A rigorous method to compare interpretability of rule-based algorithms.

Vincent Margot

A rigorous method to compare interpretability.

Vincent Margot
Chief Algorithm and Data Officer

Advestis, 69 boulevard Haussmann, F-75008 Paris

Abstract

Interpretability is becoming increasingly important in predictive model analysis. Unfortunately, as mentioned by many authors, there is still no consensus on that idea. The aim of this article is to propose a rigorous mathematical definition of the concept of interpretability, allowing fair comparisons between any rule-based algorithms. This definition is built from three notions, each of which being quantitatively measured by a simple formula: predictivity, stability and simplicity. While predictivity has been widely studied to measure the accuracy of predictive algorithms, stability is based on the Dice-Sorensen index to compare two sets of rules generated by an algorithm using two independent samples. Simplicity is based on the sum of the length of the rules deriving from the generated model. The final objective measure of the interpretability of any rule-based algorithm ends up as a weighted sum of the three aforementioned concepts. This paper concludes with the comparison of the interpretability between several rule-based algorithms, CART, RuleFit, Node harvest, Coverin Algorithm and SIRUS for the regression settings and CART, PART and RIPPER for the classification settings.

Keywords: Interpretability, Transparency, Explainability, Predictivity, Stability, Simplicity, Rule-based algorithms, Machine Learning.

1 Introduction

The widespread use of machine learning (ML) in many sensible areas such as healthcare, justice or asset management has underlined the importance of interpretability in the decision-making process. In recent years, the number of publications on interpretability has increased exponentially. For a complete overview of the interpretability in ML, see the book [33]. Usually, two main ways can be distinguished for the production of interpretable predictive models. The first one relies on the use of an uninterpretable machine learning algorithm to create predictive models, and then to take them up again to create a so-called post-hoc interpretable model, for example LIME [37], DeepLIFT [39], SHAP [28]. These explanatory models try to measure the importance of a feature on the prediction process (see [19] for an overview of existing methods). However, as outlined in [38], the explanations may not be sufficient for a sensitive decision-making process.

The other way is to use an intrinsically interpretable algorithm to directly generate an interpretable model. There exists two familly of intrinsically interpretable algorithm: the tree-based algorithms such as CART [3], ID3 [35], C4.5 [36], M5P [42], LMT [25] and the rule-based algorithms such as RIPPER [4], FORS [24], M5 Rules [22], RuleFit [13], Ender [6], Node Harvest [32] or more recently SIRUS [1, 2], RIPE [29] and Covering Algorithm [30].

These algorithms are based on the notion of rule. A rule is a If-Then statement of the form

$$\begin{aligned} \text{IF} \quad & c_1 \text{ And } c_2 \text{ And } \dots \text{ And } c_k \\ \text{THEN} \quad & \text{Prediction} = p, \end{aligned} \tag{1}$$

The condition part If is a logical conjunction, where c_i's are tests that check whether the observation has the specified properties or not. The number k is called the length of the rule. If all c_i's are fulfilled the rule is said activated. And the conclusion part Then is the prediction of the rule if it is activated.

For now, there is no rigorous mathematical foundation for the concept of interpretability. This is because behind the interpretability are hiding many different concepts as explained in [27], [7], [43] and [34]. However, some concepts such as fairness, ethics and morals which are linked to specific applications such as justice, mortgage or even healthcare, cannot be quantitatively measured.

As proposed in [43, 1], we present an interpretability measure for any tree algorithms and rule-based algorithms based on the triptych predictability, stability, simplicity: Predictability measures the accuracy of the predictive model. The accuracy ensures a trustfulness in the generated model. Stability quantifies the noise sensitivity of an algorithm. It allows to evaluate the robustness of the algorithm. Simplicity could be construed as the capacity to audit the prediction easily. A simple model ensures that it is easy to evaluate some qualitative criteria such as ethics and morals.

Hence, by measuring theses three quantities we are able to classify levels of interpretability of a set of algorithms for a given problem.

2 Predictivity

The aim of a predictive model is to predict the value of a variable of interest $Y \in \mathcal{Y}$, given features $X \in \mathcal{X}$ where $\mathcal{X}$ is a d-dimensional space. Formally, we set the standard regression setting as follows: Let (X, Y) be a couple of random variables in $\mathcal{X} \times \mathcal{Y}$ of unknown distribution $\mathbb{Q}$ such that

$$Y = g^*(X) + Z, \tag{2}$$

where $\mathbb{E}[Z] = 0$ and $\mathbb{V}(Z) = \sigma^2$ and g^* is a measurable function from $\mathcal{X}$ to $\mathcal{Y}$.

We denote by $\mathcal{G}$ the set of all measurable functions from $\mathcal{X}$ to $\mathbb{R}$. The accuracy of a regression function $g \in \mathcal{G}$ is measured by its risk, defined as

$$\mathcal{L}(g) := \mathbb{E}_{\mathbb{Q}} \left[\gamma \left(g; (X, Y) \right) \right], \tag{3}$$

where $\gamma : \mathcal{G} \times (\mathcal{X} \times \mathcal{Y}) \to [0, \infty[$ is called a contrast function. The risk measures the average discrepancy, given a new observation (X, Y) from the distribution $\mathbb{Q}$, between $g(X)$ and Y.

The choice of γ depends on the nature of Y. For example, if $Y \in \mathbb{R}$, one generally uses the quadratic contrast with $\gamma \left(g; (X, Y) \right) = \left(g(X) - Y \right)^2$. The minimizer of the risk (3) with the quadratic contrast is called the regression function $\eta \in \mathcal{G}$, defined by

$$\eta(x) := \mathbb{E}_{\mathbb{Q}} \left[Y \mid X = x \right].$$

If $Y \in \{0, 1\}$, one uses the $0-1$ contrast function $\gamma \left(g; (X, Y) \right) := \mathbf{1}_{g(X) \neq Y}$. The minimizer of the risk (3) with the $0-1$ contrast function is called the Bayes classifier $s \in \mathcal{G}$ defined by

$$\eta(x) := \mathbf{1}_{\mathbb{Q}(Y=1 \mid X=x) \geq 1/2}.$$

Given a sample $D_n = ((X_1, Y_1), \ldots, (X_n, Y_n))$, we aim at predicting Y conditionally on X. The observations (X_i, Y_i) are independent and identically distributed (i.i.d) from the distribution $\mathbb{Q}$.

To do so, we consider a statistical algorithm $\mathcal{A}$ which is a measurable mapping defined by

$$\begin{array}{cccc} \mathcal{A} & : & (\mathcal{X} \times \mathcal{Y})^n & \to & \mathcal{G}_n^{\mathcal{A}} \\ & & D_n & \mapsto & g_n^{\mathcal{A}}, \end{array} \tag{4}$$

where $\mathcal{G}_n^{\mathcal{A}} \subseteq \mathcal{G}$.

The purpose of an algorithm $\mathcal{A}$, is to generate a measurable function $g_n^{\mathcal{A}}$ that minimizes the risk (3). To carry out this minimization, the algorithms use the Empirical Risk Minimization principle (ERM) [41], meaning that

$$g_n^{\mathcal{A}} = \underset{g \in \mathcal{G}_n^{\mathcal{A}}}{\arg\min} \, \mathcal{L}_n(g), \tag{5}$$

where $\mathcal{L}_n(g) = \frac{1}{n} \sum_{i=1}^{n} \gamma(g, (X_i, Y_i))$ is the empirical risk.

Hence, according to the ERM principle, the choice of γ determines the function that an algorithm, $\mathcal{A}$ tries to estimate, and thus the function $g_n^{\mathcal{A}}$.

The notion of predictivity is based on the ability of an algorithm to provide an accurate predictive model. This notion has been well studied since years. In this paper, the predictivity is defined as follows:

$$\mathcal{P}(g_n^{\mathcal{A}}, \gamma) := 1 - \frac{\mathcal{L}(g_n^{\mathcal{A}})}{\mathcal{L}(h_\gamma)}, \tag{6}$$

where h_γ is the trivial constant predictor according to γ. For example, for the quadratic contrast $h_\gamma = \mathbb{E}[Y]$ and for the $0-1$ contrast $h_\gamma = median(Y)$.

This quantity as a measure of the accuracy is independent from the range of Y. We may assume that it is a positive number between 0 and 1. Indeed, the risk (3) is a positive function and if $\mathcal{P}_n(g_n^{\mathcal{A}}, \gamma) > 1$, it means that the predictor $g_n^{\mathcal{A}}$ is worse than the trivial constant predictor.

3 q-Stability

In [1, 2], authors have proposed a measure of the stability of a rule-based algorithm built upon the following definition:

A rule learning algorithm is stable if two independent estimations based on two independent samples result in two similar lists of rules.

The notion of q-stability is based on the same definition. This notion appears to be fairer for algorithms that do not use features discretization and operate on real values rather than integer values. In fact, if the feature is continuous, the probability that a decision tree algorithm cuts on the same exact value for the same rule, given two independent samples is null. For this reason, the pure stability appears too penalizing in this case.

In order to avoid this unfairness, we discretize all continuous features. Features discretization is a common solution for controlling the complexity of a rule generator. In [10], for example, the authors use the entropy minimization heuristic to discretize the features and for the algorithms BRL (Bayesian Rule Lists) [26], SIRUS [1] and RICE [31], authors have used the empirical quantiles of features to discretize them. See [8] for an overview of usual discretization methods.

Let $q \in \mathbb{N}$ be the number of quantiles considered for the discretization and let X be a continuous feature. An integer $p \in \{1, \dots, q\}$, named bin, is associated to each interval $[x_{(p-1)/q}, x_{p/q}]$, where $x_{p/q}$ is p-th q-quantile of X. A discrete version of a feature X, denoted $Q_q(X)$, is designed by replacing each value by its corresponding bins. in other words, a value p_a is associated for all $a \in X$ such that $a \in [x_{(p_a-1)/q}, x_{p_a/q}]$.

This discretization process can be extended to a rule set by replacing for all rules, the intervals' bound of each test c_i by their corresponding bins. For example, the test $X \in [a, b]$ becomes $Q_q(X) \in [\![p_a, p_b]\!]$, where p_a and p_b are such that $a \in [x_{(p_a-1)/q}, x_{p_a/q}]$ and $b \in [x_{(p_b-1)/q}, x_{p_b/q}]$.

The formula of the q-stability is based on the so-called Dice-Sorensen index. Let $\mathcal{A}$ be a rule-based algorithm and let D_n and D'_n two independent samples of n i.i.d observations drawn from the same distribution $\mathbb{Q}$. And let $R_n^{\mathcal{A}}$ and $R_n'^{\mathcal{A}}$ be the sets of rules generated by $\mathcal{A}$, given D_n and D'_n respectively. Then, the q-stability is calculated by

$$\mathcal{S}_n^q(\mathcal{A}) := \frac{2\left|Q_q(R_n^{\mathcal{A}}) \cap Q_q(R_n'^{\mathcal{A}})\right|}{|Q_q(R_n^{\mathcal{A}})| + |Q_q(R_n'^{\mathcal{A}})|}, \tag{7}$$

where $Q_q(R)$ is the discretized version of the rule-set R and with the convention that $0/0 = 0$. The discretization process is performed using D_n and D'_n

respectively.

This quantity is the ratio of common rules in $Q_q(R_n^{\mathcal{A}})$ and $Q_q(R_n'^{\mathcal{A}})$. It is a positive number between 0 and 1: If $Q_q(R_n^{\mathcal{A}})$ and $Q_q(R_n'^{\mathcal{A}})$ have no common rules, then $\mathcal{S}_n^q(\mathcal{A}) = 0$ while if $Q_q(R_n^{\mathcal{A}})$ and $Q_q(R_n'^{\mathcal{A}})$ have the same rules, then $\mathcal{S}_n^q(\mathcal{A}) = 1$.

4 Simplicity

In [30], authors have introduced a notion of interpretability index based on the sum of the length of all the rules constituting the predictive model. Such interpretability index is not to be confused with the broader concept of interpretability developed in this paper. As will be stated in Section 5 the former will be construed as one of the components of the latter.

Definition 4.1. *The interpretability index of an estimator g_n generated by a set of rules R_n is defined by*

$$Int(g_n) := \sum_{r \in R_n} length(r). \tag{8}$$

Even if (8) seems naive, we consider it as good measure of simplicity of a tree algorithm or a rule-based algorithms. Indeed, if the number of rule or the length of the rules increase so do $Int(g_n)$. The fewer the number of rules and their length the better the understanding.

Furthermore, the value (8), which is a positive number, cannot be directly compared to the values from (6) and (7), which are between 0 and 1.

The measure of the simplicity is based on the definition 4.1. The idea is to compare (8) relatively to a set of algorithms $\mathcal{A}_1^m = \{\mathcal{A}_1, \dots, \mathcal{A}_m\}$. Hence the simplicity of an algorithm $\mathcal{A}_i \in \mathcal{A}_1^m$ is defined in relative terms as follows:

$$\mathbb{S}_n(\mathcal{A}_i, \mathcal{A}_1^m) = \frac{min\{Int(g_n^A : A \in \mathcal{A}_1^m)\}}{Int(g_n^{\mathcal{A}_i})}. \tag{9}$$

Like the previous ones, this quantity is a positive number between 0 and 1: If $\mathcal{A}_i$ generates the simplest predictor among the set of algorithms $\mathcal{A}_1^m$ then $\mathbb{S}_n(\mathcal{A}_i, \mathcal{A}_1^m) = 1$. Then, the simplicity of other algorithms in $\mathcal{A}_1^m$ are evaluated relatively to $\mathcal{A}_i$.

5 Interpretability

In [7], the authors define interpretability as "the ability to explain or to present in understandable terms to a human". We claim that an algorithm with a high predictivity (6), stability (7) and simplicity (9) are interpretable in the sense of [7]. Indeed, high predictivity ensures trust, high stability ensures robustness and a high simplicity ensures that the generated model can be easily understood by humans because there are few rules with small lengths.

The main idea underlying the proposed definition of interpretability, is the use of a weighted sum of the predictivity (6) the stability (7) and the simplicity (9). Let $\mathcal{A}_1^m$ be a set of algorithms, the interpretability of any algorithm $\mathcal{A}_i \in \mathcal{A}_1^m$ is defined by:

$$\mathcal{I}(\mathcal{A}_i, D_n, D'_n, \gamma, q) = \alpha_1 \mathcal{P}(g_n^{\mathcal{A}_i}, \gamma) + \alpha_2 \mathcal{S}_n^q(\mathcal{A}_i) + \alpha_3 \mathbb{S}_n(\mathcal{A}_i, \mathcal{A}_1^m), \tag{10}$$

where the coefficients α_1, α_2 and α_3 have been chosen according to the statistician's desiderata such that $\alpha_1 + \alpha_2 + \alpha_3 = 1$. If a statistician tries to understand and to describe a phenomenon then simplicity and predictivity are more important than stability.

It is important to notice that the definition of interpretability (10) depends on the set of rule-based algorithms and the choice of a given regression setting. Therefore, the interpretable value only makes sense within that set of algorithms and for a specific regression setting. As the statistician's desiderata get refined, it is naturally possible to broaden the choice of $\mathcal{I}$ and consider other choices for $\mathcal{I}$ under constraint of possitive partial derivatives for each of the components.

6 Application

The aim of this application is to compare several algorithms considered as interpretable: Regression Tree [3], RuleFit (RF) [13], NodeHarvest (NH) [32], Covering Algorithm (CA) [30] and SIRUS [2] for regression settings and RIPPER [4], PART [12] and Classification Tree [3] for classification settings[1].

6.1 Brief overview of chosen algorithms

RIPPER (Repeated Incremental Pruning to Produce Error Reduction), is a sequential coverage algorithm. It is based on the "separate-and-conquer" approach. This means that for a chosen class, it searches for the best rule according to a criterion and removes the points covered by that rule. Then it looks for the best rule for the remaining points. And so on until all points of that class are covered. Thereafter it proceeds to the next class. Classes are examined in growing size.

PART is also a "separate-and-conquer" rule learner. The main difference is that to create the "best rule", the algorithm uses a pruned Decision Tree and keeps the leaf with the largest coverage.

RuleFit is a very accurate rule-based algorithm. First it generates a list of rules by considering all nodes and leaves of a boosted tree ensemble ISLE [14]. Then rules are used as additional features in a sparse linear regression model Lasso [40].

Node harvest also based uses a tree ensemble as rule generator. The algorithm considers all nodes and leaves of a Random Forest as rules and solves a

[1]We have excluded algorithms designed only for binary classification such as M5Rules [23], NodeHarvest [32], SIRUS [1]

linear quadratic problem to fit a weight to each node. Hence, the estimator is convex combination of nodes.

Covering Algorithm has been designed to generate a very simple model. The algorithm extracts a sparse set of rules considering all nodes and leaves of a tree ensembles (using Random Forest algorithm, Gradient Boosting algorithm [15] or Stochastic Gradient Boosting algorithm [16]). Rules are selected according to theirs statistical properties to form a "quasi-covering". The covering is then turned into a partition using the so-called partitioning trick [29] to form a consistent estimator of the regression function.

SIRUS (Stable and Interpretable RUle Set) has been designed to be a stable predictive algorithm. SIRUS uses a modified Random Forest to generate a large number of rules and selects rules with a redundancy greater than a tuning parameter p_0. To be sure to have redundancy, features are discretized.

For a review of rule-based algorithms see [18, 17] and for a review of interpretable machine learning see [33].

6.2 Datasets

We have used public databases from the UCI Machine Learning Repository [9] and from [21]. We have chosen eight datasets for regression summarized in Table 1 and four datasets for classification summarized in Table 2. For the dataset Student for regression we have removed variables $G1$ and $G2$ which are the first and the second grade respectively because the target attribute $G3$ has a strong correlation with attributes $G2$ and $G1$. In [5] authors say that it is more difficult to predict $G3$ without $G2$ and $G1$, but such prediction is much more useful.

Name	$(n \times d)$	Description
Ozone	330×9	Prediction of atmospheric ozone concentration from daily meteorological measurements [21].
Machine	209×8	Prediction of published relative performance [9].
MPG	398×8	Prediction of city-cycle fuel consumption in miles per gallon [9].
Boston	506×13	Prediction of the median price of neighborhoods, [20].
Student	649×32	Prediction of the final grade of student based on attributes collected by reports and questionnaires [5].
Abalone	4177×7	Prediction of the age of abalone from physical measurements [9].

Table 1: Presentations of the public regression datasets used in this paper.

Name	$(n \times d)$	Description
Wine	4898×11	Classification of withe wine quality from 0 to 10 [9].
Student	131×21	Classification of the end semester percentage good, average or poor, based on different social, economic and academic attributes [9].
Covertype	581012×54	Classification of forest cover type $[1, 7]$ based on cartographic variables [9].
Speaker	329×12	Classification of accent, six possibilities, based on features extracted from 1s of reading of a word [11].

Table 2: Presentations of the public classification datasets used in this paper.

6.3 Execution

For each dataset, we realize 10 experiments. For each experiment, the data are randomly split into a training set and a test set, with a ratio of 80%/20%, respectively. The algorithms parameter settings is summarized in Table 3. For each algorithm, a model is fitted on the training set to calculate the simplicity (8) and we evaluate the predictivity (6) on the test set. Then the training set is randomly split into two sets of the same length and two models are fitted to evaluate the stability (7). At the end, we calculate the interpretability score (10) with $\alpha_1 = \alpha_2 = \alpha_3 = 1/3$ for each experiment. The code is a mix between Python and R. It is available on GitHub https://github.com/Advestis/Interpretability.

Algorithm	Parameters
CART	$max\_leaf\_nodes = 20$.
RuleFit	$tree\_size = 4$, $max\_rules = 2000$.
Node harvest	$max.inter = 3$.
CA	$generator\_func = RandomForestRegressor$, $n\_estimators = 500$, $max\_leaf\_nodes = 4$, $alpha = 1/2 - 1/100$, $gamma = 0.95$, $k\_max = 3$
SIRUS	$max.depth = 3$, $num.rule = 10$.

Table 3: Algorithms parameter settings.

6.4 Results for regression

The averaged scores are summarized in Table 4. As expected RuleFit is the most accurate algorithm. Nevertheless, RuleFit is nor stable nor simple. SIRUS is the stablest algorithms and Covering Algorithm one the simplest. But, for all datasets SIRUS seems to be the most interpretable algorithm among this selection of algorithms and according to our measure (10) with $\alpha_1 = \alpha_2 = \alpha_3 = 1/3$.

6.5 Results for classification

The averaged scores are summarized in Table 5. All chosen algorithms have the same accuracy on all datasets. Nevertheless, RIPPER and PART are both very stable algorithms. And RIPPER is the simplest of the three. Hence, for this datasets and among this three algorithms, RIPPER is the most interpretable algorithm according to our measure (10) with $\alpha_1 = \alpha_2 = \alpha_3 = 1/3$.

7 Conclusion and perspectives

In this paper, a quantitative criterion for interpretability of tree-based algorithms and rule-based algorithms was presented. This measure is based on the triptych: Predictivity (6), Stability (7) and Simplicity (9) as proposed in [43, 1]. This concept of interpretability has been thought to be fair and rigorous. It can be adapted to the various desiderata of the statistician by choosing appropriate the coefficients in the interpretability formula (10).

The methodology described in our paper for comparing the interpretability of tree-based algorithms and rule-based algorithms seems to provide a fair measure for classifying the interpretability of a set of algorithms. In fact, it is composed of three different scores that allow to integrate the main goals of interpretability. It can be seen that the q-stability score and the simplicity score appear to be stable regardless of the data sets. In fact, these scores are linked to how the algorithms work. An algorithm designed to be accurate, stable or simple should keep this property whatever the dataset.

However, according to Definition 4.1, 100 rules of length 1 have the same simplicity that one single rule of length 100, which is debatable. Moreover, the stability measure is purely syntactic and rather restrictive. Indeed, if some features are duplicated, two rules may have two different syntactic conditions but by otherwise identical based in their activations. One way of relaxing this stability criterion could be to compare the rules, based on their activation sets (i.e. by looking to observations where conditions are met simultaneously).

In a future work we will extend the measure of interpretability to other well-known machine learning algorithms such as linear regression or other machine learning problems such as clustering or dimension reduction such as the Principal Component Analysis (PCA) and the Non Negative Matrix/Tensor

Dataset	$\mathcal{P}_n$				
	RT	RuleFit	Node harvest	CA	SIRUS
Ozone	0.60	**0.75**	**0.73**	0.62	0.65
Machine	**0.78**	**0.82**	0.69	0.46	0.66
MPG	**0.81**	**0.88**	**0.85**	0.72	0.76
Boston	0.73	**0.88**	0.78	0.56	0.67
Student	0.0	**0.23**	**0.25**	0.14	0.25
Abalone	0.47	**0.56**	0.44	0.40	0.33

Dataset	$\mathcal{S}_n^q$				
	RT	RuleFit	Node harvest	CA	SIRUS
Ozone	0.88	0.12	**0.97**	0.18	**0.96**
Machine	0.77	0.16	**0.96**	0.03	**0.97**
MPG	0.92	0.18	**0.95**	0.33	**1.0**
Boston	0.83	0.16	**0.95**	0.15	**0.95**
Student	0.84	0.17	**0.99**	0.08	**1.0**
Abalone	0.87	0.18	**0.97**	0.22	**1.0**

Dataset	$\mathbb{S}_n$				
	RT	RuleFit	Node harvest	CA	SIRUS
Ozone	0.03	0.01	0.01	**0.87**	0.40
Machine	0.07	0.04	0.04	**0.96**	0.49
MPG	0.04	0.01	0.02	**1.0**	0.31
Boston	0.06	0.01	0.02	**1.0**	0.60
Student	0.10	0.06	0.09	0.42	**1.0**
Abalone	0.15	0.02	0.06	0.59	**1.0**

Dataset	$\mathcal{I}$				
	RT	RuleFit	Node harvest	CA	SIRUS
Ozone	0.50	0.29	0.57	0.56	**0.67**
Machine	0.51	0.34	0.54	0.48	**0.71**
MPG	0.59	0.36	0.61	**0.69**	**0.69**
Boston	0.54	0.35	0.58	0.57	**0.74**
Student	0.31	0.46	0.44	0.21	**0.75**
Abalone	0.50	0.25	0.49	0.40	**0.78**

Table 4: Average of predictivity score ($\mathcal{P}_n$), stability score ($\mathcal{S}_n^q$), simplicity score ($\mathcal{S}_n$) and interpretability score ($\mathcal{I}$) over a 10-fold cross-validation of usual interpretable algorithms for various regression public datasets. Best values are in bold, as well as values within 10% of the maximum for each dataset.

Factorization (NMF, NTF). Another interesting extension would be to add a semantic analyst of the variables involved in the rules. Indeed, using NLP methods one could measure a distance between the target and these variables in a text corpus. This distance could be interpreted as the relevance of using such

Dataset	$\mathcal{P}_n$			$\mathcal{S}_n^q$		
	CART	RIPPER	PART	CART	RIPPER	PART
Wine	0.53	0.53	0.58	0.93	**1.0**	**0.99**
Student	0.39	0.38	0.41	0.85	**1.0**	**1.0**
Covertype	0.53	0.53	0.58	0.93	**1.0**	**0.99**
Speaker	0.67	0.66	0.69	0.88	**1.0**	**1.0**

Dataset	$\mathbb{S}_n$			$\mathcal{I}$		
	CART	RIPPER	PART	CART	RIPPER	PART
Wine	0.79	**0.97**	0.03	0.75	**0.83**	0.53
Student	0.03	**1.0**	0.09	0.42	**0.79**	0.50
Covertype	0.78	**0.97**	0.03	**0.75**	**0.83**	0.53
Speaker	0.16	**1.0**	0.43	0.57	**0.89**	0.71

Table 5: Average of predictivity score ($\mathcal{P}_n$), stability score ($\mathcal{S}_n^q$), simplicity score ($\mathbb{S}_n$) and interpretability score ($\mathcal{I}$) over a 10-fold cross-validation of usual interpretable algorithms for various classification public datasets. Best values are in bold, as well as values within 10% of the maximum for each dataset.

variables to describe the target.

References

[1] C. Bénard, G. Biau, S. Da Veiga, and E. Scornet. Sirus: making random forests interpretable. arXiv preprint arXiv:1908.06852, 2019.

[2] C. Bénard, G. Biau, S. Da Veiga, and E. Scornet. Interpretable random forests via rule extraction. arXiv preprint arXiv:2004.14841, 2020.

[3] L. Breiman, J. Friedman, R. Olshen, and C. Stone. Classification and Regression Trees. CRC press, 1984.

[4] W. Cohen. Fast effective rule induction. In Machine Learning Proceedings 1995, pages 115–123. Elsevier, 1995.

[5] P. Cortez and A. M. G. Silva. Using data mining to predict secondary school student performance. In Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008), 2008.

[6] K. Dembczyński, W. Kotłowski, and R. Słowiński. Solving regression by learning an ensemble of decision rules. In International Conference on Artificial Intelligence and Soft Computing, pages 533–544. Springer, 2008.

[7] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608, 2017.

[8] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and unsupervised discretization of continuous features. In Machine Learning Proceedings 1995, pages 194–202. Elsevier, 1995.

[9] D. Dua and C. Graff. Uci machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

[10] U. M. Fayyad and K. B. Irani. Multi-interval discretization of continuous-valued attributes for classification learning. In Proc. 13th Int. Joint Conf. on Artificial Intelligence, pages 1022–1027, 1993.

[11] E. Fokoue. UCI machine learning repository, 2020. URL [WebLink].

[12] E. Frank and I. H. Witten. Generating accurate rule sets without global optimization. 1998.

[13] J. Friedman and B. Popescu. Predective learning via rule ensembles. The Annals of Applied Statistics, pages 916–954, 2008.

[14] J. Friedman, B. Popescu, et al. Importance sampled learning ensembles. Journal of Machine Learning Research, 94305, 2003.

[15] J. H. Friedman. Greedy function approximation: a gradient boosting machine. Annals of statistics, pages 1189–1232, 2001.

[16] J. H. Friedman. Stochastic gradient boosting. Computational statistics & data analysis, 38(4):367–378, 2002.

[17] J. Fürnkranz and T. Kliegr. A brief overview of rule learning. In International Symposium on Rules and Rule Markup Languages for the Semantic Web, pages 54–69. Springer, 2015.

[18] J. Fürnkranz, D. Gamberger, and N. Lavrač. Foundations of rule learning. Springer Science & Business Media, 2012.

[19] R. Guidotti, A. Monreale, F. Turini, D. Pedreschi, and F. Giannotti. A survey of methods for explaining black box models. arXiv preprint arXiv:1802.01933, 2018.

[20] D. Harrison Jr and D. L. Rubinfeld. Hedonic housing prices and the demand for clean air. Journal of environmental economics and management, 5(1): 81–102, 1978.

[21] T. Hastie, J. Friedman, and R. Tibshirani. The Elements of Statistical Learning, volume 1. Springer series in statistics Springer, Berlin, 2001.

[22] G. Holmes, M. Hall, and E. Prank. Generating rule sets from model trees. In Australasian Joint Conference on Artificial Intelligence, pages 1–12. Springer, 1999.

[23] K. Hornik, C. Buchta, and A. Zeileis. Open-source machine learning: R meets Weka. Computational Statistics, 24(2):225–232, 2009. doi: 10.1007/s00180-008-0119-7.

[24] A. Karalič and I. Bratko. First order regression. Machine Learning, 26 (2-3):147–176, 1997.

[25] N. Landwehr, M. Hall, and E. Frank. Logistic model trees. Machine learning, 59(1-2):161–205, 2005.

[26] B. Letham, C. Rudin, T. McCormick, and D. Madigan. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. The Annals of Applied Statistics, 9(3):1350–1371, 2015.

[27] Z. C. Lipton. The mythos of model interpretability. arXiv preprint arXiv:1606.03490, 2017.

[28] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. In Advances in Neural Information Processing Systems, pages 4765–4774, 2017.

[29] V. Margot, J.-P. Baudry, F. Guilloux, and O. Wintenberger. Rule induction partitioning estimator. In International Conference on Machine Learning and Data Mining in Pattern Recognition, pages 288–301. Springer, 2018.

[30] V. Margot, J.-P. Baudry, F. Guilloux, and O. Wintenberger. Consistent regression using data-dependent coverings. 2019.

[31] V. Margot, J.-P. Baudry, F. Guilloux, and O. Wintenberger. Rule induction covering estimator : A new data dependent covering algorithm. 2020.

[32] N. Meinshausen et al. Node harvest. The Annals of Applied Statistics, 4 (4):2049–2072, 2010.

[33] C. Molnar. Interpretable Machine Learning. Lulu.com, 2020.

[34] W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, and B. Yu. Interpretable machine learning: definitions, methods, and applications. arXiv preprint arXiv:1901.04592, 2019.

[35] J. R. Quinlan. Induction of decision trees. Machine learning, 1(1):81–106, 1986.

[36] J. R. Quinlan. C4.5: Programs for machine learning. 1993.

[37] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you?: Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pages 1135–1144. ACM, 2016.

[38] C. Rudin. Please stop explaining black box models for high stakes decisions. arXiv preprint arXiv:1811.10154, 2018.

[39] A. Shrikumar, P. Greenside, and A. Kundaje. Learning important features through propagating activation differences. arXiv preprint arXiv:1704.02685, 2017.

[40] R. Tibshirani. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society. Series B (Methodological), pages 267–288, 1996.

[41] V. Vapnik. The nature of statistical learning theory. Springer science & business media, 2013.

[42] Y. Wang and I. H. Witten. Induction of model trees for predicting continuous classes. 1997.

[43] B. Yu and K. Kumbier. Three principles of data science: predictability, computability, and stability (pcs). arXiv preprint arXiv:1901.08152, 2019.