Pamukkale Univ Muh Bilim Derg, 27(4), 513-519, 2021

Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi

Pamukkale University Journal of Engineering Sciences

Performance analysis of set partitioning formulations on the rule extraction from random forests

Rastgele ormanlardan kural çıkarmada küme bölüntüleme formülasyonlarının performans analizi

Mert EDALI[1,2*]

[1]Department of Industrial Engineering, Faculty of Mechanical Engineering, Yildiz Technical University, Istanbul, Turkey.
medali@yildiz.edu.tr

[2]Department of Medicine, University of Chicago, 5841 South Maryland Avenue, MC 6092 Chicago, IL 60637, USA.
edali@uchicago.edu

Received/Geliş Tarihi: 01.07.2020 Revision/Düzeltme Tarihi: 26.10.2020 doi: 10.5505/pajes.2020.05926
Accepted/Kabul Tarihi: 23.11.2020 Research Article/Araştırma Makalesi

Abstract

Random Forests is a widely used machine learning algorithm for classification and regression problems from different domains. Although they are generally accurate, their interpretability is low compared to their building blocks: single decision trees. Using the fact that each member of a Random Forest is a decision tree, we propose different set partitioning formulations to extract interpretable if-then rules from Random Forests. Our experiments on well-known classification and regression datasets show that the original set partitioning model formulation significantly reduces the number of rules while keeping the accuracy at acceptable levels. We also propose a modification to the problem's objective function, which aims to reduce the number of extracted rules further. We observe a further reduction in the number of extracted rules while the accuracy values stay nearly the same. Although the set partitioning problem is NP-hard, we obtain optimal results for most datasets within twenty minutes.

Keywords: Random forests, Rule extraction, Set partitioning, Classification, Regression, Interpretability.

Öz

Rastgele Ormanlar farklı alanlardaki sınıflandırma ve regresyon problemleri için sıklıkla kullanılan bir yapay öğrenme algoritmasıdır. Yüksek başarım göstermelerine rağmen, yapıtaşları olan karar ağaçlarına kıyasla yorumlanabilirlikleri oldukça düşüktür. Her bir üyesinin bir karar ağacı olduğu gerçeğinden yola çıkarak, Rastgele Ormanlardan yorumlanabilir eğer-ise tipinde kurallar çıkarmak için farklı küme bölüntüleme formülasyonları öneriyoruz. Literatürde sıklıkla kullanılan sınıflandırma ve regresyon veri setleri üzerinde yaptığımız deneylerin sonuçları göstermektedir ki orijinal küme bölüntüleme model formülasyonu, başarımı kabul edilebilir seviyelerde tutarak kural sayısını önemli ölçüde düşürebilmektedir. Çıkarılan kural sayısını daha da düşürebilmek için problemin amaç fonksiyonuna bir değişiklik öneriyoruz. Bu değişiklikle birlikte, çıkarılan kural sayısında daha da düşüş gözlemlerken başarımın aynı seviyelerde kaldığını gözlemliyoruz. Küme bölüntüleme problemi NP-zor olmasına rağmen, çoğu veri seti için yirmi dakika içinde en iyi çözümü buluyoruz.

Anahtar kelimeler: Rastgele ormanlar, Kural çıkarma, Küme bölüntüleme, Sınıflandırma, Regresyon, Yorumlanabilirlik.

1 Introduction

Random Forests (RFs) have been extensively used to solve classification and regression problems in a broad range of domains such as bioinformatics [1], medicine [2],[3] remote sensing [4], and time series modeling [5]. Basically, an RF is an ensemble of many decision trees. Each tree in the forest returns a prediction, which is a categorical value in classification and a numerical value in regression, and the final prediction is obtained by combining these individual predictions (i.e., majority voting in classification and taking the mean in regression). The power of RFs stems from incorporating two different training mechanisms; bagging (bootstrap aggregation) and random feature selection during split generation. These mechanisms allow for growing uncorrelated trees, yielding more stable and accurate predictions compared to individual decision trees [6].

Although growing many decision trees and combining their predictions increase prediction accuracy significantly, the interpretability of RFs is quite low compared to individual decision trees. While visualization and rule extraction (in the form of *if-then* rules) are two possible ways of interpreting decision trees, it is not beneficial to use these tools directly for RF interpretation. The main reason behind this is that each individual tree in an RF is trained on a bootstrapped version of the training data. As a result, each tree in an RF only covers a part of the training data, giving an idea about how inputs relate the outputs for only those data points. In addition, an RF potentially contains hundreds of trees and, thus, thousands of decision rules. As a result, directly visualizing or listing excess number of rules will not contribute to the interpretability of RF models. Therefore, techniques extracting these rules in a more distilled way to enhance understanding are needed. However, extracting rules from an RF is challenging due to two main reasons. First, as previously mentioned, an RF contains a large number of rules, which makes the rule extraction process time-consuming. Second, extracted rules should be as accurate as possible and collectively give an idea about how inputs relate to outputs for the whole domain of the problem at hand. In other words, the union of extracted rules should cover all of the training set instances.

*Corresponding author/Yazışılan Yazar

The accuracy vs. interpretability tradeoff arising in RF models leads to some attempts in the literature to increase the interpretability of RFs by extracting an accurate set of *if-then* rules. Mashayekhi and Gras [7] propose a hill-climbing algorithm for rule extraction from RF classification models. They assign a score to each rule of an RF model by considering the number of correctly and incorrectly classified instances in the training set. They propose another rule score formulation which also considers rule length. Their experimental results show that hill-climbing coupled with both rule scoring formulations is capable of extracting rules from RF models with fewer rules and minimal loss of accuracy. Rule scoring formulation which incorporates rule length further reduces the number of extracted rules. However, their algorithm does not guarantee the coverage of all training set instances, and rules covering the same training set instances might be selected. In an extension to their study [8], the authors propose new algorithms based on sparse group Lasso methods for both regression and classification problems. They conclude that the multiclass sparse group Lasso method achieves the least number of extracted rules for most datasets with a lower accuracy loss. However, they also note that this specific method is not applicable to regression problems. It is also important to note that the authors limit the initial number of rules to be at most 1000 prior to the rule extraction step. Liu et al. [9] propose a combined rule extraction and feature selection method (CRF) based on a linear programming model utilizing a 1-norm regularization. They only focus on classification problems for several biological datasets. Experimental results show that CRF significantly reduces the number of rules compared to the original RF model's rules while preserving classification accuracy. Adnan and Islam [10] develop an algorithm, ForEx++, which extracts rules based on their accuracy, coverage, and length. For each class, the algorithm first selects rules that have the accuracy and coverage values greater than the average accuracy and coverage values calculated by considering all the rules in the RF. In addition, for each class, they also select rules having the length less that the average length calculated over all of the rules. At the final step, the intersection of the rules selected by considering these three criteria is presented as the extracted rules. They run ForEx++ on two different medical classification problems. Although the results are satisfactory in terms of accuracy, they conclude that the rules extracted with ForEx++ may not cover all instances in the training set. Besides, their algorithm may not guarantee the diversity of rules, i.e., a set of extracted rules that might be similar. Phung et al. [11] establish a two-step greedy algorithm, ExtractingRuleRF, to extract rules from RFs dealing with classification problems. In the first step, rule refinement, rules obtained from an RF are first ranked according to some criteria such as accuracy and coverage. Then, the rules are processed to remove redundant conditions in a rule, duplicate rules, rules that are covered by other rules. At the end of this step, rules preserve their accuracy while having higher interpretability. In the second step, rule extraction, the authors use two different rule extraction policies, top-town or bottom-up, according to the weights of rules calculated in the first step. While the former extracts a rule set with high coverage but with lower accuracy, the latter returns compact but accurate rule sets having lower coverage. They perform experiments with a single dataset. Meinshausen [12] proposes a quadratic programming-based rule extraction scheme from tree ensembles. While the objective function of the quadratic optimization model minimizes the prediction error, constraints ensure that each training set instance is covered by only one rule. The approach is capable of dealing with both regression and classification problems. In the experiments, the author keeps only 1000 rules from RFs prior to solving the optimization problem. Friedman and Popescu [13] develop RuleFit, which uses a linear model with Lasso penalty to extract rules from rule-based ensembles such as RFs. However, in the linear model, the authors use both rules of the ensemble and original input variables in the training set as independent variables. Although the Lasso penalty minimizes the number of selected rules and input variables, the resulting model can return a mixture of rules and original model input variables. Therefore, RuleFit may not be regarded as a direct rule extraction method. Deng [14] presents a framework called inTrees (interpretable trees) for interpreting tree-based ensembles. The framework incorporates a set of tools such as rule listing, rule pruning, and rule selection. The framework is also capable of generating a Simplified Tree Ensemble Learner (STEL), which is obtained by a greedy and iterative selection of rules based on their accuracy, length, and coverage. The STEL serves as a new classifier where the rules are distilled from an RF model. For breast cancer diagnosis, Wang et al. [15] develop Improved Random Forest-Based Rule Extraction (IRFRE) method which considers both accuracy and interpretability of rules in a multi-objective optimization scheme. The authors use a multi-objective evolutionary algorithm to solve the optimization problem. Although their approach shows promising results, it is assessed only on three different datasets and tailored to classification problems. Besides, evolutionary algorithm dictates the selection of some parameters such as population size, crossover probability, and mutation probability.

Although the literature review reveals a multitude of rule extraction approaches from RFs and tree ensembles, most of them also have some disadvantages. For example, most of the studies only deal with classification problems (e.g., [7], [9]-[11],[15]). Besides, some of them are developed for specific problems (e.g., [11],[15]) or tested on specific datasets (e.g., [10]). Therefore, the extent of their generalizability to problems from other domains is unproven. We also observe that, for the methods utilizing Lasso penalty and evolutionary methods (e.g., [8],[9],[13],[15]), a parameter selection step is needed, which emerges as a disadvantage because these parameters must be optimized for each dataset separately. Finally, we also see that most of the methods do not guarantee the coverage of all training instances (e.g., [7],[8],[10]).

In this study, we propose a collection of set partitioning formulations to extract rules from RF models. Our approach is capable of handling both classification and regression problems. Furthermore, the proposed approach does not require any preprocessing step, meaning that the rules obtained from an RF can be directly fed to the rule extraction problem. The most beneficial characteristic of our approach is that it is fully parameter-free. We also guarantee the coverage of each training instance to give a comprehensive view about the relationship between input variables (features) and outputs. In addition, the set partitioning formulation enables us to prevent the intersection of rules as much as possible.

The remainder of the article is organized as follows: Section 2 gives preliminary background information about RFs and the set partitioning problem formulation. Section 3 presents the experimental design and the results. Section 4 concludes the study.

2 Preliminaries and proposed method

In this section we provide the formal definitions of Random Forests and set partitioning problem formulations.

2.1 Random Forests

Let $D = \{(\boldsymbol{x}_i, y_i): i = 1, \cdots, n\}$ be a dataset used to train an RF. Here, $\boldsymbol{x}_i = (x_{i1}, \cdots, x_{ip})$ is the input (feature) vector with p features, and y_i is the corresponding output. When the output is continuous, the problem is considered a regression problem. In contrast, if y_i is categorical, the problem is called a classification problem. n is the number of rows in D, and is generally called the size of the dataset.

An RF is an ensemble of T decision trees $\{g_t, t = 1, \cdots, T\}$. Each tree in the forest is trained on a dataset selected from D by using bootstrapping (random sampling with replacement). In addition, at each split generation in the tree fitting process, only a random subset of inputs is used. These two techniques enable to generate a diverse set of trees. In classification, the output of an input vector is predicted by applying the majority voting rule over all the predictions returned from trees. In regression, the mean of the predictions returned by each tree is assigned as the prediction [6].

Since it is possible to express a decision tree as a set of *if-then* rules, an RF can also be considered as a large set of rules. Each tree in an RF can be converted to a rule set by tracing the path from the root to each leaf node. Figure 1 shows a decision tree arbitrarily selected from an RF trained on a dataset with two input variables (i.e., x_1 and x_2) and two categorical outputs (i.e., A and B).

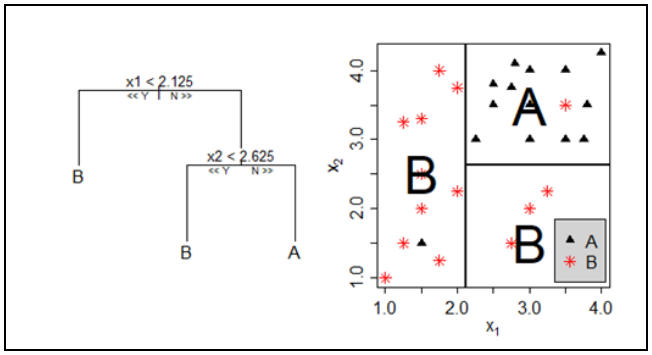

Figure 1. A tree of an RF (left) and the corresponding partition in the input space (right).

Table 1 lists all the rules obtained from the tree given in Figure 1.

Table 1. Rules listed from the tree given in Figure 1.

Rule Number	Rule
1	IF $x_1 \leq 2.125$ THEN $y = B$
2	IF $x_1 > 2.125$ AND $x_2 \leq 2.625$ THEN $y = B$
3	IF $x_1 > 2.125$ AND $x_2 > 2.625$ THEN $y = A$

As mentioned in the Introduction, one will obtain a large number of rules from an RF. This excess number of rules does not improve the interpretability of an RF. Therefore, we propose a set partitioning formulation to extract rules, whose details are given in the following subsection.

2.2 The set partitioning problem

The set partitioning problem has a long history in the optimization literature. It has been extensively used to model some problems such as crew scheduling [16] and vehicle routing [17]. In the problem, the objective is to select a set of columns of a binary matrix so that the row sums of the selected columns are exactly equal to 1. Since each column is associated with a cost value, the aim is to select those columns while minimizing the total cost. The problem can be formally defined as follows [18]:

$$minimize \sum_{j=1}^{m} c_j x_j \tag{1}$$

$$\sum_{j=1}^{m} a_{ij} x_j = 1 \quad i = 1, \cdots, n \tag{2}$$

$$x_j \in \{0,1\} \quad j = 1, \cdots, m \tag{3}$$

Assume that A is an $n \times m$ binary matrix such that $a_{ij} \in \{0,1\}$, $i \in \{1, \cdots, n\}$ and $j \in \{1, \cdots, m\}$. In the objective function (1), c_j is the cost of selecting column j of A, and x_j is the decision variable which is equal to 1 if column j is selected and 0 otherwise. Equation (2) shows the constraint set of the problem, which ensures that row sums of the selected set of columns are equal to 1. Equation (3) ensures that each decision variable must be binary (i.e., either 0 or 1).

2.3 Proposed approach

In order to formulate the rule extraction problem as a set partitioning problem, we first need to define the components of Equations (1)-(3) in the context of rule extraction.

The first step in the proposed rule extraction scheme is to generate matrix A and cost vector c. Each column j of A corresponds to a rule of the RF. In addition, each row i of A corresponds to each instance of D. Here, we use the binary encoding approach presented in Liu et al. [9]. The algorithm is presented in detail in Figure 2. The inputs for the algorithm are a random forest F and a training set D. There are two outputs of the algorithm, namely A and c. In the first step, all the rules contained in F are transformed to a list of rules (RL). As mentioned before, this can simply be achieved by tracing the path from the root to each leaf node for all trees in F. n is the number of instances in D, and m is the number of rules contained in F (and thus in RL).

Input: F, D
Output: A, c
1: Generate a set of rules, RL, listed from F.
2: $n \leftarrow size(D)$ /* Number of instances in D */
3: $m \leftarrow size(RL)$ /* Number of rules in RL */
4: **for** $j = 1$ **to** m **do**
5: **for** $i = 1$ **to** n **do**
6: **if** rule j covers $\boldsymbol{x}_i$ **then**
7: $a_{ij} = 1$
8: **else**
9: $a_{ij} = 0$
10: **end if**
11: **end for**
12: $c_j \leftarrow error(j)$
13: **end for**

Figure 2. Algorithm for generating A and c for an RF F.

The algorithm generates A column by column. At each iteration of the outer for loop, a new column is added to A. In the inner for loop, for each row i of column j, we set $a_{ij} = 1$, if rule j

covers training instance i (i.e., $\boldsymbol{x}_i$), and we set $a_{ij} = 0$ otherwise. After setting column values, we also calculate the error of rule j, which corresponds to the "cost" of incorporating that rule in the extracted rule set. In classification, $error(\cdot)$ function can be any appropriate error measure such as misclassification rate. In regression problems, it may be one of the error measures such as Root Mean Square Error (RMSE), Mean Square Error (MSE), Mean Absolute Error (MAE), etc. After obtaining A and c, one can solve the rule extraction problem.

Figure 3 shows the rules extracted from an RF trained on the dataset shown in Figure 1. We see that the set partitioning formulation ensures the coverage of all training instances while avoiding that an instance is covered by more than one rule. It is also obvious from the figure that this may not mean that the rules cannot intersect. We also see that the set partitioning formulation does not guarantee that the extracted rules cover the entire input space, especially when some subspaces of the input space lack data instances. Therefore, the set of extracted rules cannot be represented as a tree. For that reason, the extracted rule set cannot be directly used as a classification or regression model.

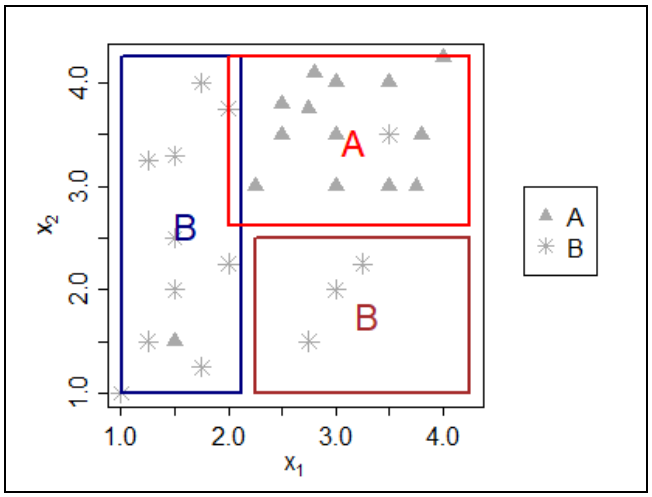

Figure 3. Visualization of the extracted rules from an RF trained on the dataset presented in Figure 1.

Table 2 shows the extracted rules, which are visualized in Figure 3.

Table 2. Extracted rules from an RF trained on the dataset presented in Figure 1.

Rule Number	Rule
1	IF $x_1 \leq 2.125$ THEN $y = B$
2	IF $x_1 > 2$ AND $x_2 > 2.625$ THEN $y = A$
3	IF $x_1 > 2.25$ AND $x_2 \leq 2.5$ THEN $y = B$

2.4 Different objective function formulations

As mentioned in the Introduction, we aim to propose a set of set partitioning formulations for rule extraction from RFs. For this purpose, we modify the objective function (1) accordingly. For classification, we use the original objective function (Equation (1)) and also a modified version is given below for experimentation:

$$minimize \sum_{j=1}^{m} x_j + \sum_{j=1}^{m} c_j x_j = \sum_{j=1}^{m} (1 + c_j) x_j \qquad (4)$$

The main difference between Equation (1) and (4) is that the former only aims to reduce the total "cost" while the latter also aims to reduce the number of extracted rules. For regression, we use Equation (1) and a modified version is given below as objective functions for experimentation:

$$minimize \sum_{j=1}^{m} x_j + \sum_{j=1}^{m} \frac{c_j}{\max_j c_j} x_j = \sum_{j=1}^{m} \left(1 + \frac{c_j}{\max_j c_j}\right) x_j \qquad (5)$$

As one can see, the main aim of introducing Equation (5) as the alternative objective function is to reduce the number of extracted rules. In addition, we normalize c to prevent it from dominating the first term during optimization. This normalization is used because most of the error measures for regression problems have the same scale as the output y.

3 Experimental design and results

In this section, we provide the experimental design and the results of these experiments. We use a Windows 10 64-bit operating system with 8 GB RAM, dual-core CPU (i7-7500U 2.70 GHz). We select five classification (Table 3) and five regression (Table 4) datasets frequently used in machine learning studies. All of the datasets are taken from UCI Machine Learning Repository [19], except for boston [20] and mammography [21] datasets.

Table 3. Characteristics of the datasets for classification.

Dataset	Number of Features	Number of Classes	Number of Instances
iris	4	3	150
mammography	6	2	11183
glass	10	6	214
WDBC	30	2	569
liver	6	2	345

Table 4. Characteristics of the datasets for regression.

Dataset	Number of Features	Number of Instances
boston	13	506
wine_white	11	4898
auto_mpg	7	392
airfoil	5	1503
concrete	8	1030

For RF training, we use randomForest package (version 4.6-12) [22] in R (version 3.5.1) [23]. We use 10 × 5 nested and stratified cross-validation design for hyperparameter optimization. We consider the following subsets of the hyperparameters: $ntree \in \{50, 100, 150\}$ and $maxnodes \in \{10, 25, 50\}$. Here, *ntree* is the number of trees in the forest, and *maxnodes* is the maximum number of terminal (leaf) nodes for each tree in the forest [22]. Although it is known that RFs perform well under default hyperparameter settings, we perform hyperparameter tuning over both *ntree* and *maxnodes* since both of them directly affect the complexity of the rule extraction problem by determining the number of columns of A. Besides, these two hyperparameters also affect the structure of the rules: (i) the higher *ntree* is, the higher the chance of generating a diverse set of rules is, (ii) the higher *maxnodes* is, the more confined the rules in the forest are. To calculate vector c (i.e., cost coefficients in Equation (1)), we use misclassification error for classification and RMSE for regression.

To solve each set partitioning problem, we use the R implementation of Gurobi solver [24]. Since the set partitioning problem is an integer programming problem (see Equation (3)), Gurobi uses branch-and-bound algorithm. However, we also note that the set partitioning problem is NP-hard, implying that the problem is not solvable in polynomial time for large instances [25],[26]. Therefore, we set a limit of 1200 seconds (i.e., 20 minutes). If the optimal solution is not found within this time limit, Gurobi solver returns the incumbent solution (i.e., the best known feasible solution during the execution of the branch-and-bound algorithm).

While presenting the results, we also provide the accuracy/error of the rules when they are used as a classification or regression model. Since both mammography and glass datasets are imbalanced, we report macro-F1 values for all classification datasets. Macro-F1 is calculated by taking the averages of F1 values obtained for each class. For regression datasets, we report RMSE values. While assessing the accuracy of rule sets, we use the following approach: If a test instance is covered by more than one rule, we perform majority voting over the classes of those rules for classification, and we take the mean of the outputs of those rules for regression. If no rules cover a test instance, we consider the fraction of the conditions satisfied in each rule for that test instance. Then, we follow the same procedure that we have followed for the case where a test instance is covered by more than one rule. We also report the fraction of the missed points for each dataset. These reported values provide evidence about the input space coverage capability of the extracted rules. As mentioned before, although the set partitioning model ensures the coverage of all training instances, there still might be some subspaces of the input space which are left uncovered due to the lack of training instances in those subspaces (see Figure 3). If the fraction of the missed points is low, there is a high probability that there will be one or more rule covering each test instance. However, if it is high, the rule set does not cover the input space well, leaving some test instances uncovered. In the latter case, extracted rules may perform poorly when they are used as a classification or regression model. In addition, one may miss the information about how inputs relate to outputs when the fraction of missed points is high.

3.1 Results for classification problems

We first experiment with the original set partitioning problem formulation (Equations (1)-(3)) for classification problems. The results are presented in Table 5. All the reported results are averages over 10 folds, and the numbers in parenthesis are standard deviations. We see that RF performs well on the iris, mammography, and WDBC datasets. However, the number of rules contained in RFs is very high for each dataset, which significantly degrades the interpretability. We observe that the proposed set partitioning formulation dramatically reduces the number of rules while keeping the macro-F1 values at acceptable levels for most datasets. One critical issue with the approach is that the macro-F1 value of the extracted rules is highly dependent on the macro-F1 value of the corresponding RF. For example, the macro-F1 value significantly deteriorates for glass and liver datasets, where the initial RF models do not perform well. However, this result is not surprising because the fraction of the test points missed by the extracted rules is also higher for these datasets. In addition, we know that the glass and mammography datasets are imbalanced in terms of class distribution. Therefore, the class imbalance problem should be handled carefully before the RF training and rule extraction processes.

We also observe that the average runtime of the branch-and-bound algorithm for all datasets is quite low. The highest runtime is observed for the mammography dataset, which has 11183 instances. However, the average runtime is less than six minutes. We also note that all the instances are solved to optimality within the time limit.

We then run experiments with the modified objective function (Equations (4), (2), and (3)). As mentioned before, this new objective function is introduced to reduce the number of extracted rules further. The results are summarized in Table 6. The first and most critical observation is that the number of extracted rules is significantly reduced with the modified objective function (i.e., Equation (4)). In contrast, the same level of macro-F1 value of the extracted rules is maintained compared to the previous formulation (i.e., Equation (1)). We also see an improvement in terms of coverage of the test instances (see the sixth column in Table 6). However, we notice an increase in the runtimes of the branch-and-bound algorithm. Some instances for mammography and liver datasets cannot be solved within the time limit. In addition, we observe high standard deviations in runtimes for these datasets. When we scrutinize these cases, we see that, for some replications, the number of rules in RFs is high, which results in a high number of columns of A, and thus, the high number of decision variables. Since the set partitioning problem is NP-hard, any increase in the dimension of the problem exponentially increases the runtime of the solution procedure.

3.2 Results for regression problems

We run the second set of experiments for regression datasets. We first use the original set partitioning problem formulation (Equations (1)-(3)). The numerical results are given in Table 7. We note that the scale of the error values depends on the scale of the outputs of each dataset. Therefore, these numbers are specific for each dataset. We first observe that RFs tend to generate large number of rules compared to classification datasets. We also see that the original set partitioning formulation helps us to reduce the number of rules while allowing a slight increase in error values. However, for boston, auto_mpg, and concrete datasets, we still have large number of extracted rules, which results in reduced interpretability. Except for wine_white dataset, all of the instances are solved to optimality. Another important result is that the coverage of the extracted rules is satisfactory (see the sixth column in Table 7).

Table 8 shows the results when we use the set partitioning model with the modified objective function (i.e., Equations (5), (2), and (3)). We observe a significant reduction in the number of extracted rules compared to the case where we use the original set partitioning formulation. While reducing the number of rules, we also see a slight reduction in the error values. However, the reduction in the number of rules comes at a cost; we observe high runtimes for all datasets. For boston, wine_white, and concrete datasets, we detect some instances which cannot be solved to optimality within the given time limit. However, incumbent solutions still provide accurate results for those datasets.

Table 5. Results of the experiments for classification problems with Equations (1)-(3).

Dataset	Random Forest		Extracted Rules			
	Macro-F1	Number of Rules	Macro-F1	Number of Rules	Fraction of Missed Points	Runtime (sec)
iris	0.95 (0.05)	442.20 (136.14)	0.93 (0.07)	16.50 (3.41)	0.04 (0.03)	0.02 (0.02)
mammography	0.81 (0.04)	5250.00 (2486.07)	0.77 (0.04)	80.10 (16.31)	0.00 (0.00)	346.88 (393.78)
glass	0.70 (0.19)	3951.50 (1672.59)	0.44 (0.24)	110.20 (27.03)	0.21 (0.07)	0.08 (0.03)
WDBC	0.96 (0.04)	1547.56 (753.17)	0.90 (0.02)	48.11 (11.14)	0.06 (0.04)	0.26 (0.21)
liver	0.72 (0.10)	4978.36 (2596.59)	0.56 (0.16)	179.73 (94.86)	0.20 (0.12)	4.41 (12.00)

Table 6. Results of the experiments for classification problems with Equations (4), (2), and (3).

Dataset	Random Forest		Extracted Rules			
	Macro-F1	Number of Rules	Macro-F1	Number of Rules	Fraction of Missed Points	Runtime (sec)
iris	0.95 (0.05)	498.20 (270.98)	0.93 (0.06)	3.30 (0.48)	0.00 (0.00)	0.02 (0.01)
mammography	0.82 (0.04)	6000.00 (1748.01)	0.78 (0.07)	36.40 (5.08)	0.00 (0.00)	798.10 (463.41)
glass	0.71 (0.12)	3877.20 (1466.59)	0.50 (0.23)	15.80 (2.49)	0.07 (0.08)	4.62 (4.32)
WDBC	0.96 (0.04)	2185.00 (756.90)	0.94 (0.04)	12.20 (1.23)	0.02 (0.02)	18.87 (13.30)
liver	0.70 (0.08)	5124.70 (2389.96)	0.58 (0.10)	35.60 (11.25)	0.10 (0.07)	989.64 (448.47)

Table 7. Results of the experiments for regression problems with Equations (1)-(3).

Dataset	Random Forest		Extracted Rules			
	Error	Number of Rules	Error	Number of Rules	Fraction of Missed Points	Runtime (sec)
boston	3.27 (0.74)	6000.00 (2108.19)	5.12 (1.41)	121.70 (26.97)	0.12 (0.04)	3.87 (2.77)
wine_white	0.71 (0.03)	5750.00 (2058.18)	0.78 (0.04)	56.50 (12.00)	0.00 (0.00)	798.07 (512.02)
auto_mpg	2.77 (0.36)	5500.00 (2297.34)	3.74 (0.81)	89.80 (25.61)	0.10 (0.08)	0.15 (0.06)
airfoil	4.16 (0.20)	5744.50 (2367.92)	6.68 (0.35)	7.60 (4.93)	0.00 (0.00)	0.72 (0.25)
concrete	7.23 (0.56)	6250.00 (1767.77)	9.69 (1.24)	62.30 (14.37)	0.02 (0.01)	25.29 (23.72)

Table 8. Results of the experiments for regression problems with Equations (5), (2), and (3).

Dataset	Random Forest		Extracted Rules			
	Error	Number of Rules	Error	Number of Rules	Fraction of Missed Points	Runtime (sec)
boston	3.27 (0.74)	6000.00 (2108.19)	4.73 (0.76)	24.90 (2.51)	0.04 (0.03)	943.00 (461.52)
wine_white	0.71 (0.03)	5750.00 (2058.18)	0.77 (0.03)	37.80 (3.97)	0.00 (0.00)	841.27 (472.28)
auto_mpg	2.77 (0.36)	5500.00 (2297.34)	3.69 (0.65)	18.40 (3.69)	0.03 (0.03)	40.10 (50.51)
airfoil	4.16 (0.19)	4747.50 (2185.50)	6.67 (0.37)	8.90 (3.78)	0.00 (0.00)	1.22 (2.13)
concrete	7.24 (0.66)	5500.00 (1972.03)	9.67 (0.58)	26.80 (3.74)	0.01 (0.01)	758.26 (572.62)

4 Conclusion

In this study, we propose a collection of different set partitioning formulations to extract rules from Random Forest classification and regression models. The proposed approach does not require a preprocessing step and is parameter-free. In addition, it aims to extract accurate rules whose union covers the input space of the problem as much as possible while keeping the intersections at a minimum.

First, we use the original set partitioning formulation for experimentation. Although the problem is NP-hard, we obtain optimal solutions for most of the datasets within the given time limit. We observe a significant reduction in the number of rules with acceptable deterioration in macro-F1 values. However, for imbalanced classification problems, we see that the macro-F1 value of the extracted rules is reduced when the initial Random Forest models does not perform well. Therefore, one might need to incorporate some mechanisms to handle class imbalance to obtain an accurate set of extracted rules. We also observe that the accuracy of the extracted rules is low if the fraction of the missed test instances is high.

We also obtain very promising results when we modify the objective function to reduce the number of extracted rules further. The modified formulation not only reduces the number of rules but also increases the coverage of test instances without loss of accuracy. Therefore, we can conclude that the modified objective functions yield better accuracy, number of rules, and coverage. However, these improvements come at a cost; we observe increased runtimes when we use the modified objective functions within the set partitioning problem. We note that the runtimes are still at acceptable levels, and incumbent solutions can still provide satisfactory solutions.

We also provide example R programs for classification and regression to enable researchers to implement the approach proposed in this paper [27].

5 Author contribution statements

In the scope of this study, Mert EDALİ contributed to the formation of the idea, the literature review, the design and analysis of computer experiments, and the writing of the manuscript.

6 Ethics committee approval and conflict of interest statement

There is no need to obtain permission from the ethics committee for the article prepared.

There is no conflict of interest with any person / institution in the article prepared.

7 References

[1] Boulesteix AL, Janitza S, Kruppa J, König IR. "Overview of random forest methodology and practical guidance with emphasis on computational biology and bioinformatics". *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(6), 493-507, 2012.

[2] Masetic Z, Subasi A. "Congestive heart failure detection using random forest classifier". *Computer Methods and Programs in Biomedicine*, 130, 54-64, 2016.

[3] Jog A, Carass A, Roy S, Pham DL, Prince JL. "Random forest regression for magnetic resonance image synthesis". *Medical Image Analysis*, 35, 475-488, 2017.

[4] Belgiu M, Drăguţ L. "Random forest in remote sensing: A review of applications and future directions". *ISPRS Journal of Photogrammetry and Remote Sensing*, 114, 24-31, 2016.

[5] Baydogan MG, Runger G, Tuv E. "A bag-of-features framework to classify time series". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11), 2796-2802, 2013.

[6] Breiman L. "Random forests". *Machine Learning*, 45(1), 5-32, 2001.

[7] Mashayekhi M, Gras R. "Rule extraction from random forest: the RF + HC methods". *Canadian Conference on Artificial Intelligence*, Halifax, NS, Canada, 2-5 June 2015.

[8] Mashayekhi M, Gras R. "Rule extraction from decision trees ensembles: new algorithms based on heuristic search and sparse group lasso methods". *International Journal of Information Technology & Decision Making*, 16(6), 1707-1727, 2017.

[9] Liu S, Patel RY, Daga PR, Liu H, Fu G, Doerksen RJ, Chen Y, Wilkins DE. "Combined rule extraction and feature elimination in supervised classification". *IEEE Transactions on Nanobioscience*, 11(3), 228-236, 2012.

[10] Adnan MN, Islam MZ. "Forex++: A new framework for knowledge discovery from decision forests". *Australasian Journal of Information Systems*, 2017. https://doi.org/10.3127/ajis.v21i0.1539

[11] Phung LTK, Chau VTN, Phung NH. "Extracting rule RF in educational data classification: from a random forest to interpretable refined rules". *2015 International Conference on Advanced Computing and Applications (ACOMP)*, Ho Chi Minh City, Vietnam, 23-25 November 2015.

[12] Meinshausen N. "Node harvest". *The Annals of Applied Statistics*, 4(4), 2049-2072, 2010.

[13] Friedman JH, Popescu BE. "Predictive learning via rule ensembles". *The Annals of Applied Statistics*, 2(3), 916-954, 2008.

[14] Deng H. "Interpreting tree ensembles with inTrees". *International Journal of Data Science and Analytics*, 7(4), 277-287, 2019.

[15] Wang S, Wang Y, Wang D, Yin Y, Wang Y, Jin Y. "An improved random forest-based rule extraction method for breast cancer diagnosis". *Applied Soft Computing*, 86, 105941, 1-18, 2020.

[16] Marsten RE, Shepardson F. "Exact solution of crew scheduling problems using the set partitioning model: Recent successful applications". *Networks*, 11(2), 165-177, 1981.

[17] Baldacci R, Christofides N, Mingozzi A. "An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts". *Mathematical Programming*, 115(2), 351-385, 2008.

[18] Garfinkel RS, Nemhauser GL. "The set-partitioning problem: Set covering with equality constraints". *Operations Research*, 17(5), 848-856, 1969.

[19] Dua D, Graff C. "UCI Machine Learning Repository". http://archive.ics.uci.edu/ml (08.07.2020).

[20] Carnegie Mellon University. "StatLib-Datasets Archive". http://lib.stat.cmu.edu/datasets/boston (08.07.2020).

[21] Woods KS, Doss CC, Bowyer KW, Solka JL, Priebe CE, Kegelmeyer Jr WP. "Comparative evaluation of pattern recognition techniques for detection of microcalcifications in mammography". *International Journal of Pattern Recognition and Artificial Intelligence*, 7(6), 1417-1436, 1993.

[22] Liaw A, Wiener M. "Classification and Regression by randomForest". *R News*, 2(3), 18-22, 2002.

[23] R Foundation for Statistical Computing. "R: A language and environment for statistical computing". https://www.R-project.org/ (08.07.2020).

[24] Gurobi Optimization LLC. "Gurobi Optimizer Reference Manual". http://www.gurobi.com (08.07.2020).

[25] Lewis M, Kochenberger G, Alidaee B. "A new modeling and solution approach for the set-partitioning problem". *Computers & Operations Research*, 35(3), 807-813, 2008.

[26] Rasmussen MS. Optimisation-Based Solution Methods for Set Partitioning Models. PhD Thesis, Technical University of Denmark, Kgs. Lyngby, Denmark, 2011.

[27] RuleExtractionfromRFs. "Example Scripts for the Manuscript". https://github.com/mertedali/RuleExtractionfromRFs (25.10.2020).