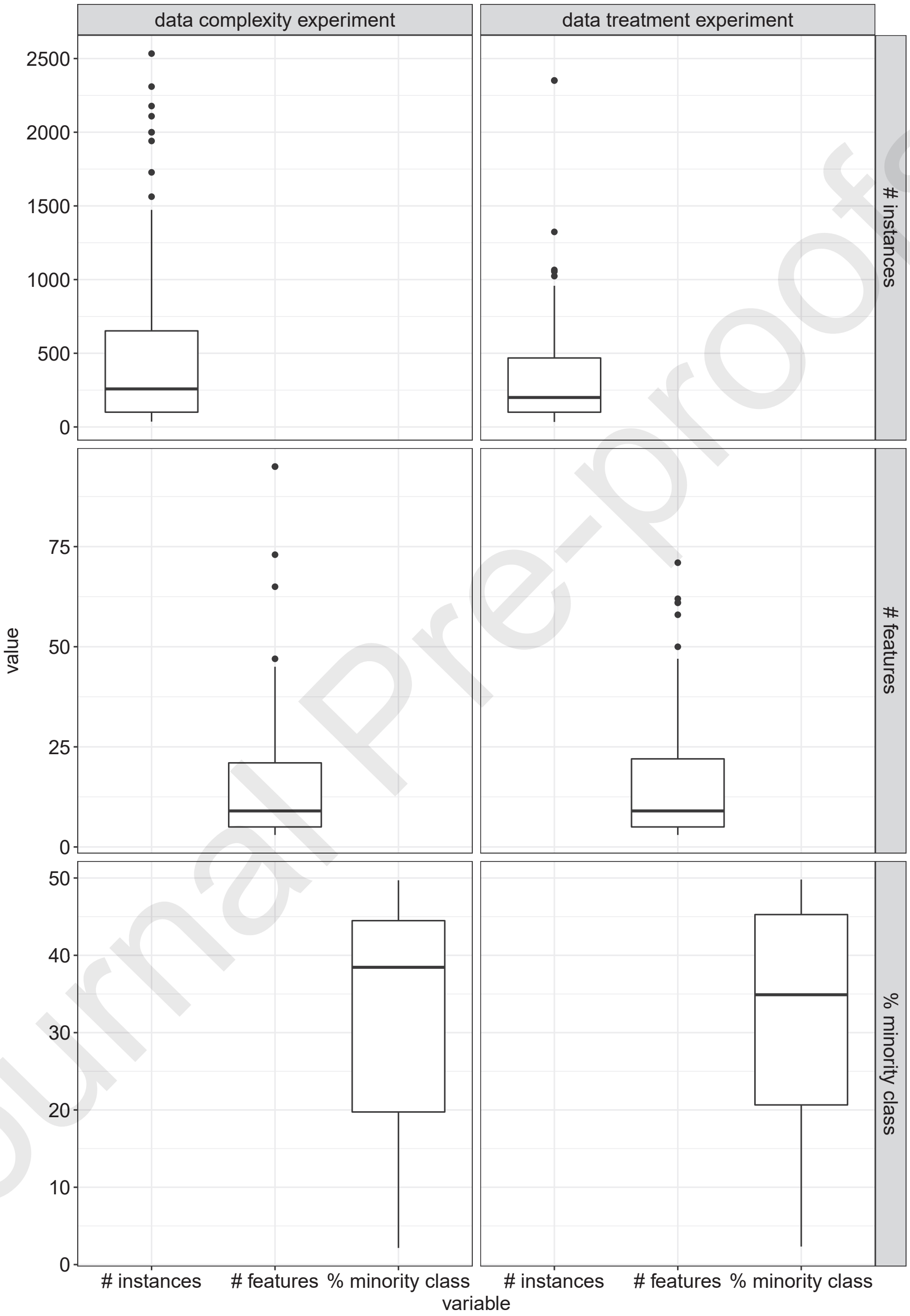

Figure 12: Comparison of the characteristics of both groups of datasets