Journal of Data Science 19 (4), 593–614
October 2021

DOI: 10.6339/21-JDS1025
Statistical Data Science

Predictive Comparison Between Random Machines and Random Forests

Mateus Maia[1], Arthur R. Azevedo[2], and Anderson Ara[3,*]

[1]*Department of Math & Statistics, Maynooth University, Maynooth, Ireland*
[2]*Department of Statistics, Federal University of Bahia, Salvador-BA, Brazil*
[3]*Department of Statistics, Federal University of Paraná, Curitiba-PR, Brazil*

Abstract

Ensemble techniques have been gaining strength among machine learning models, considering supervised tasks, due to their great predictive capacity when compared with some traditional approaches. The random forest is considered to be one of the off-the-shelf algorithms due to its flexibility and robust performance to both regression and classification tasks. In this paper, the random machines method is applied over simulated data sets and benchmarking datasets in order to be compared with the consolidated random forest models. The results from simulated models show that the random machines method has a better predictive performance than random forest in most of the investigated data sets. Three real data situations demonstrate that the random machines may be used to solve real-world problems with competitive payoff.

Keywords *bagging; ensemble; support vector machines*

1 Introduction

Ensemble methods are machine learning algorithms that use multiple models in order to combine them and build a stronger one (Dietterich, 2000). In general, the strategy of combination of the models is made can be defined by two types of ensembles: i) the bagging approach (Breiman, 1996) based on independent bootstrapping models aggregated by the majority vote (classification tasks), or using the average of models predictions (regression tasks), and ii) the boosting approach (Freund and Schapire, 1997) which generates sequentially aggregated models using different weights based on their previous errors.

The bagging procedure is widely used in many real-world applications and there are several studies showing the effectiveness of this approach (Liang et al., 2011; Syarif et al., 2012; Zareapoor et al., 2015; Bhavan et al., 2019). When Breiman (1996) first introduced the bagging procedure, he emphasizes that the success of this method relies on the strength and the instability of single models that compose the bagging algorithm. The strength of the model can be explained as the predictive capacity of each model. The instability concept was also explored by Breiman et al. (1996). This characteristic is defined through the idea that if small changes in bootstrap replications of a sample of observations produce large changes in the bootstrap models, then, it can be considered unstable.

Although both of these characteristics are important to obtain robust results through the bagging procedure, it is not simple to optimize them simultaneously. Ho (1998) presents this trade-off between the strength and instability of models. Generally, strong models, i.e.: high

*Corresponding author. Email: alsouzara@gmail.com.

Received August 5, 2021; Accepted September 19, 2021

accuracy, are more stable, which implies a greater correlation between models of this type, and vice versa. Considering this aspect, Ho (1998) proposed the use of a random subspace method for constructing decision forests, where a random selection of features to split each node is made. Her work showed that it was possible to reduce the correlation of tree models without reducing their accuracy.

Later, this main idea of using random subspace was formalized by Breiman (2001), who presented the random forest (RF) method. Breiman (2001) showed that RF procedure creates base models which were strong (i.e.: high predictive power) and uncorrelated, resulting in a robust and consistent ensemble model (Scornet et al., 2015). The flexibility is also shown through the articles that present the use of random forests in handling missing data (Tang and Ishwaran, 2017), and its robustness to estimate class' probabilities (Sage et al., 2020). The effectiveness of RF is also demonstrated in literature with diverse real-world applications (Pal, 2005; Bosch et al., 2007; Statnikov et al., 2008; Futoma et al., 2015; Rodriguez-Galiano et al., 2015; Ouedraogo et al., 2019).

Support vector machines (SVM) are very efficient and popular tools for classification and regression with several perks. SVMs are rooted in the statistical learning theory (Vapnik, 1999) and this method has a globally optimal solution which is obtainable by solving a convex optimization problem, while the problems of local minima disrupt other common contemporary approaches, such as neural networks. A SVM also handles high-dimensional data since it considers non-linearity inherent in data through incorporation of kernel functions (Moguerza and Muñoz, 2006; Shivaswamy et al., 2007; Land and Schaffer, 2020; Kim and Kim, 2020). Despite its great efficiency, the choice of kernel function is crucial in SVM applications, and also it can be overlapped by ensemble models as random forest (Fernández-Delgado et al., 2014; Huo et al., 2016).

Inspired by this concept of random subspace from random forests, the ensemble approach random machines (RM) was designed. The RM method is a new ensemble procedure which uses the SVM as a base learner and applies an innovative random sampling of kernel functions to add instability and benefit the bagging structure. Ara et al. (2021) demonstrated that this algorithm successfully reduced the correlation between base learners while maintaining their strength, resulting in a better predictive performance than the traditional SVM and ensemble of SVMs.

In this paper, the RM were compared with RF, to show that this recent ensemble approach is competitive and can even result in better predictive performance than a robust and consolidated method such as RF in classification and regression tasks. In Section 2, an overview of the methodology of each algorithm is presented. In Section 3 and Section 4, both methods are applied and compared over simulated and benchmarking datasets, respectively. Section 5 reports the application of RM to solve successfully three real-world problems. Section 6 section closes the paper with the final comments.

2 Methodology

2.1 Random Forests

The RF predictor is composed of multiple tree models $f_i(x), i = 1, \ldots, B$, where f_i is a tree estimated based on a random subset of size m, $m < p$, of a p-dimensional base-learner vector $x \in R^p$ for an outcome variable y. Each tree is built on a bootstrap sample and B is the total number of them. Breiman (2002) refers to m as *mtry* and suggests that these values should be

equal to $p/3$ for regression tasks and $\sqrt{p}$ to classification tasks. Other parameters are *nodesize* – the minimum number of observations inside a terminal node – and number of trees B – also named as *ntree* – that compose the model.

The final prediction of the random forest is given by the collection of trees, and changes depending on prediction task. For the regression context, the final prediction to a new observation is given by

$$F(\boldsymbol{x}) = \frac{1}{B}\sum_{i=1}^{B} f_i(\boldsymbol{x}, \theta).$$

For the classification context, it is given by

$$F(\boldsymbol{x}) = \text{mode}\left\{f_i(\boldsymbol{x}, \theta)\right\}_i^B.$$

2.2 Random Machines

The RM method (Ara et al., 2021) uses SVMs (Cortes and Vapnik, 1995; Drucker et al., 1997) as base learners in the bagging procedure with a random sample of kernel functions to build them. The methodology of this ensemble procedure differs for regression and classification assignments. For a classification task, given the observations $\{\boldsymbol{x_i}, y_i\}_{i=1}^n$, with $y_i \in \{-1, 1\}$, $i = 1, \ldots, n$, where n is the sample size, SVM (Cortes and Vapnik, 1995) calculates an optimal hyperplane that separate the observation's classes. Its coordinates $\boldsymbol{w}$ are given by the minimization of the Equation (1),

$$\min \frac{1}{2}||\boldsymbol{w}||^2 + C\frac{1}{n}\sum_{i=1}^{n}\varepsilon_i, \tag{1}$$

with the constraints $y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) - (1 - \varepsilon_i) \geqslant 0$, $\varepsilon_i = \max\{0, 1 - y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b)\}$ for $i = 1, \ldots, n$, and where $C > 0$ is a regularization parameter.

The solution using the Lagrangian dual optimization for the soft margin problem (Fletcher, 2013), is given by

$$\max_{\boldsymbol{\alpha}} \left(\sum_{i}^{n}\alpha_i - \frac{1}{2}\sum_{i}^{n}\sum_{j}^{n}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i \cdot \boldsymbol{x}_j\right), \tag{2}$$

$$\text{s.t} = \begin{cases} \sum_i^n \alpha_i y_i = 0, \\ 0 \leqslant \alpha_i \leqslant C. \end{cases}$$

The predictions values for new observations $\boldsymbol{x^*}$ are given by,

$$f(\boldsymbol{x^*}) = \text{sgn}\left(\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x_i} \cdot \boldsymbol{x^*} - b\right),$$

where sgn(.) is the sign function.

To deal with the non-linearity in support vector models, the *kernel trick* is used to transform the data from the input space into a high-dimensional space where the observations are linearly separable. This transformation is made through the kernel functions $K(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\phi}(\boldsymbol{x}) \cdot \boldsymbol{\phi}(\boldsymbol{y})$. The most common kernel functions in SVM applications were used in this paper, and are presented in the Table 1, where $\gamma \in \mathbb{R}^+$, $d \in \mathbb{N}$.

The use of different kernel functions is one of the main ideas that support the efficiency of RM and differentiates it from traditional ensemble approaches. Through random sampling,

Table 1: Kernel functions.

Kernel	$K(\boldsymbol{x}, \boldsymbol{y})$	Parameters
Linear Kernel	$\gamma(\boldsymbol{x} \cdot \boldsymbol{y})$	γ
Polynomial Kernel	$(\gamma(\boldsymbol{x} \cdot \boldsymbol{y}))^d$	γ, d
Gaussian Kernel	$e^{-\gamma\|\|\boldsymbol{x}-\boldsymbol{y}\|\|^2}$	γ
Laplacian Kernel	$e^{-\gamma\|\|\boldsymbol{x}-\boldsymbol{y}\|\|}$	γ

it is possible to have a broader representation of the data, since each kernel visits different feature spaces during the bagging procedure. Also, the proportions of visits to these feature spaces are defined by weights based on the predictive capacity of every single kernel. Finally, a model averaging is realized using the weights based on out-of-bag accuracy. Therefore, the random selection of feature spaces increases the diversity of the base learners without decrease its accuracy. The idea of increase in diversity and maintaining the accuracy in bagging was also demonstrated in works that use kNN classifiers as base models (Gul et al., 2018). In the following we explain in detail the entire process used by RM.

The classification RM algorithm initializes generating support vectors models $h_r(\boldsymbol{x})$, where $r = 1, \ldots, R$ is the number of different kernels functions that will be used over a training set $\{(\boldsymbol{x_i}, y_i)\}_{i=1}^{n}$. Afterwards, each model will be validated over a test set $\{(\boldsymbol{x_i}, y_i)\}_{i=1}^{T}$, and an accuracy vector, $\boldsymbol{ACC} \in \mathbb{R}^R$, is obtained. So, each coordinate refers to the predictive performance of the support vector machine model with the respective kernel function. For instance, in this paper we consider the four kernel functions presented in Table 1 ($R = 4$), and then the vector $\boldsymbol{ACC}$ would be given by $\boldsymbol{ACC} = \{ACC_{Lin.}, ACC_{Pol.}, ACC_{Gau.}, ACC_{Lap.}\}$.

Subsequently, a vector of probabilities $\boldsymbol{\lambda} \in \mathbb{R}^R$ is calculated using Equation (3) in order to weight a random selection of the kernel functions that will be used in the bootstrap SVM base learners. Each term of $\boldsymbol{\lambda}$ is given by,

$$\lambda_r = \max\left\{0, \frac{\log\left(\frac{ACC_r}{1-ACC_r}\right)}{\sum_{j=1}^{R}\log\left(\frac{ACC_j}{1-ACC_j}\right)}\right\}, \qquad \forall r = 1, \ldots, R. \tag{3}$$

In order to model the base learners that compose the RM, B bootstrap samples are generated and B support vector models $g_b(\boldsymbol{x})$ are estimated based on these samples using the weighted random kernel functions sampled with probability λ_r. The probabilities λ_r are higher if some kernel function used in $h_r(\boldsymbol{x})$ predicted correctly observations from test set. Therefore, the kernel functions with higher accuracy will appear often when the random kernel selection for each bootstrap model is made. If any kernel function applied in $h_r(\boldsymbol{x})$ does not do better than a random choice, then ACC_r will be closer to 0.5 in binary cases and the probability of select that kernel function is near to zero.

Using the out-of-bag samples as test set, the predictive performance of each classifier $g_b(\boldsymbol{x})$ is evaluated, with $b = 1, \ldots, B$, which generates a new vector of accuracy $\boldsymbol{\Omega} = (\Omega_1, \ldots, \Omega_B) \in \mathbb{R}^B$. Therefore, a weight is calculated to each model prediction using the Equation (4),

$$w_b = \frac{1}{(1-\Omega_b)^2}, \quad b = 1, \ldots, B. \tag{4}$$

The final classification is given by Equation (5),

$$G(\boldsymbol{x_i}) = \operatorname{sgn}\left(\sum_{b}^{B} w_b g_b(\boldsymbol{x_i})\right), \quad i = 1, \ldots, N. \tag{5}$$

Considering the multi-class case, where K is the number of classes, the final decision model is given by

$$G(\boldsymbol{x_i}) = \arg\max_{k}\left(\sum_{b}^{B} w_b \mathbb{1}_{[g_b(\boldsymbol{x_i})=k]}\right), \quad i = 1, \ldots, N,$$

where $\mathbb{1}_{[x]}$ it the indicator function.

In the regression tasks, the target variable is no longer categorical, but continuous. Therefore, the RM's approach needs some modifications in the general procedure. Thus, the support vector regression (SVR) method (Drucker et al., 1997) is used as a base learner, and the measure used to evaluate is no longer the accuracy, but the root mean square error ($RMSE$). The equation that defines the probabilities vector of sample a kernel function becomes $\boldsymbol{\Lambda} = (\Lambda_1, \ldots, \Lambda_R)$ and is now given by Equation (6),

$$\Lambda_r = \frac{e^{-\beta\delta_r}}{\sum_{j=1}^{R} e^{-\beta\delta_j}}, \tag{6}$$

with $\forall r = 1, \ldots, R$. The $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_R)$ represents the standardized (i.e.: divided by its standard deviation) $RMSE$ from the support vector regression models $h_r(\boldsymbol{x})$ over the test set, and β is the correlation coefficient that tunes the penalty of the generalization error of each model. The probabilities Λ_r are higher if determined kernel function used in $h_r(\boldsymbol{x})$ has lower generalization error measured from the calculated $RMSE$ over the test set. Consequently, the models with lower δ_r will frequently appear when the random kernel selection for each bootstrap model is performed.

Algorithm 1 Random machines algorithm.

Input: Training Data, Test Data, B, Kernel Functions
for each $KernelFunction_r$ **do**
  Calculate the model h_r;
  Calculate the accuracy ACC_r or $RMSE_r$.
Calculate the probabilities λ_r;
Generate B bootstrap samples;
for b in B **do**
  Model $g_b(\boldsymbol{x})$ by sampling a kernel function with probability λ_r or Λ_r;
  Assign the weight using $OOBG_b$ samples.
Calculate $G(\boldsymbol{x})$

Both approaches are summarized in pseudo-codes to classification and regression tasks in Algorithm 1. The random selection of kernel functions enables visiting multiple kernel spaces, improving the representation of algorithm's learning, and turning out RM as a different ensemble method when compared with traditional SVM ensemble approaches.

3 Artificial Data Application

To compare RM and RF concerning their predictive capacity, both methods were applied to different simulated scenarios. The validation technique used was a repeated holdout, with thirty repetitions and a split ratio of 70%–30% of the training-test. This validation setting was selected to measure the generalization capacity to predict new observations consistently (Larsen and Goutte, 1999). Also, in order to achieve the best of each algorithm, a grid search was realized to select the best hyperparameters. For the RF method, the elements that were selected to compose the grid search were:

- *mtry*: number of variables randomly sampled as candidates at each split.
  - Classification: $mtry \in \{0.25\sqrt{p}; 0.5\sqrt{p}; \sqrt{p}; 2\sqrt{p}; 4\sqrt{p}\}$
  - Regression: $mtry \in \{0.25\frac{p}{3}; 0.5\frac{p}{3}; \frac{p}{3}; 2\frac{p}{3}; 4\frac{p}{3}\}$
- $nodesize \in \{5; 10; 25\}$; as the minimum size of observations in terminal nodes.
- $ntree \in \{100; 500; 1000\}$; as the number of tree collections in a random forest.

The choice of these hyperparameters was justified regarding with Probst et al. (2019), which evaluated that they were the most influential parameters in the RF algorithm. Concerning the RM grid search, the hyperparameters range in $C = \{0.1; 0.5; 1; 5\}$ (i.e.: cost parameter), $\gamma_{Gau.} = \{0.1; 0.5; 1; 5\}$ (i.e.: γ parameter for the Gaussian kernel presented in Table 1) and $\gamma_{Lap.} = \{0.1; 0.5; 1; 5\}$. The other parameters as the polynomial degree $d = 2$ and SVR parameter $\varepsilon = 0.1$ and $\beta = 2$ were chosen as default, since those values yielded reasonable good results for most of datasets, being not necessary to use a grid search to select them.

3.1 Classification Task

In classification context, three scenarios were generated with the objective to experiment different data behaviors. Simulation 1 regards a binary classification problem, were $y \in \{1, -1\}$ and each class is sampled from a different multivariate normal distribution. The Class 1 observations are sampled from a distribution with mean vector $\boldsymbol{\mu}_1 = \mathbf{0_p}$, with $\mathbf{0}_p$ as p-dimensional zero vector, and covariate matrix $\boldsymbol{\Sigma_1} = 4\boldsymbol{I}_p$. The Class -1 observations are sampled from a normal multivariate that has mean vector $\boldsymbol{\mu}_{-1} = 4 \times \mathbf{1_p}$, with $\mathbf{1}_p$ as p-dimensional vector of ones, and covariate matrix $\boldsymbol{\Sigma_{-1}} = \boldsymbol{I}_p$. The Simulation 1 configuration presents a setting where the two groups are easily linearly separable. The Simulation 2 follows the same pattern as Simulation 1, however the parameters of normal multivariate distribution from each class are different. Considering the Class 1 instances, they are sampled from a distribution with mean vector $\boldsymbol{\mu}_1 = \mathbf{0_p}$ and covariate matrix $\boldsymbol{\Sigma_1} = 4\boldsymbol{I}_p$ and the Class -1 observations are sampled from a normal multivariate that has mean vector $\boldsymbol{\mu}_{-1} = 2 \times \mathbf{1_p}$ and covariate matrix $\boldsymbol{\Sigma_{-1}} = \boldsymbol{I}_p$. At this second scenario, the two classes are no longer easily linearly separable by a hyperplane as was in the first one. The Simulation 3 data set explores the non-linearity behavior from a binary classification task through a circle uniformly distributed inside in the middle a p-dimensional cube.

All of scenarios varied the parameters: $n = \{100, 500, 1000\}$ number of observations, the $p = \{2, 10, 50\}$ dimension, and the $r = \{0.1, 0.5\}$ ratio observations in each class. The evaluation was realized considering the following metrics:

- **Accuracy (ACC)**: estimates the ratio of correctly classified observations to total observations from the sample. From a standard binary confusion matrix, we would have the quantities of true positives (TP), true negatives (TN), false positives (FP) and, false negatives (FN).

Table 2: Summary of mean values of ACC and MCC on Simulation 1 Dataset.

			RM						RF	
			ACC			MCC			ACC	MCC
n	p	r	$B = 25$	$B = 50$	$B = 100$	$B = 25$	$B = 50$	$B = 100$		
100	2	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.968 ± 0.025	0.805 ± 0.201
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.978 ± 0.028	0.957 ± 0.055
	10	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	50	0.1	1.000 ± 0.000	0.001 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
500	2	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.996 ± 0.006	0.975 ± 0.038
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.965 ± 0.015	0.929 ± 0.029
	10	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	50	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
1000	2	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.991 ± 0.005	0.948 ± 0.025
		0.5	0.991 ± 0.002	0.990 ± 0.004	0.990 ± 0.002	0.983 ± 0.003	0.981 ± 0.001	0.981 ± 0.003	0.970 ± 0.008	0.941 ± 0.015
	10	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	50	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

The calculation is given by

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}.$$

- **Matthew's Correlation Coefficient (MCC)**: introduced by Matthews (1975), is usually used to measure the predictive capacity of a model and it is calculated by

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}.$$

This metric can be considered a more reliable coefficient since it penalizes the false positive and false negative predictions, especially in cases of imbalanced data (Boughorbel et al., 2017). The MCC range varies from $[-1, 1]$, where 1 means a perfect prediction, 0 no better than a random prediction, and -1 is a complete reverse classification.

The result for Simulation 1 was summarized in the Table 2, where the ACC and MCC of the best hyperparameters (i.e.: resulted in maximum ACC and MCC) were obtained. From the outcome it is possible to notice that since the simplicity of the classification task both methods performed well with perfect predictions in most cases. Through this behaviour, RM did not present a good performance when compared concerning the RF when the classes are unbalanced with small sample size, as it can be seen in MCC from RM values in $r = 0.1$ and $n = 100$ scenarios, which is generally lower when compared with the RF.

The outcome for Simulation 2 is reported in Table 3. In this artificial data sets, the two classes are no longer easily linearly separable, and this characteristic is reflected in the lower values of ACC and MCC when compared with the first scenario. Despite that, in most cases, the RM outperform the random forest approach. The RM seems few accurate by to MCC measure only in cases where exists a great unbalance between the classes ratio = 0.1, and a small sample size $n = 100$.

The result of the third scenario is presented in Table 4. Simulation 3 emphasizes a non-linear classification problem, and it can be noticed from the evaluation metrics that the RM

Table 3: Summary of mean values of ACC and MCC using the RM and increasing the number of base learners over the Simulation 2 Dataset.

			RM						RF	
			ACC			MCC			ACC	MCC
n	p	r	$B = 25$	$B = 50$	$B = 100$	$B = 25$	$B = 50$	$B = 100$		
100	2	0.1	0.997 ± 0.013	1.000 ± 0.000	1.000 ± 0.000	0.979 ± 0.079	1.000 ± 0.000	1.000 ± 0.000	0.91 ± 0.032	0.383 ± 0.298
		0.5	0.947 ± 0.013	0.947 ± 0.022	0.948 ± 0.016	0.896 ± 0.027	0.896 ± 0.046	0.899 ± 0.033	0.871 ± 0.054	0.749 ± 0.104
	10	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.967 ± 0.030	0.811 ± 0.21
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.983 ± 0.023	0.967 ± 0.046
	50	0.1	1.000 ± 0.000	0.001 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	0.962 ± 0.055	0.839 ± 0.216
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
500	2	0.1	0.955 ± 0.009	0.957 ± 0.007	0.957 ± 0.007	0.801 ± 0.044	0.812 ± 0.030	0.813 ± 0.034	0.958 ± 0.012	0.744 ± 0.076
		0.5	0.917 ± 0.006	0.918 ± 0.005	0.918 ± 0.005	0.835 ± 0.012	0.836 ± 0.010	0.837 ± 0.009	0.875 ± 0.025	0.753 ± 0.049
	10	0.1	0.999 ± 0.003	0.998 ± 0.004	1.000 ± 0.000	0.994 ± 0.015	0.993 ± 0.017	1.000 ± 0.000	0.996 ± 0.005	0.978 ± 0.026
		0.5	1.000 ± 0.002	1.000 ± 0.000	1.000 ± 0.000	0.999 ± 0.004	1.000 ± 0.000	1.000 ± 0.000	0.982 ± 0.008	0.965 ± 0.015
	50	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
1000	2	0.1	0.946 ± 0.005	0.948 ± 0.006	0.949 ± 0.005	0.676 ± 0.035	0.691 ± 0.041	0.694 ± 0.034	0.951 ± 0.011	0.987 ± 0.070
		0.5	0.929 ± 0.004	0.927 ± 0.003	0.926 ± 0.003	0.858 ± 0.007	0.855 ± 0.006	0.853 ± 0.005	0.867 ± 0.013	0.738 ± 0.027
	10	0.1	0.996 ± 0.004	0.994 ± 0.005	0.998 ± 0.002	0.972 ± 0.032	0.960 ± 0.039	0.988 ± 0.017	0.999 ± 0.002	0.992 ± 0.011
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	0.994 ± 0.005	0.989 ± 0.009
	50	0.1	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
		0.5	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

Table 4: Summary of mean values of ACC and MCC on Simulation 3 Dataset.

			RM						RF	
			ACC			MCC			ACC	MCC
n	p	r	$B = 25$	$B = 50$	$B = 100$	$B = 25$	$B = 50$	$B = 100$		
100	2	0.1	0.972 ± 0.041	0.969 ± 0.001	0.967 ± 0.046	0.618 ± 0.001	0.682 ± 0.028	0.692 ± 0.001	0.918 ± 0.059	0.461 ± 0.344
		0.5	0.887 ± 0.041	0.880 ± 0.036	0.872 ± 0.034	0.794 ± 0.074	0.783 ± 0.062	0.770 ± 0.057	0.894 ± 0.054	0.796 ± 0.103
	10	0.1	0.995 ± 0.027	0.990 ± 0.001	0.987 ± 0.052	0.990 ± 0.002	0.991 ± 0.001	0.990 ± 0.002	0.889 ± 0.050	0.990 ± 0.001
		0.5	0.602 ± 0.056	0.609 ± 0.117	0.628 ± 0.045	0.565 ± 0.095	0.572 ± 0.040	0.589 ± 0.081	0.684 ± 0.079	0.537 ± 0.136
	50	0.1	1.000 ± 0.000	0.990 ± 0.001	1.000 ± 0.000	0.992 ± 0.001	0.992 ± 0.002	0.992 ± 0.001	0.889 ± 0.050	0.993 ± 0.001
		0.5	0.588 ± 0.146	0.598 ± 0.299	0.545 ± 0.102	0.516 ± 0.226	0.542 ± 0.097	0.505 ± 0.216	0.637 ± 0.084	0.687 ± 0.172
500	2	0.1	0.987 ± 0.005	0.892 ± 0.046	0.987 ± 0.008	0.892 ± 0.066	0.983 ± 0.007	0.883 ± 0.051	0.970 ± 0.015	0.811 ± 0.085
		0.5	0.975 ± 0.014	0.951 ± 0.028	0.976 ± 0.011	0.952 ± 0.021	0.929 ± 0.011	0.959 ± 0.021	0.951 ± 0.017	0.902 ± 0.034
	10	0.1	0.930 ± 0.002	0.924 ± 0.096	0.930 ± 0.002	0.911 ± 0.002	0.901 ± 0.002	0.929 ± 0.004	0.904 ± 0.019	0.918 ± 0.073
		0.5	0.915 ± 0.025	0.832 ± 0.049	0.929 ± 0.016	0.829 ± 0.031	0.830 ± 0.011	0.872 ± 0.022	0.832 ± 0.036	0.874 ± 0.070
	50	0.1	0.930 ± 0.001	0.921 ± 0.001	0.930 ± 0.001	0.921 ± 0.001	0.920 ± 0.001	0.928 ± 0.001	0.914 ± 0.018	0.901 ± 0.001
		0.5	0.700 ± 0.018	0.735 ± 0.056	0.706 ± 0.015	0.439 ± 0.037	0.453 ± 0.043	0.497 ± 0.108	0.803 ± 0.037	0.618 ± 0.068
1000	2	0.1	0.988 ± 0.003	0.933 ± 0.020	0.989 ± 0.003	0.937 ± 0.016	0.902 ± 0.003	0.941 ± 0.015	0.974 ± 0.008	0.846 ± 0.044
		0.5	0.980 ± 0.004	0.959 ± 0.008	0.980 ± 0.003	0.959 ± 0.006	0.940 ± 0.003	0.961 ± 0.005	0.960 ± 0.012	0.920 ± 0.023
	10	0.1	0.932 ± 0.023	0.934 ± 0.301	0.939 ± 0.019	0.552 ± 0.258	0.548 ± 0.008	0.574 ± 0.063	0.907 ± 0.012	0.502 ± 0.112
		0.5	0.967 ± 0.007	0.934 ± 0.015	0.971 ± 0.007	0.942 ± 0.014	0.922 ± 0.004	0.944 ± 0.008	0.856 ± 0.019	0.715 ± 0.036
	50	0.1	0.900 ± 0.001	0.901 ± 0.001	0.900 ± 0.001	0.900 ± 0.001	0.900 ± 0.001	0.901 ± 0.001	0.901 ± 0.012	0.901 ± 0.001
		0.5	0.840 ± 0.109	0.885 ± 0.072	0.917 ± 0.034	0.695 ± 0.202	0.773 ± 0.141	0.836 ± 0.068	0.841 ± 0.026	0.685 ± 0.052

surpassed the random forest in all cases. This outcome probably happens due to the support vector classifier's capacity to deal with the non-linearity through the kernel trick. Is important to observe that both methods had difficulties to classify the setting with high dimensionality ($p = 50$), unbalanced data ($r = 0.1$). This fact is reflected in both methods by lower MCC values, which is an appropriate measure to evaluate unbalanced scenarios.

The predictive capacity of each algorithm also can be compared through the number of times that a method won, i.e.: achieve higher or equal ACC or MCC, in thirty holdout repetitions. The outcome of simulation experiments is graphically summarized by Figure 1. Is remarkable

that the RM outperformed the RF in the majority of scenarios presented. Nonetheless, both methods have a slightly underfit over small sample sizes in the non-linear classification problem expressed by Simulation 3. Also, considering Simulation 2, the calculated MCC values for the RF models are higher in particular cases where the data have simultaneously unbalanced between the classes and small sample size. However, it is worth remembering that the ACC measure was used during the RM training process, which does not exhibit this same behavior.

Tables 2, 3, and 4 also express the sensibility of the RM with respect to the number of bootstrap samples B. From all simulation scenarios, it can be observed that, on average, the $RMSE$ tends to be smaller as we increase the number of base learners. However, the difference seems to be smaller from 50 bootstrap samples and 100, showing the consistency of the algorithm.

3.2 Regression Task

The artificial data generation for regression tasks considered five different scenarios to evaluate which algorithm would perform better. Simulations 1–3 are toy examples (Scornet, 2016), Simulation 4 (Van der Laan et al., 2007) and Simulation 5 (Roy and Larocque, 2012) are also simulation scenarios already tested and used in literature. All covariates $\boldsymbol{X} = (X_1, \ldots, X_p)$ from Simulations 1–4 follow a uniform distribution $[-1, 1]^p$. For the case of Simulation 5 each predictor follows an independent standard normal distribution. To appraise how each model is affected by the sample size, the values of $n = \{30, 100, 500, 1000\}$ were chosen. Moreover, the $RMSE$ was the measure selected to analyze the performance of RF and RM.

The equations of each simulation scenario are described below

- model 1: $p = 2$, $Y = X_1^2 + e^{-X_2^2} + \mathcal{N}(0, 0.25)$
- model 2: $p = 8$, $Y = X_1X_2 + X_3^2 - X_4X_7 + X_5X_8 - X_6^2 + \mathcal{N}(0, 0.5)$
- model 3: $p = 4$, $Y = -\sin(X_1) + X_2^2 + X_3 - e^{-X_4^2} + \mathcal{N}(0, 0.5)$
- model 4: $p = 6$, $Y = X_1^2 + X_2^2X_3e^{-|X_4|} + X_6 - X_5 + \mathcal{N}(0, 0.5)$
- model 5: $p = 6$, $Y = X_1 + 0.707X_2^2 + 2\mathbb{1}_{X_3>0} + 0.873\log(|X_1|)|X_3| + 0.894X_2X_4 + 2\mathbb{1}_{X_5>0} + 0.464e^{X_6} + \mathcal{N}(0, 1)$

The averages of $RMSE$ are presented in Table 5. The result achieved in all scenarios gives the evidence that the regression RM outperformed the RF, reinforcing the idea that the novel ensemble approach is competitive. It is interesting to notice that the difference between $RMSE$ values of each method is lower when the sample size is equally small, in most cases. This behaviour maybe can be interpreted as the regression RM approach can benefit even more with larger sample sizes. Figure 2 emphasizes the superiority of regression RM in that cases, where is possible to see the proportion of the number of times that the RF have greater values of $RMSE$.

4 Benchmarking Applications

Simulations are interesting to study the performance of both methods under controlled situations. However, analysis over real data sets is a valuable and essential contribution. Therefore, the comparison was also applied to real-world and benchmarking data. All data sets were retrieved from the *UCI Repository of Machine Learning* (Dua and Graff, 2017), being fifteen classification tasks and being fifteen regression tasks, summing up a total of thirty different data sets. All of them were chosen in order to diversify the sample size, dimensionality, and domain application. The validation technique, hyperparameter tuning, and metrics rating used in the benchmarking application were the same as used in Section 3.

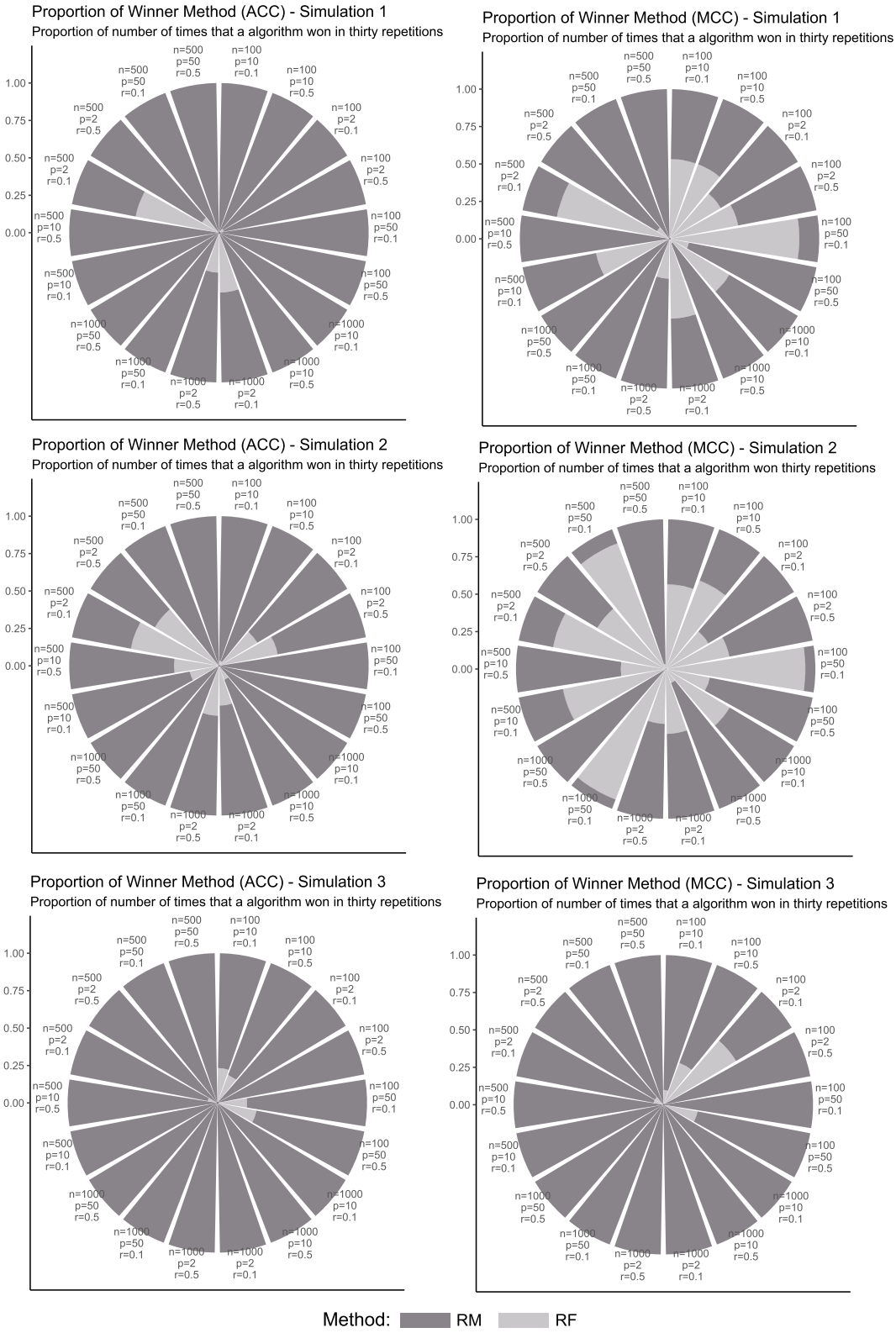

Figure 1: Proportion of the number times that a method won, considering MCC and ACC, in 30 holdout repetitions, calculated for all scenarios.

Table 5: Mean and standard deviation of $RMSE$ for all simulation settings.

		RM			RF
	n	$B = 5$	$B = 25$	$B = 50$	
Simulation 1	30	0.573 ± 0.190	0.562 ± 0.166	0.556 ± 0.178	0.568 ± 0.289
	100	0.509 ± 0.092	0.508 ± 0.082	0.502 ± 0.080	0.520 ± 0.103
	500	0.515 ± 0.035	0.510 ± 0.035	0.510 ± 0.035	0.528 ± 0.050
	1000	0.491 ± 0.019	0.490 ± 0.020	0.490 ± 0.020	0.498 ± 0.033
Simulation 2	30	1.136 ± 0.393	1.102 ± 0.365	1.101 ± 0.359	0.877 ± 0.262
	100	1.025 ± 0.140	1.000 ± 0.139	1.006 ± 0.142	0.964 ± 0.218
	500	0.796 ± 0.062	0.781 ± 0.055	0.771 ± 0.054	0.959 ± 0.093
	1000	0.762 ± 0.037	0.744 ± 0.034	0.741 ± 0.035	0.939 ± 0.070
Simulation 3	30	0.896 ± 0.282	0.888 ± 0.304	0.885 ± 0.306	0.789 ± 0.332
	100	0.845 ± 0.139	0.827 ± 0.117	0.824 ± 0.118	0.806 ± 0.220
	500	0.736 ± 0.069	0.725 ± 0.056	0.723 ± 0.058	0.754 ± 0.083
	1000	0.737 ± 0.039	0.732 ± 0.040	0.732 ± 0.041	0.779 ± 0.045
Simulation 4	30	0.991 ± 0.332	0.932 ± 0.247	0.939 ± 0.262	1.085 ± 0.356
	100	0.794 ± 0.131	0.799 ± 0.117	0.783 ± 0.121	0.822 ± 0.198
	500	0.774 ± 0.052	0.763 ± 0.056	0.761 ± 0.052	0.793 ± 0.062
	1000	0.754 ± 0.032	0.745 ± 0.029	0.744 ± 0.029	0.758 ± 0.038
Simulation 5	30	2.626 ± 0.983	2.511 ± 0.908	2.552 ± 0.936	2.448 ± 1.301
	100	2.125 ± 0.402	2.033 ± 0.401	2.021 ± 0.384	2.281 ± 1.021
	500	1.730 ± 0.203	1.691 ± 0.190	1.681 ± 0.197	1.946 ± 0.271
	1000	1.573 ± 0.122	1.547 ± 0.120	1.541 ± 0.122	1.730 ± 0.169

Table 6: Description of fifteen binary benchmarking.

Data Set	n	p	Class Proportion	Data Set	n	p	Class Proportion
audit risk	774	24	305/470	*ionosphere*	351	33	126/225
banknote	1371	4	610/762	*kidney disease*	154	24	41/114
clean1	476	166	207/269	*parkinson*	194	22	48/147
column 2C	310	6	100/210	*spirals*	500	2	250/250
cryotherapy	90	6	42/48	*thoraric*	470	14	70/400
dataR2	116	9	52/64	*vehicle*	846	18	218/628
diabetes	769	8	268/500	*whosale*	440	6	142/298
german	1000	24	300/700				

4.1 Classification Cases

The description of classification data sets is given in Table 6 with the number of observations (n), that range from 90–1371, the number of predictors (p), that range from 2–166, and the class proportion, i.e.: the ratio between the numbers of observations in each class. All of them are cases of binary classification.

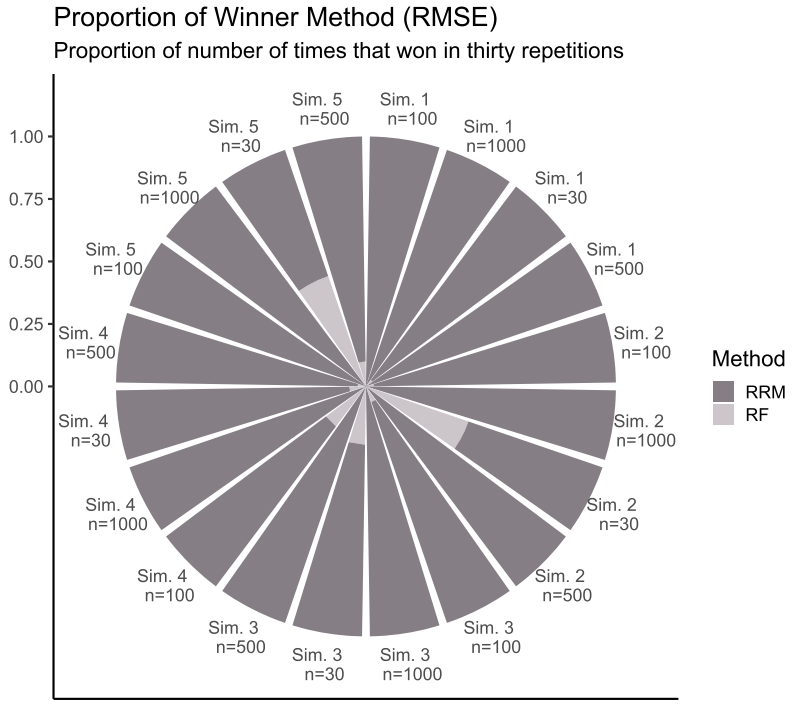

Figure 2: Proportion the number times that a method won base on the $RMSE$ for all simulation scenarios ($B = 25$).

The average calculated values of ACC and MCC are presented in Table 7. Faced with benchmarking considered scenarios, the RM had greater performance than the random forest algorithm, since it achieved higher mean values of ACC and MCC in 12 of 15 data sets. Studying the three cases (*german*, *vehicle*, *thoraric*) where the random forest obtained better MCC values, can be noticed that all of them are cases of imbalance between the classes. Highlighting the *thoraric* data set, it presents the largest difference of the MCC's mean value between those three and also have the smaller ratio class ratio. The RM's efficiency can also be observed graphically in Figure 3 which shows the proportion that a method won in thirty holdout repetitions.

4.2 Regression Cases

The characterization of regression benchmarking is featured in Table 8 that presents the number of observations that ranges (n) from 23–4177, the number of predictors that ranges from 1–60. Also, the mean value of the predictive variable is provided.

The values of Root Mean Squared Error obtained for regression RM and RF are in Table 9. The result given by Table above reveals that the regression RM produced a lower generalization error (i.e.: lower $RMSE$) in majority of data sets tested. Comparing both columns, it can be noticed that regression RM column is, in average, 8.5% smaller than the random forest one. Also, in absolute terms, regression RM won in 12 of 15 regression scenarios, losing only in: *friedman#3*, *pyrim*, and *triazine*. Investigating deeply, the high ratio p/n value from these two last data might be the reason of RF performed better since it can take more advantage of high dimensionality, and small sample size than the regression RM. Despite these three cases, the superiority of regression RM also is depicted in Figure 4.

Table 7: Average values of the accuracy and Matthew's Correlation Coefficient for benchmarking.

Dataset	RM		RF	
	ACC	*MCC*	*ACC*	*MCC*
audit risk	1.000 ± 0.002	0.999 ± 0.005	0.999 ± 0.002	0.998 ± 0.005
banknote	1.000 ± 0.001	1.000 ± 0.001	0.991 ± 0.005	0.982 ± 0.010
clean1	0.920 ± 0.028	0.837 ± 0.057	0.883 ± 0.027	0.765 ± 0.054
column 2C	0.855 ± 0.027	0.658 ± 0.057	0.834 ± 0.031	0.612 ± 0.066
cryotherapy	0.931 ± 0.043	0.867 ± 0.085	0.886 ± 0.064	0.779 ± 0.133
dataR2	0.749 ± 0.063	0.505 ± 0.128	0.727 ± 0.073	0.457 ± 0.146
diabetes	0.771 ± 0.023	0.476 ± 0.036	0.766 ± 0.019	0.476 ± 0.039
german	0.766 ± 0.019	0.406 ± 0.044	0.768 ± 0.020	0.417 ± 0.043
ionosphere	0.947 ± 0.019	0.885 ± 0.040	0.926 ± 0.025	0.841 ± 0.053
kidney disease	1.000 ± 0.001	1.000 ± 0.001	0.999 ± 0.004	0.998 ± 0.010
parkinson	0.924 ± 0.041	0.796 ± 0.106	0.895 ± 0.047	0.718 ± 0.129
spirals	0.997 ± 0.009	0.963 ± 0.182	0.936 ± 0.032	0.853 ± 0.166
thoraric	0.853 ± 0.028	0.002 ± 0.038	0.853 ± 0.028	0.013 ± 0.084
vehicle	0.982 ± 0.007	0.955 ± 0.019	0.984 ± 0.007	0.959 ± 0.017
whosale	0.919 ± 0.020	0.814 ± 0.046	0.917 ± 0.022	0.808 ± 0.049

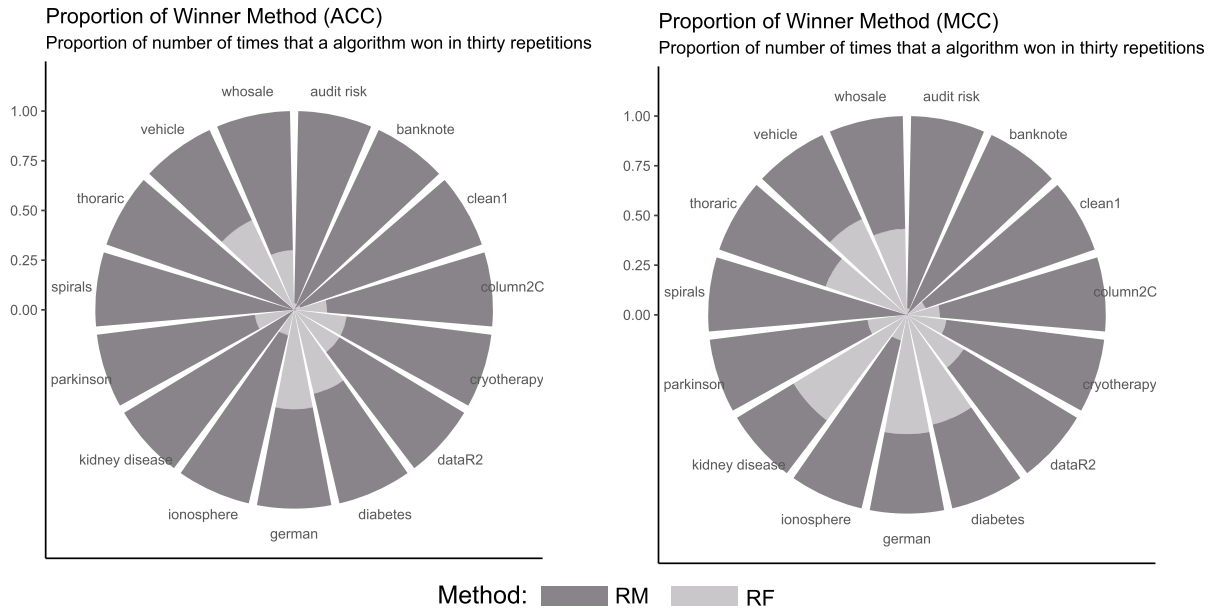

Figure 3: Proportion of the number times that a method won, considering *MCC* and *ACC*, in 30 holdout repetitions, calculated for all benchmarking.

Table 8: Description of the fifteen regression benchmarking.

Data Set	n	p	$\bar{y}$	Data Set	n	p	$\bar{y}$
abalone	4177	7	9.934	*mpg*	399	6	23.515
boston housing	506	13	22.533	*ozone*	330	8	11.776
cars	50	1	42.980	*pyrim*	74	27	0.659
friedman#1	500	10	14.417	*slump*	103	7	36.039
friedman#2	500	4	447.791	*space ga*	3107	6	−0.576
friedman#3	500	4	1.311	*stormer*	23	2	84.722
geysers	299	1	3.461	*triazine*	186	60	0.652
machines	209	7	99.330				

Table 9: Average values of $RMSE$ from regression RM and RF calculated on regression benchmarking.

Data Set	$RMSE$	
	RM	RF
abalone	2.113 ± 0.058	2.159 ± 0.054
boston housing	3.284 ± 0.384	3.357 ± 0.474
cars	15.549 ± 3.472	15.712 ± 3.531
friedman#1	2.081 ± 0.107	2.315 ± 0.123
friedman#2	129.157 ± 7.484	134.394 ± 9.499
friedman#3	0.149 ± 0.017	0.138 ± 0.017
geysers	0.883 ± 0.039	0.967 ± 0.059
machines	34.982 ± 12.355	37.729 ± 22.943
mpg	2.722 ± 0.275	2.847 ± 0.273
ozone	3.926 ± 0.254	4.024 ± 0.283
pyrim	0.099 ± 0.038	0.091 ± 0.047
slump	1.896 ± 0.431	3.815 ± 0.740
space ga	0.106 ± 0.009	0.117 ± 0.010
stormer	19.572 ± 9.309	40.606 ± 12.222
triazine	0.136 ± 0.017	0.125 ± 0.014

5 Three Real-World Applications

In this section, the RM approach was used to solve three different novel real-world problems: predict a defaulting status from companies, classify people's gender from their definition of love, and predict the rate of use of a Brazilian social assistance programme by municipality. The result was also compared with Linear and Gaussian SVM, and the RF. The descriptive analysis of these applications are shown in the Supplementary Material B.

5.1 Default Status from Business Companies

The dataset is composed of 66 observations where each instance represents a determined company. The outcome is a binary variable y_i, where if $y_i = 1$, this indicates that the corporation

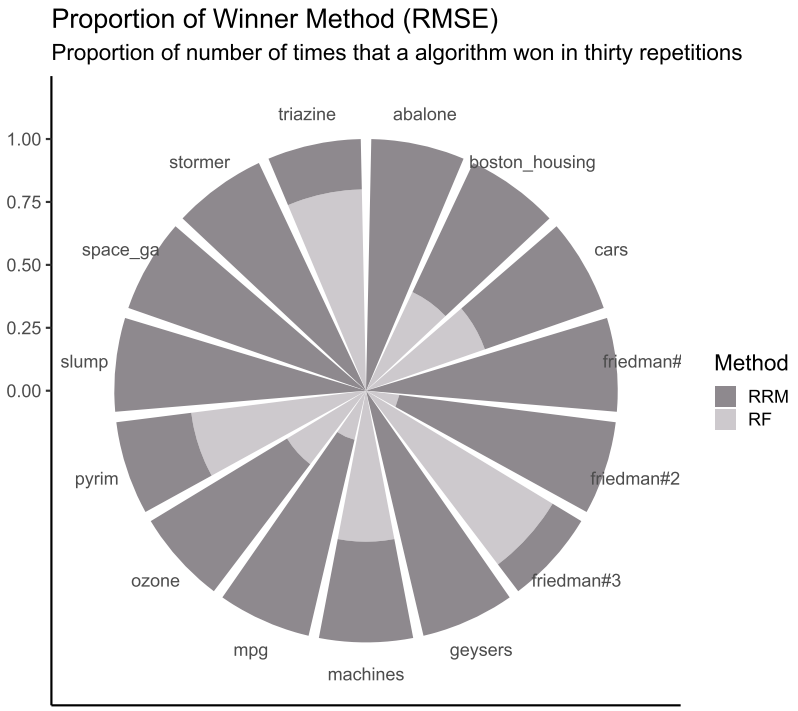

Figure 4: Proportion of the number times that a method won, considering $RMSE$, in 30 holdout repetitions, calculated for all regression benchmarks.

is compliant with their salaries, on the other hand, if $y_i = -1$, then it indicates a company in default. There are seven continuous covariates that describes each company. Two of them are the current liquid ratio (CLR) and the dry liquidity ratio (DLR), respectively. The other five are Kanitz indexes (Callado, 2003) that can indicate the possibility of business bankruptcy.

The proportion ratio between the number of instances from categories $y_i = 1$ and $y_i = -1$ is 27/39. To validate the performance of RM, RF, and SVM, a 100 repeated holdout validation was used with the split ratio of 70–30% of training-test set.

The tuning of the hyperparameters was also applied for all algorithms to achieve the better results for each of them. A grid search was realized over all possible hyperparameters combination for each method being respectively: RM: $C = \{0.1; 0.5; 1; 5\}$, $\gamma_{Gau.} = \{0.1; 0.5; 1; 5\}$, $\gamma_{Lap.} = \{0.1; 0.5; 1; 5\}$; RF: $mtry = \{1; 3; 6\}$, $nodesize = \{5; 10; 25\}$, $ntree = \{100; 500; 1000\}$; Linear Support Vector Machine, $C = \{0.1; 0.5; 1; 5\}$, $\gamma = \{0.1; 0.5; 1; 5\}$, $\gamma = \{0.1; 0.5; 1; 5\}$; Gaussian Support Vector Machine, $C = \{0.1; 0.5; 1; 5\}$, $\gamma = \{0.1; 0.5; 1; 5\}$.

To evaluate the predictive capacity, ACC and MCC were calculated over the test set. The result is given in Figure 5.

The mean of the average accuracy values for SVM.Lin, SVM.Gau, RF and RM are 80%, 85%, 90%, 90%, respectively. Considering the average MCC values are given by 65.51%, 66.66%, 77.87% and 77.20%, respectively. Interpreting these results, we can infer that RM surpassed the support vector models, and obtained an equivalent performance when compared with the RF.

It is important to emphasize that the RM performed competitively with the robust approach of RF in this application, resulting in great values of ACC and MCC. The optimal hyperparameters configuration were: $mtry = 6$, $nodesize = 5$ and $ntree = 1000$ for RF; $C = 5$,

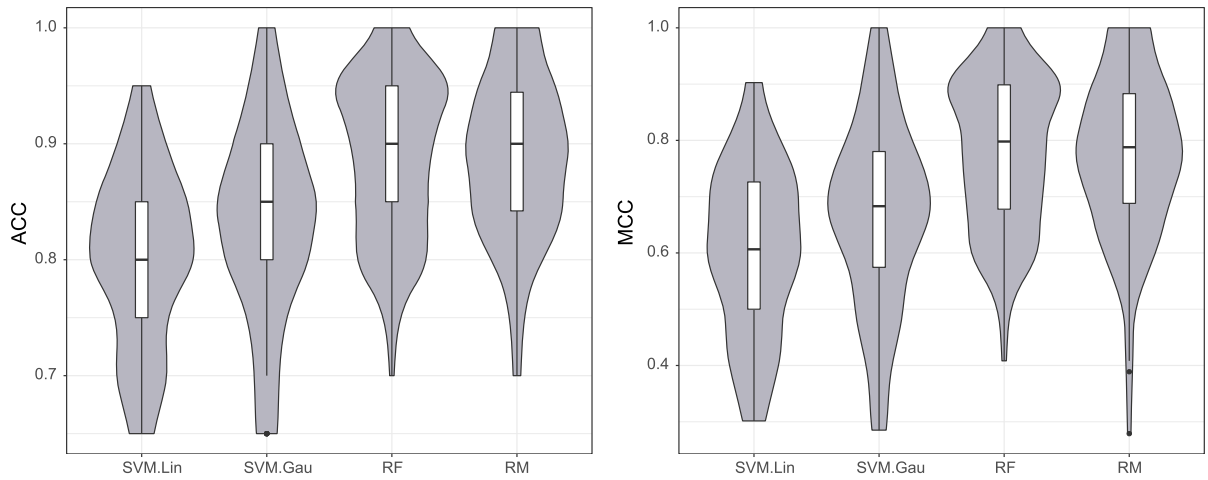

Figure 5: Violin-boxplots of the average values of ACC and MCC over the companies test set.

$\gamma_{Gau.} = 0.5$, $\gamma_{Lap.} = 0.1$ for RM; $C = 1$, $\gamma = 5$, for SVM with Linear Kernel and $C = 5$, $\gamma = 0.5$ for SVM with Gaussian Kernel.

5.2 Gender Prediction by the Love Interpretation

This database consists of a collection of statements gathered from 581 people, from different gender and age groups, about associations and feelings about what is love. The data was gathered in order to study and explore the concept of love based on a Brazilian sample (Td, 2017). The transcripts of the responses were analyzed by psychologists and psychiatrists that created 14 different categories to indicate a specific type of love perception. Beside it, a score is associated with each category to quantify how much of that feeling is present in the respective answer. For this classification task, the outcome will be the biological gender of each person, which is defined as a binary target $y_i \in \{Male, Female\}$.

To state the model the 14 categories of love and the age were selected as independent variables. The gender was defined as the dependent variable y_i. Therefore, the RM, random forest and, support vector models were applied in order to build a model capable to predict the gender. To evaluate the performance a 100 repeated holdout validation was used with the split ratio of 70–30% of training-test set.

The tuning of hyperparameters was also realized following the same grid search configuration presented in the Section 4.5.1, with the exception of the range of $mtry$ parameter that changed to $mtry = \{1, 2, 4, 8\}$.

The results of models performance are summarized in Figure 6. From the violin-boxplots of the average ACC values, it can be noticed that the performance of all models considering the accuracy is almost the same. However, since the classes are unbalanced the evaluation of the predictive capacity through MCC is more meaningful. When only MCC is observed is clear that the Regression machines performed slightly better than the other models. The median of the one hundred MCC means values for SVM.Lin, SVM.Gau, RF and RM are 0.162, 0.145, 0.163 and 0.186, respectively.

Counting the number of times that RM predicted better than RF (i.e.: yielded a lower $RMSE$ in a holdout repetition), we observed that occurred over 60 of 100 repetitions. The

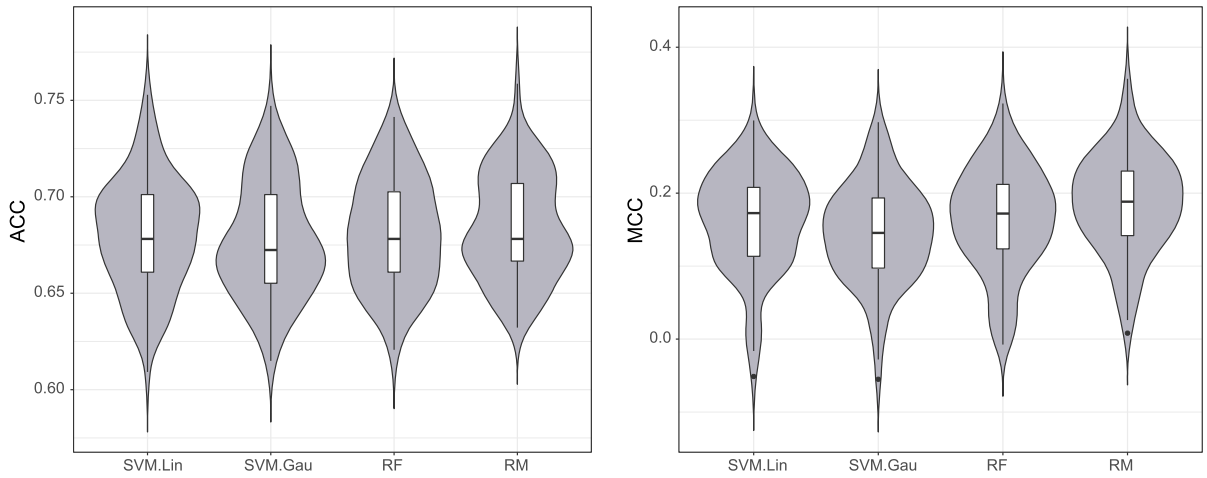

Figure 6: Violin-boxplots of the average values of ACC and MCC over test set.

optimal hyperparameters configuration were: $mtry = 4$, $nodesize = 25$ and $ntree = 1000$ for random forest; $C = 5$, $\gamma_{Gau.} = 0.5$, $\gamma_{Lap.} = 0.5$ for RM; $C = 5$, $\gamma = 5$, for SVM with Linear Kernel and $C = 1$, $\gamma = 0.1$ for SVM with Gaussian Kernel.

5.3 Forecasting the Rate of Use of a Brazilian Social Programme

Government public administration aims to provide support to its population through assistance programs that promote the reduction of poverty and inequality. In this sense, the Brazilian government has a social program for direct income distribution. The database from this application contains the collection of Brazilian cities and their rate of use of this benefit. This *rate of use* (y_i) is defined as the number of people who enjoy the assistance divided by the total population of that city.

Is important to emphasize that the capacity of creating models that can help to predict the rate of use social program like the *Bolsa-Família* can guide the Government to better manage resources, and provide better support in directing public policies. The data was retrieved from the Brazilian organizational site called *Transparency Portal*, and bring information about 5564 counties and their socioeconomic indexes.

Also, setting Y as target variable and the other variables as predictors, regression models were applied using the regression RM, RF and support vector regression models using the Linear and Gaussian kernel functions. Their performances were evaluated using the Root Mean Squared Error which was calculated through a validation technique of 100 repeated holdouts with a split ratio of 70–30% training-test. The tuning of hyperparameters was also realized following the same grid search configuration presented in the Section 4.5.1. The $\epsilon = 0.1$ parameter of SVR models was set as default.

The average values of $RMSE$ obtained in each algorithm are summarized in Figure 7. From the result is clear that the regression RM outperforms all of models since it presents the lower generalization error among them. The median of average values for SVM.Lin, SVM.Gau, RF and RM are respectively 0.0156, 0.0150, 0.0150 and 0.0147. The behavior of RM shows that it can be a competitive robust ensemble model as RF.

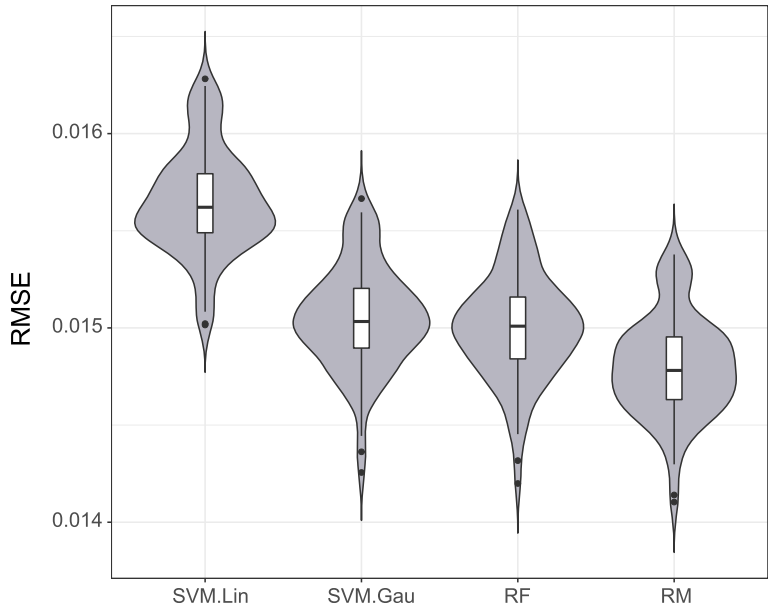

Figure 7: Violin-boxplots of the average values of $RMSE$ over test set.

Another way to compare and emphasizes the superiority of RM for this regression task is through counting the number of times that the proposed algorithm produced a lower $RMSE$ in a holdout repetition. Regarding all the repetitions, for SVM.Lin that happened 100 times, for the SVM.Gau 99 times, and for the RF 93 times. The optimal hyperparameters configuration were: $mtry = 6$, $nodesize = 25$ and $ntree = 1000$ for random forest; $C = 0.1$, $\gamma_{Gau.} = 5$, $\gamma_{Lap.} = 0.5$ for RM; $C = 1$, $\gamma = 0.1$, for SVM with Linear Kernel and $C = 1$, $\gamma = 0.1$ for Gaussian Kernel.

6 Final Comments

This paper proposes an empirical comparison between the recent ensemble learning approach called RM, and the consolidated tree ensemble method called RF. Both models were evaluated in classification and regression tasks over several simulated data sets and benchmark data. The results obtained show that the new RM procedure is strong competitive and produce better performance over the majority of the presented cases. Despite its good overall performance, the computational cost of this approach still larger when compared with RF.

For both simulated and real databases RM generally has an overall better performance than RF when classes are balanced or with not small samples. This behavior could be explained by the transformation of the feature space provided by the kernel functions. The use of kernel trick in support vector models leads to the different non-linear decision boundaries that may give a better representation of the learning problem than linear split rules from treed models. Moreover, the random sampling of kernel functions can increase the diversity necessary in ensemble approaches more than the random subspace sampling of predictions from RF. Another key aspect that can explain this phenomenon is the base learners that compose each ensemble approach, since SVM are generally stronger, i.e.: have a better predictive capacity, than simple tree models (Huang et al., 2003).

Additionally, RM were considered in thirty benchmark datasets and in three novel real-world applications, which were explored in this paper. In this case, the RM promoted better results in two of them and presented similar results in one of them. These facts reinforcing the predictive capacity the RM. On the other hand, the RM did not show a high predictive capacity via MCC in some situations where exist imbalanced classes with small sample size. Traditionally, SVM drops the predictive capacity in these cases (Wu and Chang, 2003). Some authors reported modifications on traditional SVM models to deal with this problem, and obtain greater results (Wang and Japkowicz, 2010; Batuwita and Palade, 2013).

RM and RF are considered ensemble learning methods, which different procedures based on bagging modelling. A principal difference of them is the base learner, RM considers support vector models as base learners and random forest considers decision trees as base model. Random machines consider different kernel functions to map the complete feature space. Random forest considers different feature subspace. In terms of the computational complexity, SVM has a time complexity $\mathcal{O}(n^3)$ and decision trees has a time complexity $\mathcal{O}(n \times p^2)$ (Al-Rajab et al., 2017). For this reason the RM is more computational complex than RF, specially in situations with large sample sizes. Our experience shows that a learning time with 60 seconds on random forest is equivalent a 400 seconds on RM.

For future works, new procedures to accelerate the learning time of RM may be considered. Also, may it interesting to use these adaptations jointly with its workflow to obtain better results in more simulation scenarios, different kernels and different weighing functions as well as an exhaustive investigation of the computational costs.

Supplementary Material A

The RM was also implemented in R language and it can be used through the rmachines package, available and documented at GitHub https://github.com/MateusMaiaDS/rmachines. To a overall description of how to reproduce the results from this article just access the README at https://mateusmaiads.github.io/rmachines_and_randomforest/.

Supplementary Material B

Exposes a descriptive analysis of the three real-world applications displayed in Section 5 and additional results around the comparison of RM and RF.

Acknowledgement

The authors are thankful to the unknown reviewers who had reviewed and suggested changes which improve the final version of this manuscript.

Funding

The authors gratefully acknowledge the financial support of the Brazilian research funding agencies CAPES (Federal Agency for the Support and Improvement of Higher Education). M.M.'s work was supported by a Science Foundation Ireland Career Development Award Grant 17/CDA/4695.

References

Al-Rajab M, Lu J, Xu Q (2017). Examining applying high performance genetic data feature selection and classification algorithms for colon cancer diagnosis. *Computer Methods and Programs in Biomedicine*, 146: 11–24.

Td A (2017). O conceito de amor: um estudo exploratório com uma amostra brasileira, Ph.D. thesis, Universidade de São Paulo.

Ara A, Maia M, Louzada F, Macêdo S (2021). Random machines: A bagged-weighted support vector model with free kernel choice. *Journal of Data Science*, 19(3): 409–428.

Batuwita R, Palade V (2013). Class imbalance learning methods for support vector machines. In: *Imbalanced learning: Foundations, Algorithms, and Applications*, 83–99.

Bhavan A, Chauhan P, Shah RR, et al. (2019). Bagged support vector machines for emotion recognition from speech. *Knowledge-Based Systems*, 184: 104886.

Bosch A, Zisserman A, Munoz X (2007). Image classification using random forests and ferns. In: *2007 IEEE 11th International Conference on Computer Vision*, 1–8. IEEE.

Boughorbel S, Jarray F, El-Anbari M (2017). Optimal classifier for imbalanced data using Matthews correlation coefficient metric. *PloS ONE*, 12(6): e0177678.

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

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

Breiman L (2002). Manual on setting up, using, and understanding random forests v3.1. Statistics Department University of California. Berkeley, CA, USA, 1:58.

Breiman L, et al. (1996). Heuristics of instability and stabilization in model selection. *The Annals of Statistics*, 24(6): 2350–2383.

Callado ALC (2003). Estudo sobre insolvência entre empresas paraibanas: uma aplicação do termômetro de kanitz. *Anais do Encontro Nordestino de Contabilidade–ENECON*.

Cortes C, Vapnik V (1995). Support-vector networks. *Machine Learning*, 20(3): 273–297.

Dietterich TG (2000). Ensemble methods in machine learning. In: *International Workshop on Multiple Classifier Systems*, 1–15. Springer.

Drucker H, Burges CJ, Kaufman L, Smola AJ, Vapnik V (1997). Support vector regression machines. In: *Advances in Neural Information Processing Systems*, 155–161.

Dua D, Graff C (2017). UCI machine learning repository.

Fernández-Delgado M, Cernadas E, Barro S, Amorim D (2014). Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, 15(1): 3133–3181.

Fletcher R (2013). *Practical Methods of Optimization*. John Wiley & Sons.

Freund Y, Schapire RE (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1): 119–139.

Futoma J, Morris J, Lucas J (2015). A comparison of models for predicting early hospital readmissions. *Journal of Biomedical Informatics*, 56: 229–238.

Gul A, Perperoglou A, Khan Z, Mahmoud O, Miftahuddin M, Adler W, et al. (2018). Ensemble of a subset of knn classifiers. *Advances in Data Analysis and Classification*, 12(4): 827–840.

Ho TK (1998). The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell*, 20(8): 1–22.

Huang J, Lu J, Ling CX (2003). Comparing naive Bayes, decision trees, and svm with auc and accuracy. In: *Third IEEE International Conference on Data Mining*, 553–556. IEEE.

Huo J, Shi T, Chang J (2016). Comparison of random forest and svm for electrical short-term load forecast with different data sources. In: *2016 7th IEEE International Conference on*

Software Engineering and Service Science (ICSESS), 1077–1080. IEEE.

Kim S, Kim C (2020). Influence diagnostics in support vector machines. *Journal of the Korean Statistical Society*, 1–22.

Land WH, Schaffer JD (2020). The support vector machine. In: *The Art and Science of Machine Intelligence*, 45–76. Springer.

Larsen J, Goutte C (1999). On optimal data split for generalization estimation and model selection. In: *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No. 98TH8468)*, 225–234. IEEE.

Liang G, Zhu X, Zhang C (2011). An empirical study of bagging predictors for different learning algorithms. In: *Twenty-Fifth AAAI Conference on Artificial Intelligence*.

Matthews BW (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2): 442–451.

Moguerza JM, Muñoz A (2006). Support vector machines with applications. *Statistical Science*, 21(3): 322–336.

Ouedraogo I, Defourny P, Vanclooster M (2019). Application of random forest regression and comparison of its performance to multiple linear regression in modeling groundwater nitrate concentration at the african continent scale. *Hydrogeology Journal*, 27(3): 1081–1098.

Pal M (2005). Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, 26(1): 217–222.

Probst P, Wright MN, Boulesteix AL (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(3): e1301.

Rodriguez-Galiano V, Sanchez-Castillo M, Chica-Olmo M, Chica-Rivas M (2015). Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. *Ore Geology Reviews*, 71: 804–818.

Roy MH, Larocque D (2012). Robustness of random forests for regression. *Journal of Nonparametric Statistics*, 24(4): 993–1006.

Sage AJ, Genschel U, Nettleton D (2020). Tree aggregation for random forest class probability estimation. *Statistical Analysis and Data Mining: The ASA Data Science Journal*. 13(2): 134–150.

Scornet E (2016). Random forests and kernel methods. *IEEE Transactions on Information Theory*, 62(3): 1485–1500.

Scornet E, Biau G, Vert JP, et al. (2015). Consistency of random forests. *The Annals of Statistics*, 43(4): 1716–1741.

Shivaswamy PK, Chu W, Jansche M (2007). A support vector approach to censored targets. In: *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, 655–660. IEEE.

Statnikov A, Wang L, Aliferis CF (2008). A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC Bioinformatics*, 9(1): 319.

Syarif I, Zaluska E, Prugel-Bennett A, Wills G (2012). Application of bagging, boosting and stacking to intrusion detection. In: *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 593–602. Springer.

Tang F, Ishwaran H (2017). Random forest missing data algorithms. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 10(6): 363–377.

Van der Laan MJ, Polley EC, Hubbard AE (2007). Super learner. *Statistical Applications in Genetics and Molecular Biology*, 6(1).

Vapnik VN (1999). An overview of statistical learning theory. *IEEE Transactions on Neural*

Networks, 10(5): 988–999.

Wang BX, Japkowicz N (2010). Boosting support vector machines for imbalanced data sets. *Knowledge and Information Systems*, 25(1): 1–20.

Wu G, Chang EY (2003). Class-boundary alignment for imbalanced dataset learning. In: *ICML 2003 Workshop on Learning from Imbalanced Data Sets II*, 49–56. Washington, DC.

Zareapoor M, Shamsolmoali P, et al. (2015). Application of credit card fraud detection: Based on bagging ensemble classifier. *Procedia Computer Science*, 48: 679–685. 2015.