Proceedings of Machine Learning Research 179:1–20, 2022 Conformal and Probabilistic Prediction with Applications

Well-Calibrated Rule Extractors

Ulf Johansson ULF.JOHANSSON@JU.SE
Tuwe Löfström TUWE.LOFSTROM@JU.SE
Niclas Ståhl NICLAS.STAHL@JU.SE
Dept. of Computing, Jönköping University, Sweden

Editor: Ulf Johansson, Henrik Boström, Khuong An Nguyen, Zhiyuan Luo and Lars Carlsson

Abstract

While explainability is widely considered necessary for trustworthy predictive models, most explanation modules give only a limited understanding of the reasoning behind the predictions. In pedagogical rule extraction, an opaque model is approximated with a transparent model induced using original training instances, but with the predictions from the opaque model as targets. The result is an interpretable model revealing the exact reasoning used for every possible prediction. The pedagogical approach can be applied to any opaque model and use any learning algorithm producing transparent models as the actual rule extractor. Unfortunately, even if the extracted model is induced to mimic the opaque, test set fidelity may still be poor, thus clearly limiting the value of using the extracted model for explanations and analyses. In this paper, it is suggested to alleviate this problem by extracting probabilistic predictors with well-calibrated fitness estimates. For the calibration, Venn-Abers with its unique validity guarantees, is employed. Using a setup where decision trees are extracted from MLP neural networks, the suggested approach is first demonstrated in detail on one real-world data set. After that, a large-scale empirical evaluation using 25 publicly available benchmark data sets is presented. The results show that the method indeed extracts interpretable models with well-calibrated fitness estimates, i.e., the extracted model can be used for explaining the opaque. Specifically, in the setup used, every leaf in a decision tree contains a label and a well-calibrated probability interval for the fidelity. Consequently, a user could, in addition to obtaining explanations of individual predictions, find the parts of feature space where the decision tree is a good approximation of the MLP and not. In fact, using the sizes of the probability intervals, the models also provide an indication of how certain individual fitness estimates are.

Keywords: Rule extraction, Fidelity, Interpretability, Explainability, Calibration, Venn-Abers predictors

1. Introduction.

When using machine learning for data analysis, predictive models are often required to provide explanations for predictions produced. This can be necessary to comply with regulatory measures, in domains such as finance, law enforcement or medicine; or to follow ethics guidelines like (High-Level Expert Group on AI, 2019). Today, a majority of the research about explanation methods focuses on creating explanations for single instance predictions, e.g., LIME (Ribeiro et al., 2016) and SHAP (Lundberg and Lee, 2017). These methods explain how the different features contribute to one prediction, thus aiding a human decision maker in understanding the reasons behind that specific prediction. However, in many

scenarios, where a more global understanding of the model and the underlying relationship is needed, these single prediction explanations will not suffice.

Rule extraction is a technique for approximating global models with interpretable models, e.g., decision trees or rule sets. Such interpretable approximations of the opaque model admit extensive inspection and analysis of the relationships found by the opaque model, enabling global explanations and providing insights into the underlying data and the domain. Given an extracted interpretable model, it is of course also straightforward to obtain detailed explanations for single predictions, i.e., local explanations. Depending on the exact situation, the extracted model may either be used to make the actual predictions, or simply to explain the predictions made by the opaque model.

One way to obtain the interpretable model is to have a machine learning technique (that produces transparent models) learn the input-output relationship of the opaque model. This procedure is referred to as *pedagogical* or *black-box* rule extraction. Pedagogical rule extraction is *model agnostic*, in the sense that it may by used on any type of opaque model. Another option is to employ *open-box* techniques which extract a transparent model based on the inner workings of the opaque model. These techniques rely on specific characteristics, regarding e.g., the architecture of the opaque model, and are thus tailored to a specific type of models, most often a neural network.

In rule extraction, *fidelity* measures the extent to which an extracted model makes the same predictions as the opaque model upon which it is based. For classification, this simply means the proportion of instances where the opaque and transparent models agree. Pedagogical rule extraction results in transparent models that approximate the opaque model, similarly to the way a model approximates a data set in inductive learning, whereas open-box techniques will produce exact, but possibly very complex, transparent representations. Thus, pedagogical rule extraction has the distinct advantage of being model agnostic, but provides no fidelity guarantees. Open-box methods, in contrast, often per design obtains perfect fidelity, but is restricted to a certain type of opaque models.

In order to be used for explaining opaque models, the extracted models need to have high fidelity. A low-fidelity transparent model is, at best, not very useful, and at worst misleading, since it might produce predictions that differ substantially from the opaque model. Naturally, most pedagogical rule extraction techniques are designed to somehow optimize fidelity, but similar to any inductive model generated to optimize predictive performance during training, there are no guarantees that fidelity on training data will carry over to new unseen data. Furthermore, a single measure of model fidelity on a test set only indicates the average infidelity rate, but does not give any indication of whether a particular instance can be expected to be predicted identically to the opaque model or not.

2. Background.

2.1. Rule extraction

Rule extraction is the process of generating transparent models, typically trees or rule sets, from opaque models, such as neural networks or ensembles. Rule extraction is a large field in machine learning, and a wealth of algorithms exist. A good introduction and survey can be found in (Huysmans et al., 2006). One reason for performing rule extraction is to complement predictions from an opaque model with explanations to explicitly show a

decision maker the relationships found by the model and utilized in the prediction. Another situation where rule extraction is used is when a transparent model is needed for the actual predictions. Rather than using a technique directly producing a transparent model, such as a decision tree, from data, an interpretable model can be extracted from a high-performance opaque model. When doing this, the underlying assumption is that the opaque model provides a better basis for construction of the transparent model than the original training data, i.e., that the opaque model is a better representation of the relationship between inputs and targets than the examples in the data set. The main motivation for this assumption is that a properly trained opaque model will have "smoothed out" irregularities in the data, and learned a more general function than the one expressed by the examples in the data set. Experimental evaluation supports this claim by showing that transparent models generated by rule extraction often outperform state-of-the-art decision tree algorithms like CART and C4.5, see e.g., (Johansson, 2007). Rule extraction with guarantees on fidelity – but without the addition and communication of fidelity estimations – has previously been studied by Johansson et al. (2014) for classification, and by Johansson et al. (2022), for regression.

2.2. Probabilistic prediction

Probabilistic predictors output not only a label but a probability distribution. If these probability distributions perform well against statistical tests based on subsequent observations of the labels, the probabilistic predictor is said to be *valid*. In this paper, we will focus on one aspect of validity, i.e., that the predictor should be *well-calibrated*. If p^{c_j} is the probability estimate for class j, the probability estimate for the predicted label (i.e., the confidence) should match the observed accuracy.

$$p(c_j \mid p^{c_j}) = p^{c_j} \tag{1}$$

Here it must be noted that if a transparent model capable of producing probability estimates is generated by pedagogic rule extraction, the probability estimates will in fact be for fidelity (i.e., making the same prediction as the opaque model) and not accuracy. Specifically, if the extracted model is a *probability estimation tree (PET)* (Provost and Domingos, 2003), the fidelity estimates can be calculated using *relative frequencies*; i.e., the proportion of training instances corresponding to a specific class in the leaf where the test instance falls:

$$p_i^{c_j} = \frac{g(i,j)}{\sum_{k=1}^{C} g(i,k)} \tag{2}$$

where $g(i,j)$ is the number of instances belonging to class j (i.e., instances predicted as class j by the opaque model) that falls in the same leaf as instance i, and C is the number of classes. For simplicity, we call such decision trees extracted from opaque models *fidelity estimation trees (FETs)*.

Unfortunately, PETs are notorious for being very overconfident, see e.g., (Johansson et al., 2018), requiring external calibration to become well-calibrated. While the standard approaches Platt scaling (Platt, 1999) and isotonic regression (Zadrozny and Elkan, 2001) can be used for this calibration, several recent studies (Johansson et al., 2019a,b, 2021) suggest applying *Venn predictors* (Vovk et al., 2004) instead. When used for calibration, Venn prediction uses a separate calibration set, similar to Platt scaling and isotonic regression.

In this paper, we will investigate using Venn predictors for calibrating FETs, arguing that if well-calibrated, these models will be a very good basis for explanations of the opaque model, specifically alleviating the problem with the potentially low test set fidelity present in all pedagogic rule extraction. While the test set fidelity may still be low, the proposed method gives an exact description of the fidelity to expect in different parts of feature space.

2.3. Venn and Venn-Abers predictors

Venn predictors are multi-probabilistic predictors with proven validity properties (Vovk et al., 2005). Calibrating on a separate labeled data set not used for training the underlying model, the key idea of Venn prediction is to divide all calibration instances into a number of *categories*. These categories are typically somehow based on the predictions from the underlying model, and the specific division into categories is called a *Venn taxonomy*, where different Venn taxonomies produce different Venn predictors. When predicting a test instance, the category is determined using the taxonomy and the underlying model, exactly as for the calibration instances. The probability distribution over the possible labels is then calculated as the relative frequencies of the labels for calibration instances belonging to that category. To obtain validity, this calculation must include the test instance to be predicted, where the true label is, of course, not known. Consequently, all possible labels for the test instance must be considered, leading to a set of C label probability distributions, where C is the number of possible labels. These C probability distributions are the actual output of Venn prediction, but to make the predictions easier to interpret, they are often converted into a predicted label together with a probability interval, see e.g., (Lambrou et al., 2015).

While the multiprobability predictions produced by Venn predictors are automatically valid, regardless of the taxonomy used, the chosen taxonomy affects both the accuracy of the Venn predictor and the sizes of the probability intervals. Ideally, the probability estimates should, of course, be close to zero or one, and the intervals as tight as possible. While taxonomies with more categories lead to more specific predictions, using too many categories will result in larger intervals, since each interval will be based on few examples. For two-class problems, this trade-off can be handled automatically by using so-called *Venn-Abers predictors* (Vovk and Petej, 2012) where an optimized taxonomy is found using isotonic regression. Since Venn-Abers predictors are Venn predictors, they inherit the validity guarantees, while the optimized taxonomy leads to more specific predictors.

Venn-Abers predictors require *scoring classifiers* as underlying models, i.e., a test prediction from the underlying model must be a *prediction score* $s(x)$, where a higher value indicates a larger belief in the label 1. With access to a calibration set $\{z_{q+1}, \ldots, z_l\}$, a multi-probabilistic prediction from a Venn-Abers predictor for x_{l+1} is produced as follows:

Let s_0 be the scoring function for $\{z_{q+1}, \ldots, z_l, (x_{l+1}, 0)\}$ and s_1 be the scoring function for $\{z_{q+1}, \ldots, z_l, (x_{l+1}, 1)\}$.

Let g_0 be the isotonic calibrator for

$$\{(s_0(x_{q+1}), y_{q+1}), \ldots, (s_0(x_l), y_l), (s_0(x_{l+1}), 0)\} \tag{3}$$

and g_1 be the isotonic calibrator for

$$\{(s_1(x_{q+1}), y_{q+1}), \ldots, (s_1(x_l), y_l), (s_1(x_{l+1}), 1)\} \tag{4}$$

Then the valid probability interval for $y_{l+1} = 1$ is

$$(p_0, p_1) = (g_0(s_0(x_{l+1})), g_1(s_1(x_{l+1}))) \tag{5}$$

It should be noted that this valid probability interval is more informative than the corresponding point estimation. Specifically, the width of the interval is an indication of the uncertainty, i.e., it exhibits the confidence in the estimation.

3. Method.

All experimentation was carried out using scikit learn, keras and tensorflow. Both single- and multi-layer MLPs were used as opaque models. The activation functions in the hidden and output layers were ReLU and sigmoid, respectively. Since all problems are two-class, one single output unit was used. The number of hidden units h was chosen as $h = \lfloor \frac{2}{3}a \rfloor$ where a is the number of attributes. The loss function was set to cross entropy, and Adam was used as the optimizer. Standard decision trees were used as the extracted models. All parameter values were left at default, with the exception that the minimum number of training instances in each leaf was set to 5.

Since Venn-Abers needs a separate labeled data set for the calibration, two different MLPs were trained for every fold; one using all training instances and one dividing the training instances into a proper training set (2/3) and a calibration set (1/3). All in all, the following setups were used:

- **ANNa**: MLPs trained using all training data.
- **ANNt**: MLPs trained using 2/3 of the training data.
- **Uncal**: Rule extraction with decision trees induced on the original training data but with the predictions from ANNa as the targets.
- **VA**: Rule extraction with decision trees induced on the original training data but with the predictions from ANNt as the targets, and then calibrated with Venn-Abers on the calibration set.

For the actual evaluation, 10x10-fold cross validation was used, so all results are averaged over the 100 folds.

In the analysis, we first look at model results, i.e., accuracy, fidelity and sizes. After that, we evaluate the quality of the fidelity calibration, starting with the expected calibration error (ECE) as used by e.g., Guo et al. (2017). Here it must be noted that while ECE and the other calibration metrics described below are normally used to evaluate accuracy estimations, we use them to evaluate *fidelity* estimations. In the rule extraction scenario, the probability estimates from the interpretable models are of course, as described above, for the predictions from the opaque model, not the true targets. Consequently, the predicted labels from the opaque model are also used instead of the true targets in the calculations.

When evaluating Venn-Abers for calibration, a scalar probability (here fidelity) estimate is necessary. For this, we follow the suggestion by Vovk and Petej (2012) and use a regularized value p instead of the center of the interval (p_0, p_1):

$$p = \frac{p_1}{1 - p_0 + p_1} \tag{6}$$

ECE partitions the fidelity estimates into M equally-sized bins and then calculates the actual fidelity of each bin. Let $\text{fid}(B_m)$ be the fidelity and $\text{est}(B_m)$ the mean fidelity estimate in bin m. Then

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} \Big| \text{fid}(B_m) - \text{est}(B_m) \Big| \tag{7}$$

where n is the total number of calibration instances. In the experimentation, we used 5 bins.

While ECE is easily interpretable, the calculation only involves the mean of the predictions falling in a specific bin. With this in mind, the quality of the fidelity calibration was also evaluated using two loss functions that operate on individual predictions.

The *log loss* function is defined as:

$$\lambda_{log} = \begin{cases} -\log p & \text{if } \hat{y} = 1 \\ -\log(1-p) & \text{if } \hat{y} = 0 \end{cases} \tag{8}$$

where log is the natural logarithm, $\hat{y}$ is the prediction from the opaque model and p is the probability estimate for the label 1 from the interpretable model. It should be noted that while the log loss is infinite when an estimate is categorical but wrong, the log loss calculation in scikit learn avoids this by clipping the probability estimates so they are not exactly 0 or 1.

Similarly, the *Brier loss*, which punishes large errors less severely, is defined as:

$$\lambda_{Br} = (\hat{y} - p)^2 \tag{9}$$

The 25 benchmarking data sets used (see Table 1) are all two-class problems, publicly available from either the UCI repository (Bache and Lichman, 2013) or the PROMISE Software Engineering Repository (Sayyad Shirabad and Menzies, 2005).

Table 1: Dataset descriptions

Data set	#inst	#attrib	Source	Data set	#inst	#attrib	Source
colic	328	23	UCI	kc2	522	22	Promise
creditA	690	16	UCI	kc3	325	39	Promise
diabetes	768	9	UCI	liver	345	7	UCI
german	1000	21	UCI	pc1req	320	9	Promise
haberman	306	4	UCI	pc4	1458	38	Promise
heartC	303	13	UCI	sonar	208	61	UCI
heartH	270	12	UCI	spect	218	22	UCI
heartS	270	14	UCI	spectf	348	45	UCI
hepati	155	20	UCI	transfusion	748	5	UCI
iono	351	35	UCI	ttt	958	10	UCI
je4042	274	9	Promise	vote	435	17	UCI
je4243	363	8	Promise	wbc	699	10	UCI
kc1	2109	22	Promise				

4. Results.

In this section we first present a case study from the drug discovery domain before showing and analyzing the results from the benchmarking study.

4.1. Drug discovery case study

Table 2: Molecule feature descriptions

Feature name	Feature description
Weight	Molecular weight in Dalton.
LogP	Partition Coefficient, which describes how easily each molecule is dissolved into water.
HDonors	Number of hydrogen donors.
HAcceptors	Number of hydrogen acceptors.
AromaticRings	Number of aromatic rings.
TPSA	The topological polar surface area, which is the surface sum over all polar polar parts of the molecule.
RotatableBonds	Number of bonds which allow free rotation around themselves.
HeavyAtomCount	Number of non-hydrogen atoms.
FractionCSP3	The fraction of C atoms that are SP3 hybridized.
RingCount	Number of rings.

The discovery of new pharmaceuticals drugs is a long and costly process. This process has become more and more data-driven where models, both simulation- and machine learning based, are used to predict the properties of proposed molecules.

A common process is to perform a virtual screening of large sets of virtually generated molecules and only continue the discovery process with those molecules that have promising predictions, see (Reddy et al., 2007). In these cases, additional value can be gained by not only removing those molecules with negative predictions, but also removing positive predictions, where either the prediction or the logic of the predictive model is uncertain. This allows the future discovery process to focus on a few promising candidates with clearly understood predictions. To achieve this, it is necessary to dissect the model and understand what it bases its decisions on, enabling domain experts to couple established knowledge about the problem to the model decision process. Furthermore, such analysis of the model may result in increased understanding of the underlying chemical mechanics that causes certain properties and, consequently, which chemical sub-spaces future drug discovery projects should target. To highlight the benefits that can be gained through extracting rules that demonstrate the underlying reasoning of the model, we conduct a study where a deep artificial neural network model is trained to predict the inhibition of the cytochrome P450 2C19 enzyme. This is an enzyme that is involved in the human metabolism and an inhibition would lead to slow metabolism of the molecule, as well as other drugs. Hence, the inhibition of this enzyme may lead to a decrease of effectiveness of other drugs and a harmful collection of substances in the body.

To train the model, the data set that was collected by Veith et al. (2009), consisting of 12 665 instances, is used. The molecules in this data set are represented by 10 molecular descriptors that are both commonly used and human-understandable. The features are listed in Table 2 and are selected in order to make explanations of the opaque model interpretable and informative from a chemical perspective.

The opaque model used in this study is a deep neural network with 5 hidden layers and a total of 91 456 free parameters to fit. When trained and evaluated, using 10-fold cross validation, the neural network achieves an accuracy of 76.4 %. The extracted model, which is a decision tree, is then trained to mimic the network, and when evaluated it gets an accuracy of 71.8 % on the original data. The accuracy of both these models are, however, of lesser importance as long as we could explain the predictions of some of the examples with great certainty and correctly understand how the model is reasoning in these cases. For this, the method presented above is applied, resulting in the well-calibrated and interpretable FET in Fig 1.

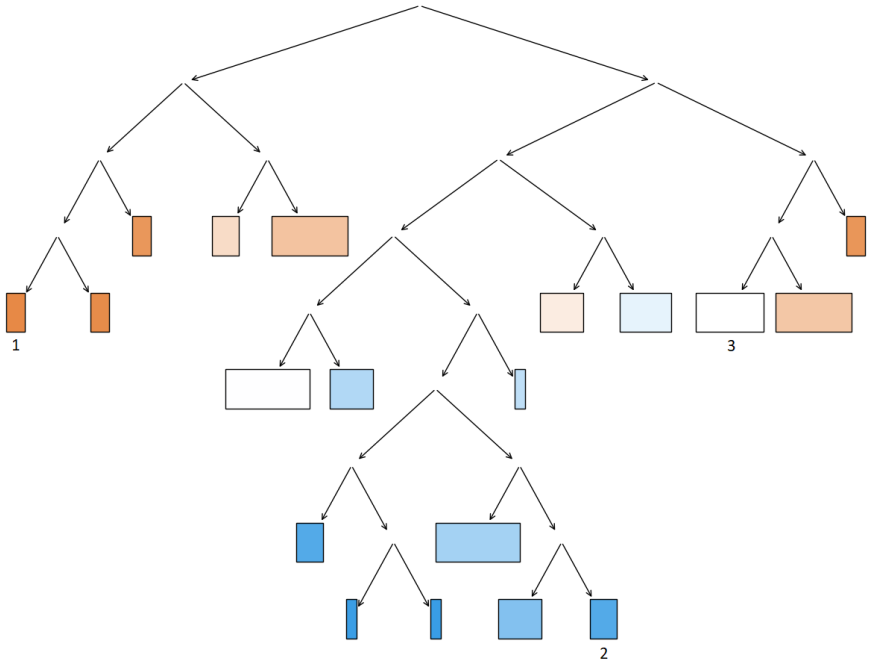

Figure 1: A well-calibrated and interpretable FET for the cytochrome P450 2C19 enzyme. Colors represent the target classes of molecules with either no CYP2C19 inhibition (orange) or CYP2C19 inhibition (blue), the intensity of the colors corresponds to the estimated fidelity while the width of the leaves give the sizes of the Venn-Abers probability intervals, indicating how certain the fidelity estimates are.

The FET, consequently, conveys two things:

- Its fidelity to the opaque model in different parts of the input space, represented by the colour intensity of the leaves.
- How certain it is about its own fidelity, represented by the width of the leaves.

The entire left branch of the tree in our example is more or less strongly dedicated to the prediction of no CYP2C19 inhibition, with some leaves being certain that they are performing exactly as the underlying model (having small intervals and strong color intensity), whereas other leaves are less likely to explain the underlying model well, or indicate the model being less confident about its fidelity. The central part of the tree is dedicated to the prediction of molecules with CYP2C19 inhibition with different degrees of fidelity and certainty, with some leaves having high fidelity with certainty and others having a lot of uncertainty. In the right-most part of the tree, most leaves have both low fidelity and high uncertainty.

Each leaf also represent a description, in the form of a series of conjunctive conditions defined for the input attributes, of the instances in that part of the instance space. Example of rules, in the form of conjunctive conditions, that describe the instance space of the three numbered leaves in Fig. 1 are listed below in Fig. 2. The interval at the end of each rule is the Venn-Abers interval for the fidelity.

1) LogP ≤ 1.2
$\rightarrow$ **No CYP2C19 inhibition** $[0.961, 1.0]$

2) LogP > 4.1
& FractionCSP3 ≤ 0.34
& AromaticRings ≤ 6
& RingCount ≤ 3
& RotatableBonds ≥ 5
$\rightarrow$ **CYP2C19 inhibition** $[0.929, 0.995]$

3) LogP > 2.5
& $0.47 <$ FractionCSP3 ≤ 0.56
$\rightarrow$ **Indecisive** $[0.451, 0.656]$

Figure 2: Rules for the three leaves in Figure 1

Note that the rules above, and the uncertainty that is attributed to each of these predictions, represent the FET's explanation of how the underlying opaque model, in this case a deep neural network, behaves. Hence, these rules describe the approximate reasoning of the opaque model, showing in which regions of the input space the FET detects that it follows general patterns, such as shown in rules 1) and 2). These rules can easily be verified with domain knowledge to determine whether the opaque model is behaving rationally. For instance, the first rule strongly indicates that if the logP is low, then the underlying model will predict that there will be no CYP2C19 inhibition. There are also regions in the input space where the FET could not discover any general patterns in the behaviour of the opaque model, for example rule 3) above. This does not imply that the model would intrinsically fail for inputs in this region, just that no easily obtained explanations exist there. Finally, Fig. 3 shows the reliability plot for the CYP2C19 data set.

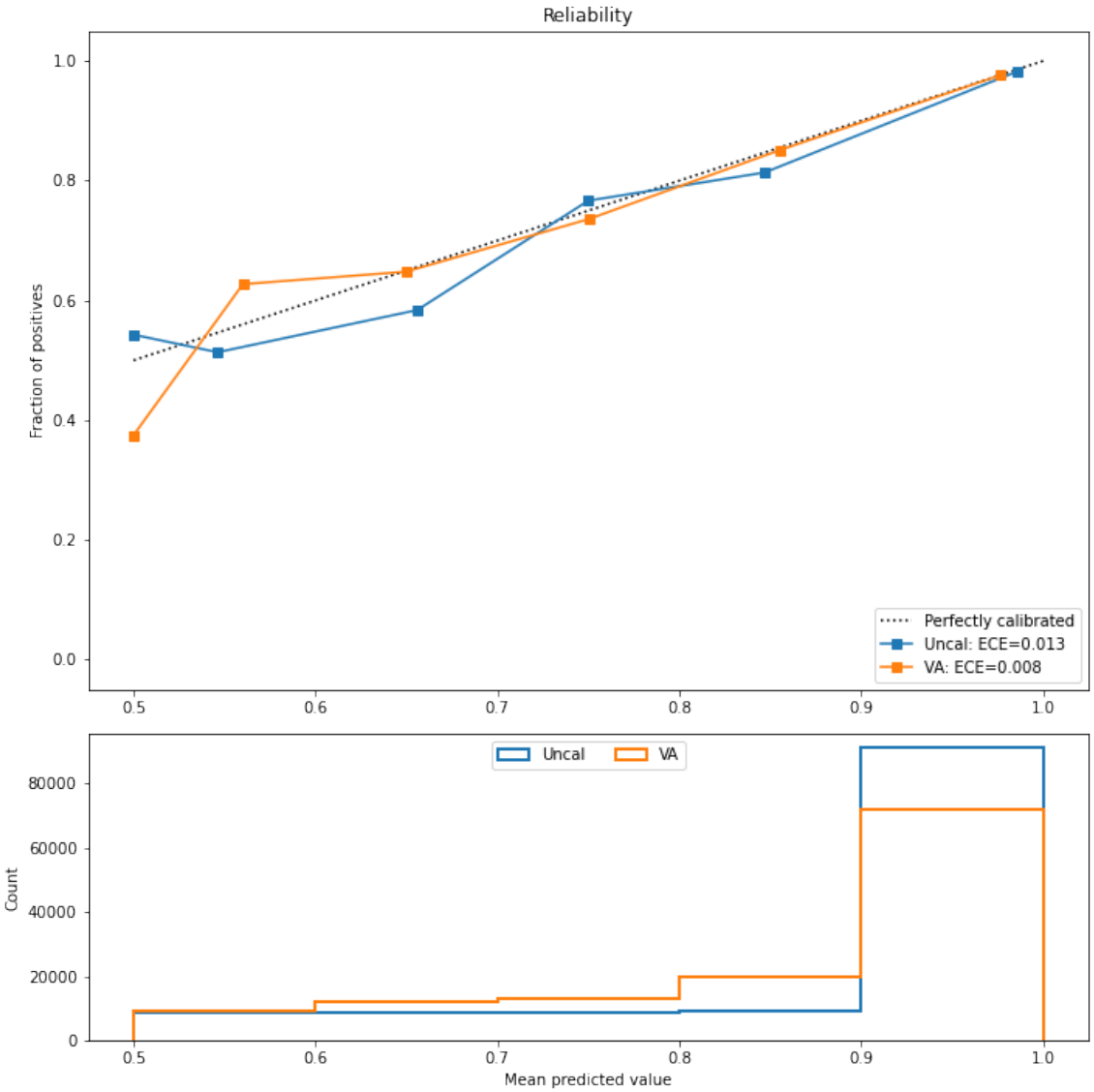

Figure 3: Fidelity reliability plot for CYP2C19.

In this and following examples, only five bins are used. Even if both models have low ECEs in this example, it is evident that the uncalibrated FET is slightly over-confident in its fidelity estimate, whereas the calibrated FET is very accurate in its fidelity estimate.

In summary, the possibility to extract a well-calibrated FET from an opaque model like a deep neural network provides multiple benefits. Firstly, the FET provides a global explanation of the opaque model, clearly indicating where the extracted model performs similarly to the opaque model and where it does not. Secondly, each leaf clearly indicates both a fidelity level and an estimate of the confidence in the fidelity level. Thirdly, each leaf and the path to that leaf provide a specific explanation of the logic behind those predictions of the opaque model.

4.2. Results on benchmarking data sets

Table 3 below shows the model results on the benchmark data. Starting with accuracy, we see that the loss in accuracy associated with using an interpretable model is as low as just two percentage points, on average. Looking at mean ranks, it is obvious that the access to more training data is beneficial. Using a Wilcoxon signed ranks tests with $\alpha = 0.05$, ANNa is actually significantly more accurate than ANNt. For the extracted models, though, there are no such differences, despite the fact that VA is trained on a smaller data set than Uncal. Regarding fidelity, the differences are again very small, and not significant at $\alpha = 0.05$. Finally, it may be noted that using the smaller training set, and having a fixed setting for the minimum instances in each leaf, as expected, leads to smaller trees. While some of the trees are rather complex, most of them are small enough to manually inspect and analyze.

Table 3: Model results

	Accuracy				Fidelity		Size	
	ANNa	ANNt	Uncal	VA	Uncal	VA	Uncal	VA
colic	.804	.789	.779	.804	.837	.824	49.2	33.1
creditA	.850	.848	.842	.844	.882	.874	65.8	46.1
diabetes	.765	.760	.748	.747	.886	.874	66.7	47.7
german	.649	.650	.647	.671	.823	.827	149.4	106.5
haberman	.713	.719	.714	.720	.981	.983	6.4	4.0
heartC	.819	.815	.780	.777	.857	.829	38.1	27.1
heartH	.828	.829	.784	.774	.866	.865	32.1	22.9
heartS	.832	.828	.774	.779	.851	.841	31.7	22.2
hepati	.848	.843	.783	.800	.835	.859	17.4	11.8
iono	.917	.915	.871	.872	.857	.870	28.7	20.4
je4042	.714	.711	.719	.702	.900	.894	25.8	16.3
je4243	.626	.625	.618	.612	.912	.885	32.7	24.6
kc1	.762	.759	.753	.750	.935	.931	55.0	39.6
kc2	.793	.791	.797	.793	.942	.937	16.7	11.9
kc3	.871	.867	.874	.870	.942	.948	18.4	12.8
liver	.686	.640	.610	.597	.772	.793	53.3	34.1
pc1req	.683	.654	.691	.639	.853	.822	16.7	12.0
pc4	.904	.902	.879	.880	.908	.919	87.6	60.2
sonar	.841	.816	.717	.697	.715	.733	29.8	19.5
spect	.883	.881	.865	.884	.948	.975	19.1	9.9
spectf	.791	.788	.749	.782	.803	.829	33.3	22.3
transfusion	.749	.752	.746	.745	.975	.974	14.0	9.8
ttt	.981	.960	.913	.909	.912	.903	84.9	68.5
vote	.860	.856	.862	.846	.910	.902	53.2	36.7
wbc	.971	.970	.954	.952	.971	.968	19.9	15.4
Mean	**.806**	**.799**	**.779**	**.778**	**.883**	**.882**	**41.8**	**29.4**
Mean rank	**1.16**	**1.84**	**1.40**	**1.60**	**1.40**	**1.60**	**2.00**	**1.00**

Before looking at aggregated results regarding the calibration, we discuss some typical results on the data set level. Figure 4 below shows a reliability plot for the diabetes data set. On this data set, the uncalibrated model is constantly too confident, i.e., the empirical fidelity is lower than the confidence for all bins. Specifically, there is a large group of instances where the uncalibrated model is very certain about the label predicted by the opaque model. Calibration with Venn-Abers, however, produces a very well-calibrated model.

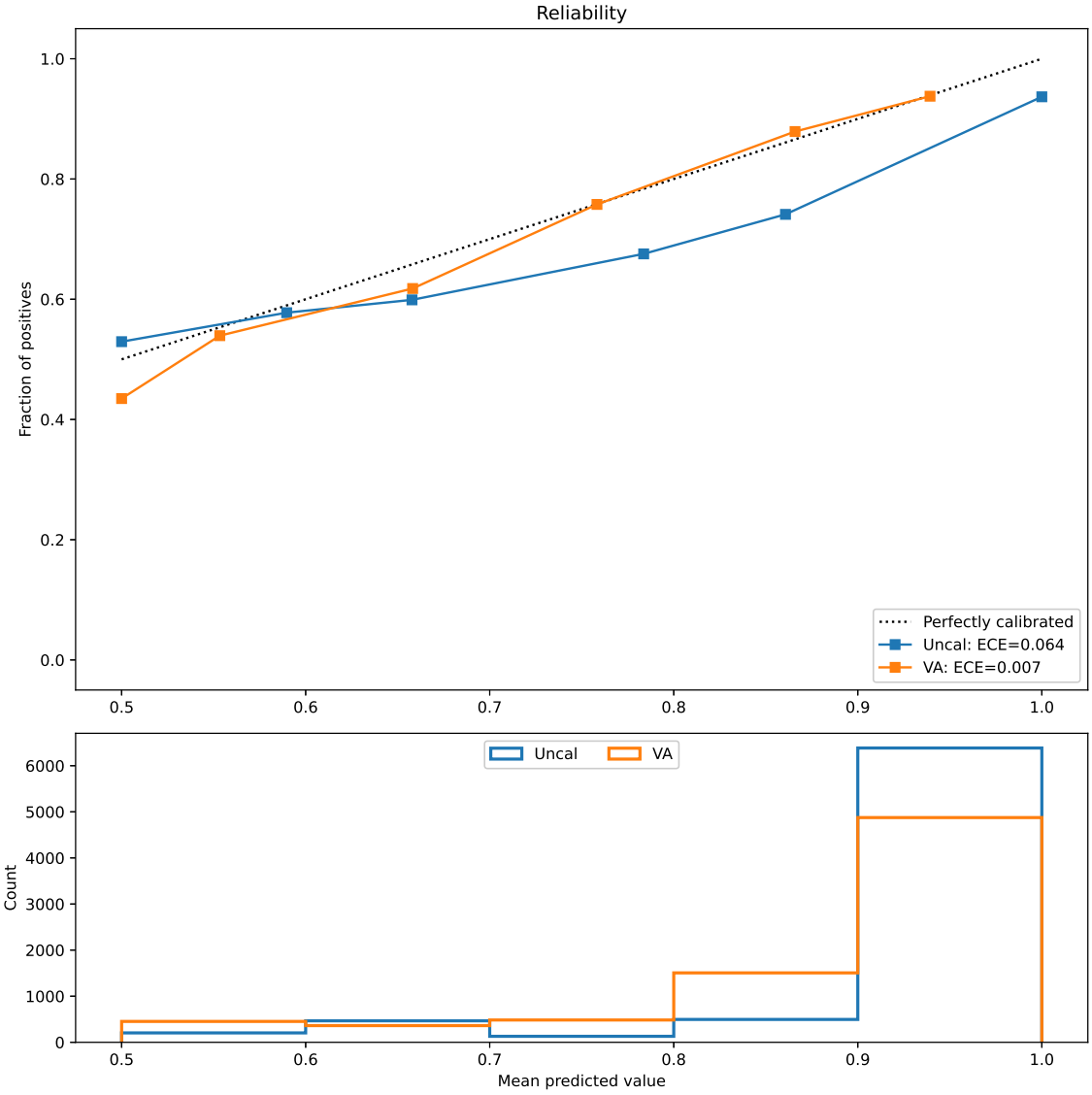

Figure 4: Fidelity reliability plot for diabetes

Figure 5 below shows a similar example where the calibration leads to a model which is less certain, which of course makes sense since the fidelity on this specific data set is lower than 0.8. Looking at the reduction in ECE, the external calibration is very successful.

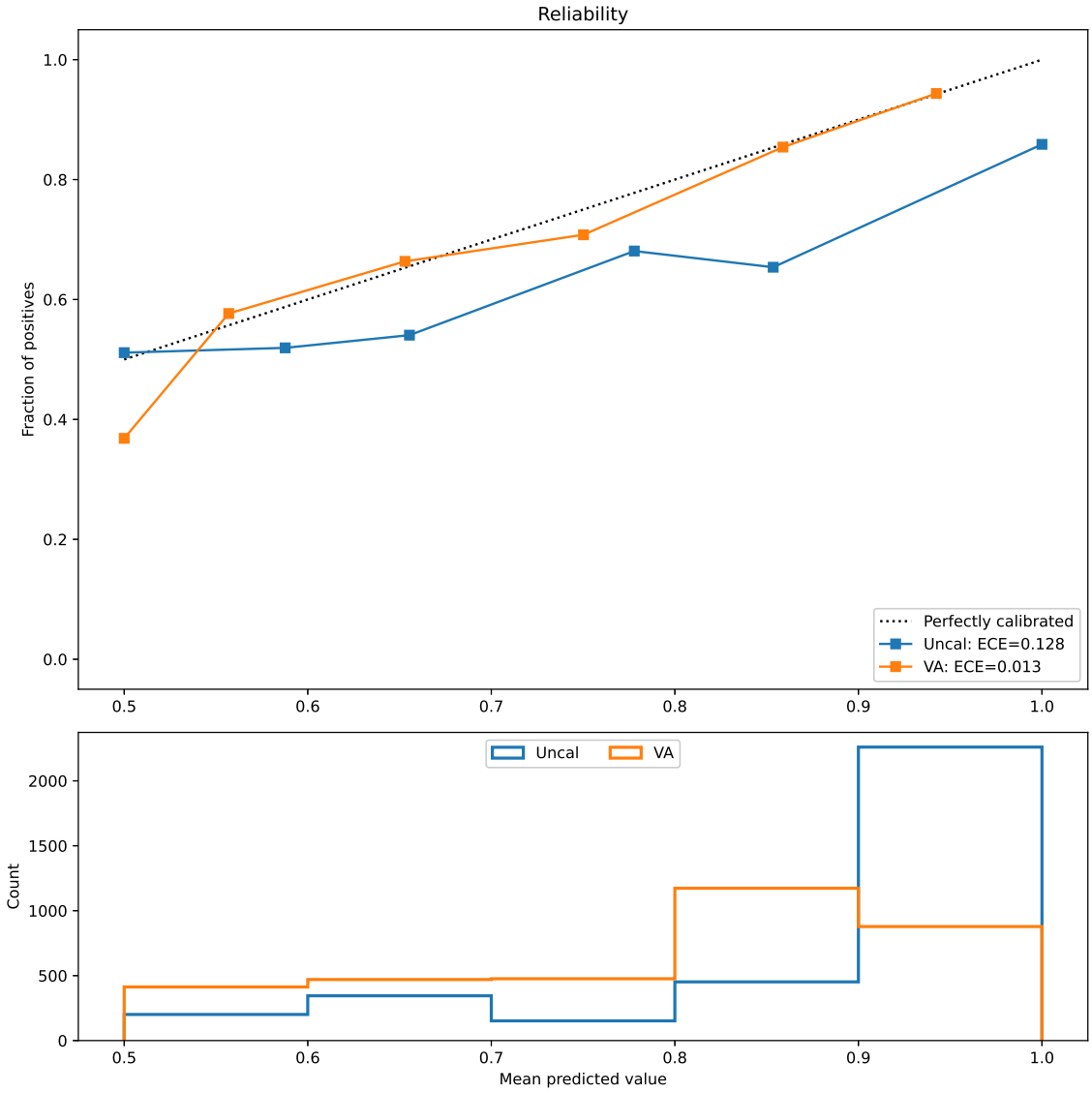

Figure 5: Fidelity reliability plot for liver

Figure 6 below shows one of the most extreme data sets, where the uncalibrated model is exceptionally poorly calibrated, specifically much too overconfident. Again, the external calibration is very successful, although even the calibrated model is still slightly overconfident.

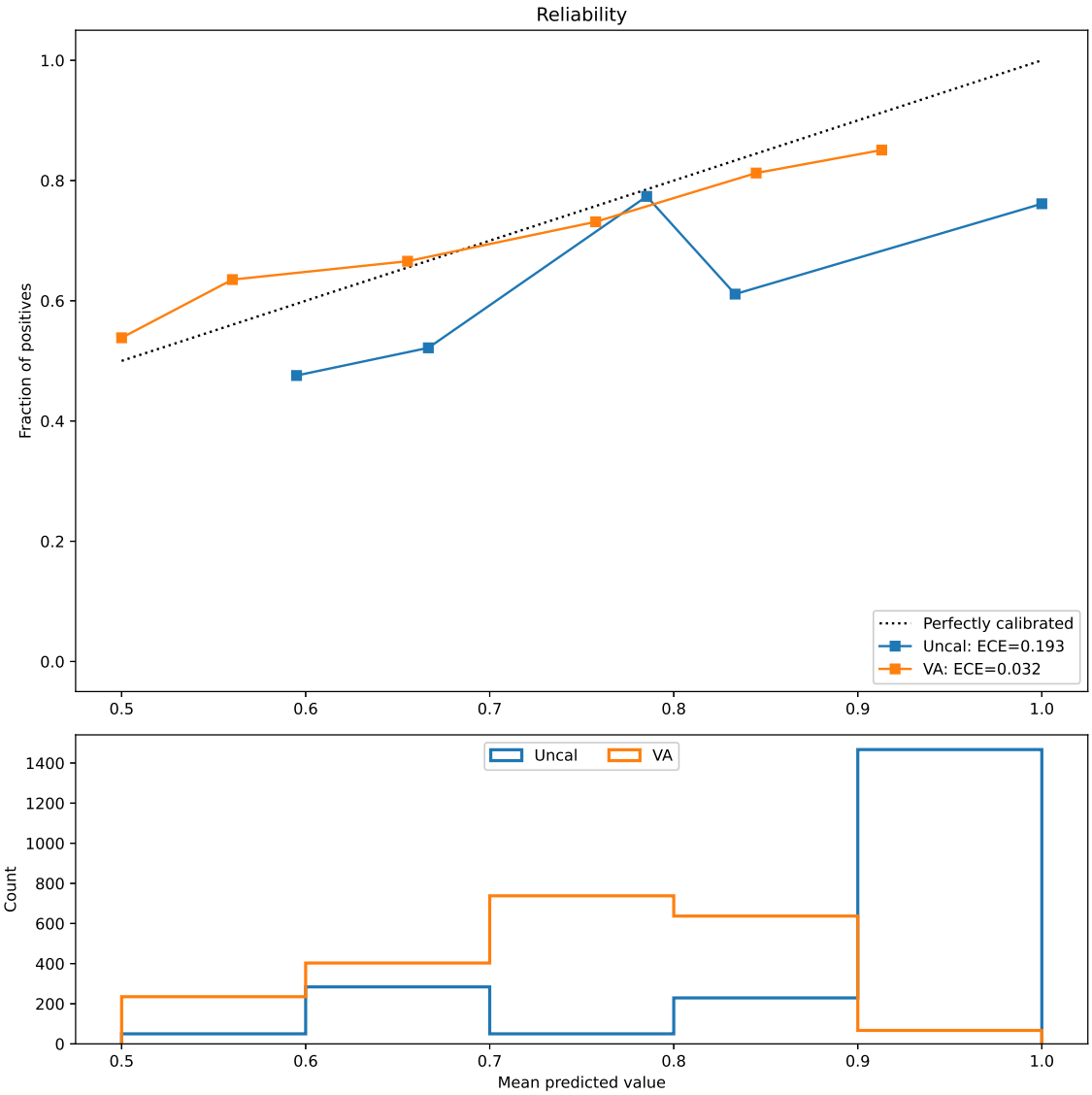

Figure 6: Fidelity reliability plot for sonar

Figure 7 below illustrates a data set where calibration is able to improve an already reasonably well-calibrated model.

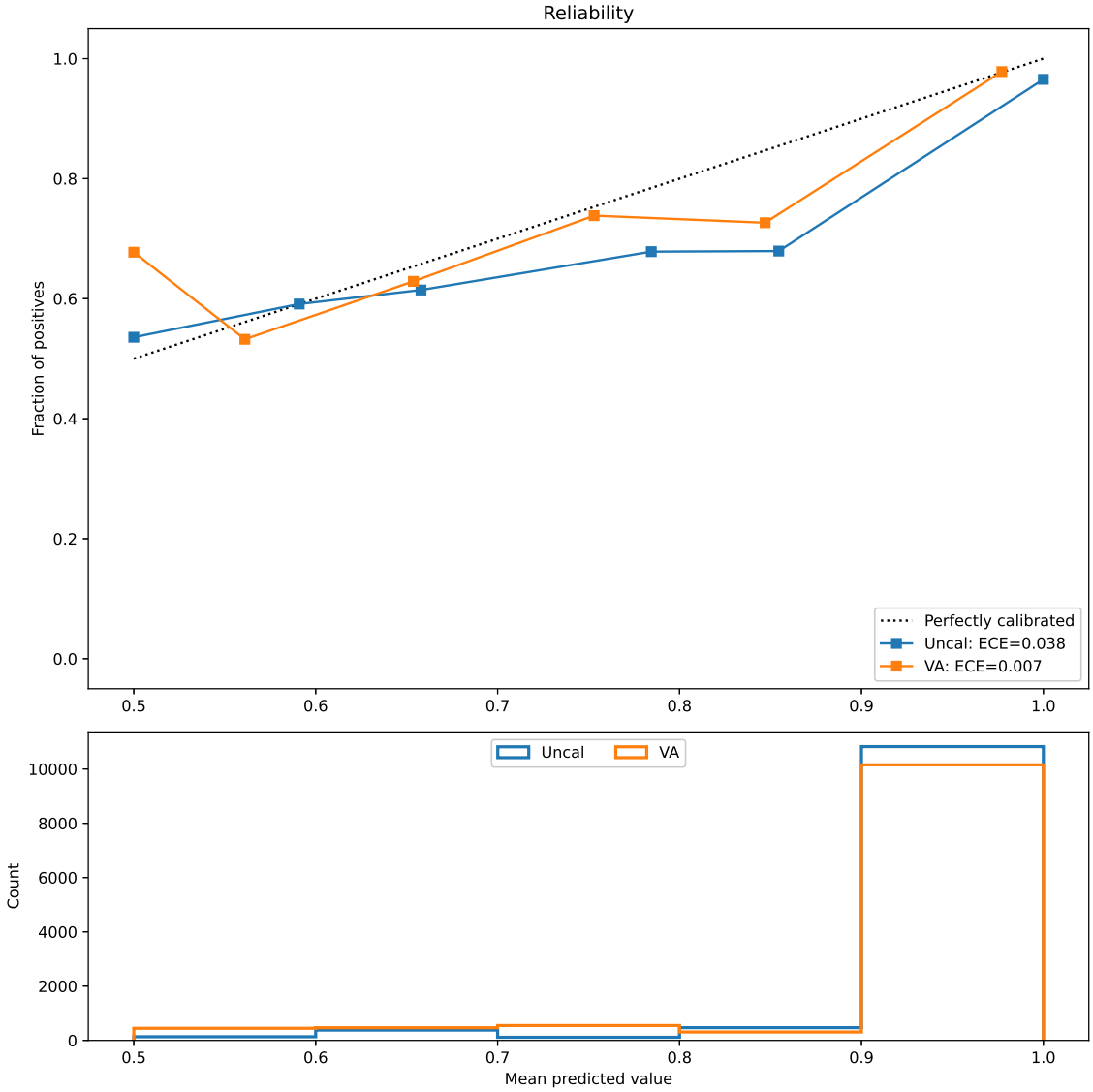

Figure 7: Fidelity reliability plot for kc1

Figure 8 below, finally, shows one of the two data sets where the ECE is actually larger after the calibration than before. Here, where the fidelity is over 0.98, the calibration makes the model slightly underconfident.

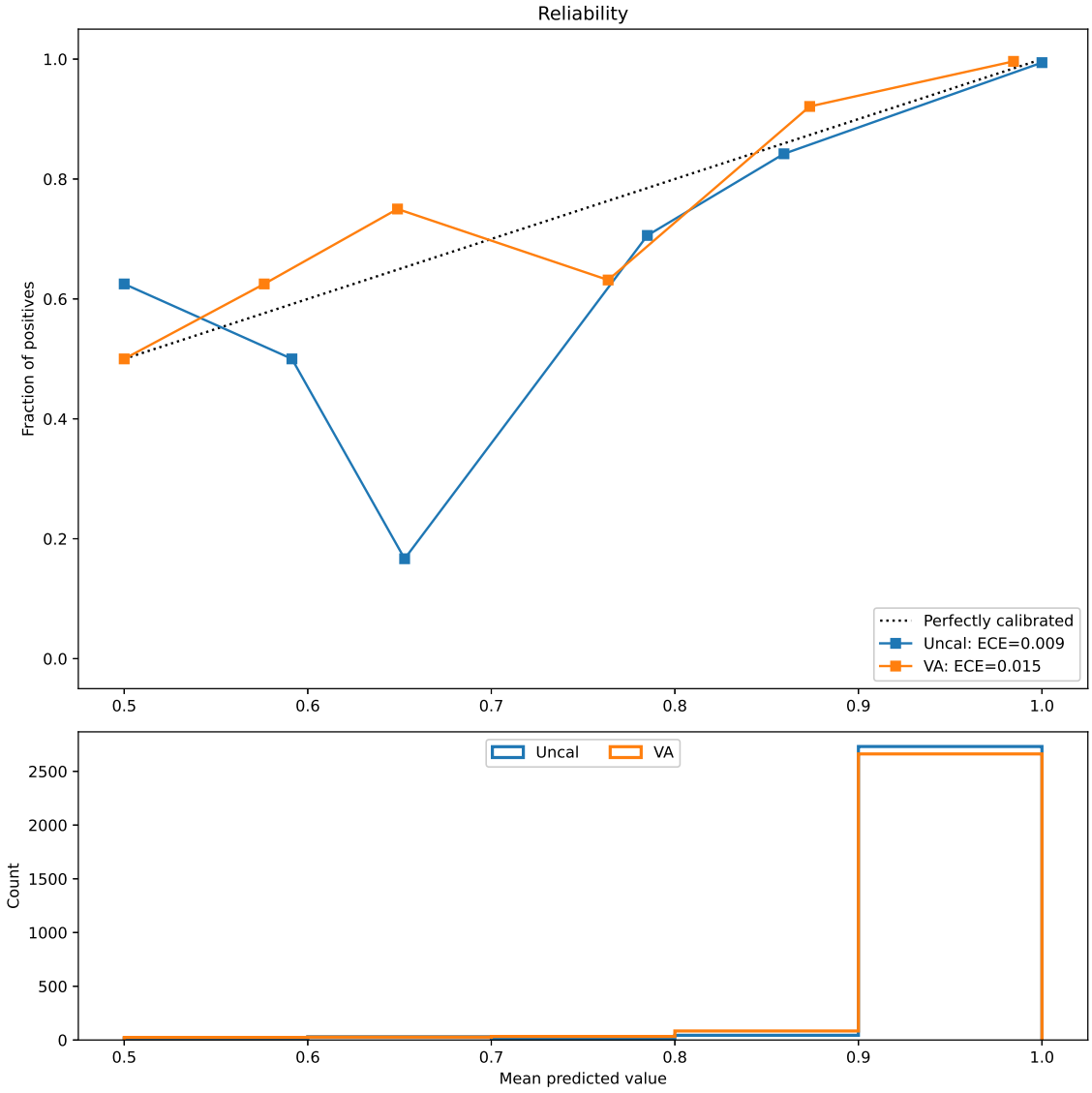

Figure 8: Fidelity reliability plot for haberman

Turning to the aggregated calibration results in Table 4 below, the first two columns show the average difference between the estimated and empirical fidelities. Interestingly enough, for Uncal the average difference over all data sets is 0.06, i.e., uncalibrated fidelity trees are on average six percentage points too optimistic. For VA, however, the difference is very close to zero. Comparing fidelity ECE:s, the calibration lead to lower ECE:s on all data sets but two. This is of course a significant difference using a Wilcoxon test at $\alpha = 0.05$. For log loss and Brier loss, the results are even more clear; the calibration using Venn-Abers

results in an improvement on each and every data set. In summary, these results clearly demonstrate the benefit of using calibration, often converting fidelity trees from outright misleading to well-calibrated.

Table 4: Calibration results

	Difference		ECE		Log loss		Brier loss	
	Uncal	VA	Uncal	VA	Uncal	VA	Uncal	VA
colic	.079	-.013	.096	.030	8.97	.427	.322	.132
creditA	.063	-.004	.068	.014	15.84	.336	.514	.098
diabetes	.063	.002	.064	.007	21.10	.330	.660	.096
german	.075	.002	.075	.011	2.48	.404	.149	.126
haberman	.008	-.012	.009	.015	32.48	.058	.951	.013
heartC	.073	.003	.081	.026	14.99	.395	.504	.122
heartH	.051	-.014	.056	.031	17.63	.332	.600	.098
heartS	.081	-.010	.086	.029	16.39	.392	.536	.119
hepati	.089	-.005	.089	.019	23.31	.351	.752	.105
iono	.096	-.009	.097	.016	9.14	.339	.292	.101
je4042	.051	-.017	.053	.024	16.26	.284	.520	.081
je4243	.042	-.007	.046	.017	12.16	.295	.396	.085
kc1	.038	.004	.038	.007	29.39	.183	.880	.051
kc2	.032	-.015	.033	.020	26.67	.182	.800	.048
kc3	.025	-.013	.030	.019	29.31	.137	.899	.038
liver	.127	.005	.128	.013	6.43	.455	.272	.147
pc1req	.050	-.030	.061	.038	12.18	.409	.460	.128
pc4	.054	-.002	.057	.005	27.91	.224	.853	.062
sonar	.193	.010	.193	.032	10.90	.570	.385	.192
spect	.000	-.024	.025	.027	.32	.098	.030	.022
spectf	.119	-.010	.119	.013	3.91	.411	.162	.127
transfusion	.013	-.008	.013	.008	31.99	.083	.940	.021
ttt	.031	-.002	.032	.006	9.84	.257	.322	.073
vote	.036	-.003	.036	.006	21.71	.270	.699	.077
wbc	.011	-.010	.012	.013	21.37	.103	.635	.025
Mean	**.060**	**-.007**	**.064**	**.018**	**16.91**	**.293**	**.541**	**.087**
Mean rank			**1.88**	**1.12**	**2.00**	**1.00**	**2.00**	**1.00**

Table 5 below shows the Venn-Abers fidelity estimates and the corresponding empirical fidelity values. Despite the fact that most intervals are fairly tight, typically just a few percentage points, the empirical error rate falls within the interval for every data set.

Table 5: Venn-Abers fidelity estimates and corresponding empirical fidelity values

	VA fid. est.		Fid.		VA fid. est		Fid.
	Low	High	Emp.		Low	High	Emp.
colic	.799	.845	.824	kc2	.916	.949	.937
creditA	.863	.890	.874	kc3	.925	.963	.948
diabetes	.870	.893	.874	liver	.785	.829	.793
german	.822	.845	.827	pc1req	.764	.876	.822
haberman	.968	.990	.983	pc4	.914	.928	.919
heartC	.820	.871	.829	sonar	.726	.783	.733
heartH	.837	.893	.865	spect	.944	.982	.975
heartS	.818	.874	.841	spectf	.806	.852	.829
hepati	.835	.907	.859	transfusion	.962	.980	.974
iono	.851	.891	.870	ttt	.896	.916	.903
je4042	.866	.917	.894	vote	.891	.925	.902
je4243	.869	.910	.885	wbc	.955	.976	.968
kc1	.931	.945	.931	**Mean**	**.865**	**.905**	**.882**

5. Concluding remarks.

We have in this paper introduced and evaluated rule extractors with well-calibrated fitness estimations. In the specific setup used in the empirical study, Venn-Abers was used for calibrating standard decision trees generated from pedagogic rule extraction. The result is a very informative model where each leaf in the tree contains a well-calibrated fidelity estimation probability interval. In our opinion, this solves the inherent problem with the potentially low test fidelity always present in black-box rule extraction. Using this representation language, a user would always know exactly how well the extracted model is able to approximate the opaque model, for every instance. Obviously, the extracted model can also be used to understand the opaque in many more ways than just explaining individual predictions. Specifically, it clearly identifies the parts of feature space where it is a good approximation of the opaque model and not. Looking at the sizes of the probability intervals, a user also gets an indication of the confidence in individual fitness estimations.

For future work, dedicated rule extraction algorithms could be used, instead of decision trees. More generally, we suggest outright comparisons between external explanation modules and well-calibrated rule extraction, investigating the quality of the explanations. Finally, it should be noted that the fidelity trees introduced here, just like all pedagogic rule extractors, are of course agnostic to whether the opaque model is correct or not. But an extracted model calibrated using a separate labeled data set can actually include information about the performance of the opaque model on these instances. We believe that investigating the exact construction and usability of such accuracy/fidelity estimation models would be very interesting.

Acknowledgements

The authors acknowledge the Swedish Knowledge Foundation, Jönköping University, and the industrial partners for financially supporting the research through the AFAIR project with grant number 20200223, as part of the research and education environment SPARK at Jönköping University, Sweden.

References

Kevin Bache and Moshe Lichman. UCI machine learning repository, 2013.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017.

High-Level Expert Group on AI. Ethics Guidelines for Trustworthy AI. Report, European Commission, Brussels, April 2019.

J. Huysmans, B. Baesens, and J. Vanthienen. Using rule extraction to improve the comprehensibility of predictive models. FETEW Research Report KBI 0612, K. U. Leuven, 2006.

Ulf Johansson. *Obtaining Accurate and Comprehensible Data Mining Models - An Evolutionary Approach*. PhD thesis, Linköping University, Institute of Technology, Department of Computer and Information Science, 2007.

Ulf Johansson, Rikard König, Henrik Linusson, Tuve Löfström, and Henrik Boström. Rule extraction with guaranteed fidelity. In *IFIP International Conference on Artificial Intelligence Applications and Innovations*, pages 281–290. Springer, 2014.

Ulf Johansson, Tuve Löfström, Håkan Sundell, Henrik Linusson, Anders Gidenstam, and Henrik Boström. Venn predictors for well-calibrated probability estimation trees. In *COPA*, pages 1–12. PMLR, 2018.

Ulf Johansson, Tuwe Löfström, and Henrik Boström. Calibrating probability estimation trees using venn-abers predictors. In *Proceedings of the 2019 SIAM International Conference on Data Mining, SDM 2019, Calgary, Alberta, Canada, May 2-4, 2019.*, pages 28–36, 2019a.

Ulf Johansson, Tuwe Löfström, Henrik Boström, and Cecilia Sönströd. Interpretable and specialized conformal predictors. In *COPA*, pages 3–22. PMLR, 2019b.

Ulf Johansson, Tuwe Löfström, and Henrik Boström. Calibrating multi-class models. In *Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications*, volume 152 of *Proceedings of Machine Learning Research*, pages 111–130. PMLR, 08–10 Sep 2021.

Ulf Johansson, Cecilia Sönströd, Tuwe Löfström, and Henrik Boström. Rule extraction with guarantees from regression models. *Pattern Recognition*, page 108554, 2022. ISSN 0031-3203. doi: https://doi.org/10.1016/j.patcog.2022.108554. URL https://www.sciencedirect.com/science/article/pii/S0031320322000358.

Antonis Lambrou, Ilia Nouretdinov, and Harris Papadopoulos. Inductive venn prediction. *Annals of Mathematics and Artificial Intelligence*, 74(1):181–201, 2015.

Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st international conference on neural information processing systems*, pages 4768–4777, 2017.

John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

Foster Provost and Pedro Domingos. Tree induction for probability-based ranking. *Mach. Learn.*, 52(3):199–215, 2003.

A Srinivas Reddy, S Priyadarshini Pati, P Praveen Kumar, HN Pradeep, and G Narahari Sastry. Virtual screening in drug discovery-a computational perspective. *Current Protein and Peptide Science*, 8(4):329–351, 2007.

Marco Tulio Ribeiro, Sameer Singh, and Carlos 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, 2016.

J. Sayyad Shirabad and T.J. Menzies. The PROMISE Repository of Software Engineering Databases. School of IT and Engineering, Univ. of Ottawa, Canada, 2005.

Henrike Veith, Noel Southall, Ruili Huang, Tim James, Darren Fayne, Natalia Artemenko, Min Shen, James Inglese, Christopher P Austin, David G Lloyd, et al. Comprehensive characterization of cytochrome p450 isozyme selectivity across chemical libraries. *Nature biotechnology*, 27(11):1050–1055, 2009.

Vladimir Vovk and Ivan Petej. Venn-abers predictors. *arXiv preprint arXiv:1211.0025*, 2012.

Vladimir Vovk, Glenn Shafer, and Ilia Nouretdinov. Self-calibrating probability forecasting. In *Advances in Neural Information Processing Systems*, pages 1133–1140, 2004.

Vladimir Vovk, Alex Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer-Verlag New York, Inc., 2005.

B. Zadrozny and C. Elkan. Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *ICML*, pages 609–616, 2001.