REBAGG: REsampled BAGGing for Imbalanced Regression

Paula Branco PAULA.BRANCO@DCC.FC.UP.PT
LIAAD-INESC TEC DCC-FCUP, University of Porto Porto, Portugal

Luís Torgo LTORGO@DAL.CA
Faculty of Computer Science - Dalhousie University, Halifax, Canada
LIAAD-INESC TEC DCC-FCUP, University of Porto Porto, Portugal

Rita P. Ribeiro RPRIBEIRO@DCC.FC.UP.PT
LIAAD-INESC TEC DCC-FCUP, University of Porto Porto, Portugal

Editors: Luís Torgo, Stan Matwin, Nathalie Japkowicz, Bartosz Krawczyk, Nuno Moniz, and Paula Branco

Abstract

The problem of imbalanced domains is important in multiple real world applications. This problem has been thoroughly studied for classification tasks. In particular, the adaptation of ensembles to tackle imbalanced domains has shown important advantages in a classification context. Still, for imbalanced regression problems only a few solutions exist. Moreover, the capabilities of ensembles for dealing with imbalanced regression tasks are yet to be explored. In this paper, we present the REsampled BAGGing (REBAGG) algorithm, a bagging-based ensemble method that incorporates data pre-processing strategies for addressing imbalanced domains in regression tasks. The extensive experimental evaluation conducted shows the advantage of our proposal in a diverse set of domains and learning algorithms.

1. Introduction

Several real world applications involve learning from imbalanced domains. Although being a problem more studied in a classification context, other tasks, such as regression, data streams or multi-label, also suffer from this problem (Krawczyk, 2016; Branco et al., 2016b). In imbalanced regression tasks, the user is more interested in being accurate in a subset of the continuous target variable that, although being more important, is underrepresented in the available data. For instance, in a financial context, when predicting the return of an asset the high and low values are typically scarce. However, in this case, it is particularly important for an agent to obtain accurate forecasts both in the high and low values because those values can lead to either heavy losses or large missed profits.

The research community has been working intensively in the problem of imbalanced domains for over two decades. Still, the majority of solutions proposed are concentrated in the problem of class imbalance. Recently, some attention has been given also to imbalanced regression tasks although this problem is still an open challenge (Krawczyk, 2016). This paper addresses imbalanced regression tasks.

Over the past years several different aspects of the problem of learning from imbalanced domains have been addressed (He and Ma, 2013). One essential challenge is related with

© 2018 P. Branco, L. Torgo & R.P. Ribeiro.

performance evaluation. In these domains, the use of traditional performance assessment metrics is not recommended as they fail to capture what is relevant to the user (Ribeiro, 2011). Therefore, we need to use evaluation metrics that are suitable for imbalanced regression problems. Another major challenge is the inability of standard learners to focus on the most important and rare cases. Typically, learning algorithms focus on the most frequent cases, exhibiting a poor predictive accuracy on the rare and most interesting cases for the end-user of these applications. To deal with this issue four types of methods were put forward: data pre-processing, development of special purpose learners, prediction post-processing or hybrid methods (Branco et al., 2016b).

One approach that produced promising results for solving the class imbalance problem, is the use of ensemble methods together with data pre-processing strategies (e.g. Liu et al. (2009)). In general, these approaches aim at training an ensemble method where the diversity among the members of the ensemble is achieved through the use of different data samples obtained by a given pre-processing strategy. For a more complete review on ensemble methods for imbalanced classification see Galar et al. (2012).

The success of the use of ensemble methods with pre-processing strategies in imbalanced binary classification tasks led to its extension to other tasks such as multiclass (Lango and Stefanowski, 2018). Also the use of bagging-based strategies has show advantages in pursuing this direction (e.g. Khoshgoftaar et al. (2011); Błaszczyński and Stefanowski (2015)). Still, no similar attempt has been made for tackling imbalanced regression tasks. This motivated the work we present in this paper whose goal is to study the incorporation of pre-processing strategies with ensemble methods in imbalanced regression tasks. In particular, we describe the REBAGG (REsampled BAGGing) algorithm. This method integrates data pre-processing strategies with bagging. REBAGG is able to generate a diverse set of models by taking advantage of different ways of resampling the training data. We show that our proposal is effective for tackling imbalanced regression problems when using different base learners and in a diversity of domains. The main contributions of our work are as follows: i) we propose the first ensemble method for tackling the problem of imbalanced regression; and ii) we demonstrate the advantage of our algorithm in a diversity of domains and for multiple learning algorithms.

This paper is organised as follows. In Section 2 the problem definition is presented. Section 3 provides an overview of the related work. Our proposal is described in Section 4 and the results of an extensive experimental evaluation are discussed in Section 5. Finally, Section 6 presents the main conclusions.

2. Problem Definition

Standard predictive tasks aim at obtaining a model $m(\mathbf{x})$ that provides a good approximation of an unknown function $Y = f(\mathbf{x})$ that maps a set of p features into the values of the target variable Y with domain $\mathcal{Y}$. This goal is achieved by using a training set $\mathcal{D} = \{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{N}$ with N examples. When the target variable is nominal, the predictive task is called a classification task, and when we have a numeric target variable, we face a regression problem.

Imbalanced regression tasks are a special class of regression tasks. This type of problems can be described by the two following assertions: i) the user assigns non-uniform prefer-

ences across the target variable domain $\mathcal{Y}$; and ii) the most important cases are scarcely represented in the available training data. Regarding the first assertion, this means that the predictive performance of a model m has a different importance for the user on different locations of the target variable domain. The second assertion concerns the lack of representation in the available training set $\mathcal{D}$ of the most important ranges of the target variable. The conjugation of these two factors is responsible for a degradation in the models performance on the most important cases for the user. In fact, standard learning algorithms are ineffective in this context because they are unable to focus on the most important and rare cases (Ribeiro, 2011; Branco et al., 2016b).

For solving the problem of defining the numeric target variable importance, Torgo and Ribeiro (2007) and Ribeiro (2011) proposed the concept of a **relevance function**, $\phi : \mathcal{Y} \rightarrow [0, 1]$. This function maps the variable domain into a scale of relevance, where 1 corresponds to maximal relevance and 0 to the minimum relevance. Ideally, this function should be defined by domain experts that have the necessary background knowledge to precisely quantify the mapping of the variable domain into a relevance scale. Still, there are several difficulties related with domain experts: i) often there are no domain experts available, ii) they represent an high investment, and iii) they require a considerable amount of time to convert domain knowledge into a **relevance function**. Given these difficulties, Ribeiro (2011) presented an automatic method for obtaining this function. This method is based on the assumption that higher levels of rarity correspond to the most interesting ranges of the domain, which is the most usual setting. Moreover, the automatic method also assumes that the rare and interesting values of the target variable are concentrated on the extremes of the distribution which is also a common setting. Function $\phi(y)$ is estimated using the target variable sample distribution by assigning more relevance to the rare and most extreme cases. This is achieved by using the quartiles and the inter-quartile range of the target variable estimated from the training data (further details available in Ribeiro (2011)).

Using the **relevance function** and a user defined threshold on the relevance values, t_R, we are able to define two disjunct subsets of $\mathcal{D}$: the set containing the rare and important values, $\mathcal{D}_R$, and the set $\mathcal{D}_N$ with the normal and uninteresting values. Formally these sets are defined as follows: $\mathcal{D}_R = \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) \geq t_R\}$ and $\mathcal{D}_N = \{\langle \mathbf{x}, y \rangle \in \mathcal{D} : \phi(y) < t_R\}$.

When handling imbalanced domains, it is very important to consider suitable evaluation measures. Standard evaluation metrics were shown to be unsuitable, potentially leading to incorrect conclusions concerning the models expected performance (e.g. He and Ma (2013); Ribeiro (2011)). Therefore, it is necessary to use adequate measures when assessing the performance of imbalanced domains. This issue was addressed for both classification and regression tasks. Torgo and Ribeiro (2009) and Ribeiro (2011) proposed a utility framework for obtaining precision and recall for regression tasks that is able to capture the key features of precision and recall in classification problems but is also capable to take into account the magnitude of the errors that is important in regression. Based on the utility framework (Torgo and Ribeiro, 2009; Ribeiro, 2011), Branco (2014) proposed the following definitions of precision ($prec^{\phi}$) and recall (rec^{ϕ}) for imbalanced regression tasks:

$$prec^{\phi} = \frac{\sum_{\phi(\hat{y}_i) > t_R} (1 + u(\hat{y}_i, y_i))}{\sum_{\phi(\hat{y}_i) > t_R} (1 + \phi(\hat{y}_i))} \quad (1) \qquad rec^{\phi} = \frac{\sum_{\phi(y_i) > t_R} (1 + u(\hat{y}_i, y_i))}{\sum_{\phi(y_i) > t_R} (1 + \phi(y_i))} \quad (2)$$

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 signalling the cases that are relevant for the user, and $u(\hat{y}_i, y_i)$ is the normalised utility of predicting $\hat{y}_i$ for the true value y_i, as proposed by Torgo and Ribeiro (2009) and Ribeiro (2011).

In this paper we use as main evaluation metric the F_1-measure adapted for regression tasks, F_1^ϕ (cf. Equation 3), that depends on $prec^\phi$ and rec^ϕ measures (cf. Equations 1 and 2).

$$F_1^\phi = \frac{2 \cdot prec^\phi \cdot rec^\phi}{prec^\phi + rec^\phi} \tag{3}$$

3. Related Work

Ensembles methods are techniques that involve building several different models that are combined using a certain aggregation strategy. Ensembles main goal is to provide more accurate predictions when compared with the use of a single model (Zhou, 2012). These methods have shown this ability in a diversity of real world problems and competitions (e.g. Netflix Competition (Koren, 2009), KDD-Cup (Yu et al., 2010)). We use the term ensemble to refer to methods that combine different hypothesis generated by a selected base learning algorithm. For obtaining ensembles that are accurate it is important to have diversity in the models that compose it. In regression problems, ensembles diversity has been thoroughly studied in the well-known "bias-variance" (Ueda and Nakano, 1996) and "ambiguity" decomposition (Krogh and Vedelsby, 1994).

Ensemble methods by themselves are not sufficient to tackle the problem of imbalanced domains. In fact, they are not able to overcome the problem that each single model suffers: focusing on the average cases and neglecting the rare and more important cases (Galar et al., 2012). However, their combination with other strategies for addressing imbalanced domains has shown positive results in classification tasks. Namely, the integration of ensembles with data pre-processing strategies has shown good results when dealing with the class imbalance problem. Regarding imbalanced classification, a large number of proposals address this problem using ensemble methods (e.g. Liu et al. (2009); Błaszczyński and Stefanowski (2015)). We refer the interested reader to a survey dedicated to this particular issue (Galar et al., 2012). However, in a regression context, as far as we know, no attempt has been made to combine ensemble methods with other strategies designed for imbalance domains. Only one study was conducted for assessing the impact in the performance of standard ensemble methods (Moniz et al., 2017).

In this paper we are focused on bagging-based ensembles. Bagging (bootstrap aggregating) was proposed by Breiman (1996) and consists of building models using bootstrap samples of the original training data. These methods require two main steps: i) the generation of k different models using bootstrap samples of the training set, and, ii) the aggregation of the models predictions. The latter step is typically achieved through averaging the prediction in regression problems. Algorithm 1 displays the standard bagging method. More detailed information regarding bagging can be found in Kuncheva (2004).

Input: $\mathcal{D}$ - original regression data set
k - size of the bootstrap samples
m - number of models to train
$\mathcal{L}$ - learning algorithm
$\mathcal{L}.pars$ - learning algorithm parameters
Output: Predictions for a *Test* set
Learning Phase
for $i \leftarrow 1$ ***to*** m **do**
  $S_i \leftarrow$ bootstrap sample of $\mathcal{D}$ of size k
  $M_i \leftarrow$ model trained using S_i data and applying algorithm $\mathcal{L}$ with parameters $\mathcal{L}.pars$
end
Prediction Phase
foreach $x_j \in Test$ **do**
  $M^*(x_j) = \frac{\sum_{i=1}^{m} M_i(x_j)}{m}$; // Aggregation through models averaging
end
return $M^*(x_j), \forall x_j \in Test$

Algorithm 1: Standard Bagging Algorithm (BAGG).

4. Bagging-based Strategies for dealing with Imbalanced Regression Tasks

In this section, we describe our proposal regarding the integration of bagging-based ensembles with data pre-processing methods. The diversity is a key aspect in ensemble learning. Therefore, we propose an algorithm that allows to obtain diversity on the generated models while simultaneously biasing them towards the least represented and more important cases. We propose the REsampled BAGGing (REBAGG) algorithm, which works in two main steps: i) build a number of models using pre-processed samples of the training set; and ii) use the trained models to obtain predictions on unseen data by applying an averaging strategy. Algorithm 2 describes the REBAGG algorithm pseudocode.

Regarding the first step, we developed four main types of resampling methods to apply on the original training set: balance, balance.SMT, variation, and variation.SMT. The key distinguishing feature of these methods is related with: i) the ratio between the number of rare and normal cases used in the new sample; and, ii) how new rare cases are obtained. On the resampling methods labelled with the prefix "balance", the new modified training set will have the same number of rare and normal cases. On the other hand, for resampling methods with the prefix "variation", the ratio of rare to normal cases in the new training set will vary. Specifically, a ratio is randomly selected among 5 possible choices $(1/3, 2/5, 1/2, 3/5, 2/3)$. The selected ratio is used as the target percentage of rare cases to include in the new training set while the remaining cases are obtained from the set of normal examples. This provides a higher diversity in the modified training sets which will include samples that can be either balanced, or more favourable to the rare or normal cases. Another important issue concerns the new rare cases that are added in some strategies. When the resampling method has no suffix appended, then the new cases are obtained by using exact copies of randomly selected rare cases. However, it is possible to use different approaches to add new synthetic cases. We use the SmoteR algorithm (Torgo et al., 2013) to generate new synthetic cases rare cases in the methods that have as suffix "SMT".

The use of the previously described resampling methods allow to obtain new training sets that are capable of both biasing the learners towards the rare and important cases and

Input: $\mathcal{D}$ - original regression data set
k - size of the bootstrap samples
m - number of models to train
$\mathcal{L}$ - learning algorithm
$\mathcal{L}.pars$ - learning algorithm parameters
$\phi()$ - a relevance function
t_R - threshold on the relevance values
$resamp$ - the resampling method for obtaining the data samples
Output: Predictions for a *Test* set
Learning Phase
$D_N \leftarrow \{< \mathbf{x}, y > \in \mathcal{D} : \phi(y) < t_R\}$
$D_R \leftarrow \{< \mathbf{x}, y > \in \mathcal{D} : \phi(y) \geq t_R\}$
for $i \leftarrow 1$ ***to*** m; // Build biased resampled samples
do
  if *resamp = "balance"* **then**
    $R_i \leftarrow$ bootstrap sample of D_R with size $k/2$
    $N_i \leftarrow$ bootstrap sample of D_N with size $k/2$
  else if *resamp = "balance.SMT"* **then**
    $R_i \leftarrow D_R \bigcup \{$ new cases generated with SmoteR until $|R_i| = k/2\}$
    $N_i \leftarrow$ bootstrap sample of D_N with size $k/2$
  else if *resamp = "variation"* **then**
    $p \leftarrow$ `SAMPLE` $(1, \{1/3, 2/5, 1/2, 3/5, 2/3\})$
    $R_i \leftarrow$ bootstrap sample of D_R with size $p \times |D_R|$
    $N_i \leftarrow$ bootstrap sample of D_N with size $k - p \times |D_R|$
  else if *resamp = "variation.SMT"* **then**
    $p \leftarrow$ `SAMPLE` $(1, \{1/3, 2/5, 1/2, 3/5, 2/3\})$
    if $nRare \leq |D_R|$ **then**
      $R_i \leftarrow$ sample of D_R without replacement with size $p \times |D_R|$
    else
      $R_i \leftarrow D_R \bigcup \{$new cases generated with SmoteR until $|R_i| = p \times |D_R|\}$
    end
    $N_i \leftarrow$ bootstrap sample of D_N with size $k - p \times |D_R|$
  end
  $S_i \leftarrow R_i \bigcup N_i$
  $M_i \leftarrow$ model trained with algorithm $\mathcal{L}$ with parameters $\mathcal{L}.pars$ using S_i data
end
Prediction Phase
foreach $x_j \in Test$ **do**
  $M^*(x_j) = \frac{\sum_{i=1}^{k} M_i(x_j)}{k}$; // Aggregation through models averaging
end
return $M^*(x_j), \forall x_j \in Test$

Algorithm 2: REBAGG Algorithm.

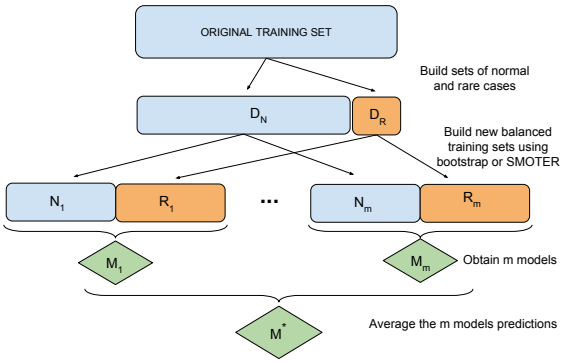

Figure 1: REBAGG with balanced resampled train sets.

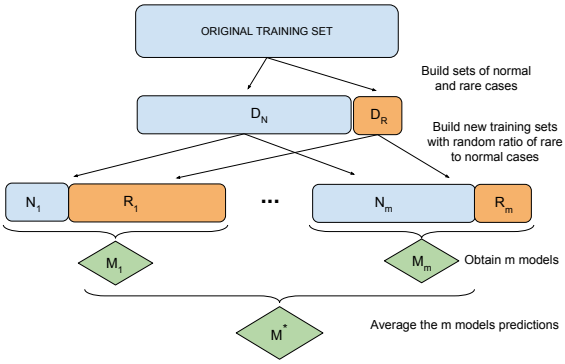

Figure 2: REBAGG with train sets of varying ratio of normal to rare cases.

generating diversity in the ensemble models. Regarding the models aggregation strategy, we implemented a simple models' predictions averaging for obtaining the final predictions. Figure 1 shows REBAGG algorithm with new balanced train sets, and Figure 2 displays the REBAGG strategy that builds new train sets with varying ratio of rare to normal cases.

5. Experimental Evaluation

This section presents the experimental evaluation carried out and a discussion of the main results. To enable an easy replication of our work, the data, the experiments code and the results are available at https://github.com/paobranco/REBAGG. We have used the free open source R environment (R Core Team, 2018) in our experiments. The main goal of our experiments is to assess the effectiveness of REBAGG algorithm in the context of imbalanced regression problems.

Table 1: Data sets information by descending order of rare cases percentage. (N: nr cases; *tpred*: nr predictors; *p.nom*: nominal predictors; *p.num*: numeric predictors; $nRare$: nr. cases with $\phi(y) > 0.8$; $\%Rare$: $nRare/N \times 100$).

Data Set	N	tpred	p.nom	p.num	nRare	% Rare	Data Set	N	tpred	p.nom	p.num	nRare	% Rare
servo	167	4	2	2	34	20.4	a5	198	11	3	8	21	10.7
a6	198	11	3	8	33	16.7	fuelCons	1764	38	12	26	164	9.3
Abalone	4177	8	1	7	679	16.3	availPwr	1802	16	7	9	157	8.7
machCpu	209	6	0	6	34	16.3	cpuSm	8192	13	0	13	713	8.7
a3	198	11	3	8	32	16.2	maxTorq	1802	33	13	20	129	7.2
a4	198	11	3	8	31	15.7	dAiler	7129	5	0	5	450	6.3
a1	198	11	3	8	28	14.1	bank8FM	4499	9	0	9	288	6.4
a7	198	11	3	8	27	13.6	ConcrStr	1030	8	0	8	55	5.3
boston	506	13	0	13	65	12.8	Accel	1732	15	3	12	89	5.1
a2	198	11	3	8	22	11.1	airfoild	1503	5	0	5	62	4.1

Data Sets We selected 20 regression data sets from different domains whose main characteristics are described in Table 1. For each of these data sets we obtained a relevance function through the automatic method proposed in Ribeiro (2011) which assigns higher

Table 2: Base regression learners, parameter variants, and respective R packages.

Learner	Parameter Variants	R package
RPART	$minsplit = \{20, 50, 100, 200\}, cp = \{0.01, 0.05\}$	`rpart` Therneau et al. (2017)
MARS	$nk = \{10, 17\}, degree = \{1, 2\}, thresh = \{0.01, 0.001\}$	`earth` Milborrow (2012)
SVM	$cost = \{10, 150, 300\}, gamma = \{0.01, 0.001\}$	`e1071` Dimitriadou et al. (2011)
RF	$mtry = \{5, 7\}, ntree = \{500, 750, 1500\}$	`randomForest` Liaw and Wiener (2002)
GBM	$distribution = gaussian, n.trees = \{300, 450, 600\}$, $shrinkage = \{0.01, 0.1\}, interaction.depth = \{1, 2\}$	`gbm` with contributions from others (2017)

relevance to high and low extreme values of the target variable. We considered a relevance threshold of 0.8 in all data sets.

Learning Algorithms All experiments were carried out in the R environment and we tested the five following types of learning algorithms: regression trees (RPART), Multivariate Adaptive Regression Splines (MARS), Support Vector Machines (SVM), Random Forests (RF) and Generalized Boosted Regression Models (GBM). The learning algorithms, respective R packages and the used parameter variants are displayed in Table 2. We tested the original learning algorithms, as well as the original BAGG algorithm and our proposed REBAGG approach each using all the mentioned learners as base learners. Two variants of BAGG were tested considering $m \in \{10, 40\}$. The REBAGG approach was tested for the following parameter variants: $m \in \{10, 40\}$, $resamp \in \{balance, balance.SMT, variation, variation.SMT\}$, k equal to original training set size, $t_R = 0.8$. REBAGG uses internally the SmoteR algorithm for the generation of synthetic cases. In this algorithm we used the following parameters: HEOM distance, 5 nearest neighbours. Therefore, 8 variants of REBAGG were tested.

Results and Discussion We applied each of the 40 learning approaches (8 RPART + 8 MARS + 6 SVM + 6 RF + 12 GBM) to each of the 20 regression data sets. We tested the original learning approaches, the two variants of BAGG algorithm and 8 variants of REBAGG approach. Thus 8800 ($40 \times 20 \times 11$) combinations were tested. We appended to REBAGG an abbreviation representing the *resamp* parameter used (B for balance, V for variation, B.SMT for balance with SmoteR , and V.SMT for variation with SmoteR).

The performance was evaluated using the F_1^ϕ measure for regression whose values were estimated by 2 repetitions of a 10-fold stratified cross validation process and the statistical significance of the observed paired differences was measured using the non-parametric Friedman Test and the post-hoc Nemenyi Test as recommended by Demšar (2006). The experiments were carried out using the following R packages: `performanceEstimation` (Torgo, 2014) for the experimental infra-structure; `uba` (available at `http://www.dcc.fc.up.pt/~rpribeiro/uba/`) for the relevance function and F_1^ϕ metric; and `UBL` (Branco et al., 2016a) for the implementation of SmoteR algorithm and REBAGG strategies. Tables 3 to 7 show the average F_1^ϕ results obtained for each learning algorithm by data set and strategy applied. Globally, the advantage of using REBAGG algorithm is clear for all learners. Only for RPART learner, our proposal presents a less striking advantage.

To assess the statistical significance of the differences observed we applied the Friedman test which allowed to reject the null hypothesis that the performance of the different algorithms was equivalent. We then proceeded to the post-hoc Nemenyi test. These results are displayed by the CD diagrams proposed by Demšar (2006) in Figures 3 and 4 for a significance levels of 0.05. Figures 3 and 4 show the CD diagrams of the base learners, BAGG

Table 3: RPART average F_1^ϕ results by data set and strategy (bold represents the best performance by data set).

		m = 10					**m = 40**				
			REBAGG					**REBAGG**			
	NONE	**BAGG**	**B**	**V**	**B.SMT**	**V.SMT**	**BAGG**	**B**	**V**	**B.SMT**	**V.SMT**
servo	0.413	0.446	0.576	0.574	0.571	0.575	0.445	**0.578**	**0.578**	0.576	0.576
a6	0.269	0.362	0.535	0.531	0.535	0.543	0.335	0.536	0.536	0.541	**0.544**
Abalone	**0.701**	0.341	0.679	0.678	0.676	**0.678**	0.338	0.678	0.680	0.677	**0.678**
machCpu	0.458	0.512	0.660	0.665	0.660	0.658	0.526	**0.666**	0.665	0.662	0.660
a3	0.213	0.163	0.533	0.536	0.534	0.532	0.149	0.537	0.536	0.536	**0.538**
a4	0.367	0.344	0.583	0.581	0.594	0.585	0.320	0.583	0.582	0.597	**0.599**
a1	0.151	0.236	0.546	0.554	0.543	0.544	0.231	0.551	**0.555**	0.549	0.546
a7	0.232	0.184	0.384	0.390	0.407	0.404	0.190	0.394	0.393	**0.414**	0.412
boston	0.761	0.833	0.868	0.868	0.852	0.863	0.842	**0.871**	**0.871**	0.854	0.863
a2	0.122	0.061	0.424	0.425	0.425	**0.434**	0.033	0.419	0.430	0.424	0.426
a5	0.099	0.035	0.404	**0.423**	0.405	0.409	0.028	0.410	0.412	0.415	0.409
fuelCons	**0.818**	0.713	0.778	0.779	0.771	0.780	0.726	0.783	0.782	0.773	0.779
availPwr	**0.868**	0.849	0.853	0.856	0.840	0.844	0.852	0.855	0.858	0.840	0.844
cpuSm	0.497	**0.567**	0.381	0.386	0.387	0.389	0.564	0.383	0.385	0.388	0.392
maxTorq	**0.893**	0.881	0.846	0.852	0.852	0.859	0.887	0.847	0.852	0.854	0.860
dAiler	0.710	0.340	0.710	0.723	0.711	0.722	0.338	0.710	0.722	0.712	**0.724**
bank8FM	**0.916**	0.854	0.845	0.851	0.866	0.870	0.875	0.846	0.852	0.868	0.874
ConcrStr	0.418	0.095	0.854	0.861	0.850	**0.864**	0.073	0.857	0.862	0.853	0.863
Accel	**0.875**	0.822	0.854	0.855	0.854	0.859	0.863	0.854	0.856	0.852	0.859
airfoild	0.108	0.019	0.187	0.187	0.204	0.206	0.018	0.190	0.192	**0.207**	0.205
Mean±sd	0.494±0.3	0.433±0.3	0.625±0.2	0.629±0.2	0.627±0.2	**0.631±0.2**	0.432± 0.32	0.627±0.2	0.630±0.2	0.630±0.2	**0.633±0.2**

Table 4: MARS average F_1^ϕ results by data set and strategy.

		m = 10					**m = 40**				
			REBAGG					**REBAGG**			
	NONE	**BAGG**	**B**	**V**	**B.SMT**	**V.SMT**	**BAGG**	**B**	**V**	**B.SMT**	**V.SMT**
servo	0.645	0.659	0.670	0.672	0.667	0.667	0.656	0.670	**0.673**	0.669	0.669
a6	0.459	0.484	0.548	0.543	0.549	0.550	0.472	**0.554**	0.547	0.547	0.548
Abalone	0.708	0.712	**0.739**	0.738	0.738	0.737	0.713	**0.739**	**0.739**	0.738	0.738
machCpu	0.797	0.786	0.784	0.791	0.795	**0.799**	0.778	0.787	0.782	0.793	**0.799**
a3	0.498	0.430	0.547	0.553	0.556	**0.558**	0.441	0.551	0.551	0.549	**0.558**
a4	0.481	0.475	0.568	0.557	0.580	0.568	0.515	0.563	0.562	**0.584**	0.580
a1	0.573	0.550	0.723	0.721	0.734	0.734	0.580	0.727	0.735	**0.737**	0.735
a7	0.294	0.305	0.351	0.354	0.377	0.377	0.308	0.365	0.364	**0.381**	0.374
boston	0.894	0.895	0.892	0.890	0.894	0.892	**0.898**	0.894	0.894	0.894	0.893
a2	0.260	0.235	0.532	0.529	0.537	0.546	0.223	**0.547**	0.546	0.539	0.545
a5	0.146	0.260	0.540	0.543	0.544	0.532	0.163	0.548	**0.551**	0.538	0.538
fuelCons	0.853	0.859	**0.877**	0.875	0.870	0.871	0.855	0.875	0.872	0.870	0.871
availPwr	0.902	0.904	0.907	**0.908**	0.903	0.905	0.904	0.905	0.904	0.902	0.902
cpuSm	0.142	0.142	0.169	**0.174**	0.173	0.173	0.144	0.170	**0.174**	0.173	**0.174**
maxTorq	0.954	0.962	0.974	0.975	0.970	0.968	0.963	**0.976**	**0.976**	0.971	0.969
dAiler	0.736	0.733	0.756	**0.760**	0.756	0.758	0.734	0.756	0.758	0.756	0.757
bank8FM	0.943	0.945	0.948	**0.949**	0.948	**0.949**	0.945	0.948	**0.949**	0.948	**0.949**
ConcrStr	0.886	0.887	0.901	0.901	0.900	0.900	0.888	0.903	**0.904**	0.901	0.902
Accel	0.895	**0.896**	0.893	0.889	**0.896**	0.894	0.889	0.887	0.885	0.885	0.887
airfoild	0.116	0.107	0.199	0.195	0.213	**0.217**	0.105	0.191	0.190	0.210	0.213
Mean±sd	0.609±0.29	0.611±0.29	0.676±0.24	0.676±0.24	**0.680±0.24**	**0.680±0.24**	0.609±0.29	0.678±0.24	0.678±0.24	**0.679±0.24**	0.680±0.24

Table 5: SVM average F_1^ϕ results by data set and strategy.

		m = 10					m = 40				
			REBAGG					REBAGG			
	NONE	BAGG	B	V	B.SMT	V.SMT	BAGG	B	V	B.SMT	V.SMT
servo	0.366	0.385	0.651	0.637	0.648	0.635	0.382	**0.652**	0.642	**0.652**	0.640
a6	0.229	0.258	0.537	0.540	0.547	0.548	0.245	0.538	0.539	**0.549**	0.546
Abalone	0.712	0.712	**0.738**	**0.738**	0.736	0.736	0.712	0.737	**0.738**	0.736	0.737
machCpu	0.780	0.781	0.788	0.791	0.792	0.792	0.780	0.792	**0.795**	0.792	0.791
a3	0.181	0.198	0.542	0.550	0.547	0.549	0.195	0.542	0.547	0.545	**0.551**
a4	0.252	0.327	0.566	0.572	0.584	**0.586**	0.308	0.571	0.572	0.585	0.581
a1	0.113	0.138	0.718	0.710	0.724	0.727	0.150	0.725	0.723	**0.728**	0.725
a7	0.107	0.102	0.333	0.331	**0.355**	0.346	0.103	0.341	0.340	0.353	0.354
boston	0.883	0.885	0.898	0.898	0.898	0.899	0.884	**0.900**	0.899	0.898	0.899
a2	0.232	0.253	0.517	0.516	0.521	0.515	0.230	0.514	0.516	**0.528**	0.520
a5	0.139	0.128	0.554	**0.560**	0.545	0.557	0.117	0.551	**0.560**	0.553	0.558
fuelCons	**0.908**	0.903	0.903	0.904	0.907	0.906	0.903	0.905	0.905	0.907	0.907
availPwr	0.935	0.934	**0.942**	**0.942**	0.940	0.939	0.934	**0.942**	**0.942**	0.939	0.939
cpuSm	0.161	0.158	0.182	0.184	0.183	**0.185**	0.159	0.183	**0.185**	0.182	**0.185**
maxTorq	0.973	0.975	0.978	**0.979**	0.976	0.976	0.976	**0.979**	**0.979**	0.976	0.976
dAiler	0.728	0.727	0.759	**0.761**	0.759	0.760	0.728	0.759	0.760	0.759	0.760
bank8FM	0.947	0.946	**0.950**	**0.950**	**0.950**	**0.950**	0.946	**0.950**	**0.950**	**0.950**	**0.950**
ConcrStr	0.840	0.839	0.906	0.907	**0.912**	**0.912**	0.841	0.907	0.908	**0.912**	**0.912**
Accel	0.872	0.864	0.901	0.903	0.902	0.903	0.871	0.902	0.903	0.902	**0.904**
airfoild	0.158	0.152	0.240	0.238	0.240	0.241	0.154	0.237	0.240	0.242	**0.244**
Mean±sd	0.526±0.35	0.533±0.34	0.680±0.24	0.681±0.24	**0.683±0.24**	**0.683±0.24**	0.531±0.35	0.681±0.24	0.682±0.24	**0.684±0.24**	**0.684±0.24**

Table 6: RF average F_1^ϕ results by data set and strategy.

		m = 10					m = 40				
			REBAGG					REBAGG			
	NONE	BAGG	B	V	B.SMT	V.SMT	BAGG	B	V	B.SMT	V.SMT
servo	0.761	0.761	0.743	0.754	0.761	0.757	**0.765**	0.762	0.761	0.758	0.758
a6	0.527	**0.539**	0.529	0.536	0.533	0.538	0.533	0.529	0.532	**0.539**	0.533
Abalone	0.719	0.719	**0.733**	0.732	0.729	0.729	0.719	**0.733**	**0.733**	0.730	0.730
machCpu	0.797	0.794	0.796	0.796	0.796	**0.799**	0.796	0.796	0.797	0.798	**0.799**
a3	0.453	0.400	0.561	0.560	0.562	0.562	0.385	0.558	0.559	0.562	**0.565**
a4	0.506	0.513	0.569	0.568	**0.587**	0.582	0.523	0.558	0.564	0.586	**0.587**
a1	0.621	0.603	0.738	**0.747**	0.737	0.742	0.606	0.746	0.745	0.746	0.744
a7	0.303	0.296	0.380	0.387	0.402	0.395	0.296	0.379	0.384	**0.406**	0.399
boston	0.902	0.898	**0.905**	0.904	0.899	0.899	0.897	**0.905**	**0.905**	0.897	0.899
a2	0.243	0.196	0.571	0.573	0.574	0.573	0.200	0.551	0.564	0.574	**0.578**
a5	0.209	0.186	0.550	0.546	0.561	0.556	0.127	0.551	0.551	**0.560**	0.558
fuelCons	0.918	0.901	0.930	0.928	0.929	0.928	0.903	**0.932**	0.931	0.927	0.928
availPwr	0.964	0.955	**0.977**	0.976	0.968	0.967	0.956	**0.977**	0.976	0.968	0.968
cpuSm	**0.508**	0.505	0.503	0.503	0.494	0.495	0.505	0.503	0.502	0.496	0.495
maxTorq	0.967	0.958	**0.974**	**0.974**	0.968	0.968	0.958	**0.974**	**0.974**	0.968	0.968
dAiler	0.735	0.736	0.744	0.745	0.752	**0.756**	0.737	0.744	0.745	0.755	0.755
bank8FM	0.946	0.945	0.947	0.947	**0.948**	**0.948**	0.945	0.947	0.947	**0.948**	**0.948**
ConcrStr	0.907	0.885	**0.955**	0.954	0.941	0.943	0.884	**0.955**	**0.955**	0.940	0.944
Accel	0.934	0.927	**0.950**	0.949	0.944	0.944	0.927	**0.950**	**0.950**	0.945	0.944
airfoild	0.219	0.142	0.219	0.218	0.193	0.197	0.136	0.220	**0.223**	0.195	0.196
Mean±sd	0.657±0.27	0.643±0.28	0.714±0.22	**0.715±0.22**	0.714±0.22	0.714±0.22	0.640±0.29	0.714±0.22	**0.715±0.22**	**0.715±0.22**	**0.715±0.22**

and REBAGG algorithms when using 10 and 40 models respectively. These results confirm that, overall, REBAGG algorithm displays a better performance with statistical significance in the majority of the situations. When using 10 models, REBAGG variants are always significantly better than the alternatives with the exception of RPART and RF learners. For these learners, although REBAGG variants exhibit a lower rank, the differences observed are not statistically significant. Regarding the results of the algorithms using 40 models, REBAGG variants are always significantly better than the use of the base learner, with the exception of the RPART learner.

One of the main conclusions is the overwhelming advantage of using REBAGG algorithm. We also observed that the higher is the number of models used, the better is the performance achieved by REBAGG. However, using 10 models in REBAGG algorithm is globally sufficient to obtain results significantly better than the alternatives. Overall, no variant of REBAGG stands out. However, we highlight that the performance achieved by

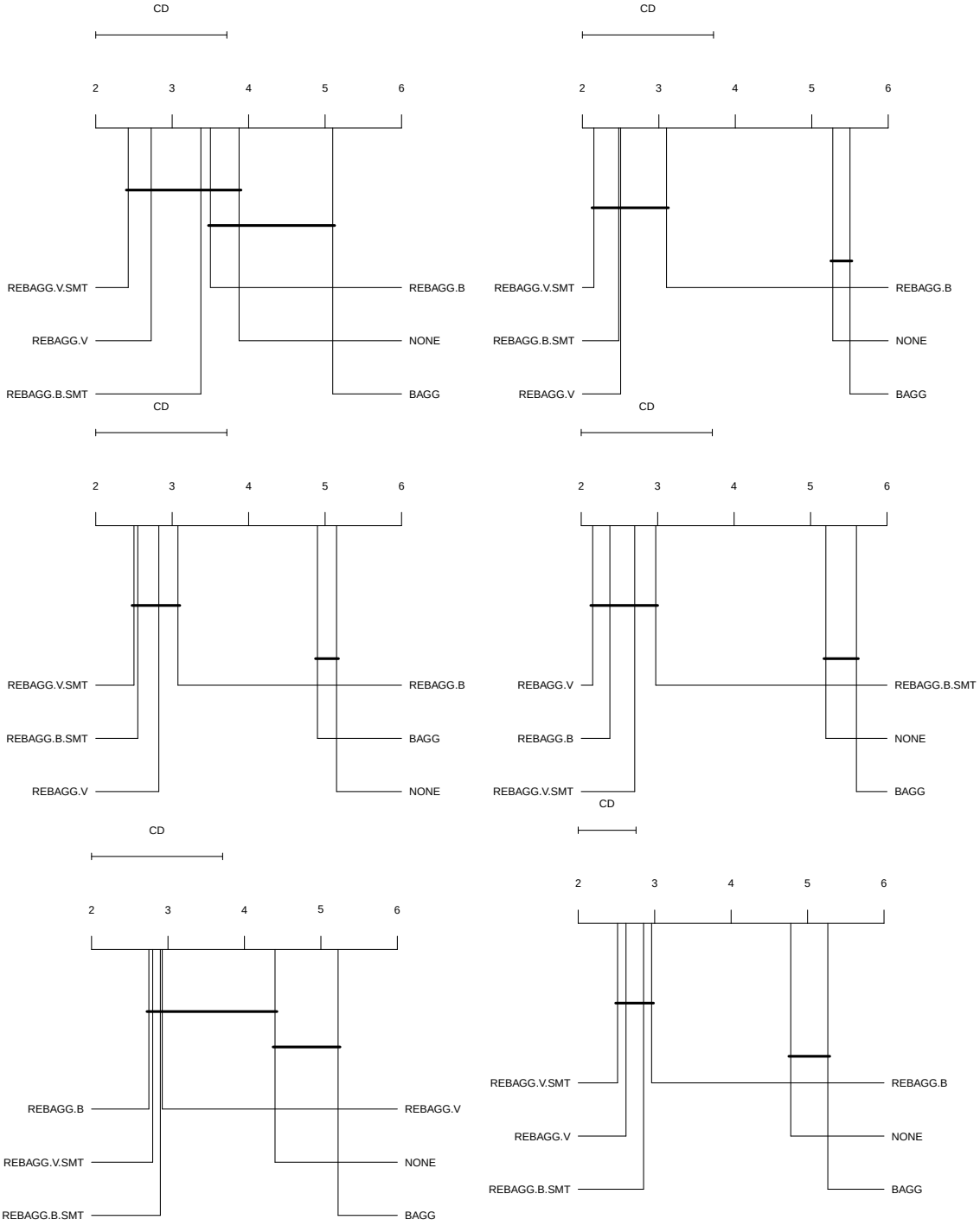

Figure 3: CD diagrams by learner, for ensembles built with 10 models (learners from left to right and top to bottom: RPART, SVM, MARS, GBM, RF, all).

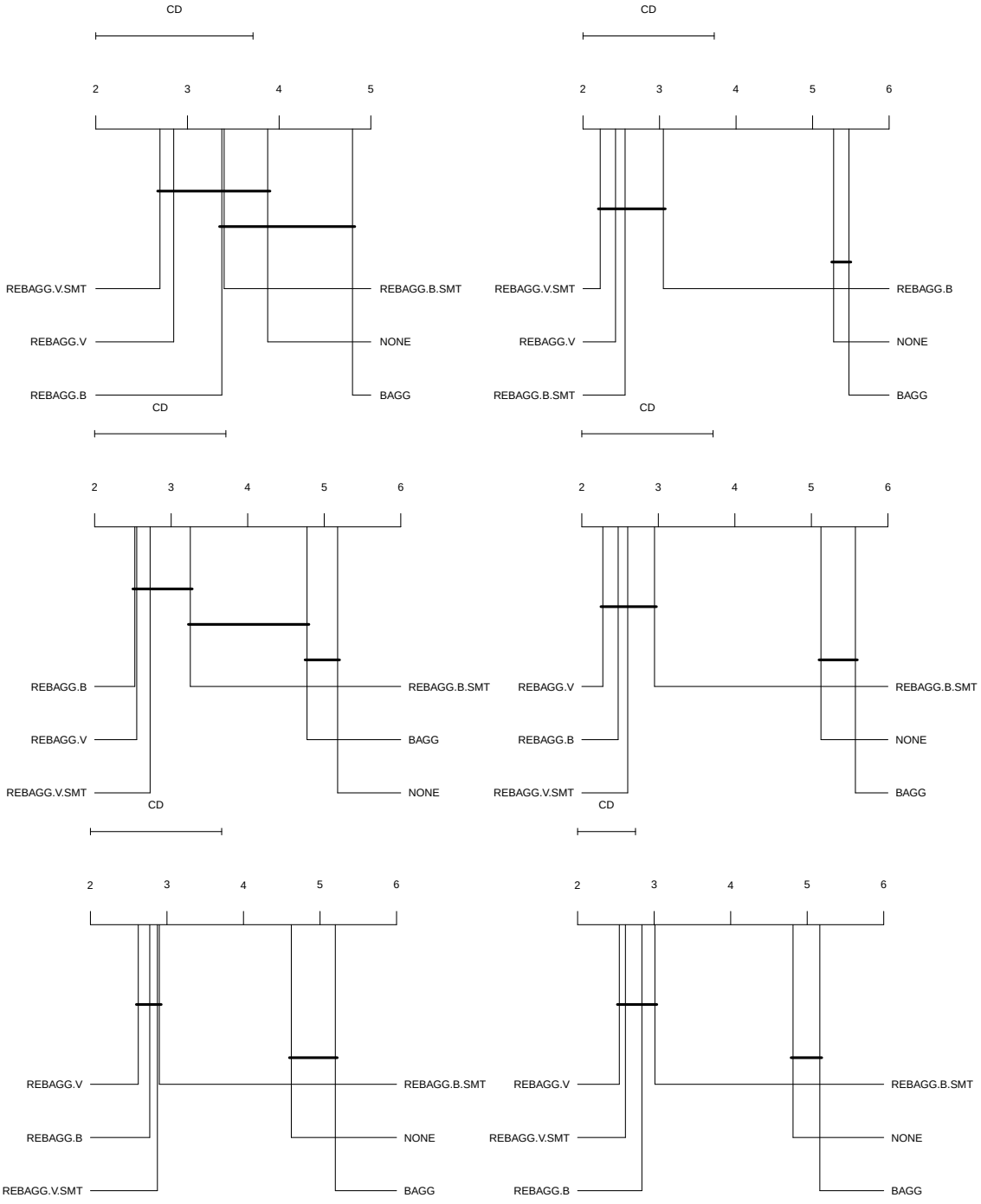

Figure 4: CD diagrams by learner, for ensembles built with 40 models (learners from left to right and top to bottom: RPART, SVM, MARS, GBM, RF, all).

Table 7: GBM average F_1^{ϕ} results by data set and strategy.

		m = 10					m = 40				
			REBAGG					REBAGG			
	NONE	BAGG	B	V	B.SMT	V.SMT	BAGG	B	V	B.SMT	V.SMT
servo	0.653	0.646	0.684	**0.685**	0.682	0.684	0.648	0.684	0.684	0.684	0.682
a6	0.523	0.529	0.514	0.519	0.536	**0.539**	0.530	0.516	0.518	0.536	0.538
Abalone	0.631	0.636	**0.709**	**0.709**	0.707	0.707	0.639	**0.709**	**0.709**	0.708	0.708
machCpu	0.722	0.722	0.734	0.740	0.740	0.748	0.726	0.737	0.741	0.742	**0.749**
a3	0.423	0.390	0.545	0.547	0.559	**0.560**	0.395	0.550	0.550	0.556	0.558
a4	0.535	0.532	0.539	0.539	**0.557**	0.554	0.522	0.532	0.529	**0.557**	0.554
a1	0.629	0.622	0.732	**0.735**	0.724	0.731	0.620	0.734	**0.735**	0.729	0.731
a7	0.317	0.312	0.359	0.357	0.368	0.363	0.309	0.361	0.360	**0.369**	0.367
boston	0.889	0.888	0.900	0.899	0.893	0.897	0.887	0.900	**0.901**	0.894	0.895
a2	0.218	0.164	0.581	**0.585**	0.569	0.566	0.147	0.582	0.584	0.568	0.569
a5	0.311	0.269	0.567	0.568	0.563	0.560	0.259	0.566	**0.569**	0.564	0.560
fuelCons	0.828	0.823	**0.871**	0.869	0.867	0.868	0.827	**0.871**	**0.871**	0.868	0.869
availPwr	0.903	0.900	**0.918**	**0.918**	0.908	0.909	0.900	**0.918**	**0.918**	0.908	0.909
cpuSm	0.131	0.134	0.187	**0.189**	0.187	0.187	0.134	0.187	0.185	0.188	0.188
maxTorq	0.937	0.936	0.952	0.952	0.947	0.947	0.936	**0.953**	**0.953**	0.947	0.948
dAiler	0.563	0.575	0.760	0.761	0.762	**0.763**	0.580	0.760	0.761	0.762	**0.763**
bank8FM	0.913	0.913	**0.935**	**0.935**	0.933	0.933	0.913	**0.935**	**0.935**	0.933	0.934
ConcrStr	0.545	0.540	**0.916**	0.912	0.913	0.912	0.540	0.915	**0.916**	0.914	0.913
Accel	0.885	0.872	0.910	0.910	0.907	0.907	0.879	**0.911**	**0.911**	0.907	0.908
airfoild	0.072	0.083	0.170	0.163	0.136	0.146	0.082	**0.172**	0.169	0.137	0.141
Mean±sd	0.581±0.27	0.574±0.28	0.674±0.24	**0.675±0.24**	0.673±0.24	0.674±0.24	0.574±0.28	**0.675±0.24**	**0.675±0.24**	0.674±0.24	0.674±0.24

REBAGG variants were not statistically different among themselves but were significantly different from the alternatives. In the majority of cases, any of the REBAGG variants provides better results than the base learner of the standard BAGG algorithm.

6. Conclusions

In this paper we presented REBAGG, a new bagging-based ensemble method that incorporates data pre-processing strategies designed to handle imbalanced regression tasks. As far as we know, this is the first ensemble method specifically adapted to handle this type of tasks. A large set of experiments was conducted for a diverse set of domains and learning algorithms. The results obtained showed the clear advantage of REBAGG algorithm. The key contributions of this work are: i) the proposal of the first ensemble method specifically developed for dealing with imbalanced regression tasks; and ii) the experimental verification of the overwhelming advantages of the proposed algorithm in a diversity of domains and learning algorithms.

As future work, we plan to explore the behaviour of REBAGG algorithm when using a higher number of models. Given the good performance observed for the REBAGG variants with varying ratio of normal to rare cases, we intend to explore the introduction of more diversity in the generation of samples. This could be achieved for instance, by using different methods for generating synthetic cases, and different strategies for under-sampling normal cases.

Acknowledgments

This work is financed by the ERDF – European Regional Development Fund through the COMPETE Programme (operational programme for competitiveness) and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project Fotocatgraf - UTAP-ICDT/CTM-NAN/0025/2014. This work is partially funded by the ERDF through the COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT as part of

project UID/EEA/50014/2013. Paula Branco is supported by a PhD scholarship of FCT (PD/BD/105788/2014). The participation of Luís Torgo on this research was undertaken thanks in part to funding from the Canada First Research Excellence Fund for the Ocean Frontier Institute.

References

Jerzy Błaszczyński and Jerzy Stefanowski. Neighbourhood sampling in bagging for imbalanced data. *Neurocomputing*, 150:529–542, 2015.

Paula Branco. Re-sampling approaches for regression tasks under imbalanced domains. Master's thesis, Dep. Comp. Science, Fac. Sciences - Univ. Porto, 2014.

Paula Branco, Rita P Ribeiro, and Luís Torgo. UBL: an R package for utility-based learning. *arXiv preprint arXiv:1604.08079*, 2016a.

Paula Branco, Luís Torgo, and Rita P Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Computing Surveys (CSUR)*, 49(2):31, 2016b.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Evgenia Dimitriadou, Kurt Hornik, Friedrich Leisch, David Meyer, and Andreas Weingessel. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2011.

Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A review on ensembles for the class imbalance problem: bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012.

Haibo He and Yunqian Ma. *Imbalanced learning: foundations, algorithms, and applications*. John Wiley & Sons, 2013.

Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Comparing boosting and bagging techniques with noisy and imbalanced data. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(3):552–568, 2011.

Yehuda Koren. The bellkor solution to the netflix grand prize. *Netflix prize documentation*, 81:1–10, 2009.

Bartosz Krawczyk. Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5(4):221–232, 2016.

Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation and active learning. In *NIPS'94*, pages 231–238. MIT Press, 1994.

Ludmila I Kuncheva. *Combining pattern classifiers: methods and algorithms*. John Wiley & Sons, 2004.

Mateusz Lango and Jerzy Stefanowski. Multi-class and feature selection extensions of roughly balanced bagging for imbalanced data. *Journal of Intelligent Information Systems*, 50(1):97–127, 2018.

Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

Stephen Milborrow. *earth: Multivariate Adaptive Regression Spline Models. Derived from mda:mars by Trevor Hastie and Rob Tibshirani.*, 2012.

Nuno Moniz, Paula Branco, and Luís Torgo. Evaluation of ensemble methods in imbalanced regression tasks. In *LIDTA*, volume 74, pages 129–140. PMLR, 2017.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018. URL https://www.R-project.org/.

Rita P. Ribeiro. *Utility-based Regression*. PhD thesis, Dep. Computer Science, Faculty of Sciences - University of Porto, 2011.

Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2017. URL https://CRAN.R-project.org/package=rpart. R package version 4.1-11.

Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Progress in Artificial Intelligence*, pages 378–389. Springer, 2013.

Luís Torgo. An infra-structure for performance estimation and experimental comparison of predictive models in R. *CoRR*, abs/1412.0436, 2014.

Luís Torgo and Rita P. Ribeiro. Utility-based regression. In *PKDD*, volume 7, pages 597–604. Springer, 2007.

Luís Torgo and Rita P. Ribeiro. Precision and recall in regression. In *DS'09: 12th Int. Conf. on Discovery Science*, pages 332–346. Springer, 2009.

Naonori Ueda and Ryohei. Nakano. Generalization error of ensemble estimators. In *Proceedings of IEEE International Conference on Neural Networks.*, pages 90–95, 1996.

Greg Ridgeway with contributions from others. *gbm: Generalized Boosted Regression Models*, 2017. URL https://CRAN.R-project.org/package=gbm. R package version 2.1.3.

Hsiang-Fu Yu, Hung-Yi Lo, Hsun-Ping Hsieh, Jing-Kai Lou, Todd G McKenzie, Jung-Wei Chou, Po-Han Chung, Chia-Hua Ho, Chun-Fu Chang, Yin-Hsuan Wei, et al. Feature engineering and classifier ensemble for kdd cup 2010. In *KDD Cup*, 2010.

Zhi-Hua Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012.