Published in the *Proceedings of the 10th International Work-Conference on the Interplay Between Natural and Artificial Computation, IWINAC 2024, 4-7 June 2024, Olhâo, Portugal, Pt. II*, which should be cited to refer to this work.

DOI: https://doi.org/10.1007/978-3-031-61137-7_35

Fidex: an Algorithm for the Explainability of Ensembles and SVMs

Guido Bologna[0000−0002−6070−3459], Jean-Marc Boutay, Quentin Leblanc, and Damian Boquete

University of Applied Sciences and Arts of Western Switzerland
Rue de la Prairie 4, 1202 Geneva, Switzerland
Guido.Bologna@hesge.ch

Abstract. A natural way to explain neural network responses is by propositional rules. Currently, the state of the art on XAI methods presents local and global algorithms, with local techniques aiming to explain single samples in their neighbourhood. We present here Fidex, a new local algorithm, which we apply to ensembles of neural networks, ensembles of decision trees and support vector machines. The key idea behind Fidex is the precise identification of discriminating hyperplanes. Its computational complexity for a neural network is linear with respect to the product of: the dimensionality of the classification problem; the number of training samples; the maximal number of antecedents per rule; and a constant related to a particular activation function approximating a sigmoid function. Based on Fidex, we formulated a global algorithm named FidexGlo. Essentially, FidexGlo uses Fidex to generate a number of rules equal to the number of samples. Then, a heuristic is deployed to remove as many rules as possible. FidexGlo was applied to four benchmark classification problems, providing competitive results with our previous global rule extraction technique. Fidex and FidexGlo are available at https://github.com/HES-XPLAIN/dimlpfidex.

Keywords: Model Explainability · Ensembles · Rule Extraction.

1 Introduction

Before the advent of deep learning, the explainability of black-box models such as neural networks (NNs) or support vector machines (SVMs), was most of the time tackled by the extraction of propositional rules. A taxonomy characterizing rule extraction techniques into three main categories has long been valid [2]. Rule extraction strategies for SVMs that are functionally identical to multi-layer perceptrons (MLPs) were proposed in [7]. More recently, Guidotti et al. presented a review of black-box models with its "explanators", which encompasses deep models [9]. In addition, Adadi and Berrada proposed a description of explainable artificial intelligence (XAI), including neural networks [1].

Because deep models are more complicated than MLPs and SVMs, a key element of explainability approaches concentrates on the immediate region surrounding a sample [11]. This approach is defined as local. A global technique, on

the other hand, takes into account all the data that define a problem. Furthermore, many other techniques used in image classification visualize areas that are mainly relevant for the outcome [14].

Decision trees (DTs) are widely used in Machine Learning. They represent transparent models because symbolic rules are easily extracted. However, when they are combined in an ensemble, rule extraction becomes harder [12]. We introduce Fidex, a new local approach to rule extraction that can be applied to NN ensembles, DT ensembles, and SVMs. The key idea behind Fidex is the precise characterisation of discriminating hyperplanes. Its computational complexity for a single neural network is linear with respect to the product of: the dimensionality of the classification problem; the number of training samples; the maximal number of rule antecedents per rule; and a constant related to a particular activation function approximating a sigmoid function.

Based on Fidex, we have formulated a global algorithm called FidexGlo. Essentially, FidexGlo depends on Fidex to generate a number of rules equal to the number of samples. Then, a heuristic is deployed to remove as many rules as possible. We applied FidexGlo to four benchmark classification problems. The results show that FidexGlo is competitive with our previous global rule extraction algorithm [3], and that its computational complexity is more advantageous. The various models employed in this study, the Fidex and FidexGlo algorithms, the experiments, and the conclusion are presented in the following parts.

2 Models and Algorithms

In this section we present the models used in this work, which are DIMLP ensembles, Quantized Support Vector Machines, and ensembles of DTs. Subsequently we describe our local/global rule extraction algorithms.

2.1 DIMLPs

The Discretized Interpretable Multi Layer Perceptron is a particular feed-forward neural network architecture from which propositional rules are extracted by determining discriminatory hyperplanes. For a general MLP model, let us denote $x^{(0)}$ as a vector for the input layer. For layer $l+1$ $(l \geq 0)$, the activation values $x^{(l+1)}$ of the neurons are

$$x^{(l+1)} = F(W^{(l)}x^{(l)} + b^{(l)}). \tag{1}$$

$W^{(l)}$ is a matrix of weight parameters between two successive layers l and $l+1$; $b^{(l)}$ is a vector also called the bias and F is an activation function. Usually, F is a sigmoid $\sigma(x)$:

$$\sigma(x) = \frac{1}{1+\exp(-x)}. \tag{2}$$

In a DIMLP, $W^{(0)}$ is a diagonal matrix. Moreover, the activation function applied to $x^{(1)}$ is a staircase function $S(x)$. Here, $S(x)$ approximates with Θ

stairs a sigmoid function:

$$S(x) = \sigma(\alpha_{min}), \text{ if } x \leq \alpha_{min}; \tag{3}$$

α_{min} represents the abscissa of the first stair. By default $\alpha_{min} = -5$.

$$S(x) = \sigma(\alpha_{max}), \text{ if } x \geq \alpha_{max}; \tag{4}$$

α_{max} represents the abscissa of the last stair. By default $\alpha_{max} = 5$. Between α_{min} and α_{max}, $S(x)$ is:

$$S(x) = \sigma(\alpha_{min} + \left[\Theta \cdot \frac{x - \alpha_{min}}{\alpha_{max} - \alpha_{min}}\right] (\frac{\alpha_{max} - \alpha_{min}}{\Theta})); \tag{5}$$

with [ ] designating the integer-part function. For $l \geq 2$, DIMLP and MLP are the same model.

The discriminating axis-parallel hyperplanes are precisely identified thanks to the staircase activation function and to the $W^{(0)}$ diagonal matrix. Let us take an example with a step activation function, which is a simple special case. The step function is

$$t(x) = \begin{cases} 1 \text{ if } x > 0; \\ 0 \text{ otherwise.} \end{cases} \tag{6}$$

Figure 1 represents a DIMLP with an input neuron a hidden neuron and an output neuron. Below the network are samples belonging to two classes: circles and squares. Due to the step activation function of the hidden neuron we have a potential discriminant hyperplane in $-b/w$, which is equal to two. If the value of w' is zero, circles and the triangles cannot be distinguished, as the signal entering y will always be negative. With w' equal to 10 and b' equal to -5, if h is 0, the signal entering y is negative; else if h is equal to one the signal entering y is positive. We therefore have a hyperplane that allows us to define two propositional rules:

- if $x > 2$ then class is triangle;
- if $x \leq 2$ then class is circle.

With the staircase activation function, the maximum number of discriminating hyperplanes per input variable corresponds to the number of stairs Θ in equation 5. Our previous rule extraction algorithm (ORE) [3] was based on the construction of a DT with a hyperplane at each node. Each path from the root to a leaf represented a propositional rule. Finally, a greedy algorithm progressively removed rule antecedents and rules [3]. Its computational complexity is $O(d^4 \cdot \Theta^4 \cdot s^2)$ [3].

The position of the hyperplanes hp_i for each input variable x_i is

$$hp_i = \frac{1}{w_i} \cdot (\ \ (\alpha_{min} - b_i) + \frac{k}{\Theta} \cdot (\alpha_{max} - \alpha_{min})\ \ ); \tag{7}$$

with $k = 0, \ \ldots, \ \Theta$.

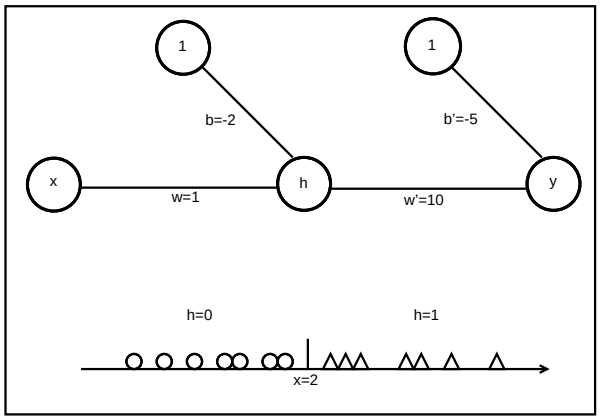

Fig. 1. Example of a DIMLP network which discriminates between two classes of data: circles and triangles at the bottom. Since the activation function of h is a step function, a hyperplane is found at $x = 2$, which corresponds to the ratio $-b/w$. The presence or absence of this hyperplane depends on the values of w' and b'.

2.2 QSVM

A Quantified Support Vector Machine (QSVM) is a DIMLP network with two hidden layers [5]. Neurons in the first hidden layer carry out a normalization of the input variables. The activation function of the neurons in the second hidden layer is related to the SVM kernel. For instance, with a dot kernel the corresponding activation function is the identity, while with a Gaussian kernel the activation function is Gaussian. The number of neurons in this layer is equal to the number of support vectors, with the incoming weight connections corresponding to the components of the support vectors.

Weights between the input layer and the first hidden are frozen during training. Specifically these weights are:

- weights: $w_i = K/\gamma_i$, with γ_i as the standard deviation of input variable x_i on the training set and K a constant equal to one;
- bias: $b_i = -K \cdot \mu_i/\gamma_i$, with μ_i as the average of input variable x_i on the training set.

After the normalization, the staircase activation function (eq. 5) is applied. During training, weights above the first hidden layer are modified according to the SVM training algorithm [13].

2.3 Ensembles

The accuracy of multiple integrated models is frequently greater than that of a single model. Two important representative learning algorithms for ensembles are bagging [6] and boosting [8]. They have been applied to both DTs and NNs. In

this work, we use DIMLP ensembles trained by bagging [6]. Specifically, bagging is rooted on resampling methods. With s training samples, bagging selects for each classifier s samples drawn with replacement from the original training set. Consequently, the individual models differ slightly, which is advantageous as compared to the whole ensemble of classifiers.

Many strategies for extracting rules from single neural networks have been proposed, but only a few authors have begun to extract rules from neural network ensembles. To generate propositional rules from single networks and ensembles, we introduced DIMLP networks [3]. In previous work, we performed rule extraction from ensembles of DTs by transforming them into ensembles of DIMLP networks [4]. In this work, we apply rule extraction to random forests (RFs) and shallow tree ensembles trained by gradient boosting (GB). We perform it in a different and more efficient way relative to our previous rule extraction technique (see Sect. 2.4).

2.4 The Fidex Algorithm

Fidelity refers to how well the extracted rules mimic the behaviour of a model. This is a measure of the accuracy with which the rules represent the decision-making process of a neural network, for example. Specifically, with s samples in a training set and s' samples for which the classifications of the rules match the classifications of the model, the fidelity is s'/s.

The Fidex local rule extraction algorithm strongly uses fidelity. We assume that we have d input variables $x_1 \dots x_d$. Furthermore, let us denote L_{xi} the list of hyperplanes for input variable x_i; L_{xi} is calculated according to eq. (7). The purpose of Fidex is to determine a propositional rule R with respect to sample S. The fidelity of R to be reached in ρ steps is ϕ, and is calculated with all the samples of the training set. The Fidex algorithm is the following:

1. Given sample S and rule $R = \emptyset$;
2. for $n = 1 \dots \rho$ do
3. for all input variables x_i randomly selected and for all L_{xi} do
4. find the hyperplane h^* allowing the highest increase of fidelity;
5. end for
6. if the increment of fidelity is strictly positive then $R = R \cup h^*$;
7. if fidelity reaches value ϕ then exit;
8. end for

Fidex computational complexity depends mainly on its loops and corresponds to $O(\rho \cdot d \cdot \Theta \cdot s)$; with s the number of training samples. Note that the s factor appears as for each new rule antecedent Fidex has to determine whether a sample is covered by R or not. Finally, Θ is due to the number of hyperplanes in each list of possible hperplanes L_{xi}.

The execution of Fidex is managed by four parameters:

- ρ: maximal number of Fidex steps (which is strongly related to the number of rule antecedents);

- ϕ: fidelity of the rule;
- p: dropout of input variables;
- q: dropout of hyperplanes.

Parameter ρ is the maximum number of iterations and is related to the maximum number of rule antecedents; its default value is equal to 10. The default value of ϕ, which corresponds to the fidelity that has to be reached is 100%. Note that Fidex could end before attaining ϕ. In that case, Fidex can simply be executed again with the same values of the parameters since it is non-deterministic, or with different values.

The execution time of Fidex can be accelerated by considering two dropout parameters: p and q. Essentially, p determines at each step the proportion of input variables that will not be taken into account. The excluded variables are selected randomly. Parameter q is similar, but with respect to excluded hyperplanes in L_{xi}. Default values for p and q are equal to 0. If variables x_i are selected with always the same order, for instance from 1 to d (see step 3) and without dropout, Fidex is deterministic.

2.5 The FidexGlo Algorithm

FidexGlo is a global rule extraction algorithm[1] that generates a ruleset for a training set of size s. It corresponds to a covering technique that calls Fidex s times. It therefore first generates s rules and then uses a heuristic to select a subset of the rule base that covers all s samples. A simple heuristic would be to select the rules randomly and stop when all the training samples are covered. Another heuristic consists of ranking the rules in descending order according to the number of samples covered, then selecting the rules in descending order until all the samples are covered. As we are using the latter heuristic, the computational complexity is $O(\rho \cdot d \cdot \Theta \cdot s^2)$. The essential reason is that Fidex is called s times.

Rule extraction can be performed with an ensemble of DIMLPs, since it can be viewed as a unique DIMLP with an additional hidden layer (the one that averages all single network responses). Hence, the list of the hyperplanes for a DIMLP ensemble is the union of all the lists related to each single network (cf. eq. 7). Hence, with ν networks in an ensemble, the computational complexity is $O(\nu \cdot \rho \cdot d \cdot \Theta \cdot s^2)$. Finally, with DT ensembles, the list of hyperplanes is compiled from the rules extracted from all the trees.

3 Experiments

3.1 Preliminaries

We applied several models to four classification problems retrieved from the Machine Learning Repository at the University of California, Irvine[2] [10]. We performed cross-validation experiments; Table 1 describes the datasets.

[1] Available at https://github.com/HES-XPLAIN/dimlpfidex
[2] https://archive.ics.uci.edu/ml/index.php

Table 1. Datasets used in the experiments. From left to right, columns designate: dataset name; number of samples; number of input features; and number of classes.

Dataset	Samples	Inputs	Classes
Breast Cancer	683	9	2
Heart Disease	270	13	2
Ionosphere	351	34	2
Spam	4601	57	2

Training datasets were normalized by Gaussian normalization. The following models were used with the four classification problems:

- RF: Random forests with FidexGlo;
- RF-ORE: Random forests with our old rule extraction algorithm [4];
- GB: Gradient boosting with FidexGlo;
- GB-ORE: Gradient boosting with our old rule extraction algorithm [4];
- QSVM-L: QSVM with linear kernel and with FidexGlo;
- QSVM-G: QSVM with Gaussian kernel and with FidexGlo;
- DIMLP-BT: DIMLP ensembles trained by bagging with FidexGlo;
- DIMLP-BT-ORE: DIMLP ensembles trained by bagging with our old rule extraction algorithm.

For RFs, GBs and SVMs we used default parameters defined in the Scikit Python Library. Specifically, for GBs the maximal depth of the trees is equal to three, while for RFs depth is not limited. All DIMLP ensembles were trained by back-propagation with default learning parameters [4] and only a hidden layer with the number of neurons equal to number of inputs. By default, a DIMLP ensemble includes 25 networks, whereas for RFs and GBs the number of trees is 100.

The Tables of the results include in the columns:

- Average predictive accuracy of the model (ratio of the number of correctly classified testing samples to the total number of testing samples);
- Average fidelity on the testing samples;
- Average predictive accuracy of the rules;
- Average predictive accuracy of the rules when rules and model agree;
- Average number of extracted rules;
- Average number of rule antecedents.

3.2 Results

Table 2 depicts the results obtained by the different models. On average, FidexGlo generated more rules than our old rule extraction algorithm, but with fewer rule antecedents. This trend is also observed in the other classification problems. Moreover, the average fidelity obtained with FidexGlo is always higher than that provided by our former rule extraction algorithm. For the "Breast Cancer" dataset, the highest predictive accuracy of the rules was obtained by

Table 2. Average results obtained on the "Breast Cancer" dataset by ten repetitions of 10-fold cross validation trials. Standard deviations are given between brackets.

Model	Tst. Acc.	Fid.	Rul. Acc. (1)	Rul. Acc. (2)	#Rul.	Avg. #Ant.
RF	97.1 (0.2)	99.1 (0.3)	96.9 (0.3)	97.4 (0.2)	30.0 (0.8)	2.6 (0.0)
RF-ORE [4]	**97.2** (0.2)	98.4 (0.5)	96.6 (0.5)	**97.7** (0.2)	24.2 (0.5)	3.4 (0.0)
GB	97.0 (0.3)	99.2 (0.4)	97.0 (0.3)	97.4 (0.4)	29.5 (0.6)	2.6 (0.0)
GB-ORE [4]	96.9 (0.4)	98.8 (0.3)	96.2 (0.5)	97.1 (0.5)	22.6 (0.6)	3.3 (0.0)
QSVM-L	97.1 (0.1)	99.5 (0.2)	**97.2** (0.2)	97.4 (0.1)	15.2 (0.4)	**2.2** (0.0)
QSVM-G	97.1 (0.2)	**99.6** (0.2)	97.0 (0.2)	97.4 (0.4)	15.0 (0.5)	2.3 (0.0)
DIMLP-BT	97.1 (0.1)	**99.6** (0.2)	97.0 (0.3)	97.2 (0.2)	15.5 (0.5)	**2.2** (0.0)
DIMLP-BT-ORE	97.1 (0.1)	98.7 (0.3)	96.4 (0.4)	97.3 (0.2)	**12.3** (0.5)	2.7 (0.1)

QSVM-L (97.2%), which is equal to the predictive accuracy of RFs. DIMLP-BT with FidexGlo and QSVM-G provided the highest fidelity with 99.6%.

As shown in Table 3, on the classification problem "Heart Disease", the highest average predictive accuracy of the rules was again obtained by QSVM-L (83.9%). DIMLP-BT ensembles reached the highest average predictive accuracy (86.1%), and the highest average predictive accuracy of the rules when ensembles and rules agree (86.4%). Finally, the highest average fidelity was provided by GB, at 96.4%.

Table 3. Average results obtained on the "Heart Disease" dataset.

Model	Tst. Acc.	Fid.	Rul. Acc. (1)	Rul. Acc. (2)	#Rul.	Avg. #Ant.
RF	82.3 (0.8)	95.5 (0.8)	80.8 (1.3)	83.1 (0.8)	52.9 (0.9)	2.9 (0.0)
RF-ORE [4]	82.7 (1.6)	93.2 (1.7)	81.4 (1.6)	84.4 (1.1)	39.7 (1.1)	4.1 (0.0)
GB	81.3 (0.8)	**96.4** (1.0)	80.6 (1.4)	82.1 (1.1)	46.7 (1.1)	2.9 (0.0)
GB-ORE [4]	79.9 (1.9)	94.1 (1.5)	79.7 (1.9)	81.7 (2.1)	34.8 (0.7)	3.8 (0.0)
QSVM-L	83.9 (0.7)	96.3 (1.0)	**83.9** (0.7)	84.7 (0.8)	27.5 (0.8)	**2.6** (0.0)
QSVM-G	82.3 (0.7)	96.0 (1.1)	81.3 (1.0)	83.1 (1.0)	32.3 (0.8)	2.7 (0.0)
DIMLP-BT	**86.1** (1.0)	95.3 (1.2)	83.3 (1.5)	**86.4** (0.9)	28.7 (0.7)	**2.6** (0.0)
DIMLP-BT-ORE	**86.1** (1.0)	94.6 (0.7)	82.3 (1.1)	86.2 (0.9)	**20.9** (0.8)	3.3 (0.0)

With the "Ionosphere" dataset the results presented in Table 4 highlight that the highest average predictive accuracy was obtained by QSVM-G (94.5%). Once again, GB provided the highest average predictive accuracy of the rules (94.2%), as well as the highest average fidelity (98.9%).

Table 5 depicts the results for the "Spam" classification problem. Random forests and the rulesets they generated provided the highest average predictive accuracies (95.4% and 95.2%, respectively). Again, GB reached the highest average fidelity (98.9%). With RF, FidexGlo generated the highest average number of rules. This trend is also observed in the previous classification problems. This could be explained by the fact that with an unlimited tree depth, the number of rules tends to be very high.

Table 4. Average results obtained on the "Ionosphere" dataset.

Model	Tst. Acc.	Fid.	Rul. Acc. (1)	Rul. Acc. (2)	#Rul.	Avg. #Ant.
RF	93.3 (0.3)	98.4 (0.6)	93.8 (0.7)	94.2 (0.5)	31.3 (0.9)	2.5 (0.0)
RF-ORE [4]	93.2 (0.4)	95.3 (1.0)	91.5 (1.3)	94.4 (0.6)	33.2 (1.2)	4.0 (0.1)
GB	93.9 (0.4)	**98.9** (0.5)	**94.2** (0.5)	94.6 (0.4)	29.3 (0.8)	2.5 (0.0)
GB-ORE [4]	92.8 (1.0)	96.5 (1.0)	91.0 (1.1)	93.4 (0.9)	29.5 (1.2)	3.5 (0.2)
QSVM-L	86.5 (0.7)	95.5 (1.0)	88.3 (0.7)	89.2 (0.9)	27.6 (1.0)	2.5 (0.0)
QSVM-G	**94.5** (0.6)	97.6 (0.6)	93.5 (0.4)	**95.1** (0.4)	26.2 (0.6)	2.5 (0.0)
DIMLP-BT	93.1 (0.4)	98.0 (0.6)	94.1 (0.4)	94.5 (0.4)	23.0 (1.1)	**2.3** (0.0)
DIMLP-BT-ORE	93.1 (0.4)	96.2 (0.9)	92.5 (0.7)	94.5 (0.4)	**19.3** (0.5)	2.9 (0.1)

Table 5. Average results obtained on the "Spam" dataset.

Model	Tst. Acc.	Fid.	Rul. Acc. (1)	Rul. Acc. (2)	#Rul.	Avg. #Ant.
RF	**95.4** (0.1)	98.7 (0.1)	**95.2** (0.2)	**95.9** (0.1)	410.9 (3.9)	2.7 (0.0)
GB	94.6 (0.1)	**99.3** (0.1)	94.5 (0.1)	94.9 (0.1)	230.2 (4.1)	2.9 (0.0)
QSVM-L	93.6 (0.1)	99.2 (0.1)	93.5 (0.1)	93.9 (0.1)	227.5 (2.5)	2.9 (0.0)
QSVM-G	94.4 (0.1)	99.1 (0.1)	94.3 (0.1)	94.8 (0.1)	247.7 (3.5)	3.0 (0.0)
DIMLP-BT	94.7 (0.1)	99.0 (0.1)	94.6 (0.1)	95.1 (0.1)	288.7 (2.1)	**2.5** (0.0)
DIMLP-BT-ORE	94.7 (0.1)	98.8 (0.2)	94.6 (0.2)	95.2 (0.2)	**90.3** (1.3)	5.5 (0.1)

4 Conclusion

To explain the classifications of the black box-models, we have introduced a new local algorithm and a new global algorithm named Fidex and FidexGlo, respectively. The key idea behind them is to determine the discriminant hyperplanes while maximising the fidelity of the underlying model. We applied them to SVMs, ensembles of DTs, and ensembles of NNs. On four classification problems, FidexGlo performed very well compared to our former global rule extraction technique, which has been compared to many other rule extraction techniques (in previous research). Our next step will be to apply Fidex to deep models such as convolutional NNs.

Acknowledgments. This work was in part conducted in the context of the Horizon Europe project PRE-ACT (Prediction of Radiotherapy side effects using explainable AI for patient communication and treatment modification), and it has received funding through the European Commission Horizon Europe Program (Grant Agreement number: 101057746). In addition, this work was supported by the Swiss State Secretariat for Education, Research and Innovation (SERI) under contract nummber 2200058. Finally, this work was in part carried out with funding from the University of Applied Sciences and Arts of Western Switzerland (HES-SO) and the Engineering and Architecture domain.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE Access **6**, 52138–52160 (2018). https://doi.org/10.1109/ACCESS.2018.2870052
2. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-based systems **8**(6), 373–389 (1995). https://doi.org/10.1016/0950-7051(96)81920-4
3. Bologna, G.: Is it worth generating rules from neural network ensembles? Journal of Applied Logic **2**(3), 325–348 (2004). https://doi.org/10.1016/j.jal.2004.03.004
4. Bologna, G.: A rule extraction technique applied to ensembles of neural networks, random forests, and gradient-boosted trees. Algorithms **14**(12), 339 (2021). https://doi.org/10.3390/a14120339
5. Bologna, G., Hayashi, Y.: Qsvm: A support vector machine for rule extraction. In: Advances in Computational Intelligence: 13th International Work-Conference on Artificial Neural Networks, IWANN 2015, Palma de Mallorca, Spain, June 10-12, 2015. Proceedings, Part II 13. pp. 276–289. Springer (2015)
6. Breiman, L.: Bagging predictors. Machine learning **24**(2), 123–140 (1996)
7. Diederich, J.: Rule extraction from support vector machines, vol. 80. Springer Science & Business Media (2008). https://doi.org/10.1007/978-3-540-75390-2_1
8. Freund, Y., Schapire, R.E.: A desicion-theoretic generalization of on-line learning and an application to boosting. In: European conference on computational learning theory. pp. 23–37. Springer (1995). https://doi.org/10.1006/jcss.1997.1504
9. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM computing surveys (CSUR) **51**(5), 1–42 (2018). https://doi.org/10.1145/3236009
10. Lichman, M.: UCI machine learning repository, university of california, irvine, school of information and computer sciences (2013), http://archive.ics.uci.edu/ml
11. Ribeiro, M.T., Singh, S., Guestrin, C.: 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. pp. 1135–1144. ACM (2016)
12. Van Assche, A., Blockeel, H.: Seeing the forest through the trees: learning a comprehensible model from an ensemble. In: ECML. vol. 7, pp. 418–429. Springer (2007). https://doi.org/10.1007/978-3-540-74958-5_39
13. Vapnik, V.N.: An overview of statistical learning theory. IEEE transactions on neural networks **10**(5), 988–999 (1999). https://doi.org/10.1109/72.788640
14. Zhang, Q.s., Zhu, S.C.: Visual interpretability for deep learning: a survey. Frontiers of Information Technology & Electronic Engineering **19**(1), 27–39 (2018). https://doi.org/10.48550/arXiv.1802.00614