Neurocomputing 343 (2019) 76–99

Contents lists available at ScienceDirect

Neurocomputing

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

Pre-processing approaches for imbalanced distributions in regression

Paula Branco[a,*], Luis Torgo[a,b], Rita P. Ribeiro[a]

[a] INESC TEC/DCC – Faculty of Sciences, University of Porto, Portugal
[b] Faculty of Computer Science, Dalhousie University, Canada

ARTICLE INFO

Article history:
Received 20 November 2017
Revised 9 July 2018
Accepted 29 November 2018
Available online 3 February 2019

Keywords:
Imbalanced distributions
Pre-processing
Regression

ABSTRACT

Imbalanced domains are an important problem frequently arising in real world predictive analytics. A significant body of research has addressed imbalanced distributions in classification tasks, where the target variable is nominal. In the context of regression tasks, where the target variable is continuous, imbalanced distributions of the target variable also raise several challenges to learning algorithms. Imbalanced domains are characterized by: (1) a higher relevance being assigned to the performance on a subset of the target variable values; and (2) these most relevant values being underrepresented on the available data set. Recently, some proposals were made to address the problem of imbalanced distributions in regression. Still, this remains a scarcely explored issue with few existing solutions. This paper describes three new approaches for tackling the problem of imbalanced distributions in regression tasks. We propose the adaptation to regression tasks of random over-sampling and introduction of Gaussian Noise, and we present a new method called ***WE**ighted **R**elevance-based **C**ombination **S**trategy* (**WERCS**). An extensive set of experiments provides empirical evidence of the advantage of using the proposed strategies and, in particular, the **WERCS** method. We analyze the impact of different data characteristics in the performance of the methods. A data repository with 15 imbalanced regression data sets is also provided to the research community.

1. Introduction

Learning from imbalanced distributions is a relevant problem with several important practical applications. In this type of tasks the main goal is to obtain a model using a data set where the most important values of the target variable are scarcely represented. Example applications span from the medical area (e.g. prediction of rare diseases), meteorology (e.g. catastrophes prediction) or financial markets (e.g. forecast of extreme returns in stock markets), among many others. The scientific community has been working intensively on this problem and several aspects related with imbalance learning have been addressed [4,11,13]. The majority of solutions proposed for tackling the problem of imbalanced domains are for classification tasks. However, this problem also arises in regression tasks. Recently, imbalanced distributions in regression problems have been gaining more attention [4] although only a few solutions exist.

Imbalanced regression tasks arise in a diversity of domains. For instance, in a financial trading context, consider the task of predicting the return of an asset. This prediction can be used by an agent for buying/selling the asset. However, this agent may invest different amounts depending on the predicted return. Obtaining accurate predictions of higher (lower) returns is highly important for the agent because these values may lead either to significant losses or large missed profits. Therefore, it is important to correctly identify the type of extreme values and to obtain accurate numeric predictions. However, the main difficulty of the task is related with the rarity of these extreme values in the asset's historical data causing a performance degradation of the models on the important cases.

Imbalanced distributions create serious problems on both the performance evaluation and the learning phases. Ribeiro [22] showed that traditionally used regression performance assessment metrics are not suitable for imbalanced domains because they fail to capture what is important to the user. These tasks require adequate performance metrics. Regarding the second issue, standard learning algorithms experience a severe performance loss because they are not able to focus on the most relevant and rare cases [4]. Therefore, we need to use approaches capable of improving the learning algorithms performance. The focus of this paper is on methods to accomplish this.

Pre-processing approaches are a possible solution for handling imbalanced domains. These solutions act before the learning stage and thus have the advantage of allowing the use of any standard

* Corresponding author.
E-mail address: pobranco@inesctec.pt (P. Branco).

https://doi.org/10.1016/j.neucom.2018.11.100
0925-2312/

learning algorithm. Moreover, the model's comprehensibility is not affected by the pre-processing strategies because they do not modify neither the model nor the obtained predictions.

The main goal of this paper is to present new pre-processing solutions to address the problem of imbalanced distributions in regression tasks. We compare the proposed methods with existing solutions and provide a data repository with the 15 imbalanced regression data sets used in this work. We also explore the relation between data characteristics and the performance of several learners and pre-processing approaches. Our main contributions are: (i) propose a new pre-processing approach (**WERCS**) for dealing with this type of problems; (ii) present the adaptation to regression tasks of two approaches developed for classification; (iii) test and compare our proposed solutions against existing alternatives; (iv) provide a repository with 15 imbalanced data sets for regression; (v) analyze the impact of different data characteristics in the predictive performance of the models; and (vi) study the effect of varying the number of important cases on the performance.

Section 2 presents the problem definition. In Section 3 a brief overview of the related work is provided. Our proposals are described in Section 4 and the results of an extensive experimental evaluation are discussed in Section 5. In Section 6 we present a use case where the techniques described in the paper are applied. Finally, Section 7 presents the main conclusions of this paper.

2. Problem definition

Imbalanced regression is a particular sub-class of regression problems. In this setting, given a training set, $\mathcal{D} = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{N}$, the goal is to obtain a model $m(\mathbf{x})$ that approximates the unknown regression function $Y = f(\mathbf{x})$. The conjunction of two specific factors causes a performance degradation on the cases that are more important for the user. These two factors are: (i) the user preferences towards a subset of the cases, and (ii) the lack of representativeness of these cases in the available training data. Thus, the problem of imbalanced regression can be described by the two following assertions as proposed by Branco et al. [4]:

- **A.1** the predictive performance of the obtained model $m(\mathbf{x})$ has a non-uniform importance in the continuous target variable domain ($\mathcal{Y}$); and
- **A.1** the more relevant cases for the user are poorly represented in $\mathcal{D}$.

In imbalanced regression, defining which are the relevant cases is not as straightforward as it is, for instance, in binary classification tasks. In a two-class setting, we typically assume that the important cases (mentioned in assertion **A.2**) are those belonging to the minority class (often referred to as the *positive* class). To address this additional difficulty the concept of a relevance function was put forward by Ribeiro [22] with the goal of providing information regarding the relevant ranges of a continuous target variable. This domain-specific function is crucial for transforming informal domain knowledge into formal, and thus usable, knowledge for predictive modeling.

Definition 1 (Relevance function)**.** The relevance function $\phi(y)$ is a function that maps the values of the target variable into a relevance scale, where 1 corresponds to the maximal relevance and 0 to minimum relevance.

$$\phi : \mathcal{Y} \rightarrow [0, 1] \tag{1}$$

To deal with imbalanced regression problems, the user is also required to define a threshold (t_R) on the relevance values. This threshold sets the boundary above which the target variable values are relevant. We use t_R to define a partition of $\mathcal{Y}$ in two complementary subsets: $\mathcal{Y}_R = \{y \in \mathcal{Y} : \phi(y) > t_R\}$ and $\mathcal{Y}_N = \mathcal{Y} \setminus \mathcal{Y}_R$. The subset $\mathcal{D}_R \subset \mathcal{D}$ contains the examples satisfying $y \in \mathcal{Y}_R$ and $\mathcal{D}_N = \mathcal{D} \setminus \mathcal{D}_R$. Using this notation, the problem of imbalanced distributions can be defined as follows [4].

Definition 2 (Problem of imbalanced distribution)**.** Given a relevance function $\phi()$, and a threshold t_R on the relevance scores, we face a problem of imbalanced distribution when the following occurs simultaneously:

- **A.1** the predictive performance of the obtained model $m(\mathbf{x})$ is more important for cases belonging to $\mathcal{D}_R$; and
- **A.2** $|\mathcal{D}_R| << |\mathcal{D}_N|$.

Ideally, the definition of the relevance function $\phi()$ should be provided by a domain expert. However, given that this is usually hard to obtain, an automatic method was proposed by Ribeiro [22] for estimating a relevance function $\phi(Y)$ for a given domain. This method is based on the assumption that the relevance is inversely proportional to the target variable probability density function, which is the common scenario in imbalanced domains. The automatic method for obtaining the relevance function assumes that the preferences of the user are biased towards the extreme and rare cases, i.e., the most important cases are underrepresented in the data.

The violation of any of the assertions established in the definition of this problem results in the elimination of the problem of imbalanced domains. If assertion **A.1** is not fulfilled, this means that either all cases are equally relevant to the user, i.e., the user has a uniform interest over the domain, or, the user preferences are biased towards the most frequent cases (D_N). In either case, this poses no problem to the learning procedure that will typically focus on the most frequent cases. If assertion **A.2** fails, this means that the distribution is approximately balanced, i.e., $|D_R| \approx |D_N|$. In this case, the learner will not have difficulties neither on the rare nor on the normal cases. For example, let us consider a data set with a continuous target variable representing the values obtained from a sensor. Suppose that assertion **A.1** is not observed. This means that the user considers all the target variable values equally important, and therefore, an error with a certain magnitude on a low, high or average target value will have the same importance for the user. In this setting, it is not important where in the target variable domain the error occurred. The user is focused in minimizing the average error, and it is not important which type of errors the model makes as long as this average is as low as possible. This means that making a prediction of 30.1 for a true value of 28 is as serious for the user as making a prediction of -30.1 for a true value of -28. In such contexts, we do not face an imbalanced domain problem, even if some values are underrepresented. Let us now suppose that the user is more interested in accurate predictions on the higher values of the target variable. This means that assertion **A.1** is observed. Let us also suppose that these higher target variable values are the most common on the training set (i.e. assertions **A.2** fails). In this case, using a standard model that minimizes the average error, will be OK because the average will be affected by the most common cases which happen to be the most important to the user. Thus, in this setting we also do not face an imbalanced domain problem. This means that only when assertions **A.1** and **A.2** are both true, we face an imbalanced learning task.

Learning from imbalanced domains raises two challenges: (i) how to correctly evaluate the performance of the models; and (ii) how to bias the learners to the most relevant cases. We discuss the issue of performance assessment in the following subsection, presenting the used evaluation framework. The second issue, of biasing the learners towards the most relevant cases, is the focus of this paper. To tackle this problem we will propose three new pre-processing methods.

2.1. Evaluation on imbalanced regression

The issue of performance assessment on imbalanced domains is of great importance. Standard evaluation metrics are inadequate for this context because they can lead to sub-optimal models [10,14,29] and to misleading conclusions [7,21]. This issue has been studied in depth for classification tasks. In this context, several metrics were proposed to overcome the difficulties detected. The same problem occurs in regression tasks with imbalanced distributions. Standard regression metrics (e.g. mean squared error) are unsuitable for these problems because they are not focused on the performance of the least represented and important examples [4,22].

In this context, Torgo and Ribeiro [25] and Ribeiro [22] proposed a framework based on the notion of utility-based learning that allows to obtain precision and recall for this type of regression tasks. At the core of utility-based regression is the definition of a non-uniform relevance function $\phi(y)$. This function allows the user to distinguish the importance assigned to different ranges of values of the target variable. The proposed utility-based framework provides the usefulness of a prediction as a function of: (i) the numeric error of the prediction, i.e. the loss function value $L(\hat{y}, y)$; and (ii) the relevance of both the predicted and true target variable values, i.e. the location and respective importance of the true and predicted values. This means that, the utility of a prediction is related with both the identification of the correct "range" and the numeric accuracy. This framework provides means for easily obtaining a utility score, that expresses the usefulness of predicting $\hat{y}$ for the true value y. In particular, a utility score is derived for each pair $(\hat{y}, y)$, by using the notions of benefits and costs of numeric predictions. This score is obtained using the utility function $U_\phi^p(\hat{y}, y)$:

$$\begin{aligned} U_\phi^p(\hat{y}, y) &= B_\phi\left(\hat{y}, y\right) - C_\phi^p\left(\hat{y}, y\right) \\ &= \phi(y) \cdot \left(1 - \Gamma_B(\hat{y}, y)\right) - \phi^p(\hat{y}, y) \cdot \Gamma_C\left(\hat{y}, y\right) \end{aligned} \tag{2}$$

where functions $B_\phi(\hat{y}, y)$ and $C_\phi^p(\hat{y}, y)$ represent the benefits and the costs incurred when predicting $\hat{y}$ for a true value y, respectively. This means that the utility of a pair of values is obtained as the net balance between the benefits and costs of a certain prediction in the context of the domain. The benefit of a prediction ($B_\phi(\hat{y}, y)$) is defined as a proportion of the relevance of the true value. This means if the prediction is perfect (i.e. equal to the true value), then the benefit is maximum and equal to the relevance of this true value. This is captured by the expression $\phi(y) \cdot (1 - \Gamma_B(\hat{y}, y))$, where $\Gamma_B(\hat{y}, y)$ is a bounded loss function that normalizes the standard loss function ($L(\hat{y}, y)$) into a [0,1] scale, so that after $L(\hat{y}, y)$ reaches a certain value the bounded loss is maximum (1). For the cost component things are slightly more complicated but the idea is similar. The cost of a prediction ($C_\phi^p(\hat{y}, y)$) is calculated as a proportion of the maximum cost that is defined as a weighted average between the relevance of the true value and the relevance of the predicted value. This weighted relevance average is given by $\phi^p(\hat{y}, y)$ that uses parameter p to defined the weights between the two relevances (0.5 gives equal importance to both). Finally, function $\Gamma_C(\hat{y}, y)$ is also a bounded loss function in the scale [0,1], similarly to $\Gamma_B(\hat{y}, y)$. Full details can be found in [22].

Two measures frequently used in imbalanced classification tasks are precision and recall. Precision and recall are defined as ratios between the correctly identified events and the signaled events (precision), or the trues events (recall). Although a high diversity of solutions has been proposed for addressing the evaluation problem in imbalanced classification, for regression tasks, the available solutions are scarce [4]. In the context of imbalanced regression, the semantics of precision and recall can be captured using the utility-based framework, while incorporating the notion of numeric accuracy necessary of regression tasks [22,26]. Based on this work, Branco [2] proposed the following definition of precision and recall for regression:

$$prec^\phi = \frac{\sum_{\phi(\hat{y}_i) > t_R} \left(1 + U_\phi^p\left(\hat{y}_i, y_i\right)\right)}{\sum_{\phi(\hat{y}_i) > t_R} \left(1 + \phi(\hat{y}_i)\right)} \tag{3}$$

$$rec^\phi = \frac{\sum_{\phi(y_i) > t_R} \left(1 + U_\phi^p\left(\hat{y}_i, y_i\right)\right)}{\sum_{\phi(y_i) > t_R} \left(1 + \phi(y_i)\right)} \tag{4}$$

where $\phi(y_i)$ is the relevance associated with the true value y_i, $\phi(\hat{y}_i)$ is the relevance of the predicted value $\hat{y}_i$, t_R is a user-defined threshold signaling the cases that are relevant for the user, and $U_\phi^p(\hat{y}_i, y_i)$ is the utility of making the prediction $\hat{y}_i$ for the true value y_i, normalized to $[-1, 1]$.

In the experimental evaluation we use as main evaluation metric the F_1-measure adapted for regression tasks, F_1^ϕ, that depends on the precision ($prec^\phi$) and recall (rec^ϕ) measures previously defined.

$$F_1^\phi = \frac{2 \cdot prec^\phi \cdot rec^\phi}{prec^\phi + rec^\phi} \tag{5}$$

In a classification setting, the F_1 measure is a suitable and frequently used measure for imbalanced tasks. This metric is solely focused on the rare cases performance. Sometimes end-users may also be interested in observing the performance of the models on the least important (normal) cases. The $G-Mean$ provides this information. This is a metric that computes the geometric mean of the accuracies of the two classes of the problem. With the goal of providing a more general overview of the impact of our proposed methods, we have adapted the $G-Mean$ metric to a regression context, which we named $G-Mean^\phi$. To achieve this, we need to adapt the concepts of sensitivity (or recall) and specificity to regression. The former is already provided through Eq. (4). We proposed the adaptation of the latter in a similar way, as follows:

$$spec^\phi = \frac{\sum_{\phi(y_i) \leq t_R} \left(1 + U_\phi^p\left(\hat{y}_i, y_i\right)\right)}{\sum_{\phi(y_i) \leq t_R} \left(1 + \phi(y_i)\right)} \tag{6}$$

We use the concepts of utility (U_ϕ^p) and relevance (ϕ) of a pair of predicted and true values of the target variable $(\hat{y}_i, y_i)$. Therefore, $spec^\phi$ is the proportion between the utility achieved by the model on the true normal cases ($\phi(y) \leq t_R$) and the maximal achievable utility (i.e., the relevance) for those cases. Using rec^ϕ and $spec^\phi$ we are able to extend the notion of $G-Mean$ to regression as follows:

$$G-Mean^\phi = \sqrt{rec^\phi \times spec^\phi} \tag{7}$$

3. Related work

Imbalanced domains are an important and well-known problem when building predictive models. This problem occurs for several task types such as: classification, regression or multi-label classification, among others (e.g. [4,5,13]). However, it has been addressed mainly in the context of classification tasks and only a few solutions exist for tackling imbalanced regression tasks.

Pre-processing strategies are the most explored type of approaches for dealing with imbalanced domains, with several solutions proposed for classification. These strategies manipulate the original data set outputting a new changed data set with a target variable distribution biased towards the most relevant cases. Examples include under-sampling (e.g. [14]), over-sampling (e.g. [1]), or

combinations of the two (e.g. [6]). Both under- and over-sampling may be performed randomly or in a more informed way that uses some criteria to sample the data. Pre-processing strategies have several advantages that make them very appealing. For instance, they can reduce the data set size (in the case of under-sampling), which makes the learning process more efficient, and they also allow the use of any standard learning algorithm. The model obtained when using these strategies is more interpretable because it is biased towards the user goals. However, a well-known drawback of these approaches is related with the difficulty of determining the optimal way of changing the distribution. Recently, this type of strategies were also applied to imbalanced regression tasks to forecast rare extreme values. The proposed methods for addressing this problem were random under-sampling and SmoteR [24,27]. We will now briefly describe them.

Random under-sampling reduces the data set size by randomly removing observations with the most common and less important target variable values. The goal is to obtain a better balance between normal (and less interesting) observations and the rare important cases. Within a regression scenario, the under-sampling strategy [24,27] uses a relevance function and a threshold on the relevance values, both defined by the user, to determine which are the common and uninteresting cases. The user also sets a percentage that represents the amount of reduction to be carried out in the uninteresting cases. The new changed data set contains all the important cases and a random sample of the uninteresting cases.

The SmoteR method, proposed by Torgo et al. [27], is an adaptation for imbalanced regression tasks of the well-known Smote [6] algorithm developed for imbalanced classification. SmoteR uses the same approach as the above-mentioned under-sampling method for defining the sets of interesting and normal cases. The normal cases are randomly under-sampled while the relevant cases are over-sampled using an interpolation strategy. The over-sampling strategy of Smote was adapted for regression tasks as follows. The key idea is to use two examples from the set of rare cases and build a new synthetic example by interpolating the predictors of these cases, with the target variable value of the synthetic example being set as a weighted average of the target variable values of the two seed examples.

4. Our proposed strategies

In this section we present three new pre-processing methods for dealing with imbalanced regression tasks where the rare cases are the most important ones. These methods act before the learning stage by changing the training set. The obtained training set is more suitable for learning on imbalanced distributions because it is biased towards the user preferences. The proposed strategies include the adaptation to regression tasks of two approaches developed for classification – *Random Over-Sampling* (**RO**) and *Introduction of Gaussian Noise* (**GN**) and a novel approach – ***WE**ighted **R**elevance-based **C**ombination **S**trategy* (**WERCS**). Our proposals are all able to handle data sets with nominal and/or numeric features.

In a classification setting it is usually straightforward to know which are the data partitions (classes) where a strategy for removing or adding examples is going to be applied. In fact, for the most common problem of binary classification, the minority class examples will be targeted with an over-sampling strategy, while the majority class examples are candidates for the application of an under-sampling strategy. In regression, we have a more complex task because the target variable is continuous. For the adaptation of **RO** and **GN** to regression we apply the same strategy used in [27] that is based in the concepts of the relevance function ($\phi()$) and the relevance threshold (t_R), to build data partitions. Each data partition includes only examples with contiguous values of the target variable having either high or low relevance based on the user-defined t_R threshold. The main idea is to build a set of bins with normal cases,[1] denoted by $Bins_N$, and another set of bins containing the relevant examples, denoted as $Bins_R$. To achieve this, we use the following procedure:

i) sort the data set by ascending order of the target variable values;
ii) scan the data, starting from the lower Y value, and create a new bin whenever the relevance of the target variable value ($\phi(Y)$) changes from below to above the relevance threshold t_R or vice-versa.

This way we build bins with consecutive examples (in terms of the target variable value) such that, all the examples in a bin have a relevance value either above or below the threshold t_R. The bins with high relevance values contain important rare cases and therefore are candidates for the application of an over-sampling strategy. On the other hand, the bins with low relevance values include normal and less interesting examples that are candidates for the application of an under-sampling strategy. Algorithm 1 implements this method of obtaining the bins. For instance, in the synthetic data set of Fig. 1 we can observe three ranges of the target variable that are important to the user. The relevance function displayed with a dashed line produces two partitions with normal cases and three partitions with rare cases. This case leads to two distinct bins in $Bins_N$ and three different bins in $Bins_R$. The described bins that are created for this synthetic data set are displayed in Fig. 2.

Algorithm 1: Construction of bins.

Input: $\mathcal{D}$ - data set with target continuous variable Y
t_R - threshold for relevance on Y values
Output: *Bins* - data set partitions into relevant and normal bins
$OrdD \leftarrow$ order $\mathcal{D}$ by ascending value of Y
$\phi() \leftarrow$ relevance function obtained from Y distribution
$Bins_N \leftarrow k$ partitions of consecutive examples $\langle \mathbf{x}_i, y_i \rangle \in OrdD$, s. t. $\phi(y_i) < t_R$
$Bins_R \leftarrow l$ partitions of consecutive examples $\langle \mathbf{x}_i, y_i \rangle \in OrdD$, s. t. $\phi(y_i) \geq t_R$
$Bins \leftarrow Bins_R \bigcup Bins_N$
return *Bins*

This construction of bins can be seen as a form of discretization of the continuous target variable into a set of bins, containing either "normal" or relevant cases. However, it should not be interpreted as a method of transforming the initial regression task into a classification problem because several other issues related with the continuous nature of the target variable remain to be addressed. For instance, the bins do not solve the problem of determining the target variable value when adding new examples through over-sampling.

Until now all the solutions proposed in the literature to tackle imbalanced regression tasks depend on some sort of domain discretization based on a relevance function and a relevance threshold. In this paper we will present a new approach (WERCS) that avoids this discretization step.

4.1. Random over-sampling

Random over-sampling is a well-known strategy for dealing with imbalanced classification tasks (e.g. [1]). It consists of randomly selecting examples from the rare class which are then replicated in a new changed data set. The goal is to provide a better balance between the number of examples of each class without discarding any information.

Our first proposal is an adaptation of the random over-sampling (**RO**) strategy to imbalanced regression tasks. To achieve this, we

[1] Cases with target value y_i such that $\phi(y_i) < t_R$.

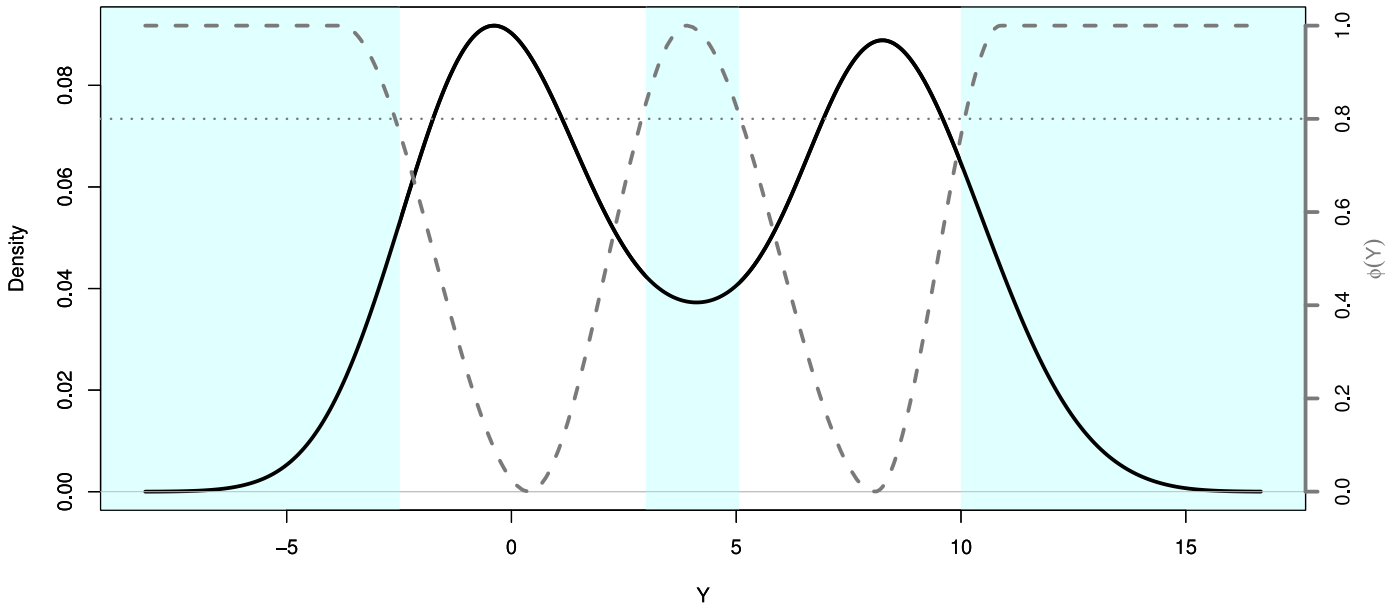


Fig. 1. Synthetic data set with three important ranges of the target variable (blue shaded regions) for a threshold t_R of 0.8 (horizontal dashed line). Density distribution (solid line) and relevance function $\phi()$ (dashed line).

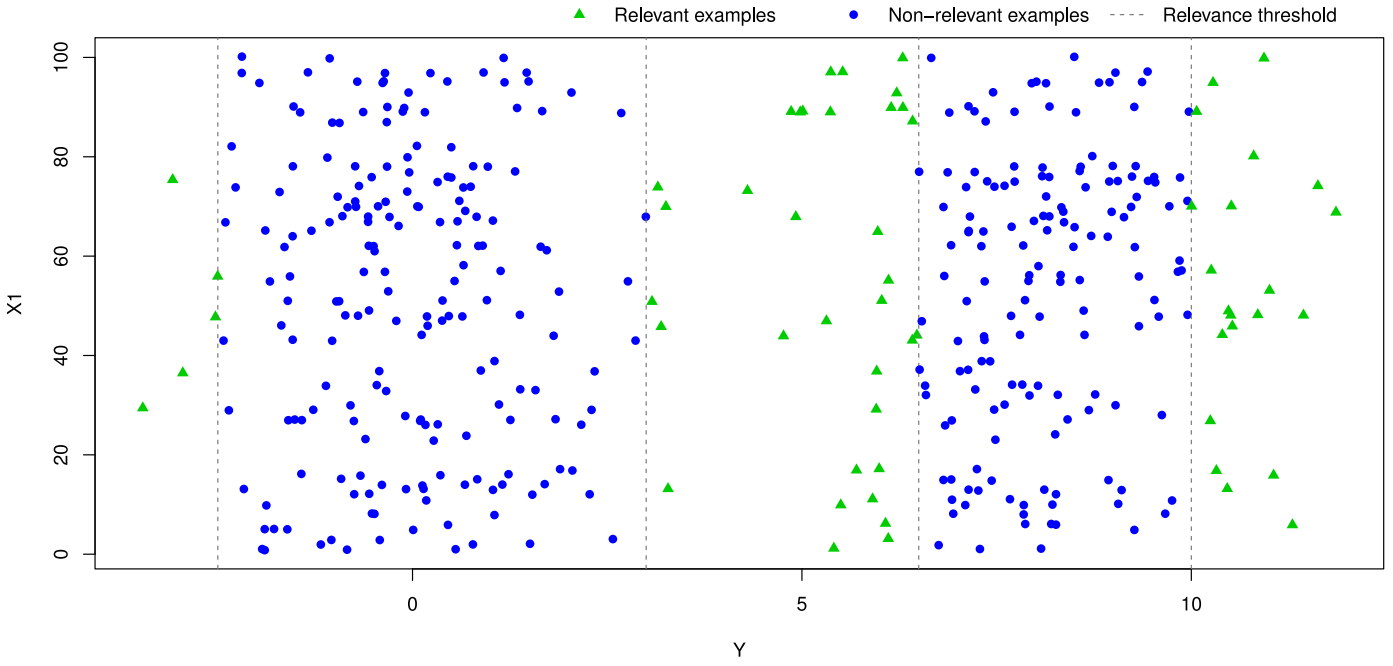


Fig. 2. Examples distribution according to the partitions obtained through the relevance function and threshold of Fig. 1.

use the relevance-based bins, $Bins_N$ and $Bins_R$, that we have described before. The examples in $Bins_N$ remain unchanged while replicas of the examples in each bin of the set $Bins_R$, are added. The number of replicas introduced in each bin in $Bins_R$ is determined by a user-defined percentage. Algorithm 2 describes this proposal. This method has the advantage of not discarding any information. However, it leads to a larger data set and the likelihood of overfitting is increased, specially for higher over-sampling percentages.

Additionally, in our implementation, we included two automatic methods of determining the amount of over-sampling that enable to balance the examples in the bins or to invert the proportion of examples in the bins. This overcomes the difficulty of asking the user to select a percentage of over-sampling. The

Algorithm 2: Random over-sampling algorithm (**RO**).

Input: $\mathcal{D}$ - data set with target continuous variable Y
t_R - threshold for relevance on Y values
$Bins$ - data partitions into relevant and normal bins
$\%o$ - percentage of over-sampling
Output: $NewD$ - a new modified data set
$NewD \leftarrow \mathcal{D}$
$Bins_R \leftarrow \{Bins_i \in Bins : \forall(\mathbf{x}, y) \in Bins_i, \phi(y) \geq t_R\}$
foreach $B \in Bins_R$ **do**
  $tgtNr \leftarrow \%o \times |B|$ // nr of replicas to add in
  $BselCases \leftarrow$ SAMPLE$(B, tgtNr)$ // add the replicas to the new data set
  $NewD \leftarrow NewD \bigcup selCases$
end
return $NewD$

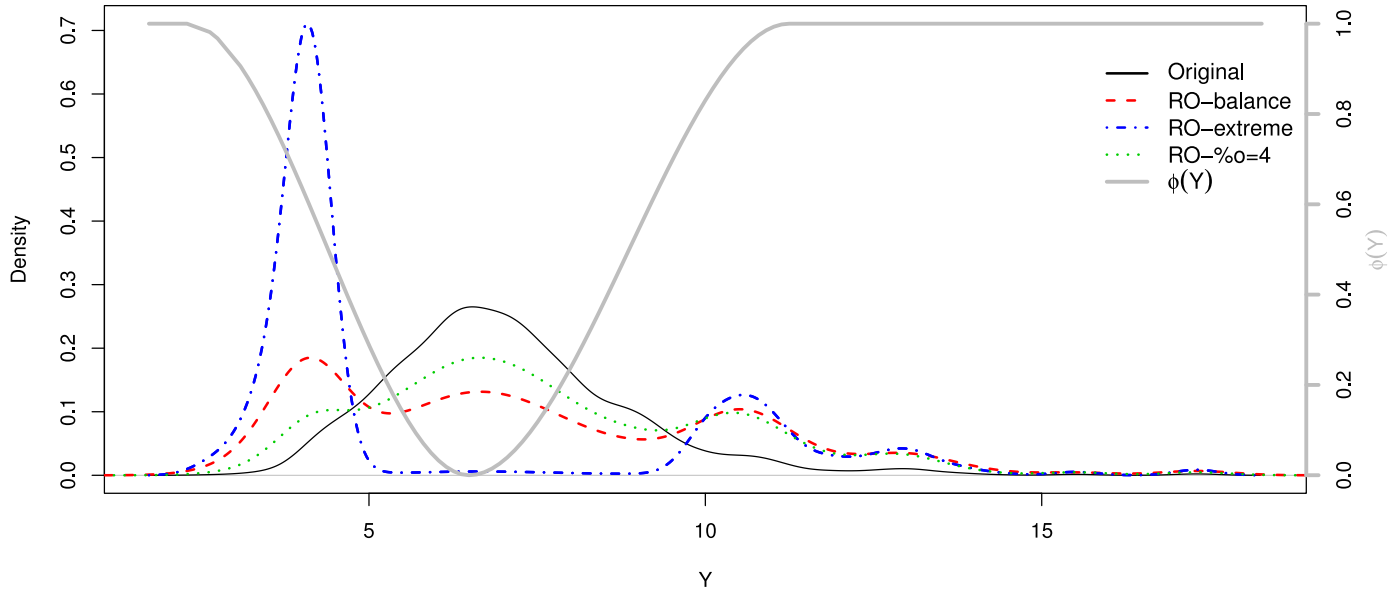


Fig. 3. Density of the target variable on the original data set and after applying random over-sampling algorithm on DS9 data set ($\phi(Y)$ automatically estimated).

option *balance* automatically determines the number of examples that must be included in each bin for them to be roughly balanced. The *extreme* option tries to transform the less represented ranges of the target variable into the most represented ones, and vice versa. As an example, consider a problem with two relevant bins, with 5 and 10 examples, and two normal bins containing 40 and 45 examples. The *balance* option changes the relevant bins frequencies to 43 $\left(\frac{\text{Sum of examples in } Bins_N}{\text{Nr. of } Bins_N} = \frac{85}{2} \approx 43\right)$ each. This means that the original frequencies 40, 45, 5 and 10 of the bins are changed to 40, 45, 43 and 43, respectively. The *extreme* option changes the bins with 5 and 10 examples to 361 and 181 examples each. These new frequencies are obtained using the following formula: $\frac{(\text{Average frequency in } Bins_N)^2}{\text{frequency of bin in } Bins_R}$. Thus, the new frequencies of the bins with 5 and 10 examples are calculated as follows: $\frac{42.5^2}{5} \approx 361$ and $\frac{42.5^2}{10} \approx 181$. Thus, in this case, the frequencies 40, 45, 5 and 10 of the bins are changed to 40, 45, 361 and 181, respectively.

Fig. 3 provides a brief overview of the impact on the density distribution of the target variable of DS9 (one of our data sets described in Table 1) for different parameter settings of the random over-sampling algorithm.

4.2. Introduction of Gaussian Noise

Our second proposal changes the data distribution using two main steps: (1) under-sampling the normal and less important cases; and (2) generating new cases with relevant target variable values. The generation of new cases uses an adaptation of the method proposed in [15,16] for classification tasks. This author described an over-sampling approach based on the generation of new cases of the minority class using the addition of normally distributed noise to existing rare cases. We use a similar strategy by over-sampling each bin of $Bins_R$. Namely, given a seed case we generate new cases by introducing a small perturbation on both the attributes and the target variable value of the seed case.

Our algorithm starts by determining the number of examples to generate for each case in a given bin with rare cases. Then, each new synthetic case is obtained as follows:

- for numeric attributes and target variable, add a perturbation based in $N(0, \delta \times sd(a))$, where δ is a parameter provided by the user controlling the amplitude of the perturbation, and $sd(a)$ is the standard deviation of attribute a estimated using the cases in the bin under consideration;
- for nominal attributes, sample from the existing values of the attribute with a probability proportional to the attribute values frequency in the considered bin.

Table 1
Data sets information for a threshold on relevance of 0.8 organized by descending order of the percentage of rare cases. (*N*: total nr. of cases; *tpred*: nr. of predictors; *p.nom*: nr of nominal predictors; *p.num*: nr. numeric predictors; *nRare*: nr. cases with $\phi(Y) > 0.8$; *%Rare: nRare*/$N \times 100$).

ID	Data Set	*N*	tpred	p.nom	p.num	nRare	% Rare
DS1	a6	198	11	3	8	33	16.7
DS2	Abalone	4177	8	1	7	679	16.3
DS3	a3	198	11	3	8	32	16.2
DS4	a4	198	11	3	8	31	15.7
DS5	a1	198	11	3	8	28	14.1
DS6	a7	198	11	3	8	27	13.6
DS7	boston	506	13	0	13	65	12.8
DS8	a2	198	11	3	8	22	11.1
DS9	fuelCons	1764	37	12	25	164	9.3
DS10	heat	7400	12	4	8	664	9.0
DS11	availPwr	1802	15	7	8	157	8.7
DS12	cpuSm	8192	12	0	12	713	8.7
DS13	maxTorque	1802	32	13	19	129	7.2
DS14	bank8FM	4499	8	0	8	288	6.4
DS15	Accel	1732	14	3	11	89	5.1

This algorithm deals with the introduction of perturbations on both nominal and numeric attributes. The generation of the target value guarantees that the synthetic cases have an high relevance. Algorithm 3 describes this method that we have named *Introduction of Gaussian Noise* (**GN**).

In our implementation of the **GN** strategy, we have also included the options *balance* and *extreme*. These options have a behavior similar to the one described for the **RO** algorithm, but ensuring that the data set size stays roughly the same. They facilitate the user's task by eliminating the need of setting the parameters for the percentages of over- and under-sampling. As an example, suppose a data set has two relevant bins with 5 and 10 examples, and two normal bins with 40 and 45 examples. The option *balance* changes all the bins frequencies to 25, i.e. to the average number of

Algorithm 3: Introduction of Gaussian Noise Algorithm (**GN**).

Input: $\mathcal{D}$ - data set with target continuous variable Y
t_R - threshold for relevance on Y values
Bins - data partitions into relevant and normal bins
$\%u$ - percentage of under-sampling
$\%o$ - percentage of over-sampling
δ - perturbation amplitude
Output: *newD* - a new modified data set
$Bins_N \leftarrow \{Bins_i \in Bins : \forall(\mathbf{x}, y) \in Bins_i, \phi(y) < t_R\}$
$Bins_R \leftarrow \{Bins_i \in Bins : \forall(\mathbf{x}, y) \in Bins_i, \phi(y) \geq t_R\}$
$newD \leftarrow Bins_R$
foreach $B \in Bins_N$ **do** `// random under-sampling procedure`
  $selNormCases \leftarrow$ randomly sample $\%u \times |B|$ elements from B
  $newD \leftarrow newD \bigcup selNormCases$
end
foreach $B \in Bins_R$ **do** `// over-sampling procedure`
  $ng \leftarrow \%o \times |B|$ `// nr of synthetic examples for each case in B`
  foreach $case \in B$ **do** `// generate synthetic examples`
    for $i \leftarrow 1$ ***to*** ng **do**
      foreach $a \in Attrs \bigcup Y$ **do**
        if a *is nominal* **then**
          $probs \leftarrow$ frequency of possible values of a
          $new[a] \leftarrow$ sample a value from the values of a with weights = $probs$
        else
          $new[a] \leftarrow$ multiply $case[a]$ with a random sample from $N(0, \delta \cdot sd(a))$
        end
      end
      $newD \leftarrow newD \bigcup \{new\}$ `// add synthetic case to newD`
    end
  end
end
return *newD*

examples considering the total number of bins. On the other hand, the *extreme* option changes the bins frequencies from 5, 10, 40 and 45 to 58, 29, 7 and 6, respectively. This is achieved in two steps. First, for each bin we use the formula: $\frac{(\text{Average nr of examples per bin})^2}{\text{frequency of the bin}}$. Then, as this provides a total number of examples different from the original data set, the obtained frequencies are rescaled (multiplied by the original data set size and divided by the sum of previously obtained frequencies). This means that in our example we would obtain in the first step 125, 63, 16 and 14 as the frequencies of bins with 5, 10, 40 and 45 examples. Then, these frequencies are multiplied by $\frac{100}{218}$.

Fig. 4 provides an overview of the impact on the density distribution of the target variable of fuelCons data set (DS9) for different parameters of the **GN** algorithm.

4.3. WERCS: WEighted Relevance-based Combination Strategy

Our third proposal, ***WE**ighted **R**elevance-based **C**ombination **S**trategy* (**WERCS**), tackles the problem of imbalanced regression through informed under- and over-sampling instead of random sampling. **WERCS** combines biased versions of under- and over-sampling strategies which are solely dependent on the relevance function provided for the data set. This method does not require the definition of bins of relevance nor the setting of a relevance threshold, using only the information of the relevance function. The key idea of **WERCS** is to use the relevance function and a modification of the defined relevance as weights for sampling examples to include and remove from the data set. Without considering a threshold on the relevance values, all examples are candidates for under-/over-sampling. Both strategies are applied in the entire domain. However, the probability of a case being selected for over- or under-sampling depends on its relevance. This process is accomplished as follows:

- a percentage of cases is randomly selected to be added as replicas considering weights proportional to the relevance function values, i.e., $w(\langle \mathbf{x}_i, y_i \rangle) = \phi(y_i)$; and
- a percentage of cases is randomly selected for being removed considering weights that are the complement of their relevance function values, i.e., $w(\langle \mathbf{x}_i, y_i \rangle) = 1 - \phi(y_i)$.

The new obtained data set will include some replicas of the most important examples, and will have some of the non-important examples removed. In both cases, examples are selected for these operations based on their relevance $\phi()$ (i.e., the importance to the user). This means that more important examples have higher probability of being replicated, whilst low relevance examples have higher probability of being removed. This facilitates the user task that only has to provide two percentages setting the level of under- and over-sampling to apply. Moreover, this new **WERCS** approach deals with the examples considering the intrinsic continuous nature of the target variable instead of partitioning the data into bins. Let us clarify this issue by considering an example. In a strategy using bins, a threshold of 0.85 would imply that an example A with 0.8 of relevance would be candidate for under-sampling while an example B with 0.9 of relevance would be candidate for over-sampling. In **WERCS**, the two examples have a high and similar probability of being selected for inclusion (0.8 for A and 0.9 for B). These examples could also be removed when under-sampling, although with a low and similar probability (0.2 for A and 0.1 for B). On the other hand, when using strategies that define bins, these two cases would be assigned to different bins: example A would belong to a Bin_N while example B would belong to a Bin_R. Therefore, example A would be treated as all other normal cases (such as normal cases with relevance 0), while example B would be treated as all relevant cases. **WERCS** prevents this counter-intuitive procedure as it uses the relevance values that are continuous. Using the relevance function for performing non-uniform sampling allows to perform an informed under-/over-sampling. Algorithm 4 implements the **WERCS** strategy. Fig. 5 shows the impact of choosing different parameters for the **WERCS** algorithm on the density distribution of the target variable of fuelCons data set (DS9).

Algorithm 4: WERCS Algorithm (**WERCS**).

Input: $\mathcal{D}$ - data set with target continuous variable Y
$\%u$ - percentage of under-sampling
$\%o$ - percentage of over-sampling
Output: *newD* - a new modified data set
$\phi() \leftarrow$ relevance function obtained from Y distribution
$newD \leftarrow \mathcal{D}$
$WOve \leftarrow \{\phi(y_i) \mid y_i \in Y\}$
$Ove \leftarrow$ sample $\%o \times |\mathcal{D}|$ cases from $\mathcal{D}$ with $WOve$ weights
`// over-sampling procedure`
$newD \leftarrow newD \bigcup Ove$
$WUnd \leftarrow \{1 - \phi(y_i) \mid y_i \in Y\}$
$Und \leftarrow$ sample $\%u \times |\mathcal{D}|$ cases from $\mathcal{D}$ with $WUnd$ weights
$newD \leftarrow newD \setminus Und$ `// under-sampling procedure`
return *newD*

5. Experimental evaluation

In this section we describe the experiments we have carried out and discuss the results. The main research questions that we try to answer are:

1. Are the proposed resampling strategies effective for dealing with the problem of imbalanced regression?
2. What is the impact of different data set characteristics in their performance?
3. How do variations in the relevance threshold impact the performance?

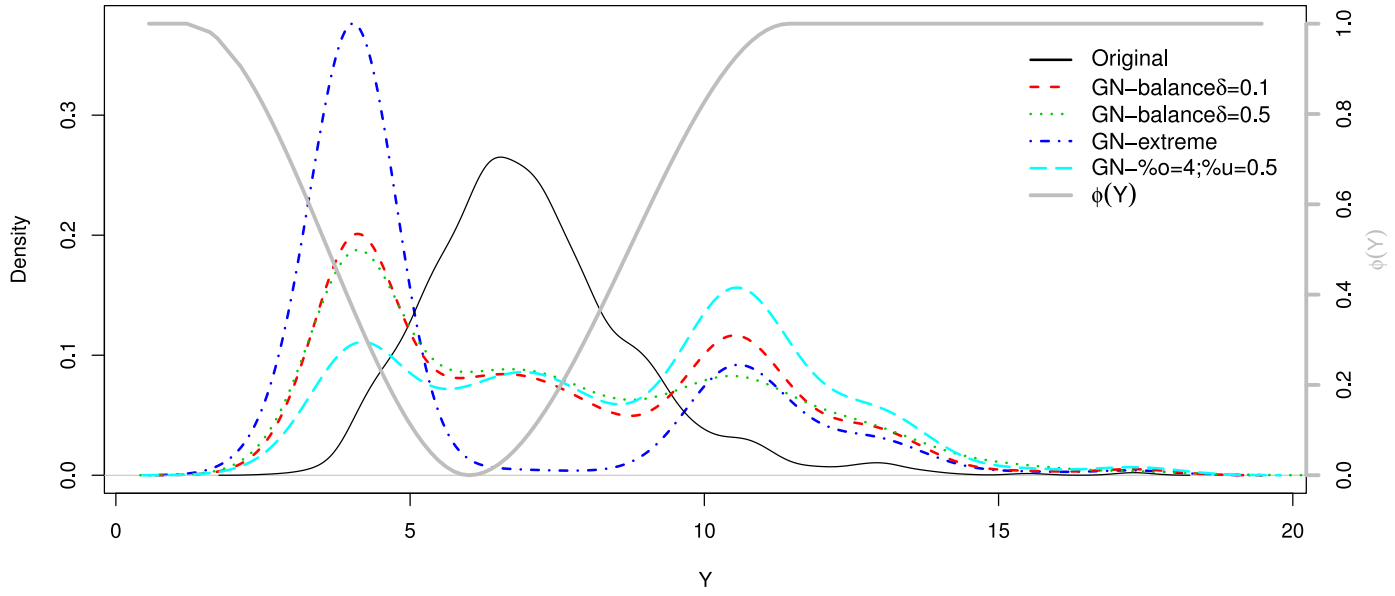


Fig. 4. Density of the target variable on the original data set and after applying Introduction of Gaussian Noise algorithm on DS9 data set ($\phi(Y)$ automatically estimated).

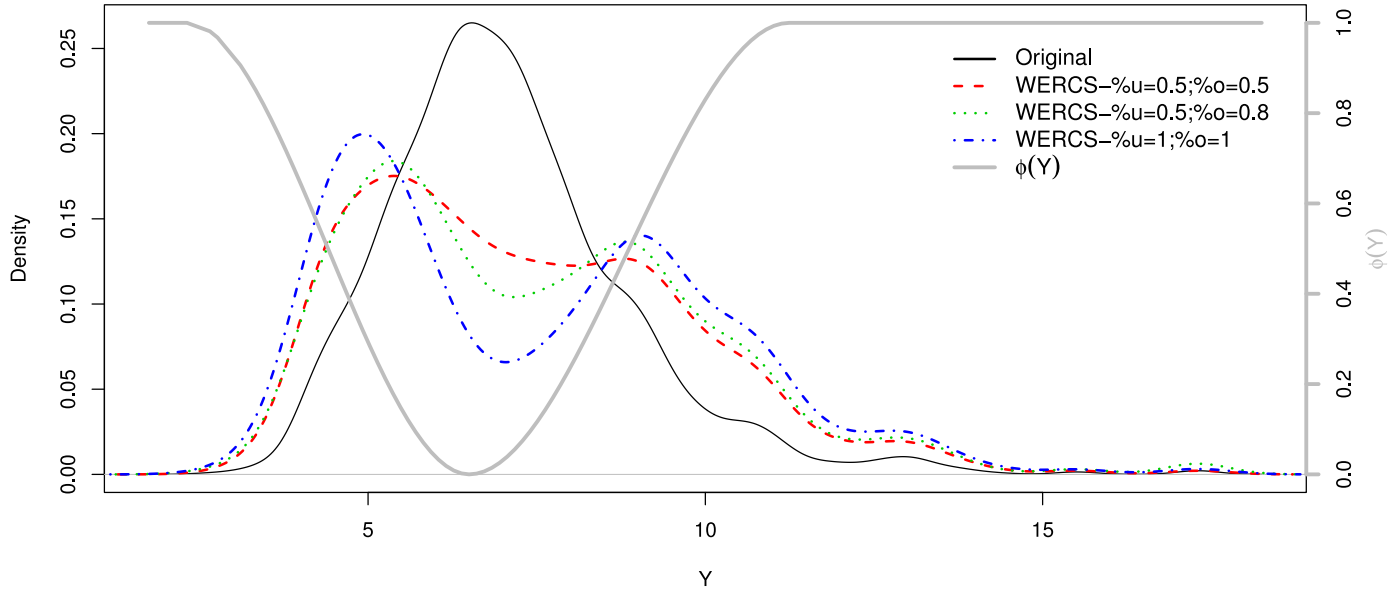


Fig. 5. Density of the target variable on the original data set and after applying **WERCS** with different parameters on DS9 data set ($\phi(Y)$ automatically estimated).

These questions are addressed in the following sections. In our experiments we used 15 regression data sets from different domains. The main characteristics of the data sets are described in Table 1. This table also displays the total number and percentage of rare cases in each data set for a relevance threshold of 0.8. To allow a more generalized use of these data sets by researchers interested in dealing with the problem of imbalanced regression we built a data repository. The 15 used regression data sets are available at https://paobranco.github.io/DataSets-IR/. This repository provides a small description of the data sets and allows the users to easily download all the data sets in three different formats: Rdata, arff and csv. The main goal of this repository is to provide a benchmark for imbalanced regression problems containing data sets with different characteristics and from different application domains.

For each of these data sets we have obtained a relevance function using the automatic method proposed by [22]. The results of this method are relevance functions that assign higher relevance to high and low rare extreme values.[2]

5.1. Evaluation of the proposed resampling approaches

The main goal of these experiments is to assess the effectiveness of the proposed strategies for dealing with the problem of imbalanced regression. We used the 15 data sets previously described, and compared the performance of the proposed strategies with the current state of the art methods for imbalanced regression: random under-sampling and SmoteR.

The experiments were carried out with 5 different types of learning algorithms: Linear Regression (LM), Neural Networks

[2] We used the R package uba that implements the automatic method for obtaining the relevance function, which is available at http://www.dcc.fc.up.pt/~rpribeiro/uba/

Table 2
Regression algorithms, parameter variants, and the respective R packages used.

Learner	Parameter variants	R package
LM		`stats` [20]
NNET	$size = \{1, 2, 5, 10\}$, $decay = \{0, 0.01\}$	`nnet` [28]
MARS	$nk = \{10, 17\}$, $degree = \{1, 2\}$, $thresh = \{0.01\}$	`earth` [19]
SVR	$cost = \{10, 150, 300\}$, $gamma = \{0.01, 0.001\}$, $epsilon = \{0.1, 0.01\}$	`e1071` [9]
RF	$mtry = \{5, 7\}$, $ntree = \{500, 750, 1500\}$	`randomForest` [17]

Table 3
Resampling strategies and respective parameters used.

Strategy		Parameter variants
None	No sampling applied	
RU	Random under-sampling	$\%u = \{balance, extreme\}$
SMT	SmoteR algorithm	$\%u/\%o = \{balance, extreme\}$
RO	Random over-sampling	%o={*balance, extreme*}
GN	Introduction of Gaussian Noise	$\%u/\%o = \{balance, extreme\}$, $\delta = \{0.05, 0.1, 0.5\}$
WERCS	WEighted Relevance-based Combination Strategy	$\%u = \{0.5, 0.8\}$, $\%o = \{0.5, 0.8\}$

(NNET), Multivariate Adaptive Regression Splines (MARS), Support Vector Regression Machines (SVR) and Random Forests (RF). Table 2 summarizes the used learning algorithms and the respective parameter variants. All experiments were carried out in the R environment, thus we include the used R packages.

We compared the following pre-processing approaches: (i) carrying out no sampling at all, i.e. use the original imbalanced data set (**None**); (ii) 2 variants of random under-sampling (**RU**); (iii) 2 variants of the SmoteRmethod (**SMT**); (iv) 2 variants of random over-sampling (**RO**); (v) 6 variants of introduction of Gaussian Noise (**GN**); and (vi) 4 variants of WERCS algorithm (**WERCS**). Table 3 describes the 17 resampling variants tested. All pre-processing approaches were implemented and are available through the R package UBL [3].

Each of the 31 learning approaches (1 LM + 8 NNET variants + 4 MARS variants + 12 SVR variants + 6 RF variants), were applied to each of the 15 regression problems using 17 different pre-processing approaches. Thus, 7905 ($31 \times 15 \times 17$) combinations were tested. The performance was evaluated according to the F_1^ϕ measure, the adaptation of F-measure for regression [2,22] and the $G-Mean^\phi$, both described in Section 2.1. We use $\beta = 1$ in the F_1^ϕ measure that assigns the same importance to both precision ($prec^\phi$) and recall (rec^ϕ). The values of F_1^ϕ, $G-Mean^\phi$ and the remaining evaluated metrics were estimated by means of a 2×10 - fold cross validation process. The experiments were carried out using the experimental infra-structure provided by the R package `performanceEstimation` [23].

Table 4 shows the F_1^ϕ mean and standard deviation results by data set, learning algorithm and sampling strategy. In bold we have the best combination of algorithm and sampling variant for each data set. The blue highlighted cells in the table represent the best results by data set. The results presented in Table 4 are rounded. Therefore, apparent ties may occur in the table, which in reality are not. This table shows that our proposed pre-processing methods achieve a better performance in the majority of cases. This happens when we compare our proposals both with the baseline of using the original imbalanced data set (**None**) and also when compared against the use of SmoteR (**SMT**) or random under-sampling (**RU**) strategies. Moreover, we also observe that the combination of random forest (RF) learners with either **RO, GN** or **WERCS** provides the best performance in 9 out of 15 data sets.

Table 5 summarizes the results of Table 4, displaying the total number of data sets where each pair of sampling strategy-learner had the best F_1^ϕ results. These results confirm that the proposed strategies present advantages in terms of performance, and **WERCS** stands out in this setting.

Table 6 shows the total number of wins/losses and significant wins/losses of each pre-processing strategy against the baseline of using the training set unchanged. The results were obtained with the Wilcoxon Signed Rank test for each data set considering a significance threshold of 95%. From this table we observe that **WERCS** algorithm has a clear advantage when compared to the use of the unchanged data sets.

Fig. 6 shows the total number of wins and losses by sampling strategy aggregated across the several learners. Although the other strategies also present advantages, the performance of **WERCS** is overwhelming.

The main conclusions to be drawn from the F_1^ϕ results are as follows: (i) generally, all the pre-processing strategies improve the performance of the tested learners; (ii) the random forest learner used in conjunction with one of the strategies proposed in this paper provides the best results in 9 out of 15 data sets; (iii) WERCS obtains the larger number of statistically significant wins against the baseline in comparison with the remaining tested strategies.

The F_1^ϕ measure provides an aggregated overview of the performance. High values of F_1^ϕ are achieved when both $prec^\phi$ and rec^ϕ values are high. Still, the observed improvement may be obtained at the cost of a penalization of the performance of $prec^\phi$ or rec^ϕ. To provide a more detailed analysis of the performance, we also show the results of $prec^\phi$ and rec^ϕ metrics. Tables 7 and 8 display the mean results of $prec^\phi$ and rec^ϕ measures achieved by data set, learner and pre-processing strategy for the best parametrization. Again, we must remark that the results presented in these tables are rounded. Therefore, different results may be rounded into the same number.

The results of Tables 7 and 8 are aggregated in Table 9 and 10, respectively. As expected, we observe a loss in the $prec^\phi$ performance achieved with the pre-processing strategies when compared to the use of the original data set. On the other hand, the rec^ϕ results show an improvement when resampling strategies are applied. This means that, the improvement verified on the F_1^ϕ scores is achieved by a loss in the $prec^\phi$ results and a gain in the rec^ϕ results. The balance provided by the F_1^ϕ shows that the loss verified on $prec^\phi$ is compensated by the gain on rec^ϕ.

This was, in fact, the expected scenario: we are able to improve the global performance measured through the F_1^ϕ by risking more on the prediction of relevant cases. This means that the pre-processed models are signaling more relevant cases in comparison to the models generated by the original data set. The increase of relevant predictions, increases the rec^ϕ score because more important cases are being detected. However, this has also a negative impact on the "False Positive", i.e., there are also more normal

Table 4
Mean and standard deviation of F_1^{ϕ} by data set for the best parameters combination of algorithm and sampling strategy applied. Results in bold highlight the best performance by data set and learner while the blue highlighted cells signals the best result by data set.

Learn	Strat	DS1	DS2	DS3	DS4	DS5	DS6	DS7	DS8	DS9	DS10	DS11	DS12	DS13	DS14	DS15
LM		0.489(0.313)	0.699(0.017)	0.350(0.307)	0.489(0.275)	0.123(0.256)	0.315(0.224)	0.817(0.031)	0.196(0.365)	**0.847(0.138)**	0.847(0.006)	**0.910(0.040)**	0.087(0.021)	**0.821(0.296)**	0.939(0.007)	**0.821(0.221)**
	RU	0.565(0.093)	0.684(0.019)	0.481(0.175)	0.524(0.116)	0.663(0.101)	0.304(0.215)	0.839(0.036)	0.272(0.280)	0.062(0.057)	0.909(0.005)	0.759(0.062)	0.126(0.024)	0.075(0.063)	0.954(0.006)	0.153(0.091)
	SMT	0.585(0.090)	0.713(0.019)	**0.491(0.174)**	0.569(0.105)	0.703(0.120)	0.314(0.220)	0.838(0.032)	**0.311(0.324)**	0.359(0.360)	0.913(0.007)	0.836(0.084)	**0.135(0.025)**	0.296(0.348)	0.952(0.005)	0.384(0.334)
	RO	0.590(0.088)	**0.719(0.019)**	0.475(0.168)	0.590(0.108)	**0.708(0.097)**	0.308(0.214)	0.837(0.029)	0.296(0.306)	0.833(0.193)	0.917(0.004)	0.909(0.053)	0.128(0.026)	0.815(0.310)	**0.955(0.005)**	0.815(0.218)
	GN	0.596(0.078)	0.712(0.019)	0.486(0.176)	0.598(0.119)	0.698(0.107)	**0.322(0.225)**	0.839(0.029)	0.309(0.320)	0.348(0.339)	0.908(0.004)	0.821(0.077)	0.130(0.027)	0.340(0.432)	**0.955(0.005)**	0.392(0.328)
	WERCS	**0.598(0.095)**	0.718(0.019)	0.486(0.177)	**0.609(0.090)**	0.674(0.194)	0.310(0.216)	**0.840(0.030)**	0.306(0.322)	0.647(0.285)	**0.920(0.004)**	0.886(0.068)	0.117(0.017)	0.590(0.356)	0.945(0.005)	0.556(0.361)
MARS		0.548(0.217)	0.715(0.021)	0.510(0.336)	0.453(0.172)	0.526(0.347)	**0.334(0.258)**	**0.901(0.038)**	0.193(0.354)	0.878(0.027)	0.934(0.006)	**0.914(0.020)**	0.130(0.034)	0.968(0.006)	0.943(0.006)	**0.912(0.016)**
	RU	0.582(0.067)	0.740(0.013)	0.488(0.175)	0.574(0.129)	0.737(0.132)	0.304(0.240)	0.868(0.037)	0.289(0.301)	0.869(0.026)	0.955(0.009)	0.908(0.022)	0.334(0.022)	**0.995(0.011)**	0.951(0.006)	0.895(0.034)
	SMT	0.589(0.121)	0.734(0.015)	0.498(0.178)	0.624(0.073)	0.757(0.122)	0.287(0.225)	0.888(0.037)	0.290(0.304)	0.864(0.022)	0.958(0.004)	0.908(0.020)	0.301(0.092)	0.981(0.021)	0.950(0.007)	0.892(0.023)
	RO	0.584(0.092)	**0.745(0.017)**	0.511(0.186)	**0.652(0.091)**	0.682(0.185)	0.280(0.262)	0.883(0.044)	0.261(0.271)	0.875(0.026)	**0.959(0.004)**	0.905(0.021)	**0.367(0.013)**	0.957(0.007)	0.952(0.006)	0.880(0.018)
	GN	**0.614(0.097)**	0.734(0.015)	**0.515(0.185)**	0.640(0.075)	**0.766(0.110)**	0.307(0.246)	0.885(0.040)	**0.306(0.321)**	0.863(0.028)	0.949(0.004)	0.908(0.019)	0.359(0.026)	**0.995(0.011)**	0.952(0.006)	0.891(0.019)
	WERCS	0.584(0.083)	0.743(0.015)	0.510(0.336)	0.632(0.117)	0.704(0.136)	0.315(0.246)	0.900(0.040)	0.285(0.300)	**0.889(0.028)**	0.954(0.004)	**0.914(0.020)**	0.144(0.027)	0.990(0.013)	**0.954(0.007)**	**0.912(0.024)**
NNET		0.541(0.266)	0.732(0.015)	0.486(0.288)	0.575(0.171)	0.340(0.372)	**0.342(0.254)**	0.821(0.033)	0.199(0.370)	0.730(0.326)	0.854(0.012)	0.922(0.020)	0.110(0.034)	0.954(0.017)	0.954(0.008)	0.893(0.022)
	RU	0.561(0.092)	0.739(0.016)	0.469(0.178)	0.529(0.122)	0.682(0.122)	0.299(0.214)	0.843(0.034)	0.270(0.279)	0.802(0.080)	0.913(0.006)	0.908(0.023)	**0.167(0.028)**	0.912(0.058)	0.952(0.007)	0.864(0.052)
	SMT	0.582(0.095)	0.740(0.015)	0.496(0.178)	0.573(0.091)	0.715(0.129)	0.319(0.223)	0.843(0.034)	0.319(0.338)	0.757(0.162)	0.917(0.006)	0.922(0.016)	0.162(0.024)	0.969(0.014)	0.952(0.008)	0.892(0.017)
	RO	0.589(0.094)	**0.745(0.013)**	0.484(0.181)	0.592(0.100)	0.708(0.104)	0.311(0.215)	0.844(0.040)	0.296(0.307)	0.734(0.219)	0.918(0.005)	0.932(0.016)	0.145(0.037)	0.972(0.010)	0.952(0.006)	0.892(0.021)
	GN	0.603(0.070)	0.744(0.017)	**0.516(0.201)**	0.604(0.110)	**0.716(0.124)**	0.320(0.223)	0.844(0.031)	**0.320(0.330)**	0.806(0.212)	0.910(0.007)	0.903(0.021)	0.165(0.040)	0.961(0.020)	0.953(0.007)	0.892(0.023)
	WERCS	**0.605(0.097)**	**0.745(0.017)**	0.510(0.183)	**0.610(0.092)**	0.707(0.129)	0.303(0.216)	**0.850(0.034)**	0.317(0.334)	**0.893(0.030)**	**0.925(0.009)**	**0.936(0.019)**	0.147(0.041)	**0.975(0.012)**	**0.955(0.006)**	**0.916(0.021)**
SVR		0.126(0.310)	0.728(0.016)	0.051(0.157)	0.329(0.347)	0.000(0.000)	0.022(0.097)	0.873(0.052)	0.099(0.306)	0.909(0.023)	0.000(0.000)	0.935(0.016)	0.159(0.027)	0.966(0.009)	0.947(0.007)	0.913(0.016)
	RU	**0.608(0.079)**	0.694(0.016)	**0.561(0.213)**	0.634(0.073)	**0.763(0.096)**	**0.338(0.238)**	0.881(0.041)	**0.321(0.331)**	0.849(0.040)	0.839(0.009)	0.924(0.018)	0.201(0.028)	0.959(0.011)	0.945(0.006)	0.916(0.018)
	SMT	0.569(0.065)	0.748(0.013)	0.506(0.199)	0.620(0.057)	0.742(0.096)	0.287(0.208)	0.894(0.036)	0.300(0.313)	0.910(0.017)	0.904(0.009)	0.940(0.011)	0.220(0.033)	0.976(0.006)	0.946(0.006)	0.893(0.019)
	RO	0.569(0.106)	**0.749(0.016)**	0.470(0.173)	0.608(0.086)	0.730(0.090)	0.287(0.217)	**0.909(0.034)**	0.293(0.306)	0.919(0.020)	**0.991(0.003)**	0.947(0.010)	0.211(0.032)	0.977(0.008)	0.945(0.006)	0.898(0.019)
	GN	0.586(0.059)	**0.749(0.014)**	0.513(0.200)	**0.640(0.075)**	0.758(0.104)	0.295(0.221)	0.900(0.035)	0.320(0.335)	0.899(0.021)	0.864(0.005)	0.928(0.016)	**0.274(0.040)**	**0.980(0.007)**	0.945(0.005)	0.907(0.019)
	WERCS	0.607(0.104)	0.748(0.018)	0.513(0.184)	0.629(0.071)	0.669(0.246)	0.278(0.216)	0.901(0.039)	0.291(0.303)	**0.920(0.022)**	0.908(0.008)	**0.951(0.010)**	0.202(0.042)	0.978(0.005)	**0.949(0.007)**	**0.919(0.020)**
RF		0.547(0.272)	0.726(0.016)	0.308(0.297)	0.488(0.268)	0.406(0.390)	0.338(0.310)	0.893(0.035)	0.119(0.296)	0.935(0.016)	0.905(0.008)	0.972(0.007)	0.510(0.035)	0.978(0.008)	0.906(0.007)	0.943(0.014)
	RU	0.614(0.073)	0.688(0.015)	0.515(0.181)	0.623(0.069)	**0.779(0.096)**	0.317(0.242)	0.883(0.029)	0.313(0.324)	0.852(0.023)	0.932(0.007)	0.958(0.015)	0.477(0.034)	0.956(0.012)	0.928(0.006)	0.907(0.020)
	SMT	0.595(0.090)	0.734(0.015)	0.543(0.194)	0.625(0.064)	0.706(0.087)	0.340(0.254)	0.898(0.031)	**0.315(0.330)**	0.930(0.012)	0.935(0.007)	0.975(0.009)	0.476(0.030)	0.977(0.009)	0.929(0.008)	0.941(0.018)
	RO	0.576(0.218)	0.731(0.017)	**0.566(0.226)**	0.556(0.224)	0.686(0.277)	0.375(0.282)	0.907(0.036)	0.201(0.328)	**0.948(0.012)**	0.934(0.007)	**0.984(0.009)**	0.505(0.040)	**0.987(0.009)**	0.916(0.008)	**0.956(0.019)**
	GN	0.606(0.104)	**0.738(0.015)**	0.519(0.189)	**0.658(0.079)**	0.760(0.111)	0.316(0.265)	0.907(0.036)	0.307(0.325)	0.928(0.018)	**0.966(0.003)**	0.978(0.008)	0.481(0.023)	0.981(0.009)	**0.937(0.008)**	0.952(0.015)
	WERCS	**0.624(0.092)**	0.737(0.015)	**0.566(0.208)**	0.636(0.105)	0.630(0.293)	**0.378(0.267)**	**0.911(0.038)**	0.249(0.325)	0.941(0.015)	0.935(0.007)	**0.984(0.010)**	**0.511(0.036)**	0.986(0.009)	0.923(0.008)	**0.956(0.018)**

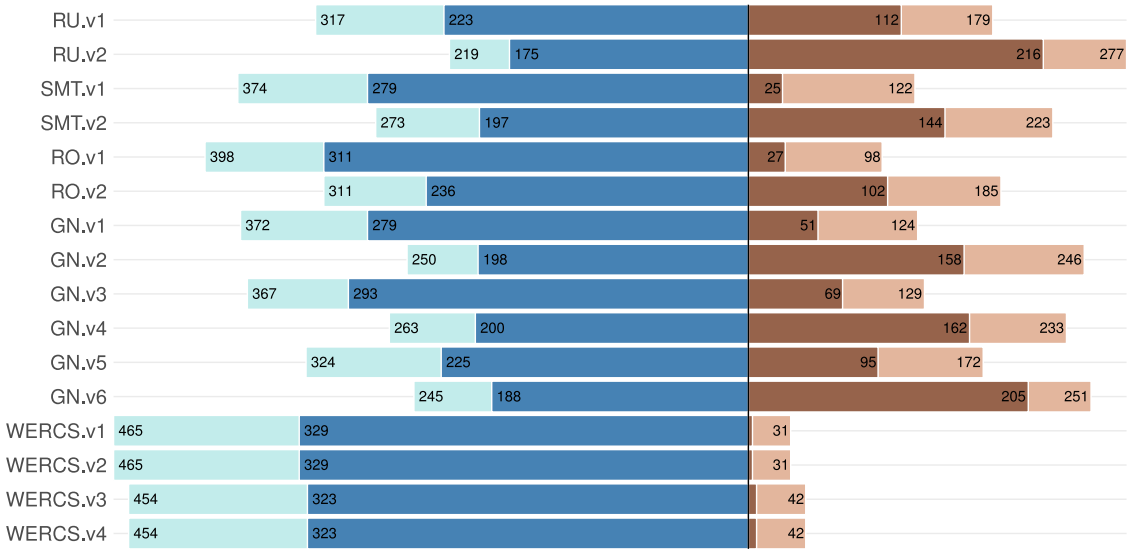


Fig. 6. Total number of F_1^{ϕ} wins/losses of sampling strategies, against using the original imbalanced data set, of all learners obtained with the Wilcoxon Signed Rank test at 95% confidence. (Left side: wins; right side: losses; darker bars: significant wins/losses.)

Table 5
Aggregation of results from Table 4. Total number of data sets where each pair of sampling strategy-learner had the best F_1^{ϕ} results.

	None	RU	SMT	RO	GN	WERCS
LM	4	0	3	3	2	4
MARS	4	1	0	4	5	4
NNET	1	1	0	1	3	10
SVR	0	5	0	3	4	4
RF	0	1	1	5	4	7
Total	9	8	4	16	18	29

and uninteresting cases that are being predicted as rare. This has a negative impact on the $prec^{\phi}$ results that tend to decrease. Nevertheless, the loss verified on the $prec^{\phi}$ results is rewarded by gains on the rec^{ϕ} results. Overall, the good performance observed on the F_1^{ϕ} results shows that this trade-off is beneficial.

We also applied the non-parametric Friedman *F*-test for assessing the statistical significance of the F_1^{ϕ} results. The *F*-test results allowed the rejection of the null hypothesis that all the tested approaches exhibit the same performance. We then proceeded with the post-hoc Nemenyi test with a significance level of 95% to verify which approaches are statistically different. Fig. 7 a–e shows the obtained critical difference diagrams (CD diagrams) proposed by Demšar [8] with the results aggregated by learner. In CD diagrams, the better is the performance of an algorithm the lower is its rank, and when two algorithms are connected by a bold horizontal line it means that the difference between their average ranks is not statistically significant. We observe that **WERCS** always displays the lower rank for all tested learners. However, the statistical significance of the results depend on the used learner. Globally, we observe that the worst performing strategies are **None** and **RU**.

To check the performance achieved in the least important cases we also evaluated the results of the $G-Mean^{\phi}$ and $spec^{\phi}$ measures that were described in Section 2.1. These results are displayed in Tables 11 and 12. The $G-Mean^{\phi}$ results show that the proposed

Table 6
F_1^{ϕ} wins/losses (significant wins/losses) of sampling strategies, against the baseline of using the original imbalanced data. Results obtained with the Wilcoxon Signed Rank test at a significant level of 95% (MWin: maximum nr of wins possible by learner).

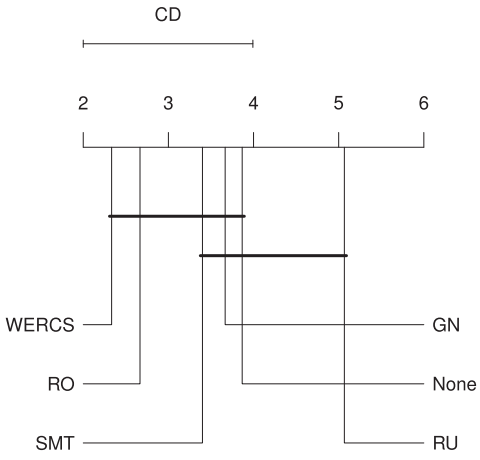
Sampl Strat	LM MWin= 16		NNET MWin= 128		MARS MWin= 64		SVR MWin= 192		RF MWin= 96	
	Win	Loss	Win	Loss	Win	Loss	Win	Loss	Win	Loss
RU.v1	9(6)	7(5)	73(48)	55(24)	49(25)	15(5)	132(108)	60(36)	54(36)	42(42)
RU.v2	6(4)	10(6)	49(34)	79(54)	32(17)	32(18)	108(108)	84(84)	24(12)	72(54)
SMT.v1	12(7)	4(4)	90(51)	38(1)	44(29)	20(2)	156(144)	36(12)	72(48)	24(6)
SMT.v2	7(5)	9(5)	52(39)	76(36)	34(21)	30(19)	132(108)	60(48)	48(24)	48(36)
RO.v1	**13(7)**	**3(0)**	96(56)	32(2)	43(26)	21(7)	168(168)	24(12)	78(54)	18(6)
RO.v2	7(5)	9(4)	54(38)	74(32)	34(19)	30(18)	144(132)	48(48)	72(42)	24(0)
GN.v1	12(7)	4(4)	85(50)	43(13)	47(24)	17(4)	144(144)	48(24)	84(54)	12(6)
GN.v2	7(5)	9(6)	53(35)	75(38)	34(20)	30(18)	108(108)	84(60)	48(30)	48(36)
GN.v3	12(7)	4(4)	83(45)	45(11)	44(25)	20(6)	144(144)	48(36)	84(72)	12(12)
GN.v4	7(5)	9(6)	55(37)	73(40)	33(21)	31(20)	120(108)	72(60)	48(30)	48(36)
GN.v5	12(6)	4(4)	77(44)	51(22)	37(25)	27(15)	132(108)	60(36)	66(42)	30(18)
GN.v6	7(3)	9(5)	52(28)	76(42)	30(19)	34(26)	108(108)	84(84)	48(30)	48(48)
WERCS.v1	12(7)	4(3)	**109(71)**	**19(0)**	**56(23)**	**8(0)**	**192(168)**	**0(0)**	**96(60)**	**0(0)**
WERCS.v2	12(7)	4(3)	**109(71)**	**19(0)**	**56(23)**	**8(0)**	**192(168)**	**0(0)**	**96(60)**	**0(0)**
WERCS.v3	12(7)	4(3)	107(73)	21(3)	53(25)	11(0)	**192(168)**	**0(0)**	90(54)	6(0)
WERCS.v4	12(7)	4(3)	107(73)	21(3)	53(25)	11(0)	**192(168)**	**0(0)**	90(54)	6(0)

Table 7
Mean and standard deviation of $prec^{\phi}$ by data set for the best parameters combination of algorithm and sampling strategy applied. Results in bold highlight the best performance by data set and learner while the blue highlighted cells signals the best result by data set.

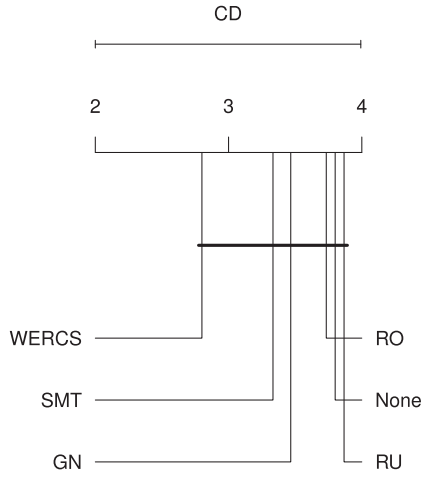
Learn	Strat	DS1	DS2	DS3	DS4	DS5	DS6	DS7	DS8	DS9	DS10	DS11	DS12	DS13	DS14	DS15
LM		0.458(0.306)	**0.724(0.024)**	0.341(0.257)	0.435(0.264)	0.118(0.255)	**0.392(0.086)**	**0.827(0.042)**	0.291(0.372)	**0.837(0.164)**	0.849(0.009)	**0.899(0.046)**	**0.088(0.022)**	**0.826(0.247)**	0.947(0.008)	**0.844(0.218)**
	RU	0.461(0.058)	0.654(0.024)	0.404(0.040)	0.461(0.066)	0.550(0.098)	0.316(0.036)	0.796(0.046)	0.423(0.078)	0.185(0.009)	0.876(0.008)	0.733(0.096)	0.204(0.005)	0.213(0.004)	0.957(0.005)	0.214(0.051)
	SMT	0.484(0.060)	0.667(0.020)	**0.407(0.039)**	0.485(0.087)	**0.604(0.146)**	0.326(0.034)	0.802(0.042)	0.483(0.082)	0.401(0.273)	0.879(0.007)	0.836(0.068)	0.208(0.005)	0.381(0.263)	0.955(0.005)	0.400(0.283)
	RO	0.487(0.058)	0.690(0.021)	0.396(0.033)	0.508(0.097)	0.590(0.102)	0.321(0.039)	0.795(0.037)	0.428(0.064)	0.808(0.167)	0.880(0.007)	0.893(0.051)	0.205(0.005)	0.817(0.265)	0.957(0.005)	0.778(0.195)
	GN	**0.495(0.046)**	0.666(0.019)	0.406(0.047)	**0.525(0.094)**	0.586(0.124)	0.336(0.043)	0.802(0.037)	**0.485(0.148)**	0.394(0.257)	0.878(0.007)	0.850(0.058)	0.205(0.005)	0.449(0.332)	**0.959(0.004)**	0.398(0.280)
	WERCS	0.487(0.061)	0.699(0.022)	0.401(0.044)	0.523(0.082)	0.574(0.194)	0.329(0.034)	0.810(0.044)	0.477(0.119)	0.629(0.271)	**0.905(0.007)**	0.87(0.058)	0.169(0.010)	0.599(0.294)	0.954(0.006)	0.552(0.342)
MARS		0.493(0.212)	**0.742(0.03)**	**0.474(0.305)**	0.383(0.16)	0.469(0.326)	0.358(0.189)	**0.902(0.043)**	0.281(0.401)	**0.881(0.042)**	0.933(0.006)	**0.914(0.023)**	0.130(0.036)	0.968(0.015)	0.95(0.006)	**0.912(0.026)**
	RU	0.477(0.067)	0.699(0.016)	0.413(0.051)	0.484(0.108)	0.645(0.155)	0.349(0.069)	0.840(0.056)	0.433(0.061)	0.832(0.037)	0.945(0.015)	0.896(0.029)	0.276(0.012)	**0.992(0.02)**	0.954(0.006)	0.853(0.058)
	SMT	0.495(0.11)	0.678(0.013)	0.421(0.055)	0.542(0.059)	0.660(0.15)	0.338(0.052)	0.866(0.053)	0.486(0.069)	0.824(0.033)	0.950(0.006)	0.892(0.028)	0.259(0.066)	0.977(0.029)	0.953(0.007)	0.853(0.037)
	RO	0.489(0.073)	0.710(0.016)	0.432(0.045)	**0.586(0.077)**	0.572(0.170)	0.349(0.056)	0.861(0.059)	0.431(0.106)	0.841(0.037)	**0.951(0.006)**	0.891(0.028)	0.294(0.009)	0.955(0.013)	0.953(0.007)	0.833(0.031)
	GN	**0.514(0.081)**	0.683(0.018)	0.432(0.062)	0.563(0.146)	**0.666(0.134)**	0.364(0.07)	0.863(0.058)	**0.508(0.070)**	0.826(0.042)	0.939(0.007)	0.893(0.024)	**0.308(0.014)**	0.991(0.021)	0.954(0.007)	0.851(0.036)
	WERCS	0.485(0.062)	**0.742(0.03)**	0.442(0.314)	0.558(0.141)	0.596(0.153)	**0.368(0.049)**	**0.902(0.043)**	0.464(0.082)	0.874(0.047)	0.948(0.005)	**0.914(0.023)**	0.147(0.027)	0.987(0.021)	**0.958(0.009)**	0.89(0.05)
NNET		**0.506(0.264)**	**0.745(0.021)**	**0.446(0.23)**	0.512(0.239)	0.310(0.372)	**0.406(0.153)**	**0.831(0.044)**	0.300(0.351)	0.748(0.286)	0.857(0.014)	0.913(0.027)	0.129(0.042)	0.948(0.025)	**0.956(0.007)**	**0.914(0.034)**
	RU	0.459(0.059)	0.693(0.018)	0.402(0.053)	0.464(0.083)	0.571(0.129)	0.315(0.041)	0.806(0.045)	0.425(0.065)	0.735(0.176)	0.882(0.008)	0.873(0.037)	0.211(0.006)	0.854(0.097)	0.951(0.01)	0.809(0.078)
	SMT	0.486(0.068)	0.680(0.019)	0.414(0.036)	0.489(0.08)	**0.624(0.149)**	0.332(0.035)	0.810(0.046)	0.486(0.083)	0.703(0.265)	0.888(0.014)	0.905(0.022)	0.215(0.006)	0.959(0.024)	0.948(0.009)	0.854(0.030)
	RO	0.488(0.06)	0.701(0.019)	0.403(0.055)	0.505(0.089)	0.591(0.12)	0.325(0.033)	0.806(0.056)	0.443(0.11)	0.7(0.276)	0.883(0.011)	0.917(0.022)	0.216(0.011)	0.96(0.017)	0.947(0.011)	0.855(0.038)
	GN	0.501(0.057)	0.706(0.02)	0.443(0.109)	**0.525(0.092)**	0.615(0.159)	0.338(0.04)	0.810(0.041)	**0.518(0.078)**	0.775(0.191)	0.884(0.013)	0.883(0.031)	**0.218(0.008)**	0.952(0.031)	0.954(0.007)	0.844(0.038)
	WERCS	0.499(0.07)	0.715(0.023)	0.422(0.041)	0.523(0.08)	0.605(0.145)	0.329(0.041)	0.819(0.045)	0.484(0.103)	**0.878(0.041)**	**0.910(0.011)**	**0.926(0.026)**	0.157(0.037)	**0.968(0.023)**	0.955(0.007)	0.906(0.031)
SVR		0.128(0.315)	**0.781(0.021)**	0.040(0.124)	0.356(0.385)	0.000 (0.000)	0.052(0.159)	0.89(0.048)	0.139(0.343)	0.919(0.023)	0.000 (0.000)	0.934(0.018)	0.161(0.027)	0.966(0.01)	**0.948(0.008)**	**0.925(0.027)**
	RU	0.513(0.074)	0.669(0.019)	0.499(0.082)	0.546(0.078)	**0.661(0.132)**	0.384(0.042)	0.861(0.055)	**0.512(0.073)**	0.833(0.055)	0.854(0.012)	0.905(0.026)	0.198(0.02)	0.947(0.02)	0.941(0.007)	0.893(0.036)
	SMT	0.484(0.144)	0.696(0.015)	0.437(0.069)	0.540(0.087)	0.640(0.125)	0.348(0.04)	0.878(0.045)	0.504(0.096)	0.891(0.021)	0.912(0.008)	0.932(0.016)	0.209(0.028)	0.973(0.007)	0.943(0.007)	0.853(0.03)
	RO	0.488(0.089)	0.718(0.017)	0.408(0.046)	0.530(0.088)	0.621(0.116)	0.342(0.060)	**0.898(0.043)**	0.469(0.072)	0.909(0.027)	**0.990(0.004)**	0.940(0.015)	0.205(0.029)	0.976(0.01)	0.939(0.007)	0.860(0.034)
	GN	0.474(0.142)	0.718(0.016)	**0.450(0.077)**	**0.570(0.091)**	0.656(0.140)	**0.385(0.070)**	0.882(0.047)	0.495(0.142)	0.874(0.027)	0.878(0.006)	0.921(0.019)	**0.255(0.035)**	**0.977(0.008)**	0.939(0.007)	0.882(0.033)
	WERCS	**0.514(0.105)**	0.739(0.021)	0.438(0.045)	0.557(0.077)	0.575(0.230)	0.348(0.074)	0.887(0.051)	0.449(0.186)	**0.920(0.026)**	0.916(0.008)	**0.950(0.012)**	0.203(0.042)	0.976(0.005)	0.947(0.008)	0.913(0.034)
RF		0.495(0.260)	**0.752(0.021)**	0.273(0.245)	0.444(0.255)	0.354(0.348)	0.393(0.222)	0.895(0.033)	0.196(0.358)	0.942(0.016)	0.913(0.007)	0.971(0.010)	**0.529(0.040)**	0.979(0.010)	0.929(0.007)	0.946(0.023)
	RU	0.493(0.069)	0.644(0.018)	0.419(0.036)	0.517(0.064)	**0.676(0.140)**	0.357(0.031)	0.862(0.035)	0.475(0.079)	0.830(0.039)	0.919(0.008)	0.944(0.027)	0.448(0.036)	0.935(0.020)	0.926(0.008)	0.864(0.037)
	SMT	0.496(0.081)	0.687(0.018)	0.467(0.055)	0.540(0.067)	0.590(0.232)	0.374(0.050)	0.886(0.044)	**0.517(0.153)**	0.910(0.019)	0.925(0.009)	0.971(0.014)	0.463(0.033)	0.971(0.014)	0.939(0.009)	0.926(0.031)
	RO	0.497(0.222)	0.737(0.022)	**0.494(0.101)**	0.493(0.217)	0.617(0.280)	0.432(0.168)	0.904(0.045)	0.37(0.341)	**0.950(0.015)**	0.924(0.008)	0.983(0.011)	0.516(0.045)	**0.988(0.011)**	0.936(0.007)	0.955(0.029)
	GN	0.511(0.121)	0.690(0.020)	0.438(0.058)	**0.600(0.119)**	0.659(0.152)	0.402(0.075)	0.901(0.042)	0.500(0.160)	0.907(0.023)	**0.961(0.005)**	0.974(0.012)	0.471(0.028)	0.978(0.013)	**0.941(0.008)**	0.947(0.025)
	WERCS	**0.520(0.086)**	0.728(0.021)	0.477(0.075)	0.568(0.135)	0.551(0.276)	**0.440(0.075)**	**0.908(0.045)**	0.423(0.279)	0.938(0.018)	0.936(0.008)	**0.983(0.011)**	0.529(0.040)	0.987(0.011)	0.940(0.008)	**0.955(0.027)**

Table 8
Mean and standard deviation of rec^{ϕ} by data set for the best parameters combination of algorithm and sampling strategy applied. (Bold signals the best performance by data set and learner while the blue cells represent the best overall result by data set.)

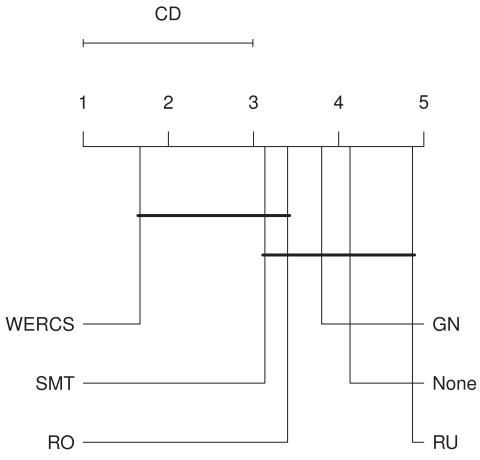
Learn	Strat	DS1	DS2	DS3	DS4	DS5	DS6	DS7	DS8	DS9	DS10	DS11	DS12	DS13	DS14	DS15
LM		0.686(0.169)	0.676(0.019)	0.660(0.258)	0.704(0.120)	0.736(0.090)	0.425(0.317)	0.811(0.043)	0.402(0.419)	0.862(0.105)	0.845(0.005)	0.921(0.036)	0.086(0.021)	**0.837(0.295)**	0.931(0.008)	0.800(0.222)
	RU	0.753(0.189)	0.718(0.019)	0.712(0.284)	0.627(0.201)	0.850(0.141)	0.491(0.375)	0.903(0.034)	0.413(0.428)	0.046(0.061)	0.944(0.003)	0.795(0.056)	0.093(0.026)	0.061(0.076)	0.957(0.005)	0.259(0.041)
	SMT	0.757(0.176)	**0.791(0.021)**	**0.751(0.282)**	0.704(0.157)	0.876(0.093)	0.496(0.371)	0.900(0.029)	0.430(0.446)	0.379(0.405)	0.956(0.004)	0.840(0.107)	**0.102(0.028)**	0.304(0.376)	0.950(0.005)	0.393(0.369)
	RO	0.767(0.172)	0.786(0.023)	0.719(0.281)	0.712(0.142)	0.897(0.105)	0.498(0.367)	**0.905(0.028)**	0.437(0.450)	**0.868(0.202)**	**0.957(0.004)**	**0.926(0.059)**	0.095(0.027)	0.834(0.308)	0.956(0.005)	**0.892(0.167)**
	GN	0.787(0.174)	0.787(0.020)	0.723(0.271)	0.726(0.172)	**0.914(0.047)**	**0.504(0.361)**	0.903(0.030)	0.427(0.442)	0.369(0.376)	0.946(0.003)	0.800(0.114)	0.097(0.029)	0.341(0.449)	**0.962(0.005)**	0.403(0.370)
	WERCS	**0.793(0.186)**	0.744(0.021)	0.734(0.294)	**0.735(0.118)**	0.895(0.094)	0.469(0.355)	0.881(0.031)	**0.448(0.462)**	0.682(0.288)	0.935(0.003)	0.904(0.083)	0.096(0.017)	0.619(0.382)	0.937(0.007)	0.563(0.380)
MARS		0.697(0.156)	0.692(0.021)	0.732(0.282)	0.661(0.088)	0.817(0.167)	0.434(0.312)	0.903(0.050)	0.396(0.416)	0.878(0.024)	0.935(0.008)	0.914(0.019)	0.129(0.033)	0.968(0.011)	0.935(0.008)	0.913(0.019)
	RU	0.785(0.145)	0.852(0.015)	0.748(0.284)	0.728(0.167)	0.909(0.052)	0.479(0.390)	0.920(0.033)	**0.423(0.452)**	**0.917(0.020)**	0.965(0.003)	0.943(0.010)	0.425(0.047)	0.998(0.002)	0.963(0.006)	0.954(0.013)
	SMT	0.762(0.147)	**0.861(0.013)**	0.760(0.280)	0.784(0.165)	0.906(0.119)	0.435(0.367)	0.917(0.025)	0.397(0.425)	0.909(0.026)	**0.966(0.004)**	0.941(0.012)	0.364(0.133)	0.986(0.018)	0.962(0.006)	0.953(0.014)
	RO	0.776(0.132)	0.849(0.016)	0.751(0.283)	0.816(0.156)	0.907(0.099)	0.442(0.444)	0.927(0.021)	0.376(0.406)	0.913(0.027)	0.966(0.003)	**0.951(0.011)**	**0.490(0.024)**	0.969(0.014)	0.963(0.006)	0.954(0.014)
	GN	**0.807(0.169)**	0.855(0.014)	**0.770(0.278)**	**0.836(0.117)**	**0.920(0.038)**	0.471(0.402)	**0.934(0.020)**	0.392(0.418)	0.912(0.038)	0.960(0.004)	0.943(0.017)	0.441(0.046)	**0.999(0.001)**	**0.963(0.006)**	**0.959(0.011)**
	WERCS	0.754(0.154)	0.773(0.019)	0.735(0.274)	0.752(0.137)	0.911(0.116)	**0.485(0.399)**	0.914(0.032)	0.398(0.423)	0.907(0.019)	0.959(0.004)	0.917(0.016)	0.142(0.028)	0.993(0.011)	0.951(0.006)	0.940(0.016)
NNET		0.704(0.185)	0.721(0.018)	0.694(0.282)	0.721(0.097)	0.766(0.118)	0.448(0.328)	0.818(0.047)	0.409(0.428)	0.730(0.331)	0.859(0.024)	0.930(0.018)	0.106(0.027)	0.960(0.014)	0.954(0.007)	0.874(0.019)
	RU	0.753(0.173)	0.854(0.016)	0.688(0.275)	0.638(0.206)	0.875(0.143)	0.485(0.374)	**0.913(0.033)**	0.423(0.444)	0.900(0.023)	**0.962(0.007)**	0.948(0.013)	0.142(0.034)	**0.986(0.004)**	0.965(0.005)	0.948(0.018)
	SMT	0.754(0.180)	0.858(0.017)	0.741(0.290)	0.727(0.136)	0.884(0.078)	**0.532(0.418)**	0.909(0.033)	0.442(0.458)	0.855(0.193)	0.961(0.005)	0.944(0.010)	**0.143(0.031)**	0.980(0.007)	0.966(0.006)	0.945(0.013)
	RO	0.776(0.184)	0.849(0.019)	0.731(0.283)	0.725(0.162)	0.904(0.094)	0.498(0.359)	0.905(0.030)	0.438(0.451)	0.833(0.245)	0.959(0.006)	**0.950(0.009)**	0.131(0.044)	0.984(0.006)	0.963(0.008)	0.949(0.008)
	GN	**0.819(0.151)**	**0.860(0.015)**	0.730(0.288)	0.742(0.126)	**0.941(0.043)**	0.502(0.364)	0.908(0.026)	0.444(0.458)	0.842(0.233)	0.947(0.004)	0.927(0.013)	0.138(0.036)	0.972(0.008)	**0.966(0.005)**	**0.960(0.012)**
	WERCS	0.787(0.181)	0.783(0.02)	**0.770(0.289)**	**0.750(0.139)**	0.893(0.091)	0.465(0.366)	0.885(0.040)	**0.452(0.466)**	**0.910(0.028)**	0.940(0.008)	0.946(0.012)	0.141(0.044)	0.982(0.005)	0.957(0.006)	0.931(0.019)
SVR		0.570(0.143)	0.682(0.020)	0.554(0.216)	0.602(0.076)	0.708(0.047)	0.338(0.246)	0.858(0.067)	0.379(0.405)	0.899(0.029)	0.600(0.006)	0.936(0.016)	0.158(0.028)	0.966(0.011)	0.946(0.008)	0.901(0.017)
	RU	**0.844(0.068)**	0.724(0.035)	**0.748(0.284)**	**0.837(0.073)**	**0.923(0.091)**	**0.482(0.365)**	0.926(0.021)	**0.449(0.463)**	0.869(0.041)	0.908(0.004)	0.946(0.011)	0.204(0.037)	0.972(0.006)	**0.964(0.004)**	**0.964(0.009)**
	SMT	0.777(0.125)	**0.860(0.013)**	0.710(0.302)	0.809(0.098)	0.901(0.091)	0.409(0.361)	0.923(0.036)	0.434(0.447)	0.931(0.020)	0.972(0.007)	0.957(0.011)	0.232(0.039)	0.980(0.005)	0.949(0.006)	0.952(0.011)
	RO	0.696(0.162)	0.845(0.014)	0.667(0.275)	0.719(0.106)	0.901(0.078)	0.381(0.326)	0.922(0.035)	0.410(0.446)	0.929(0.021)	**0.992(0.003)**	**0.961(0.012)**	0.216(0.035)	0.979(0.008)	0.950(0.007)	0.942(0.019)
	GN	0.814(0.120)	0.857(0.014)	0.702(0.289)	0.804(0.088)	0.915(0.073)	0.445(0.416)	**0.929(0.035)**	0.431(0.449)	**0.936(0.019)**	0.911(0.004)	0.943(0.015)	**0.296(0.048)**	**0.988(0.004)**	0.956(0.006)	0.960(0.010)
	WERCS	0.766(0.141)	0.766(0.021)	0.745(0.275)	0.731(0.077)	0.896(0.084)	0.365(0.299)	0.922(0.038)	0.413(0.432)	0.921(0.024)	0.901(0.009)	0.952(0.010)	0.201(0.042)	0.980(0.005)	0.952(0.008)	0.929(0.018)
RF		0.700(0.167)	0.703(0.020)	0.697(0.265)	0.682(0.085)	0.846(0.101)	0.475(0.350)	0.892(0.046)	0.379(0.394)	0.928(0.020)	0.897(0.010)	0.973(0.008)	0.492(0.033)	0.977(0.009)	0.884(0.009)	0.940(0.014)
	RU	**0.823(0.111)**	0.769(0.021)	0.802(0.284)	0.833(0.088)	**0.935(0.046)**	0.506(0.415)	0.916(0.029)	0.458(0.472)	0.876(0.021)	0.946(0.005)	0.974(0.007)	**0.509(0.033)**	0.979(0.009)	**0.954(0.005)**	0.961(0.011)
	SMT	0.755(0.131)	**0.869(0.014)**	0.779(0.280)	0.810(0.124)	0.913(0.049)	0.502(0.396)	0.916(0.034)	**0.463(0.476)**	0.960(0.010)	0.946(0.005)	0.980(0.010)	0.492(0.032)	0.984(0.007)	0.938(0.009)	0.966(0.017)
	RO	0.745(0.148)	0.726(0.02)	0.761(0.294)	0.722(0.089)	0.872(0.121)	0.475(0.358)	0.910(0.039)	0.378(0.394)	0.946(0.014)	0.946(0.005)	**0.987(0.011)**	0.494(0.038)	0.987(0.009)	0.897(0.009)	0.959(0.016)
	GN	0.817(0.097)	0.859(0.012)	0.759(0.283)	**0.839(0.096)**	0.930(0.051)	0.481(0.449)	**0.931(0.031)**	0.456(0.469)	**0.960(0.014)**	**0.973(0.004)**	0.986(0.011)	0.491(0.021)	**0.987(0.008)**	0.951(0.005)	**0.972(0.016)**
	WERCS	0.790(0.132)	0.746(0.018)	**0.804(0.293)**	0.750(0.083)	0.879(0.081)	**0.507(0.375)**	0.914(0.039)	0.403(0.418)	0.945(0.017)	0.933(0.007)	0.986(0.010)	0.495(0.035)	0.986(0.010)	0.908(0.009)	0.957(0.016)



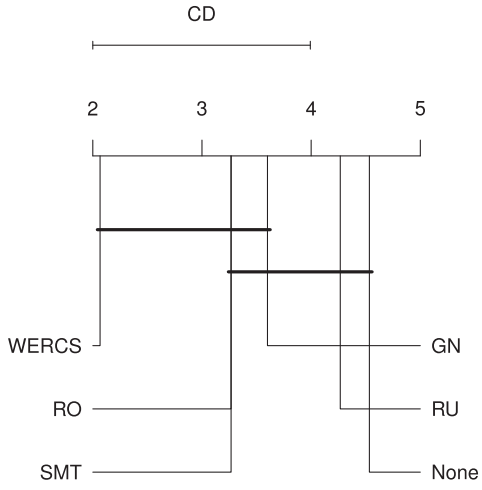
(a) F_1^{ϕ} results for LM learner.



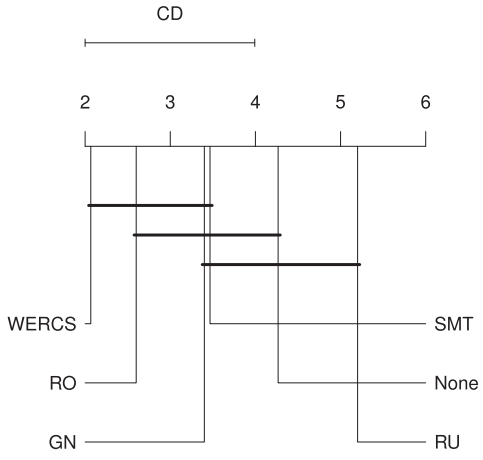
(b) F_1^{ϕ} results for MARS learner.



(c) F_1^{ϕ} results for NNET learner.



(d) F_1^{ϕ} results for SVR learner.



(e) F_1^{ϕ} results for RF learner.

Fig. 7. CD diagrams of average F_1^{ϕ} results by learner type.

Table 9
Aggregation of $prec^{\phi}$ results from Table 7. Total number of data sets where each pair of sampling strategy-learner had the best results.

	None	RU	SMT	RO	GN	WERCS
LM	8	0	2	0	4	1
MARS	6	1	0	2	4	5
NNET	7	0	1	0	3	4
SVR	3	2	0	2	5	3
RF	2	1	1	3	3	5
Total	26	4	4	7	19	18

Table 10
Aggregation of rec^{ϕ} results from Table 8. Total number of data sets where each pair of sampling strategy-learner had the best results.

	None	RU	SMT	RO	GN	WERCS
LM	1	0	3	5	3	3
MARS	0	2	2	2	8	1
NNET	0	3	2	1	5	4
SVR	0	8	1	2	4	0
RF	0	4	2	1	6	2
Total	1	17	10	11	26	10

pre-processing strategies also have an advantage. In this case, the new proposed strategies achieve the best results in 8 data sets, while in 5 data sets random under-sampling (RU) or None display the best performance. For two data sets it was not possible to determine the best performing strategy because all metric results were zero. Regarding the results of $spec^{\phi}$, we also observe that in 9 data sets the new pre-processing methods achieve the best performance while in 4 data sets the best performance is obtained by using the original data set or random under-sampling. The results of Tables 11 and 12 are summarized in Tables 13 and 14, respectively. These tables show that the results of $spec^{\phi}$ are worse than using the original data set. This was expectable because $spec^{\phi}$ is solely focused on the performance of the least important cases. However, the results of $G-Mean^{\phi}$ show that the loss in the performance at the least important cases is compensated by the gains in the more relevant cases.

For assessing the statistical significance of the $G-Mean^{\phi}$ results we applied the non-parametric Friedman *F*-test. After rejecting the null hypothesis that all tested approaches have the same performance using the *F*-test, we applied the post-hoc Nemenyi test with a significance level of 95%. This allows to verify which approaches are statistically different. Fig. 8 a–e shows the obtained CD diagrams [8] with the results aggregated by learner. In these results the **WERCS** strategy is also the one that exhibits the lower ranks for almost all the learners and hence the best performance. However, for the $G-Mean^{\phi}$ metric only the LM results display statistical significance in the observed differences. Moreover, we also observe that, globally, when considering this measure the strategy of using the original data set does not display one of the worst performances.

5.2. Impact of different data set characteristics in the performance

In this section our goal is to assess the impact of data characteristics in the performance obtained with the different pre-processing strategies. The characteristics explored are: percentage of rare cases; data set size and number of rare cases. These characteristics were selected because they are easy to determine for any data set. Moreover, they do not require further domain knowledge as would be the case with, for instance, the noise level present in the data set. Figs. 9–11 show the best F_1^{ϕ} results obtained, by learner and by sampling strategy, with the data sets sorted by a given characteristic. Fig. 9 shows the results when the data sets are ordered by decreasing percentage of rare cases. This corresponds to the same ordering provided in Table 1 by column % Rare. Fig. 10 displays the same F_1^{ϕ} results with the data sets now sorted by decreasing size, i.e., the data sets are ordered from the data set with the largest total number of examples (DS12) to the data set with the smallest total number of examples (DS8). In this case, the ties are broken by considering the decreasing percentage of rare cases. Finally, Fig. 11 presents the F_1^{ϕ} results with the data sets sorted by decreasing number of rare cases (column nRare in Table 1).

The results observed with the data sets sorted by decreasing percentage of rare cases do not show a clear tendency for any of the tested learners. This means that the percentage of rare cases in a data set may not be an important characteristic to decide which are the most difficult imbalanced regression problems. This was also observed in the context of class imbalance problems (e.g. [1,12]).

On the other hand, the figures with the data sets sorted by data set size and by number of rare cases, display a more clear tendency. In these cases, for all learners, the most difficult tasks are associated with the smaller data sets and the data sets with a smaller number of rare cases. Generally, we observe that these are the data characteristics associated with a larger performance loss across all the learners. We also observe that the sampling strategies have an increased positive impact in the performance of the smaller data sets and the data sets with a smaller number of rare cases. Only the Linear Regression learner (LM) presents some differences regarding a performance loss when some sampling strategies are applied on data sets with and intermediate size and number of rare cases.

The main conclusions of this section are that: (i) the percentage of rare cases in the data set is not a key feature for assessing the complexity inherent to an imbalanced regression task; (ii) the data set size and the total number of rare cases seem to have more impact on imbalanced regression problems.

5.3. Impact of varying the relevance threshold in the performance

In this section our goal is to assess the impact of setting different relevance thresholds in the predictive performance obtained with the different pre-processing strategies. The relevance threshold determines the values that are important, and thus the higher is the relevance threshold value, the lower is the number of rare cases. We studied the impact of varying the relevance threshold on two specific data sets. We selected data sets DS3 and DS8 because they exhibited different behaviors in the first set of experiments (cf. Table 4) and have the same number of examples. These data sets allow us to obtain percentages of rare cases ranging from 6.6% to 21.1% by varying the relevance threshold which has a direct connection with the total number of rare cases.

For each data set, all the values in the interval [0.5,1] with a step of 0.025 were considered as thresholds. From this set of values, only those that affected the number of rare cases were taken into account. Table 15 shows the number of different thresholds considered for the data sets and the percentage and number of rare cases obtained in this process.

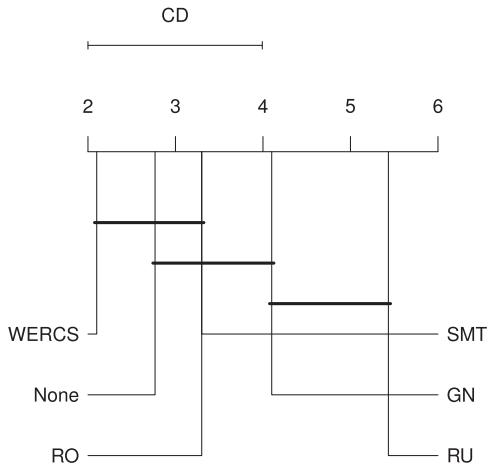
Figs. 12 and 13 show the mean rank trimmed at 20% by learner and pre-processing strategy, for a relevance threshold varying between 0.5 and 1 on data sets DS3 and DS8, respectively. The lower is the rank value, the better is the performance. The results suggest a complex relationship between the number of rare cases, the learning algorithm used and the pre-processing strategy applied.

Table 11
Mean and standard deviation of $G-Mean^{\phi}$ by data set for the best parameters combination of algorithm and sampling strategy applied. Results in bold highlight the best performance by data set and learner while the blue highlighted cells signals the best result by data set.

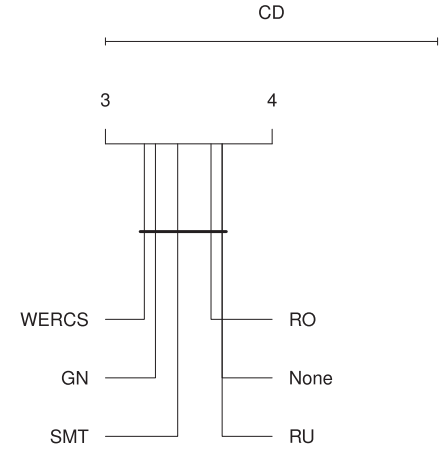
Learn	Strat	DS1	DS2	DS3	DS4	DS5	DS6	DS7	DS8	DS9	DS10	DS11	DS12	DS13	DS14	DS15
LM		0.740(0.137)	0.725(0.015)	0.701(0.260)	0.749(0.093)	0.788(0.071)	0.440(0.32)	**0.013(0.003)**	0.420(0.434)	0.000(0.000)	**0.820(0.004)**	**0.026(0.006)**	0.151(0.028)	**0.850(0.292)**	0.000(0.000)	0.832(0.207)
	RU	0.773(0.154)	0.752(0.015)	0.717(0.271)	0.668(0.167)	0.862(0.111)	0.472(0.348)	0.010(0.003)	0.413(0.427)	0.000(0.000)	0.749(0.007)	0.007(0.002)	0.150(0.031)	0.097(0.101)	0.000(0.000)	0.348(0.046)
	SMT	0.780(0.143)	0.795(0.016)	**0.749(0.270)**	0.737(0.128)	0.878(0.068)	0.478(0.346)	0.010(0.002)	0.435(0.449)	0.000(0.000)	0.786(0.010)	0.011(0.008)	**0.161(0.033)**	0.334(0.376)	0.000(0.000)	0.445(0.346)
	RO	0.789(0.139)	0.796(0.017)	0.720(0.265)	0.745(0.116)	0.901(0.080)	0.475(0.342)	0.009(0.002)	0.439(0.451)	0.000(0.000)	0.794(0.008)	0.019(0.005)	0.155(0.029)	0.845(0.301)	0.000(0.000)	**0.908(0.144)**
	GN	0.794(0.139)	**0.796(0.017)**	0.729(0.263)	0.754(0.141)	**0.904(0.034)**	**0.484(0.344)**	0.009(0.002)	0.430(0.443)	0.000(0.000)	0.759(0.011)	0.009(0.008)	0.157(0.030)	0.358(0.445)	0.000(0.000)	0.456(0.343)
	WERCS	**0.807(0.148)**	0.777(0.016)	0.733(0.276)	**0.764(0.096)**	0.901(0.072)	0.455(0.334)	0.009(0.002)	**0.447(0.460)**	0.000(0.000)	0.802(0.005)	0.018(0.007)	0.159(0.021)	0.648(0.373)	0.000(0.000)	0.609(0.347)
MARS		0.749(0.127)	0.738(0.016)	0.757(0.278)	0.717(0.070)	0.849(0.133)	0.452(0.320)	**0.014(0.003)**	0.415(0.431)	0.000(0.000)	0.880(0.015)	0.029(0.002)	0.207(0.039)	0.974(0.009)	0.000(0.000)	0.930(0.014)
	RU	0.789(0.111)	0.820(0.011)	0.740(0.27)	0.756(0.139)	0.894(0.107)	0.460(0.369)	0.010(0.003)	**0.422(0.444)**	0.000(0.000)	0.874(0.024)	0.021(0.002)	0.483(0.029)	0.998(0.001)	0.000(0.000)	0.959(0.008)
	SMT	0.776(0.122)	0.817(0.017)	0.754(0.27)	0.787(0.14)	0.914(0.093)	0.430(0.348)	0.011(0.002)	0.396(0.418)	0.000(0.000)	0.893(0.009)	0.022(0.002)	0.432(0.127)	0.988(0.014)	0.000(0.000)	0.959(0.011)
	RO	0.779(0.104)	**0.830(0.011)**	0.748(0.272)	0.812(0.129)	0.912(0.077)	0.419(0.393)	0.011(0.003)	0.382(0.404)	0.000(0.000)	0.889(0.013)	0.022(0.003)	**0.526(0.020)**	0.974(0.011)	0.000(0.000)	0.959(0.011)
	GN	**0.824(0.135)**	0.830(0.012)	**0.763(0.269)**	**0.820(0.086)**	**0.925(0.074)**	0.452(0.4)	0.011(0.002)	0.404(0.424)	0.000(0.000)	0.863(0.007)	0.022(0.002)	0.494(0.029)	**0.999(0.001)**	0.000(0.000)	**0.963(0.008)**
	WERCS	0.783(0.127)	0.800(0.015)	0.751(0.278)	0.782(0.109)	0.915(0.092)	**0.465(0.373)**	0.012(0.002)	0.415(0.431)	0.000(0.000)	**0.903(0.005)**	**0.029(0.001)**	0.221(0.033)	0.994(0.009)	0.000(0.000)	0.951(0.012)
NNET		0.753(0.148)	0.761(0.015)	0.724(0.275)	0.762(0.075)	0.811(0.093)	0.464(0.33)	**0.013(0.003)**	0.424(0.440)	0.000(0.000)	**0.824(0.005)**	**0.028(0.001)**	0.177(0.034)	0.968(0.011)	0.000(0.000)	0.901(0.015)
	RU	0.774(0.140)	0.831(0.012)	0.699(0.262)	0.675(0.172)	0.883(0.112)	0.468(0.347)	0.010(0.002)	0.416(0.432)	0.000(0.000)	0.789(0.013)	0.021(0.002)	**0.218(0.046)**	0.985(0.004)	0.000(0.000)	0.955(0.014)
	SMT	0.778(0.147)	0.827(0.014)	0.741(0.275)	0.753(0.107)	0.885(0.059)	**0.500(0.37)**	0.010(0.003)	0.444(0.458)	0.000(0.000)	0.791(0.022)	0.021(0.002)	0.212(0.035)	0.983(0.005)	0.000(0.000)	0.954(0.010)
	RO	0.795(0.151)	0.838(0.013)	0.729(0.267)	0.754(0.109)	0.900(0.083)	0.477(0.340)	0.009(0.002)	0.439(0.451)	0.000(0.000)	0.795(0.010)	0.022(0.002)	0.206(0.049)	**0.986(0.004)**	0.000(0.000)	0.957(0.006)
	GN	**0.813(0.122)**	**0.844(0.011)**	0.736(0.275)	0.762(0.100)	**0.924(0.030)**	0.482(0.344)	0.009(0.002)	0.442(0.455)	0.000(0.000)	0.764(0.012)	0.020(0.002)	0.213(0.042)	0.977(0.011)	0.000(0.000)	**0.965(0.009)**
	WERCS	0.805(0.145)	0.809(0.016)	**0.763(0.276)**	**0.775(0.112)**	0.900(0.066)	0.451(0.341)	0.009(0.003)	**0.452(0.465)**	0.000(0.000)	0.810(0.012)	0.024(0.002)	0.217(0.053)	0.985(0.003)	0.000(0.000)	0.944(0.014)
SVR		0.646(0.120)	0.731(0.017)	0.618(0.229)	0.671(0.065)	0.767(0.037)	0.379(0.265)	**0.016(0.003)**	0.400(0.421)	0.000(0.000)	0.710(0.008)	**0.032(0.001)**	0.241(0.032)	0.973(0.009)	0.000(0.000)	0.922(0.013)
	RU	**0.825(0.051)**	0.759(0.016)	**0.759(0.277)**	**0.830(0.056)**	**0.927(0.070)**	**0.472(0.350)**	0.013(0.003)	**0.448(0.461)**	0.000(0.000)	0.415(0.011)	0.023(0.002)	0.285(0.038)	0.977(0.005)	0.000(0.000)	0.953(0.009)
	SMT	0.788(0.099)	0.834(0.010)	0.727(0.286)	0.818(0.075)	0.911(0.07)	0.395(0.313)	0.015(0.002)	0.423(0.438)	0.000(0.000)	0.726(0.004)	0.026(0.002)	0.318(0.041)	0.984(0.005)	0.000(0.000)	0.960(0.008)
	RO	0.741(0.134)	0.836(0.011)	0.690(0.264)	0.757(0.084)	0.910(0.06)	0.392(0.319)	0.015(0.003)	0.417(0.443)	0.000(0.000)	**0.851(0.005)**	0.028(0.001)	0.303(0.037)	0.983(0.006)	0.000(0.000)	0.952(0.008)
	GN	0.815(0.093)	**0.841(0.011)**	0.722(0.278)	0.809(0.067)	0.923(0.056)	0.410(0.381)	0.015(0.003)	0.439(0.454)	0.000(0.000)	0.529(0.006)	0.027(0.002)	**0.381(0.047)**	**0.989(0.003)**	0.000(0.000)	**0.964(0.008)**
	WERCS	0.797(0.114)	0.798(0.016)	0.756(0.270)	0.770(0.07)	0.908(0.063)	0.381(0.3)	0.015(0.003)	0.425(0.441)	0.000(0.000)	0.756(0.004)	0.028(0.002)	0.287(0.046)	0.984(0.004)	0.000(0.000)	0.943(0.014)
RF		0.752(0.135)	0.747(0.016)	0.731(0.267)	0.736(0.069)	0.875(0.079)	0.490(0.353)	**0.016(0.003)**	0.402(0.416)	0.000(0.000)	0.889(0.005)	0.048(0.001)	0.566(0.029)	0.982(0.007)	0.000(0.000)	0.952(0.011)
	RU	**0.832(0.089)**	0.769(0.013)	0.780(0.273)	0.809(0.076)	**0.937(0.035)**	0.469(0.389)	0.011(0.003)	**0.453(0.465)**	0.000(0.000)	0.855(0.006)	0.030(0.001)	**0.577(0.028)**	0.981(0.005)	0.000(0.000)	0.963(0.010)
	SMT	0.785(0.105)	0.814(0.011)	0.781(0.275)	0.810(0.094)	0.914(0.064)	0.486(0.381)	0.013(0.003)	0.447(0.459)	0.000(0.000)	0.878(0.006)	0.045(0.001)	0.562(0.025)	0.987(0.005)	0.000(0.000)	0.971(0.013)
	RO	0.785(0.118)	0.765(0.016)	0.776(0.286)	0.768(0.069)	0.893(0.096)	0.486(0.358)	0.015(0.003)	0.401(0.415)	0.000(0.000)	0.873(0.005)	**0.049(0.001)**	0.568(0.034)	**0.990(0.007)**	0.000(0.000)	0.967(0.012)
	GN	0.814(0.075)	**0.837(0.009)**	0.764(0.276)	**0.838(0.073)**	0.927(0.064)	0.438(0.371)	0.014(0.003)	0.442(0.454)	0.000(0.000)	**0.927(0.004)**	0.046(0.001)	0.564(0.019)	0.987(0.006)	0.000(0.000)	**0.974(0.012)**
	WERCS	0.816(0.104)	0.782(0.015)	**0.800(0.284)**	0.789(0.063)	0.898(0.064)	**0.506(0.368)**	0.014(0.003)	0.420(0.433)	0.000(0.000)	0.877(0.006)	0.048(0.001)	0.568(0.030)	0.989(0.007)	0.000(0.000)	0.966(0.012)

Table 12
Mean and standard deviation of $spec^{\phi}$ by data set for the best parameters combination of algorithm and sampling strategy applied. Results in bold highlight the best performance by data set and learner while the blue highlighted cells signals the best result by data set.

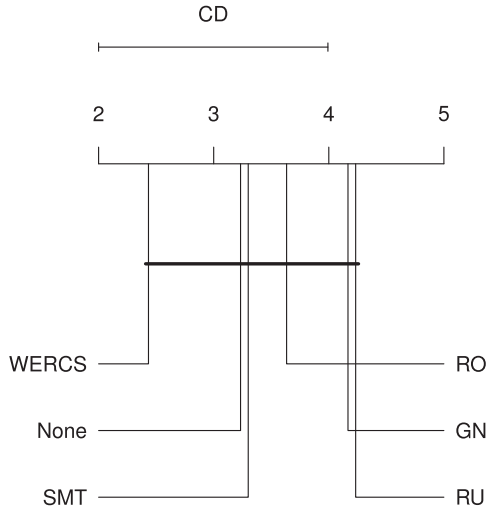
Learn	Strat	DS1	DS2	DS3	DS4	DS5	DS6	DS7	DS8	DS9	DS10	DS11	DS12	DS13	DS14	DS15
LM		0.801(0.099)	0.777(0.011)	0.746(0.264)	**0.798(0.063)**	0.844(0.050)	**0.491(0.340)**	**0.001(0.000)**	0.440(0.450)	**0.003(0.000)**	**0.860(0.003)**	**0.002(0.000)**	**0.265(0.033)**	**0.865(0.282)**	0.003(0.000)	0.869(0.180)
	RU	0.796(0.116)	0.786(0.011)	0.725(0.261)	0.717(0.127)	0.876(0.078)	0.472(0.331)	0.001(0.000)	0.415(0.425)	0.003(0.000)	0.803(0.006)	0.001(0.000)	0.250(0.021)	0.160(0.130)	0.003(0.000)	0.466(0.05)
	SMT	0.807(0.106)	0.816(0.012)	**0.748(0.261)**	0.773(0.095)	0.892(0.060)	0.479(0.332)	0.001(0.000)	0.441(0.451)	0.003(0.000)	0.826(0.008)	0.001(0.001)	0.262(0.027)	0.381(0.366)	0.003(0.000)	0.520(0.311)
	RO	0.814(0.102)	0.814(0.012)	0.724(0.254)	0.783(0.085)	0.905(0.055)	0.477(0.328)	0.001(0.000)	0.442(0.451)	**0.003(0.000)**	0.832(0.007)	0.002(0.000)	0.256(0.033)	0.859(0.288)	0.003(0.000)	**0.926(0.116)**
	GN	0.819(0.101)	**0.817(0.012)**	0.736(0.258)	0.786(0.103)	0.903(0.052)	0.486(0.335)	0.001(0.000)	0.435(0.444)	0.003(0.000)	0.809(0.009)	0.001(0.001)	0.258(0.034)	0.391(0.43)	0.003(0.000)	0.531(0.302)
	WERCS	**0.825(0.107)**	0.811(0.011)	0.735(0.262)	0.796(0.072)	**0.908(0.050)**	0.465(0.325)	0.001(0.000)	**0.449(0.458)**	0.003(0.000)	0.846(0.004)	0.001(0.000)	0.264(0.022)	0.685(0.352)	0.003(0.000)	0.671(0.296)
MARS		0.806(0.093)	0.787(0.012)	**0.785(0.277)**	0.779(0.050)	0.885(0.095)	**0.504(0.346)**	**0.001(0.000)**	**0.437(0.448)**	0.003(0.000)	0.910(0.01)	0.002(0.000)	0.331(0.042)	0.980(0.006)	0.003(0.000)	0.948(0.01)
	RU	0.812(0.082)	0.832(0.010)	0.735(0.259)	0.789(0.104)	0.909(0.075)	0.468(0.356)	0.001(0.000)	0.424(0.437)	0.003(0.000)	0.902(0.019)	0.002(0.000)	0.550(0.013)	0.999(0.001)	0.003(0.000)	0.965(0.006)
	SMT	0.808(0.094)	**0.833(0.012)**	0.750(0.259)	0.808(0.067)	0.923(0.065)	0.446(0.339)	0.001(0.000)	0.414(0.427)	0.003(0.000)	0.917(0.007)	0.002(0.000)	0.519(0.111)	0.991(0.011)	0.003(0.000)	0.965(0.007)
	RO	0.805(0.103)	0.832(0.011)	0.747(0.263)	**0.831(0.068)**	0.919(0.054)	0.422(0.379)	0.001(0.000)	0.393(0.405)	**0.003(0.000)**	0.915(0.01)	0.002(0.000)	**0.564(0.017)**	0.980(0.008)	0.003(0.000)	0.965(0.007)
	GN	**0.844(0.098)**	0.832(0.011)	0.758(0.262)	0.828(0.060)	**0.931(0.052)**	0.459(0.379)	0.001(0.000)	0.418(0.431)	0.003(0.000)	0.895(0.004)	0.002(0.000)	0.554(0.014)	**0.999(0.001)**	0.003(0.000)	**0.967(0.005)**
	WERCS	0.815(0.096)	0.829(0.010)	0.780(0.276)	0.815(0.081)	0.921(0.066)	0.484(0.326)	0.001(0.000)	**0.437(0.448)**	0.003(0.000)	**0.925(0.004)**	**0.002(0.000)**	0.344(0.035)	0.995(0.007)	0.003(0.000)	0.962(0.008)
NNET		0.808(0.104)	0.804(0.011)	**0.758(0.272)**	**0.807(0.051)**	0.860(0.065)	**0.507(0.348)**	**0.001(0.000)**	0.443(0.454)	0.003(0.000)	**0.863(0.005)**	**0.002(0.000)**	0.298(0.039)	0.976(0.008)	0.003(0.000)	0.928(0.010)
	RU	0.797(0.105)	0.837(0.011)	0.712(0.254)	0.718(0.131)	0.893(0.079)	0.470(0.331)	0.001(0.000)	0.410(0.419)	0.003(0.000)	0.826(0.009)	0.002(0.000)	**0.338(0.051)**	0.985(0.006)	0.003(0.000)	0.962(0.011)
	SMT	0.806(0.109)	0.842(0.010)	0.743(0.263)	0.782(0.074)	0.897(0.057)	0.494(0.347)	0.001(0.000)	0.447(0.457)	0.003(0.000)	0.829(0.017)	0.002(0.000)	0.316(0.036)	0.987(0.004)	0.003(0.000)	0.964(0.007)
	RO	0.817(0.113)	0.840(0.011)	0.730(0.261)	0.788(0.081)	0.905(0.056)	0.479(0.330)	0.001(0.000)	0.442(0.451)	0.003(0.000)	0.833(0.008)	0.002(0.000)	0.325(0.049)	0.987(0.003)	0.003(0.000)	0.965(0.005)
	GN	0.825(0.104)	**0.846(0.011)**	0.744(0.266)	0.791(0.093)	**0.912(0.050)**	0.482(0.334)	0.001(0.000)	0.442(0.452)	0.003(0.000)	0.813(0.018)	0.002(0.000)	0.332(0.045)	0.982(0.008)	0.003(0.000)	**0.969(0.007)**
	WERCS	**0.826(0.105)**	0.836(0.011)	0.757(0.265)	0.802(0.084)	0.910(0.047)	0.461(0.327)	0.001(0.000)	**0.453(0.462)**	**0.003(0.000)**	0.852(0.008)	0.002(0.000)	0.335(0.059)	**0.989(0.002)**	0.003(0.000)	0.958(0.010)
SVR		0.736(0.091)	0.784(0.013)	0.691(0.245)	0.749(0.051)	0.832(0.026)	0.455(0.310)	**0.001(0.000)**	0.425(0.438)	0.003(0.000)	**0.791(0.006)**	**0.002(0.000)**	0.367(0.034)	0.98(0.006)	0.003(0.000)	0.943(0.009)
	RU	**0.830(0.070)**	0.800(0.012)	**0.771(0.272)**	0.824(0.048)	**0.932(0.047)**	**0.489(0.344)**	0.001(0.000)	**0.449(0.459)**	0.003(0.000)	0.409(0.008)	0.002(0.000)	0.400(0.043)	0.982(0.004)	0.003(0.000)	0.964(0.006)
	SMT	0.804(0.070)	**0.845(0.011)**	0.748(0.274)	**0.829(0.051)**	0.921(0.049)	0.437(0.320)	0.001(0.000)	0.438(0.448)	0.003(0.000)	0.675(0.003)	0.002(0.000)	0.435(0.038)	0.987(0.003)	0.003(0.000)	0.968(0.006)
	RO	0.791(0.103)	0.836(0.012)	0.717(0.257)	0.798(0.059)	0.919(0.042)	0.434(0.321)	0.001(0.000)	0.427(0.442)	0.003(0.000)	0.814(0.004)	0.002(0.000)	0.425(0.034)	0.987(0.004)	0.003(0.000)	0.963(0.005)
	GN	0.817(0.067)	0.844(0.012)	0.746(0.269)	0.818(0.041)	0.931(0.040)	0.427(0.320)	0.001(0.000)	0.448(0.459)	0.003(0.000)	0.498(0.005)	0.002(0.000)	**0.491(0.042)**	**0.991(0.002)**	0.003(0.000)	**0.969(0.005)**
	WERCS	0.830(0.085)	0.831(0.012)	0.769(0.267)	0.813(0.053)	0.921(0.043)	0.427(0.318)	0.001(0.000)	0.439(0.450)	0.003(0.000)	0.708(0.003)	0.002(0.000)	0.411(0.045)	0.988(0.003)	0.003(0.000)	0.958(0.010)
RF		0.810(0.097)	0.795(0.012)	0.769(0.271)	0.796(0.051)	0.905(0.055)	**0.533(0.368)**	**0.001(0.000)**	0.429(0.438)	0.003(0.000)	0.917(0.004)	0.003(0.000)	0.652(0.023)	0.988(0.004)	0.003(0.000)	0.965(0.007)
	RU	0.842(0.067)	0.802(0.010)	0.759(0.262)	0.829(0.053)	**0.940(0.025)**	0.455(0.360)	0.001(0.000)	**0.448(0.457)**	0.003(0.000)	0.885(0.005)	0.002(0.000)	**0.654(0.022)**	0.984(0.004)	0.003(0.000)	0.97(0.007)
	SMT	0.818(0.076)	0.834(0.011)	0.783(0.270)	0.822(0.048)	0.928(0.044)	0.492(0.369)	0.001(0.000)	0.442(0.452)	0.003(0.000)	0.905(0.005)	0.003(0.000)	0.646(0.020)	0.990(0.003)	0.003(0.000)	0.975(0.009)
	RO	0.829(0.083)	0.807(0.011)	0.792(0.280)	0.818(0.048)	0.916(0.068)	0.526(0.368)	0.001(0.000)	0.428(0.437)	0.003(0.000)	0.900(0.004)	**0.003(0.000)**	0.653(0.026)	**0.992(0.005)**	0.003(0.000)	0.975(0.008)
	GN	0.832(0.076)	**0.837(0.009)**	0.769(0.270)	**0.837(0.051)**	0.936(0.044)	0.460(0.368)	0.001(0.000)	0.447(0.456)	0.003(0.000)	**0.945(0.003)**	0.003(0.000)	0.648(0.015)	0.991(0.004)	0.003(0.000)	**0.977(0.008)**
	WERCS	**0.845(0.074)**	0.819(0.011)	**0.796(0.277)**	0.830(0.043)	0.918(0.046)	0.530(0.369)	0.001(0.000)	0.439(0.448)	0.003(0.000)	0.902(0.005)	0.003(0.000)	0.652(0.024)	0.992(0.005)	0.003(0.000)	0.975(0.008)



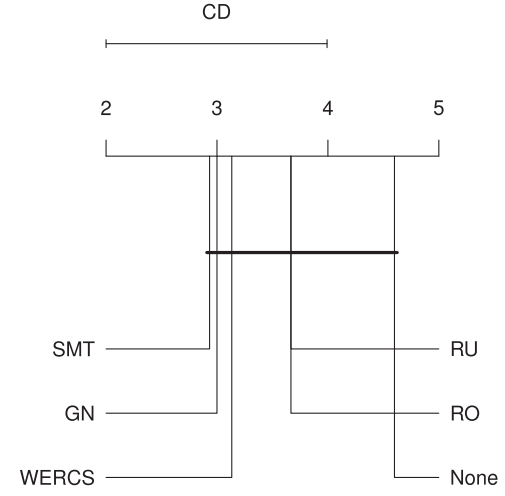
(a) $G - Mean^{\phi}$ results for LM learner.



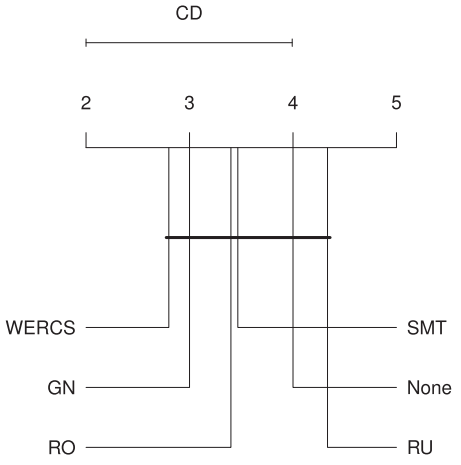
(b) $G - Mean^{\phi}$ results for MARS learner.



(c) $G - Mean^{\phi}$ results for NNET learner.



(d) $G - Mean^{\phi}$ results for SVR learner.



(e) $G - Mean^{\phi}$ results for RF learner.

Fig. 8. CD diagrams of average $G - Mean^{\phi}$ results by learner type.

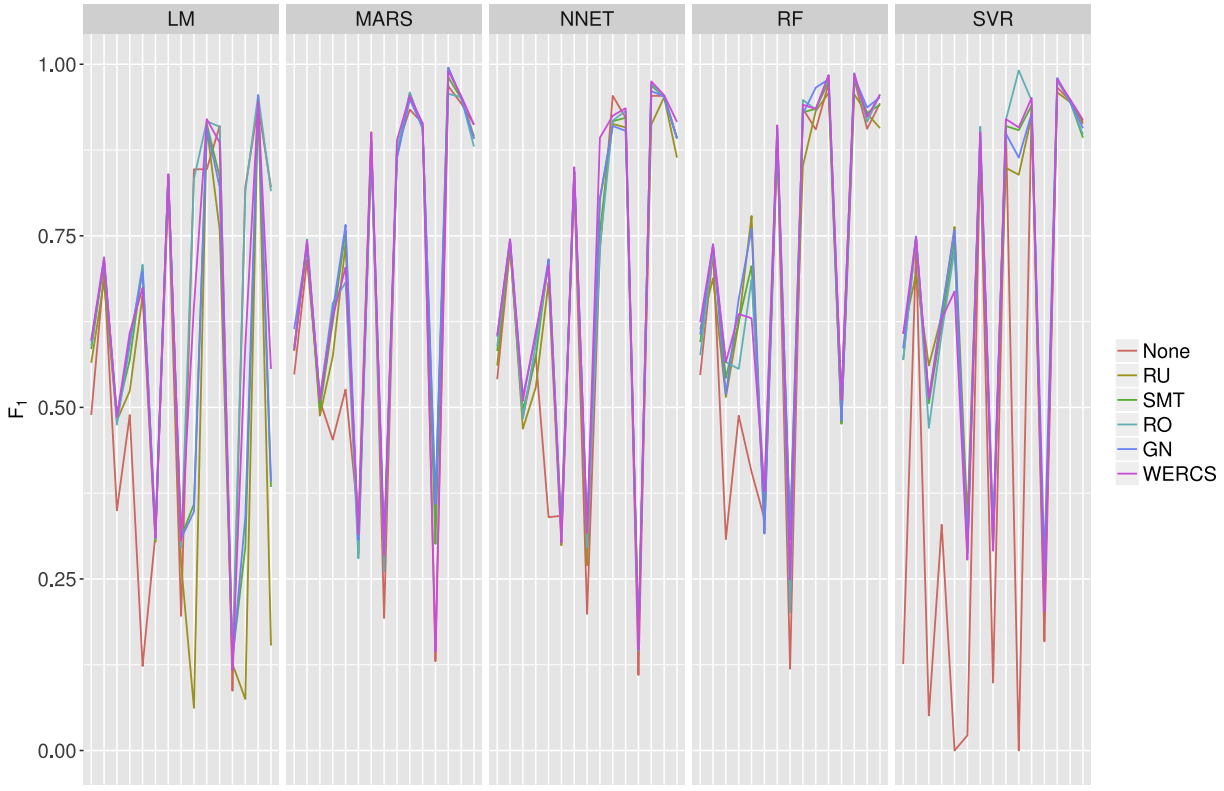


Fig. 9. Best F_1^{ϕ} results obtained by learner and by sampling strategy for each data set sorted by decreasing percentage of rare cases.

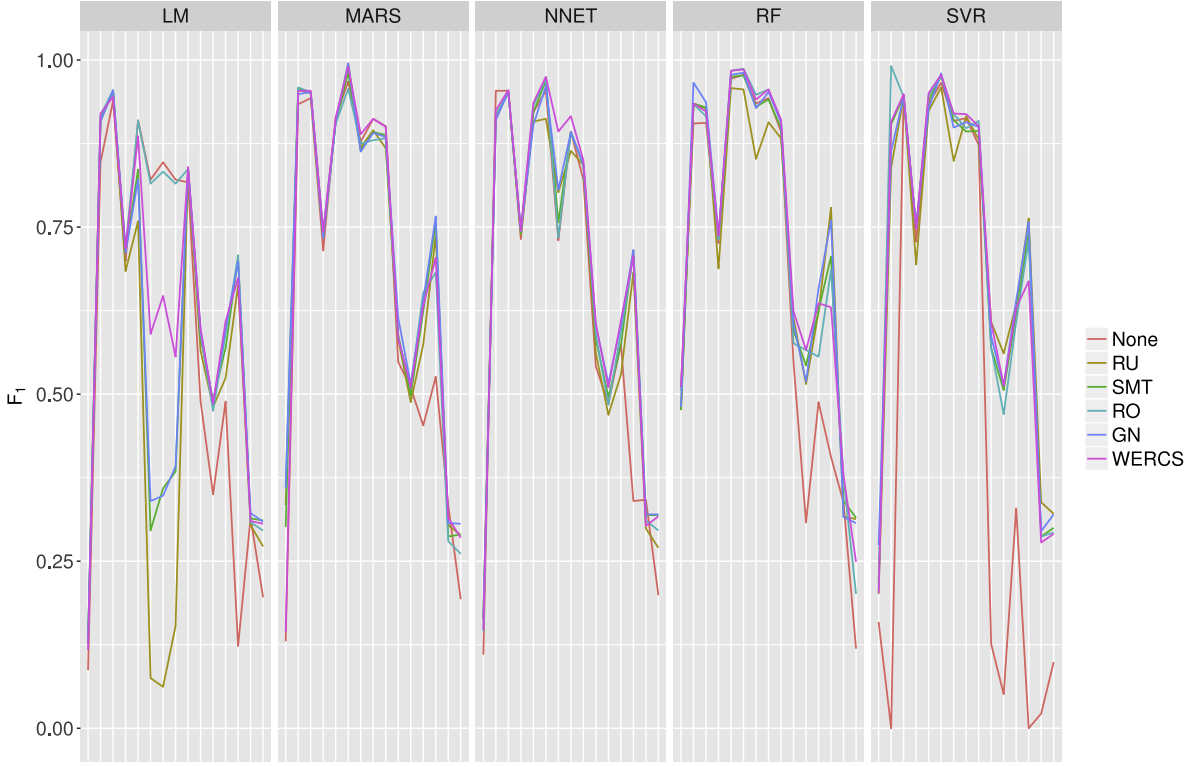


Fig. 10. Best F_1^{ϕ} results obtained by learner and by sampling strategy for each data set sorted by decreasing size.

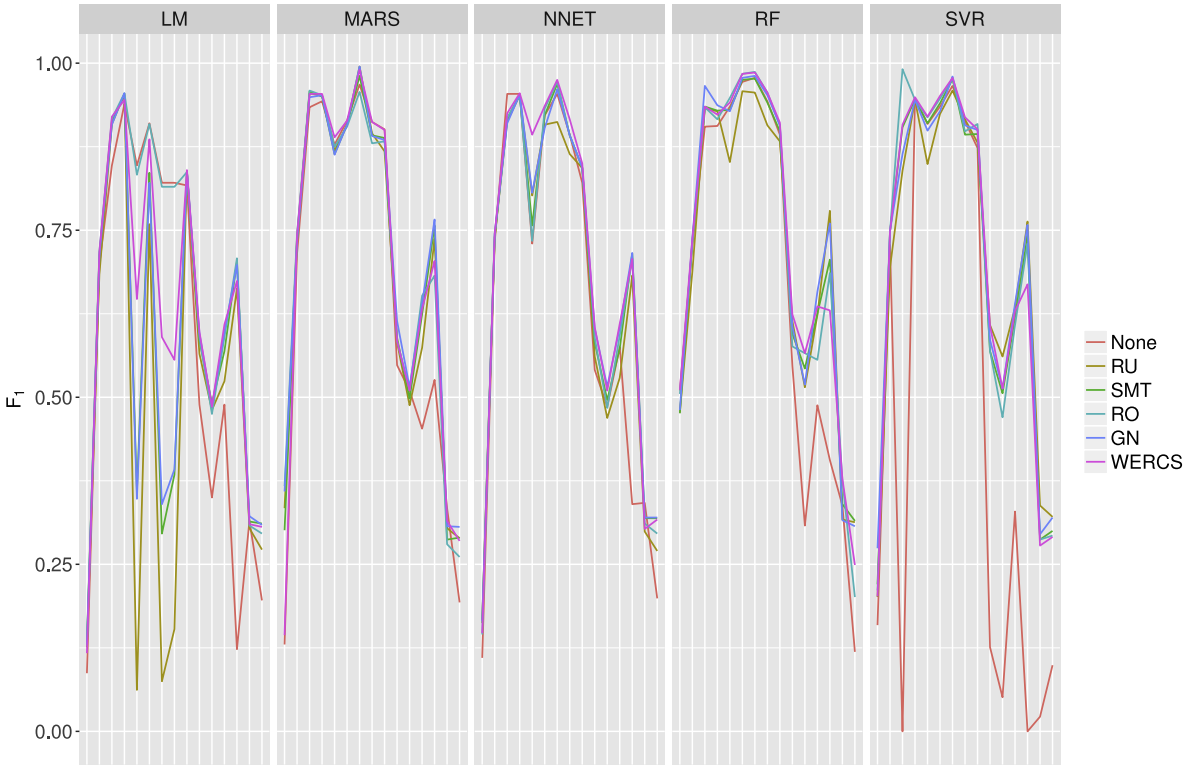


Fig. 11. Best F_1^{ϕ} results obtained by learner and by sampling strategy for each data set sorted by decreasing number of rare cases.

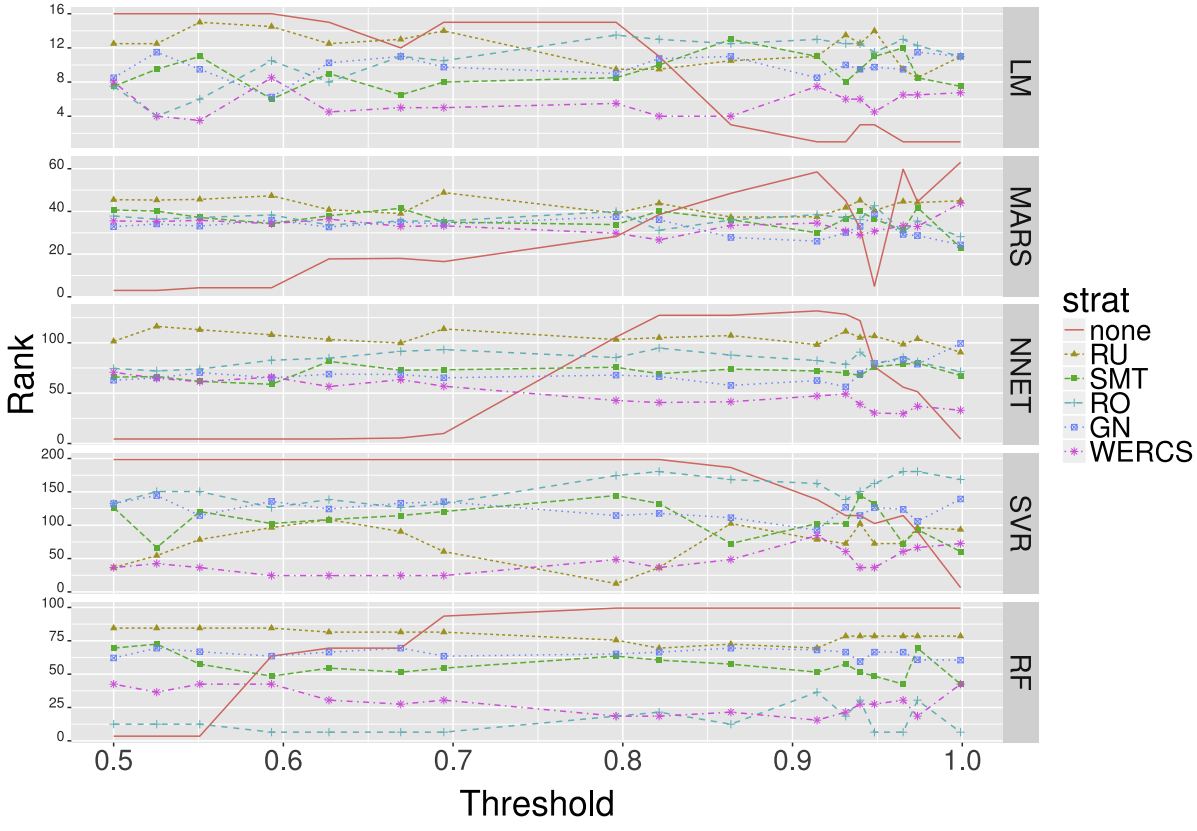


Fig. 12. Trimmed mean rank at 20% of each sampling strategy by learning algorithm on data set a3 (DS3) for relevance thresholds in [0.5,1].

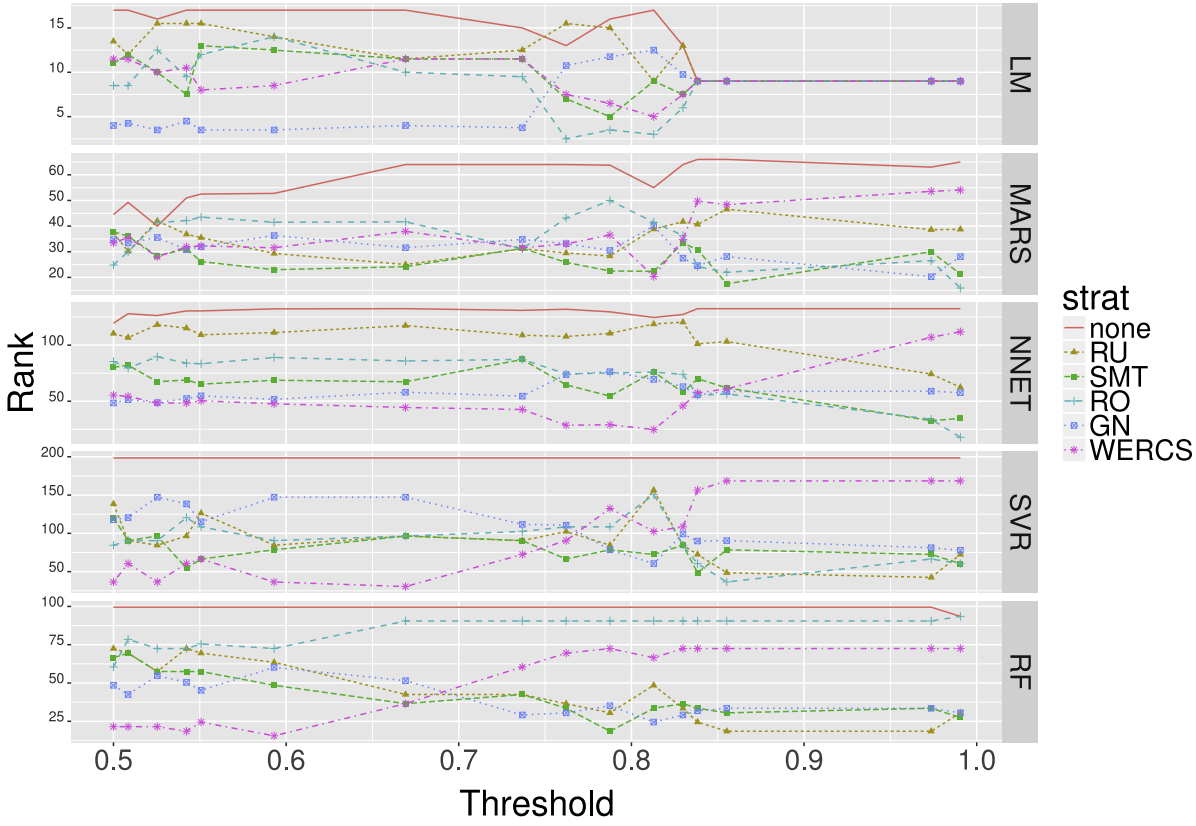


Fig. 13. Trimmed mean rank at 20% of each sampling strategy by learning algorithm on data set a2 (DS8) for relevance thresholds in [0.5,1].

Table 13
Aggregation of $G-Mean^{\phi}$ results from Table 11. Total number of data sets where each pair of sampling strategy-learner had the best results.

	None	RU	SMT	RO	GN	WERCS
LM	4	0	2	1	3	3
MARS	1	1	0	2	6	3
NNET	3	1	1	1	4	3
SVR	2	6	0	1	4	0
RF	1	4	0	2	4	2
Total	11	12	3	7	21	11

Table 14
Aggregation of $spec^{\phi}$ results from Table 12. Total number of data sets where each pair of sampling strategy-learner had the best results.

	None	RU	SMT	RO	GN	WERCS
LM	8	0	1	2	1	3
MARS	4	0	1	3	4	3
NNET	6	1	0	0	3	4
SVR	3	5	2	0	3	0
RF	2	3	0	2	4	2
Total	23	9	4	7	15	12

Table 15
Data sets evaluated with a relevance threshold varying in [0.5,1]. (Nr. Thr: Number of used thresholds; Nr. Rare: number of rare cases; % Rare: rare cases percentage).

ID	Nr. Thr	Threshold = 0.5		Threshold = 1	
		Nr. Rare	%Rare	Nr. Rare	%Rare
DS3	17	42	21.2	23	11.6
DS8	16	34	17.2	13	6.6

We can observe situations where using the original data set has a better performance than applying any of the pre-processing strategies for most of the considered thresholds (MARS on DS3) and we can also observe the opposite situation (SVR on DS8). The results seem to indicate that other characteristics of the data sets may also influence the performance of the pre-processing strategies. This is in accordance with some recent studies on imbalanced classification tasks (e.g. [18]). Still, for most of the considered situations, the application of pre-processing strategies presents advantages in terms of the performance of the learning systems when compared to using the unchanged training data.

6. A case study: predicting NO_2 emissions

Let us consider the LNO_2 emissions data set,[3] that has a continuous target variable (LNO_2). This target variable represents the hourly measured values of the logarithm of the concentration of NO_2 particles in Oslo, Norway, between October 2001 and August 2003.

In this application, the main goal is to predict the outdoor air pollution. High values of LNO_2 represent a bad air quality as opposed to lower LNO_2 values. Let us suppose that a decision maker will use the predicted LNO_2 values for determining when to impose traffic restrictions for preventing a given location from reaching a dangerous atmosphere.

The most frequent LNO_2 values do not represent a dangerous situation. However, for extremely high LNO_2 values (that are rare) the public health may be impacted and therefore it is of

[3] Data available at StatLib Datasets Archive: http://lib.stat.cmu.edu/datasets/.

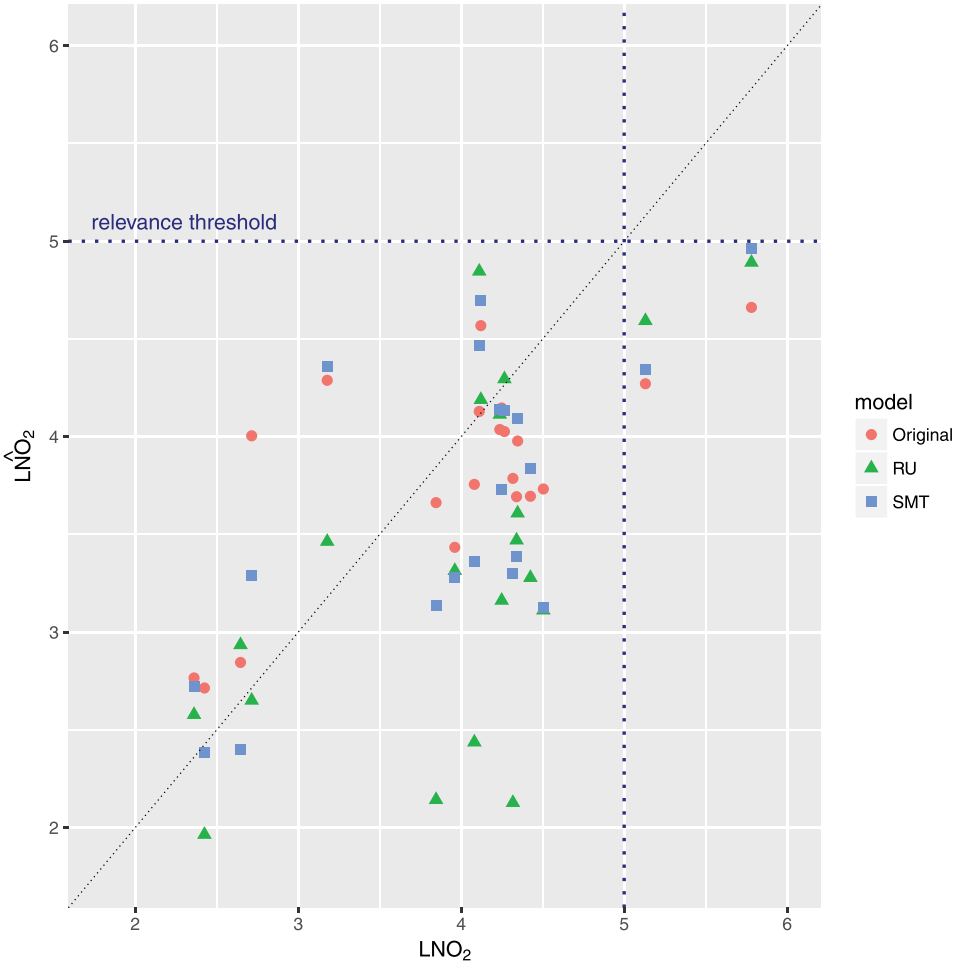


Fig. 14. SVR predictions for LNO_2 data set using the original training set, and pre-processing the training set using random under-sampling (RU) and SmoteR (SMT).

extreme importance to obtain accurate predictions on these particular cases. Given the application context described, we will show how the use of pre-processing strategies may be beneficial and help in biasing a model towards the most relevant cases.

For illustration purposes let us consider a randomly selected subset of 20 examples out of a total of 500 examples. We use this subset as test set and the remaining cases as training set. Let us build a model using this imbalanced training set. We selected a SVR with default parameters. We will consider a simple fixed setting where we use the default parameters of the learner to observe the impact in the predictions. The predictions obtained using this base model on the test set are displayed in Fig. 14 with the label "Original". This figure also displays the predictions obtained when the model is learned on a training set that was modified by applying two previously proposed resampling strategies: random under-sampling or SmoteR. In this case, neither random under-sampling nor SmoteR strategies are able to detect the two high values of LNO_2. In effect, the predictions obtained for the two higher target variable values using the pre-processing strategies are closer to the diagonal. However, these predictions are not above the relevance threshold (dashed line) which means they they could not be detected. Still we must highlight that, although not being the most favorable situation, it is noticeable that there is an improvement in the performance on these rare and important cases. Fig. 15 shows the prediction results of models obtained using the original training set and training sets pre-processed through the strategies proposed in this paper: random over-sampling, introduction of Gaussian Noise and **WERCS**. In this case, the models that use pre-processed training sets are capable of detecting one of the two high extreme examples. This means that, for this example, the predicted value is above the relevance threshold, and therefore, is predicted as a relevant case. The overall increase that is reflected by standard error metrics is not important for this application. In fact, what is important is the prediction of the higher values of LNO_2 which is accomplished in a more satisfactory way when applying pre-processing strategies to the original training set. Table 16 shows the F_1^{ϕ}, $G-Mean^{\phi}$, $prec^{\phi}$, rec^{ϕ}, $spec^{\phi}$, MAD (Mean Absolute deviation) and MSE (Mean Squared Error) results of the different models on the test set. We must highlight that this is simply an illustration of the impact of using pre-processing strategies to handle imbalanced regression problems.

Table 16
Performance estimates on LNO_2 data set for the models obtained by using the original training set and or the training set resulting from each of the pre-processing strategies.

Strategy	F_1^{ϕ}	$G-Mean^{\phi}$	$prec^{\phi}$	rec^{ϕ}	$spec^{\phi}$	MSE	MAD
Original	0	0.828	0	0.785	0.874	7.862	10.359
RU	0	0.864	0	0.838	0.890	18.619	15.115
SMT	0	0.854	0	0.821	0.888	9.696	11.987
RO	0.895	0.889	0.924	0.867	0.912	11.715	13.224
GN	0.899	0.897	0.923	0.876	0.919	7.915	10.805
WERCS	0.888	0.878	0.928	0.851	0.906	9.554	11.541

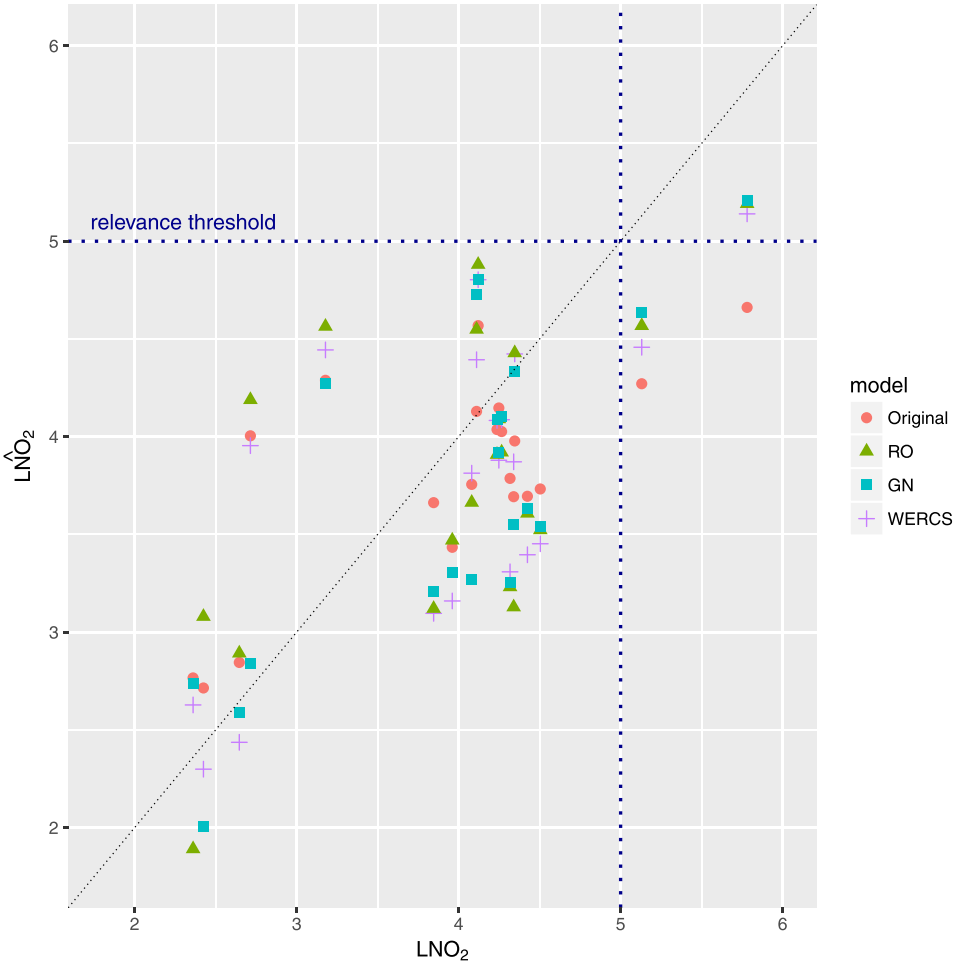


Fig. 15. SVR predictions for LNO_2 data set using the original training set, and pre-processing the training set using random over-sampling (RO), introduction of Gaussian Noise (GN) and **WERCS**.

7. Conclusions

This paper presents three new pre-processing approaches to tackle the problem of forecasting rare values of a continuous target variable. This type of strategies has the advantage of allowing the use of standard regression learning algorithms by simply manipulating the available training data. The performance obtained using our proposed strategies is very competitive on imbalanced regression tasks. Namely, the novel **WERCS** algorithm has shown considerable advantages regarding the number of wins and significant wins against the baseline of not using any sampling strategy. Moreover, this method is more user friendly than existing competitors because it does not require that the user sets a threshold on the relevance values. Additionally, **WERCS** is also computationally more efficient as it does not involve the generation of synthetic examples.

The key contributions of this paper are: (i) the proposal of a novel method (**WERCS**) for addressing the problem of imbalanced regression; (ii) the adaptation to regression tasks of two approaches developed for imbalanced classification (**RO** and **GN**); (iii) the experimental comparison of our proposals against the baseline of not performing any type of sampling and the two existing algorithms for this type of problems (random under-sampling and SmoteR); (iv) to provide a repository with 15 imbalanced regression data sets; (v) the exploration of the impact in the predictive performance of different data characteristics; and (vi) the study of the effect of changing the relevance threshold on the predictive performance of the pre-processing strategies for each learning algorithm.

Several aspects of imbalanced regression problems remain to be explored. Among them, we consider the following as the most important and interesting: (i) develop a mechanism that can automatically tune the parameters for the pre-processing methods; (ii) develop an automatic method that is able to determine which is the best pre-processing method to apply given the data set and the user preferences; (iii) explore the combination of pre-processing techniques with ensemble methods for the problem of imbalanced regression; and (iv) study the relationship between domain characteristics and the impact on performance of the pre-processing strategies. The repository with the 15 data sets used is available at https://paobranco.github.io/DataSets-IR/. For reproducibility purposes, all the figures, data and code used in these experiments is available in https://github.com/paobranco/Pre-processingApproachesImbalanceRegression.

Acknowledgments

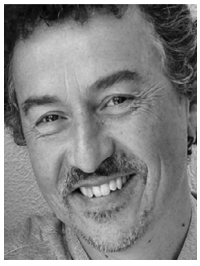
This work is financed by the ERDF European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation – COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013. Project “Coral – Sustainable Ocean

Exploitation: Tools and Sensors/NORTE-01-0145-FEDER-000036" is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF). Paula Branco is supported by a Ph.D. scholarship of FCT (PD/BD/105788/2014).

References

[1] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM SIGKDD Explor. Newslett. 6 (1) (2004) 20–29.
[2] P. Branco, Re-sampling Approaches for Regression Tasks under Imbalanced Domains, Department of Computer Science, Faculty of Sciences, University of Porto, 2014 Master's thesis.
[3] P. Branco, R.P. Ribeiro, L. Torgo, UBL: an R package for utility-based learning, 2016a. arXiv: 1604.08079.
[4] P. Branco, L. Torgo, R.P. Ribeiro, A survey of predictive modeling on imbalanced domains, ACM Comput. Surv. 49 (2) (2016b) 31.
[5] F. Charte, A.J. Rivera, M.J. del Jesus, F. Herrera, Addressing imbalance in multilabel classification: measures and random resampling algorithms, Neurocomputing 163 (2015) 3–16.
[6] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, JAIR 16 (2002) 321–357.
[7] S. Daskalaki, I. Kopanas, N.M. Avouris, Evaluation of classifiers for an uneven class distribution problem, Appl. Artif. Intell. 20 (5) (2006) 381–417.
[8] J. Demšar, Statistical comparisons of classifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.
[9] E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, A. Weingessel, e1071: Misc Functions of the Department of Statistics (e1071), TU Wien, 2011.
[10] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[11] H. He, Y. Ma, Imbalanced Learning: Foundations, Algorithms, and Applications, John Wiley & Sons, 2013.
[12] N. Japkowicz, S. Stephen, The class imbalance problem: a systematic study, Intell. Data Anal. 6 (5) (2002) 429–449.
[13] B. Krawczyk, Learning from imbalanced data: open challenges and future directions, Prog. Artif. Intell. 5 (4) (2016) 221–232.
[14] M. Kubat, S. Matwin, Addressing the curse of imbalanced training sets: one-sided selection, in: Proceedings of the 14th International Conference on Machine Learning, ICML, Morgan Kaufmann, 1997, pp. 179–186.
[15] S.S. Lee, Regularization in skewed binary classification, Comput. Stat. 14 (2) (1999) 277.
[16] S.S. Lee, Noisy replication in skewed binary classification, Comput. Stat. Data Anal. 34 (2) (2000) 165–191.
[17] A. Liaw, M. Wiener, Classification and regression by randomforest, R News 2 (3) (2002) 18–22.
[18] V. López, A. Fernández, S. García, V. Palade, F. Herrera, An insight into classification with imbalanced data: empirical results and current trends on using data intrinsic characteristics, Inf. Sci. 250 (2013) 113–141.
[19] S. Milborrow, Earth: multivariate adaptive regression spline models. Derived from MDA: Marsby Trevor Hastie and Rob Tibshirani, 2012. R package version. 2014;3:2-7.
[20] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2015.
[21] R. Ranawana, V. Palade, Optimized precision-a new measure for classifier performance evaluation, in: Proceedings of IEEE Congress on Evolutionary Computation, CEC, IEEE, 2006, pp. 2254–2261.
[22] R.P. Ribeiro, Utility-based Regression, Department of Computer Science, Faculty of Sciences, University of Porto, 2011 Ph.D. thesis.
[23] L. Torgo, An Infra-structure for Performance Estimation and Experimental Comparison of Predictive Models in R, CoRR abs/1412.0436(2014).
[24] L. Torgo, P. Branco, R.P. Ribeiro, B. Pfahringer, Resampling strategies for regression, Expert Syst. 32 (3) (2015) 465–476.
[25] L. Torgo, R. Ribeiro, Utility-based regression, in: Proceedings of Principles and Practice of Knowledge Discovery in Databases, PKDD, 7, Springer, 2007, pp. 597–604.
[26] L. Torgo, R.P. Ribeiro, Precision and recall in regression, in: Proceedings of the 12th International Conference on Discovery Science, DS'09, Springer, 2009, pp. 332–346.
[27] L. Torgo, R.P. Ribeiro, B. Pfahringer, P. Branco, Smote for regression, in: Progress in Artificial Intelligence, Springer, 2013, pp. 378–389.
[28] W.N. Venables, B.D. Ripley, Modern Applied Statistics with S, 4th ed., Springer, New York, 2002.
[29] G.M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explor. Newslett. 6 (1) (2004) 7–19.

Paula Branco is a Ph.D. student in MAPi Doctoral Programme at the Department of Computer Science of the Faculty of Sciences - University of Porto, Portugal, under the supervision of Luís Torgo and Rita Ribeiro. She is a researcher at LIAAD – INESC TEC. Her research interests include machine learning, data mining, utility-based learning and extreme values forecast. She obtained a degree in Mathematics and a Masters degree in Computer Science both from the Faculty of Sciences, University of Porto.



Luís Torgo is a Professor of Computer Science at the Faculty of Computer Science of the Dalhousie University, Canada, an Associate Professor of the Department of Computer Science of the Faculty of Sciences of the University of Porto, Portugal. He is a senior researcher of LIAAD/INESC Tec, and a current member of the board of this research lab. Luis Torgo has been doing research in the area of Data Mining and Machine Learning since 1990, and has published over 100 papers in several forums of these areas. Luis Torgo is the author of the widely acclaimed Data Mining with R book published by CRC Press in 2010 with a strongly revised second edition that appeared in January of 2017. He has been involved in many research projects under different roles and involving different types of organizations. His current broad research interests revolve around analyzing data from dynamic environments, with a particular focus on time and space-time dependent data sets, in the search for unexpected events. In terms of application domains his research is frequently linked with ecological/biological as well as financial domains. Luis Torgo is the CEO and one of the founding partners of KNOYDA a company devoted to training and consulting within data science.

Rita P. Ribeiro received her Ph.D. degree in Computer Science from the University of Porto, Portugal in 2011. She is an assistant professor at the Department of Computer Science of the Faculty of Sciences of the University of Porto and member of LIAAD-INESC TEC, the Artificial Intelligence and Decision Support Lab of University of Porto. Her main research interests include Data Mining and Machine Learning, in particular outlier detection, novelty detection, utility-based learning and evaluation issues on learning tasks. As a member of LIAAD-INESC TEC, she has been involved in several research projects concerning environmental applications, fraud detection and fault diagnosis.