What's inside the black-box? A genetic programming method for interpreting complex machine learning models

Benjamin P. Evans
School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
benjamin.evans@ecs.vuw.ac.nz

Bing Xue
School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
bing.xue@ecs.vuw.ac.nz

Mengjie Zhang
School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
mengjie.zhang@ecs.vuw.ac.nz

ABSTRACT

Interpreting state-of-the-art machine learning algorithms can be difficult. For example, why does a complex ensemble predict a particular class? Existing approaches to interpretable machine learning tend to be either local in their explanations, apply only to a particular algorithm, or overly complex in their global explanations. In this work, we propose a global model extraction method which uses multi-objective genetic programming to construct accurate, simplistic and model-agnostic representations of complex black-box estimators. We found the resulting representations are far simpler than existing approaches while providing comparable reconstructive performance. This is demonstrated on a range of datasets, by approximating the knowledge of complex black-box models such as 200 layer neural networks and ensembles of 500 trees, with a single tree.

CCS CONCEPTS

• **Computing methodologies → Genetic programming**.

KEYWORDS

Explainable Artificial Intelligence, Interpretable Machine Learning, Evolutionary Multi-objective Optimisation

ACM Reference Format:
Benjamin P. Evans, Bing Xue, and Mengjie Zhang. 2019. What's inside the black-box? A genetic programming method for interpreting complex machine learning models. In *Genetic and Evolutionary Computation Conference (GECCO '19), July 13–17, 2019, Prague, Czech Republic.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3321707.3321726

1 INTRODUCTION

A common criticism of modern machine learning techniques is the lack of interpretability/explainability. This could serve as a barrier to the wide-scale adoption of machine learning, as users are more likely to deploy a model if they can understand why particular decisions are being made. This is discussed for business in [4], for health care in [6], and more broadly in the recently deployed general data protection regulation [26] (GDPR) in the European Union, which prohibits "solely automated decisions, including profiling, which have a legal or similarly significant effect on them". This has started the push for Explainable AI (XAI). Ethical considerations play a large part in the push towards XAI, and there is no short history toward bias in machine learning techniques. As examples, Sweeney [30] exposed potentially racial bias in the suggestion algorithm used in Google AdSense, and Bolukbasi et al. [3] showed the popular word2vec algorithm can be heavily susceptible to gender bias.

XAI, more specifically, interpretable machine learning (IML), can help observe these biases, to then ideally remedy the biases. The bias can occur from a number of sources, such as the sampling of the training data, uncovering correlative but not causal relationships, or poor selection of feature sets. To uncover these biases, it's important to understand how the model is making particular decisions.

With XAI, the goal is to have the simplest rules possible without sacrificing the performance. Simplicity and performance are often conflicting objectives (motivating the need for XAI, since top performing methods are often complex). With traditional tree-based methods for XAI, such as decision tree construction, complexity is controlled by early stopping or post-pruning. These approaches suffer from limitations, such as, with early stopping (or pre-pruning), a branch is terminated when no reduction in cross-validation error is noted. However, this may be premature since additional splits further down may have reduced the error drastically (i.e. if the feature becomes more informative with the addition of another because of feature interaction). With post-pruning, leaves are shrunk by replacing parent nodes with the majority class of the leaf. If no increase in error is seen, this process continues until the error increases for each branch. Alternatively, trees are shrunk in a top-down manner, i.e. with cost-complexity pruning. Of course a major drawback to both approaches is since the tree was greedily constructed, a poor split in hindsight cannot be undone, so the pruning is limited in its ability, i.e. pruning is not going to find the optimal tree which balances complexity and accuracy, since it first greedily maximises the accuracy, then attempts to greedily reduce the complexity afterwards.

In contrast, Genetic Programming (GP) [19] is another tree construction method. Rather than trees being constructed greedily in a top-down manner, trees are evolved from a population of candidates. As well as avoiding the need for greedy construction, such population-based methods are ideal for multi-objective optimisation [10] meaning that trees can be constructed which simultaneously

GECCO '19, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6111-8/19/07…$15.00
https://doi.org/10.1145/3321707.3321726

optimise seemingly conflicting objectives. However, to the best of our knowledge, GP has not been used for constructing IML models.

In this work, a new multi-objective GP-based method for IML is proposed to overcome the aforementioned limitations. The specific objectives are to:

- Propose a simple, human-readable tree structure which can be used for reconstructing complex predictions and overcomes limitations of existing tree-based methods, such as greedy construction or need for pruning,
- Simultaneously maximise the reconstruction ability while minimising the complexity of the trees,
- Generate a frontier of trade-off solutions for user selection, and
- Finally, to evaluate the method against current state-of-the-art approaches on various datasets.

The proposed method is applicable to any black-box (arbitrarily complex) classifier and makes no further assumptions about these models (such as gradient-based, or ability to apply sparsity).

2 BACKGROUND

There are several main approaches to IML/XAI, which are briefly introduced along with the limitations here. For a more in-depth discussion, please see [15].

Firstly is exploratory data analysis [32] (EDA). While EDA can help analyse features and attributes of the data, it does not tell us anything about the model being used. For this reason, we do not consider EDA as part of IML/XAI. Rather, a preprocessing step for data explanation (and not model explanation). Likewise, feature selection can also be considered as IML [34]. However, again here we consider this a potential preprocessing step only.

Next is to use explainable models directly, with methods such as decision trees, linear models, or simple classification rules. While it is true that these models can offer high interpretability, the performance is drastically lower than the current state-of-the-art methods in ML (e.g. neural networks, random forests and boosting). This is true for most classification algorithms, where there is a trade-off between the accuracy and interpretability of the models [8]. Often, this drop in predictive ability practical is too large to consider for the increased interpretability. For this reason, simple models on their own are not an ideal approach to IML.

Sparse models are another approach. However, while sparsity does simplify models, they are still not necessarily interpretable. For example consider applying an L1 penalty to a deep neural network, despite having zeroed out some weights. This resulting model is still far from interpretable.

For deep learning methods, there are various explanation methods which can be used, such as sensitivity analysis with partial derivatives, heat maps of activation's, layer-wise methods [28], or deconvolutional networks [36]. The limitation with such methods is that they are only relevant to deep learning models (not arbitrary black-box models), and also are often local (applicable to a single prediction only) in their application.

There are other local model agnostic approaches to IML, such as LIME [27], which give information about particular predictions, but not on the global behaviour of a system. Local explanations can be useful but should be paired with a global approach to provide a fuller understanding.

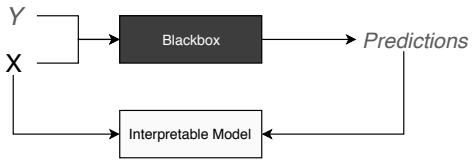


Figure 1: Model extraction process. The black-box model uses the original *y* labels for training, whereas the interpretable model uses the *predictions* from the black-box model.

A global model agnostic approach to IML is *model extraction* [1] (also called *mimic models* [18] or global surrogate models [23]). One of the trade-offs of using interpretable models was a drop in predictive performance, so one solution is to utilise two models - one black-box (complex) model for predictions, and a secondary simple model for describing the black-box. This is referred to as model extraction [1]. Rather than the secondary model being trained on the original outputs, the secondary model is trained on the predictions from the black-box model. This process is shown in Fig. 1. This is the approach we take in this work, due to the fact that the method is applicable to any black-box method and makes no further assumptions (such as gradient-based, or ability to apply sparsity).

It should also be mentioned that it is not always the case that interpretability is important, i.e. to prevent "gaming the system" [18], or in well-studied problems [14]. A model extraction approach means the standard models used and work-flows can remain the same, however, in cases where interpretability is required, a secondary model can be utilised to gather additional insights to the complex model.

2.1 Model Extraction

One of the early works in the area was [7] which used decision trees to approximate complex black-box models. Similar work was done in [1]. Both utilise decision trees as the simple method for approximating more complex models. However, an issue is ensuring these decision trees remain simple themselves, so operations such as early stopping or pruning become essential. Furthermore, due to the greedy construction of the trees, they may not be the best approximator of the more complex black-box methods. Other methods such as logistic regression can also be used, and these are compared in Section 4.

Bayesian Rule lists [20, 35] are an approach to IML which aim to achieve a good balance between complexity and accuracy, by using Bayesian optimisation to generate a set of "*if...then...*" statements which can be used for prediction. The idea is that these resulting rules are simple and easily interpretable.

Model compression was proposed in [5] (and expanded in [17]), where the authors use a neural network to compress large complex ensembles (often with thousands of base members) by training the smaller neural networks on the predictions of the ensemble. While these are not directly related to interpretable machine learning (as the neural network learnt is not necessarily interpretable), the concept is similar, and these works showed the simpler model can often achieve similar error rates to the larger, more complex ensembles, so this is promising for model extraction methods.

3 THE NEW METHOD

In this section, a novel tree-based method is proposed for XAI. Rather than the greedy construction seen with typical tree-based methods (such as CART), the proposed method uses GP to approximate an optimal tree. The tree construction process aims to maximise the reconstruction ability (mimic the predictions of a complex black-box) while minimising the complexity of the trees. The resulting trees are often far simpler while providing equivalent reconstruction ability to current approaches.

To overcome the limitations outlined in Section 1, we use NSGA-II [11], paired with strongly-typed GP (STGP) [24], to evolve decision tree-like structures, which simultaneously balance the complexity and the accuracy of the trees, by approximating a global search of the potential trees. An approximation is required as global search would not be possible for any real datasets as the tree construction process is NP-complete. Hence, the goal is to outperform greedy methods, while still being computationally feasible (at the expense of accepting near-optimal trees). Another benefit of such an approach is that rather than producing a single tree, a Pareto front of non-dominated trees is produced, which is particularly important for XAI since the user can select a tree by visualising the trade-off between the complexity and accuracy (examples of such visualisations are given in Section 5).

3.1 Overall Algorithm

The overall training algorithm is shown in Fig. 2. The black-box classifier is trained once only on the original data (x and y values), then the evolutionary process is performed based on the resulting predictions ($\hat{y}$) from this black-box model (for a total 50 generations). The evolutionary algorithm never sees the original labels (y), as this is instead attempting to recreate the predicted labels $\hat{y}$. At the end of the evolutionary run, the result is a set of Pareto optimal models/trees which approximate the complex black-box model. Only the model with the highest reconstructive ability is used here (i.e. the largest $f1$). The overall evolutionary process is similar to NSGAII. When selecting individuals, the non-dominated sorting in NSGA-II algorithm is used to rank the individuals. The two objectives are outlined in the next section.

3.1.1 Objective Functions. The two objectives are the reconstruction ability (maximisation) and the complexity (minimisation). In this case, the complexity is measured as the number of splitting points in a tree. The reconstruction ability is measured as an internal (i.e. on the training set only) 3-fold ($K = 3$) cross-validation. Meaning the two objectives are:

Objective 1:

$$maximise \frac{1}{K} \sum_{i=1}^{K} f1(predict(fold(i)), blackbox\_predict(fold(i)) \tag{1}$$

where f1 is the weighted f1-score (i.e. weighted by the number of instances per class c)

$$f1(predicted, real) = (\sum_{c \in C} |c| \times \frac{2 \times precision \times recall}{precision + recall}) / \sum_{c \in C} |c| \tag{2}$$

Objective 2:

$$minimise \sum split\_points \tag{3}$$

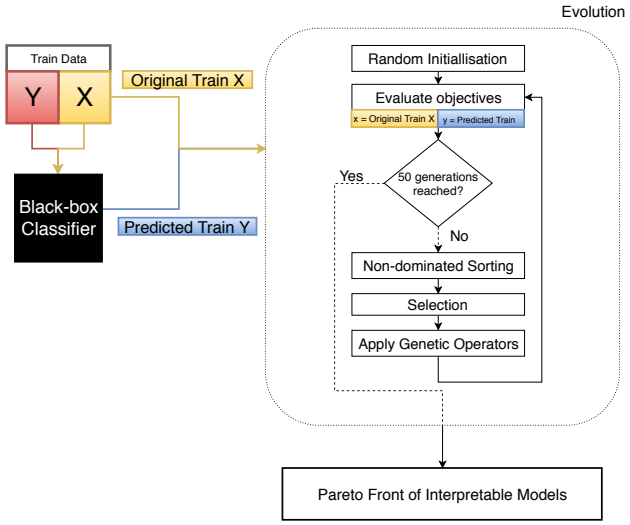


Figure 2: Evolutionary Training Process.

Utilising the average reconstruction ability of multiple folds helps create more robust trees, in that a small change to the dataset will not result in a drastic change in the resulting tree. This is in contrast to decision-tree learners which can change drastically with small changes in input. Robustness is an important concept in XAI, as if the learnt rules change drastically with the training data, this can diminish faith in the learnt rules as it implies the rules were very specific to the training data (overfit) rather than generalising well to the new/unseen cases.

3.1.2 Representation. The evolved trees are decision tree-like, meaning the internal (function) nodes are splitting points (binary for numeric data, and a branch for each category in categorical data), and the output of such nodes are probability vectors for each class. An example of such a tree is shown in Fig. 3. Numeric splitting points are treated as typed terminals (so values will only be crossed over between trees for a given feature, and not between features), and these values are sampled uniformly from feature ranges.

As the output of all nodes are probability vectors (as in decision trees), multiclass classification is supported directly. There are several other ways GP has been used for multi-class classification in the literature, but the output of probability vectors makes fewer assumptions (e.g. numeric outputs with class boundaries [37] assumes an ordering of classes), has better run time than others (e.g. a tree for each class [21], the run time scales with the number of classes), and also is the most interpretable due to the avoidance of complex numerical expressions or multiple trees, while also following closer to that of decision trees.

With decision trees, the input is at the root, and the output at a leaf. Traditionally with GP, the inputs are at the leaves (terminals), and the output is at the root. To get around this, data is passed in at the leaves but only functions are returned until the root node (only one branch will end up returning values for a given input), and then the functions are executed once we are at the root. For visualisation and use-cases, the two can be treated uniformly, but the details are given for clarity (i.e. in the visualisations in Section 5 class values are leaf nodes, but they are merely the majority class predicted for this branch, rather than being an evolvable node).

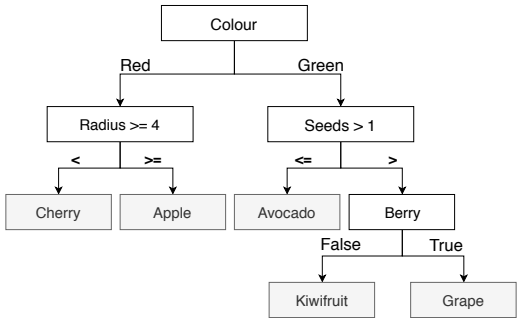
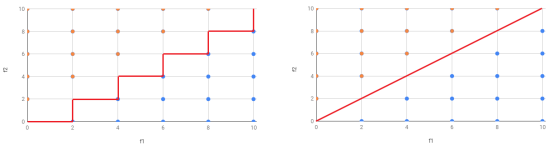


Figure 3: An example tree for fruit classification demonstrating the structure. Each node has a probability vector associated with it, but this is excluded for readability. The grey nodes indicate the predicted class (i.e. the class which had the highest probability).



(a) Decision Tree splitting criteria. **(b) GP Splitting criteria using constructed features**

Figure 4: A comparison of the splitting points when using axis-parralel methods (decision trees) and oblique methods (GP with constructed features). Oblique methods can result in simpler splits as shown here.

The trees are also designed to have the ability to construct features implicitly, from the observation that some simple patterns are not easily approximable in decision trees due to the axis-parallel splits of decision trees. Consider the simple (artificial) dataset with two features $f1, f2$, and two classes $0, 1$, where the condition is *if* $f1 >= f2$ *then class* 1 *else class* 0. With decision trees, the resulting trees would be needlessly complex, as shown in Fig. 4. In addition to the standard binary splitting nodes discussed above, we use subtrees to construct features (as mathematical expressions), then check if these constructed features are greater than or equal to zero. This encapsulates many checks for mathematical relations, i.e. f1 - f2 >=0 is equivalent to f1 >= f2 from above. Constructed features can be combined using the standard mathematical operations (+, -, *, protected /). Again, this can be important for XAI due to the dramatically simpler rules learnt, and utilising the complexity as the secondary objective prevents these constructed features from becoming overly complex.

4 EXPERIMENTS

4.1 Experiment Details

4.1.1 Datasets. 30 datasets from the OpenML repository [33] were used for comparison. These were the 20 most run binary datasets, and 10 most run multiclass datasets. The datasets were restricted to less than 15000 instances, less than 5 classes, and no missing values. These datasets are from a variety of domains, and have a varying number of features (with both categorical and numeric), classes, and instances. The datasets offer a broad range to ensure generalisability of the proposed method. These datasets are summarised in Table 1.

4.1.2 Comparison Methods. Current state-of-the-art approaches to model extraction were trialled. These were Bayesian rule lists from [35] (as *pysbrl* on pip), an h2o decision tree ([31]), a simplified scikit-learn [25] decision tree [22], and logistic regression with $\ell 1$-regularization (from scikit-learn). For the scikit-learn methods, unfortunately, these do not natively support categorical features, so a one-hot-encoding is needed to be applied prior. For Bayesian rule lists, they currently only support discrete features, so as is commonly done, multi-interval discretization as proposed in [16] was first applied. No preprocessing was required for the h2o trees. Unfortunately, to our knowledge there are no current approaches in literature to multi-objective XAI, so we only compare the best resulting solutions rather than comparing frontiers as is more commonly done with multi-objective optimisation algorithms.

4.1.3 GP Parameter Settings. The evolutionary search was run for 50 generations, with 100 trees in each generation. A larger population size could be utilised, and the results would likely improve at the expense of an increased runtime. Trees were limited to a maximum height of 17. A crossover rate of 0.8 was used, and a mutation rate of 0.2. Top performing individuals are never lost, as the $\mu + \lambda$ algorithm [2] is used, with both values set to the population size (i.e. keep the 100 best).

4.1.4 Evaluation Measures.

Classification Performance. For measuring performance, we use a weighted f1-measure where the inputs are the predicted labels and the black-box labels (rather than the real class labels). This was chosen as we assume each class is equally important, and wish to have a valid measure for both binary and multi-class classification. This measure can be roughly interpreted as "how well are we able to reconstruct the predictions of the black-box classifier for each class". For presentation, we scale this to the range $0 \ldots 100$, rather than $0 \ldots 1$.

Complexity. Measuring complexity across classifiers can be a complex task, however, here thankfully since each of the comparison methods is somewhat similar in representation, there is a natural definition of complexity. We define complexity as the number of splitting points in a tree, where in the proposed method, if a constructed feature is used as a split, this counts as multiple splits, i.e. f1+f1<=0, would be a complexity of 2, rather than 1, to provide a fair comparison. For Bayesian rule lists, the complexity is the number of rules + the number of conjunctions in these rules, i.e. *if* $f1 = 2 \wedge f2 = 0$ *then* ... counts as 2. The number of rules in logistic regression is measured as the number of non-zero coefficients. Therefore, for all methods, the minimum complexity is 0 (i.e. predict majority class, no rules learnt), and the maximum approaches ∞.

4.1.5 Black-box Methods. The aim is to have the proposed method be invariant to the black-box model used. For this reason, we chose to use three of the most common and high performing black-box models, which are random forests, gradient boosting, and a deep neural network (with 200 hidden layers with 200 neurons in each layer) all implemented in h2o [31]. 500 trees are used for random

Table 1: Summary of dataset characteristics

Dataset	Numeric Features	Categorical Features	Classes	Instances
analcatdata	70	0	4	841
autoUniv-au7-500	8	4	5	500
balance-scale	4	0	3	625
blood-transfusion	4	0	2	748
climate-model	20	0	2	540
cmc	2	7	3	1473
credit-g	7	13	2	1000
diabetes	8	0	2	768
eeg-eye-state	14	0	2	14980
GesturePhase	32	0	5	9873
hill-valley	100	0	2	1212
ilpd	9	1	2	583
iris	4	0	3	150
kc1	21	0	2	2109
kc2	21	0	2	522
kr-vs-kp	0	36	2	3196
monks-problems-1	0	6	2	556
monks-problems-2	0	6	2	601
ozone-level-8hr	72	0	2	2534
pc1	21	0	2	1109
phoneme	5	0	2	5404
qsar-biodeg	41	0	2	1055
spambase	57	0	2	4601
splice	0	60	3	3190
steel-plates-fault	33	0	2	1941
tic-tac-toe	0	9	2	958
vehicle	18	0	4	846
wall-robot-navigation	24	0	4	5456
waveform-5000	40	0	3	5000
wdbc	30	0	2	569

forests and gradient boosting, and the remaining hyperparameters kept as default. The aim is to see how well a single tree is able to reconstruct the predictions of 500 ensembles trees, or a deep network with 200 hidden layers. It is important to note these methods were not finely tuned and the relative performance of each is not of importance in this paper, as we are *not* trying to compare black-box methods (rather the ability to reconstruct their predictions).

4.1.6 Significance Tests. To compute whether the difference in reconstruction ability across datasets for each method was statistically significant, we used Friedman tests paired with Nemenyi post-hoc analysis as suggested in [12] as general performance is the main focus. Significance tests on each dataset were not used for the reasons described in [29], i.e. violations in the conclusions due to the increased probability of type-I errors, as well as the Friedman test making fewer assumptions.

To show the average reconstruction ability for each method, for each dataset, we present the average testing accuracy of a 10-fold cross-validation procedure, where each method gets the same train:test sets. The averages are also across the 3 black-box methods (so for each method, 30 runs are executed for each dataset, with the same random samples used across methods). The goal is for the extraction methods to be invariant to the black-box model used, hence the averaging.

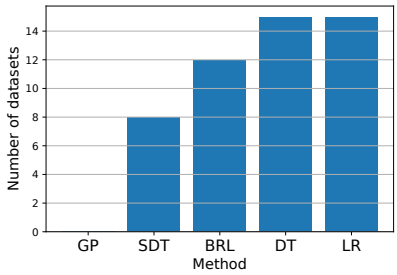


Figure 5: Number of datasets on which a method was dominated by other methods (lower the better)

4.2 Results

The results are shown in Table 2, on the two measures of concern: reconstruction ability and complexity.

As we are interested in achieving the best reconstruction ability, while also achieving the simplest representation, we compute the number of times a method was dominated across datasets. Here, the definition of non-dominated is no other method achieves either a simpler representation with the same (or improved) recreation ability, i.e., fewer dominations is a good sign.

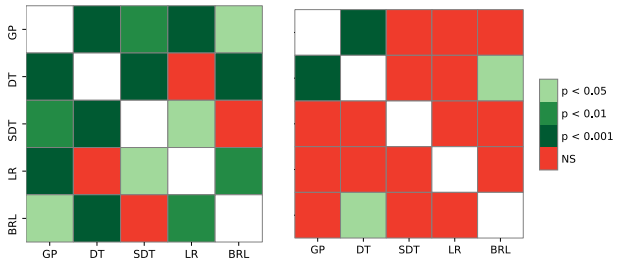
The number of times dominated was chosen as the comparison methods present only a single solution, therefore we can not compare hypervolumes of resulting frontiers (despite the proposed method returning a frontier), yet we still wish to compare the two objectives without making assumptions about the importance of either (i.e. without computing a scalar value). As the proposed method returns a frontier of solutions, only the resulting solution with the highest reconstruction ability (as measured by the fitness function on the training data, not on the unseen data), was used to represent the performance. This result is shown in Fig. 5.

GP (the proposed method) was not dominated on any of the datasets. One caveat is that analysing the dominated counts alone is not a comprehensive indicator of performance, since if the simplest possible model was used (i.e. just use the majority class, which would give a complexity of 0 as no rules were learnt), then this would never be dominated. Likewise, using the black-box model itself would achieve maximal reconstruction ability, and even though the complexity would be far greater this would still never be dominated. Another argument could be that since GP was the only method which simultaneously balanced these objectives, this measure can be seen as biased towards the proposed. This is true, but also shows the ability/usefulness of population-based techniques such as GP as they can effectively optimise multiple objectives simultaneously. This shows that multi-objective optimisation is a good choice for IML, as the objectives were optimised better than the existing approaches (in terms of dominance).

In addition to the dominated counts, the relative performance of the methods must also be considered. Friedman testing paired with Nemenyi post-hoc analysis is performed to compare whether the difference in the resulting accuracies was statistically significant across datasets. The resulting p-values are visualised in Fig. 6, where the only statistically significant differences in recreation ability are between the proposed method and the standard decision tree, and Bayesian rule lists and the decision tree. For complexity, the

Table 2: Summary of the results. The average testing performance is presented.

	Black-box Test Accuracy			Test Reconstruction Ability					Model Complexity				
	RF	GB	DL	GP	BRL	SDT	DT	LR	GP	BRL	SDT	DT	LR
Analcatdata_Authorship	99.40	98.70	99.80	83.21	92.33	92.3	91.71	98.81	5	17	6	59	122
Autouniv-Au7-500	47.50	44.52	38.30	55.69	36.83	49.49	50.64	46.57	7	1	18	163	250
Balance-Scale	82.70	86.40	96.90	72.9	73.06	77.6	81.44	80.68	6	9	13	127	12
Blood-Transfusion	66.70	73.70	70.50	81.65	79.24	86.34	87.71	78.73	4	5	8	35	4
Climate-Model	86.90	85.30	88.20	88.44	92.47	90.66	89.72	91.09	3	3	6	22	13
Cmc	54.30	54.20	45.90	60.58	51.84	58.6	68.62	67.81	6	7	11	123	150
Credit-G	73.60	75.70	72.40	76.61	57.3	76.92	77.12	80.19	7	2	8	46	76
Diabetes	74.20	73.40	71.00	77.5	79.56	78.72	78.15	79.26	4	9	9	39	8
Eeg-Eye-State	93.10	87.50	78.90	54.26	77.65	71.86	74.3	49.28	4	120	8	50	14
GesturePhase	67.00	62.80	60.00	47.51	53.16	55.94	61.35	47.14	5	81	10	270	23
Hill-Valley	35.70	52.70	64.10	80.57	76.68	84.96	82.26	81.4	4	3	18	27	100
Ilpd	65.40	66.60	70.00	67.2	51.29	63.98	70.13	67.19	4	1	8	32	118
Iris	93.30	94.64	98.30	98.93	97.37	98.46	96.89	95.4	3	3	3	15	10
Kc1	82.30	84.30	82.70	85.19	83.98	85.8	86.14	85.34	4	10	5	39	21
Kc2	77.80	80.06	86.10	86.79	86.38	87.33	85.96	85.7	3	5	6	27	18
Kr-Vs-Kp	98.80	99.40	99.00	94.22	92.25	96.62	94.48	96.8	6	15	8	16	57
Monks-Problems-1	99.80	98.90	99.10	86.63	72.44	92.9	86.34	72.44	7	2	15	14	10
Monks-Problems-2	92.30	97.10	99.80	55.11	52.85	86.3	89.93	52.42	10	0	28	43	10
Ozone-Level-8Hr	93.70	93.20	93.60	93.44	93.76	94.82	94.52	95.55	3	11	5	36	61
Pc1	91.90	92.80	91.50	92.95	93.26	93.45	94.6	92.46	4	5	6	26	20
Phoneme	91.10	88.50	90.71	81.77	85.02	82.17	88.36	76.96	6	35	6	47	5
Qsar-Biodeg	87.00	86.60	84.70	81.14	86.52	85.81	87.1	89.59	4	18	8	38	31
Spambase	95.20	95.40	93.90	81.59	94.27	89.27	92.42	94.65	4	45	5	33	54
Splice	97.30	96.20	95.10	82.09	89.66	93.62	95.47	96.99	6	24	10	100	449
Steel-Plates-Fault	99.70	94.50	99.80	88.21	99.67	99.78	99.8	99.74	5	7	6	7	20
Tic-Tac-Toe	98.80	97.40	97.60	73.07	61.3	91.1	92.25	97.83	6	3	18	33	27
Vehicle	75.20	77.30	84.30	61.14	66.36	72.65	77.9	80.59	6	17	15	112	72
Wall-Robot-Navigation	99.20	99.69	92.50	78.63	96.23	95.33	97.4	68.53	5	58	8	79	96
Waveform-5000	85.30	83.80	83.20	72.64	79.48	76.91	83.51	92.5	6	71	8	168	117
Wdbc	95.40	94.40	98.70	95.21	95.13	94.14	94.2	95.48	4	8	4	17	11



(a) Complexity (b) Reconstruction Ability

Figure 6: Statistical significance testing. Resulting p-values from Friedman test w/ Nemenyi post-hoc correction. Red indicates no significant difference. Green indicates a difference (darker=more significant).

proposed method was significantly simpler than all comparisons methods.

To analyse the results on specific results, a per dataset breakdown of the reconstruction ability vs complexity is given in Fig. 7, where the ideal position is the top left of the chart (i.e. minimal complexity, maximal reconstruction ability).

From this, we are able to conclude the proposed GP-based method consistently produces compact rules, while achieving statistically equivalent accuracy to the more complex approaches. The one exception to this was the decision tree, which was, however, on average 15× more complex than the proposed approach.

5 FURTHER ANALYSIS

To give further insights on the resulting extracted models, a comparison is given on the hill-valley dataset across the methods. The hill-valley dataset was chosen as this is the most challenging (i.e. lowest average test accuracy on the black-box models), and the deep learning recreations are used. While we would like to present all results for all methods, this would compromise 900 diagrams per method. Logistic regression is not presented, as this only a list of up-to 100 coefficients for the features, and 1 value for the intercept. These comparisons are shown in Fig. 8.

From the examples (Fig. 8), we can see that the resulting tree from the proposed method and the Bayesian rule list are by far the

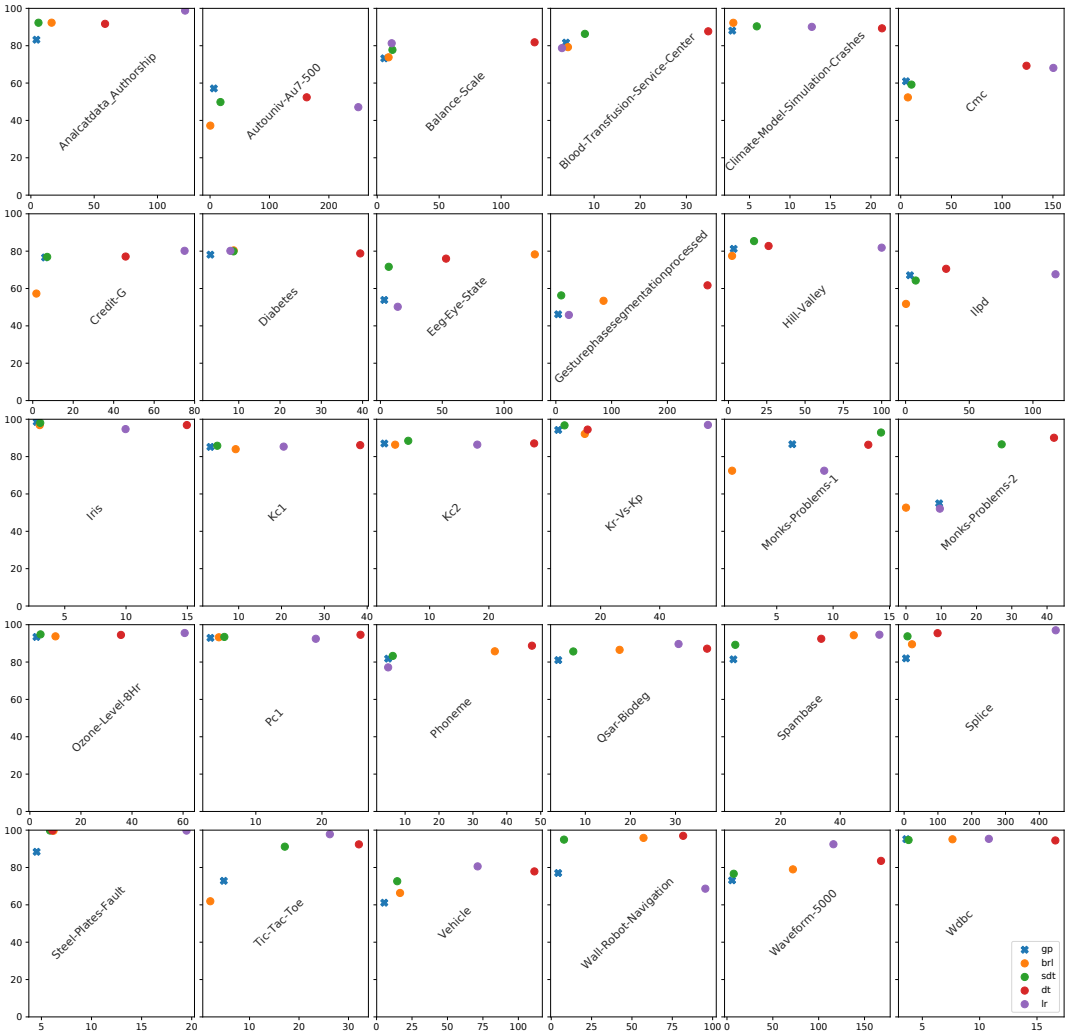


Figure 7: Testing recreation ability (y) vs Complexity (x). The ideal position is the top left corner of the graph.

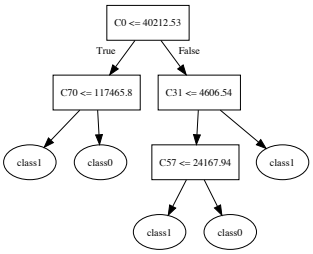
simplest of the compared methods, condensing the approximate knowledge from a 200 layer neural network into small human readable form. A similar trend is seen across the datasets when comparing the complexities. In this case, the Bayesian rule-list actually just predicts 1 class, so is overly simplistic (shown by the differences in the f-measures). To get an idea of what GP has found, the evolved tree can be consulted (Fig. 8a). The evolved tree is attempting to split the data based on four features (or points) in the dataset.

This makes sense when we consider the hill-valley dataset, which "when plotted in order (from 1 through 100) as the Y coordinate, the points will create either a Hill (a 'bump' in the terrain) or a Valley (a 'dip' in the terrain)" [13]. We can see the tree is checking the first point, and comparing to the point at 30% (i.e. the $30th$ feature), or the point at 70%, where the tree is trying to distinguish between classes by finding the common points for the hills/valleys and checking if these are high or low relative to the training data (e.g. a high point at the start, a low point at 30%, then a high point at 57% indicates a valley based on this tree).

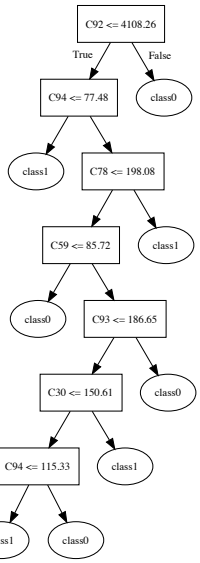
Across the board, the datasets which were most difficult to reconstruct the predictions on were: Autouniv-Au7-500, Eeg-Eye-State, Gesturephasesegmentationprocessed (GesturePhase), and Monks-Problems-2.

Two of these datasets (Autouniv-Au7-500, GesturePhase) have 5 classes. One explanation here is that as the number of classes in a dataset increases, as does the complexity necessarily with tree-based methods. For example, if we have 100 classes, we therefore require 100 leaf nodes to have a predictive branch for each class. This presents a potential area of future research, as the size of the trees could negate the explainability as the number of classes grows. Here the pressure for small trees was perhaps too strong, and this requirement would need to be relaxed in the case of a high number of classes.

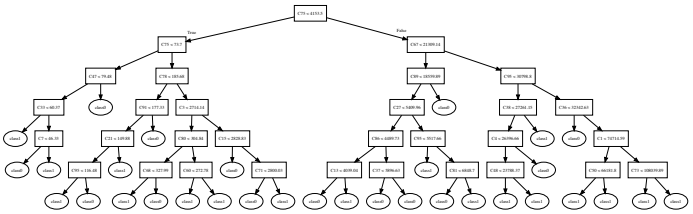
Monks-Problems-2 is entirely categorical features. In the proposed method, a categorical node has a branch for each feature -



(a) Proposed ($f_1 = 0.757$)



(b) Simplified Decision Tree ($f_1 = 0.759$)



(c) Decision Tree ($f_1 = 0.790$)

IF (X43 = 3) AND (X78 = 3) THEN: Class1
ELSE IF (X62 = 0) THEN: Class1
ELSE DEFAULT: Class1

(d) Bayesian Rule List ($f_1 = 0.704$)

Figure 8: A comparison of the resulting models on the hill-valley dataset, attempting to recreate the 200 layer deep neural network predictions. The testing recreation ability is given in brackets next to the method name.

this potentially overfits to the training data, and combining categorical features into a single branch should be considered in future work (for reference, this is done in the decision tree method, where we can see a significant improvement in reconstructive ability, and this is consistent across the datasets with all categorical features).

For eeg-eye-state, the data is sequential/time-series. The proposed method is not optimised/designed for such datasets, so this explains the lower performance.

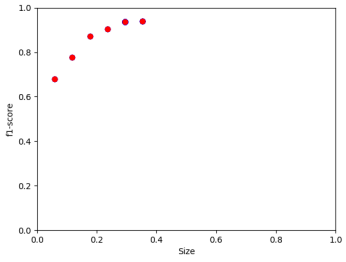


Figure 9: Resulting Pareto front.

To highlight another benefit the proposed method has over the existing IML approaches, a Pareto front is given in Fig. 9. This was the result for one of the runs on the kr-vs-kp dataset, but similar fronts are available for all datasets. In all cases, the model with the highest reconstruction ability was chosen, however, even simpler models could be used from the front if desired. Likewise, if models were overly simple, any restrictions on the height of evolved trees could be relaxed.

6 CONCLUSIONS AND FUTURE WORK

In this work, a novel model agnostic method for XAI was proposed which utilises model extraction. Multi-objective GP is used to learn a simple and interpretable representation of a complex black-box model, which is often able to effectively reproduce the black-box's predictions. This new method was compared to existing approaches for model extraction, and was found to offer drastically simpler models, with statistically equivalent test accuracy. The method is also able to handle categorical and continuous features natively, unlike some existing approaches such as Bayesian rule lists. To our best knowledge, this is the first utilisation of multi-objective optimisation in explainable AI, and follows the suggestions in [9] that "a multi-objective approach based on Pareto dominance would be more suitable to sufficiently address this trade-off" (between accuracy and interpretability). We also believe this is the first application of GP for model extraction (i.e. training on the predictions of a black-box model rather than the original labels), and shows a promising direction for future developments.

In future work, we would like to focus on three main areas. Firstly, can the recreation ability of the proposed method be improved without sacrificing simplicity by considering local search techniques for splitting points, groupings for categorical data, or making use of the black-box model for generating additional labelled instances. Secondly, here a basic complexity measure is used for evaluating the simplicity of the models. However, if the goal is human interpretation, it would be ideal to conduct a blind large scale user studies on the resulting models. And finally, related to the previous point, it is possible to guide the evolution of the models based on the human feedback, we foresee these human-in-the-loop type systems being important for XAI, and this is something that could be incorporated into the evolutionary process here by modifying the complexity measure to instead be user determined.

REFERENCES

[1] Osbert Bastani, Carolyn Kim, and Hamsa Bastani. 2017. Interpretability via model extraction. *arXiv preprint arXiv:1706.09773* (2017).

[2] Hans-Georg Beyer and Hans-Paul Schwefel. 2002. Evolution strategies–A comprehensive introduction. *Natural computing* 1, 1 (2002), 3–52.

[3] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*. 4349–4357.

[4] Indranil Bose and Radha K Mahapatra. 2001. Business data mining - a machine learning perspective. *Information & management* 39, 3 (2001), 211–225.

[5] Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. 2006. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 535–541.

[6] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. 2015. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1721–1730.

[7] Mark Craven and Jude W Shavlik. 1996. Extracting tree-structured representations of trained networks. In *Advances in neural information processing systems*. 24–30.

[8] Hoa Khanh Dam, Truyen Tran, and Aditya Ghose. 2018. Explainable Software Analytics. *CoRR* abs/1802.00603 (2018). arXiv:1802.00603 http://arxiv.org/abs/1802.00603

[9] Hoa Khanh Dam, Truyen Tran, and Aditya Ghose. 2018. Explainable software analytics. In *Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results*. ACM, 53–56.

[10] Kalyanmoy Deb. 2014. Multi-objective optimization. In *Search methodologies*. Springer, 403–449.

[11] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation* 6, 2 (2002), 182–197.

[12] Janez Demšar. 2006. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research* 7, Jan (2006), 1–30.

[13] Dua Dheeru and Efi Karra Taniskidou. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[14] Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608* (2017).

[15] Filip Karlo Došilović, Mario Brčić, and Nikica Hlupić. 2018. Explainable artificial intelligence: A survey. In *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 0210–0215.

[16] Usama Fayyad and Keki Irani. 1993. Multi-interval discretization of continuous-valued attributes for classification learning. (1993).

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531* (2015).

[18] Been Kim and Finale Doshi-Velez. [n. d.]. ICML 2017 tutorial on interpretable machine learning. http://people.csail.mit.edu/beenkim/icml_tutorial.html

[19] John R Koza. 1994. Genetic programming as a means for programming computers by natural selection. *Statistics and computing* 4, 2 (1994), 87–112.

[20] Benjamin Letham, Cynthia Rudin, Tyler H McCormick, David Madigan, et al. 2015. Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model. *The Annals of Applied Statistics* 9, 3 (2015), 1350–1371.

[21] Thomas Loveard and Victor Ciesielski. 2001. Representing classification problems in genetic programming. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, Vol. 2. IEEE, 1070–1077.

[22] Tamas Madl. 2018. Sklearn interpretable tree. https://github.com/tmadl/sklearn-interpretable-tree

[23] Christoph Molnar. 2018. Interpretable machine learning. *A Guide for Making Black Box Models Explainable* (2018).

[24] David J Montana. 1995. Strongly typed genetic programming. *Evolutionary computation* 3, 2 (1995), 199–230.

[25] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[26] General Data Protection Regulation. 2016. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46. *Official Journal of the European Union (OJ)* 59, 1-88 (2016), 294.

[27] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "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, San Francisco, CA, USA, August 13-17, 2016*. 1135–1144.

[28] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. *arXiv preprint arXiv:1708.08296* (2017).

[29] Juliet Popper Shaffer. 1995. Multiple hypothesis testing. *Annual review of psychology* 46, 1 (1995), 561–584.

[30] Latanya Sweeney. 2013. Discrimination in online ad delivery. *Queue* 11, 3 (2013), 10.

[31] The H2O.ai team. 2015. *h2o: Python Interface for H2O*. http://www.h2o.ai Python package version 3.1.0.99999.

[32] John W Tukey. 1977. *Exploratory data analysis*. Vol. 2. Reading, Mass.

[33] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. 2013. OpenML: Networked Science in Machine Learning. *SIGKDD Explorations* 15, 2 (2013), 49–60. https://doi.org/10.1145/2641190.2641198

[34] Alfredo Vellido, José David Martín-Guerrero, and Paulo JG Lisboa. 2012. Making machine learning models interpretable.. In *ESANN*, Vol. 12. Citeseer, 163–172.

[35] Hongyu Yang, Cynthia Rudin, and Margo Seltzer. 2016. Scalable Bayesian rule lists. *arXiv preprint arXiv:1602.08610* (2016).

[36] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, and Rob Fergus. 2010. Deconvolutional networks. (2010).

[37] Mengjie Zhang and Will Smart. 2004. Multiclass object classification using genetic programming. In *Workshops on Applications of Evolutionary Computation*. Springer, 369–378.