LEARNING AXIS-ALIGNED DECISION TREES WITH GRADIENT DESCENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Decision Trees are commonly used for many machine learning tasks due to their high interpretability. However, learning a decision tree from data is a difficult optimization problem, since it is non-convex and non-differentiable. Therefore, common approaches learn decision trees using a greedy growth algorithm that minimizes the impurity at each internal node. Unfortunately, this greedy procedure can lead to suboptimal trees.

In this paper, we present a novel approach for learning univariate, axis-aligned decision trees with gradient descent. This is achieved by applying backpropagation with an adjusted gradient flow on a dense decision tree representation that optimizes all decision tree parameters jointly. We show that our gradient-based optimization outperforms existing baselines on several binary classification benchmarks and achieves competitive results for multi-class tasks. To the best of our knowledge, this is the first approach that attempts to learn univariate, axis-aligned decision trees with gradient descent.

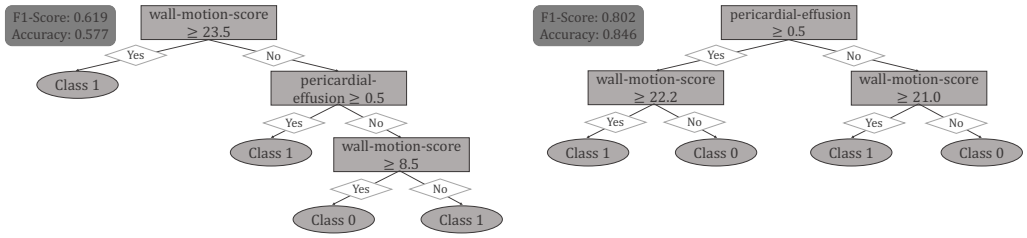
1 INTRODUCTION

Decision trees are one of the most popular machine learning models and are still frequently used today. Especially with the increasing interest in explainable artificial intelligence (XAI), decision trees received popularity. However, learning a decision tree is a difficult optimization problem, since it is non-convex and non-differentiable. Finding an optimal decision tree for a specific task is a NP-complete problem (Laurent & Rivest, 1976). Therefore, the common approach to construct a decision tree based on data is a greedy procedure (Quinlan, 1986; Breiman et al., 1984; Quinlan, 1993) that minimizes the impurity at each internal node. The algorithms that are still used today, namely CART Breiman et al. (1984) and C4.5 Quinlan (1993), date back until the 1980s and since then remained mostly unchanged. Unfortunately, a greedy algorithm only yields to locally optimal solutions and therefore can lead to suboptimal trees. We will illustrate the issues that can arise when learning decision trees with a greedy algorithm in the following example:

Example 1 The Echocardiogram dataset[1] deals with predicting one-year survival of patients after a heart attack based on tabular data from an echocardiogram. Figure 1 shows two decision trees. The left tree is learned using a greedy algorithm (CART) and the right tree is learned using our gradient-based approach. We can observe that the greedy procedure leads to a suboptimal tree with a significantly lower performance. Splitting on the *wall-motion-score* is the locally optimal split (see Figure 1a), but globally, it is beneficial to split based on the *wall-motion-score* with different values conditioned on the *pericarcial-effusion* in the second level (Figure 1b).

The contribution of this paper is a novel approach for learning univariate, axis-aligned decision trees based on global optimization with gradient descent, which we call gradient-based decision trees (GDTs). Using a gradient-based optimization, we can overcome the limitations of greedy approaches, as indicated in Figure 1b. We propose a suitable dense decision tree representation that allows a gradient-based optimization of the tree parameters (Section 3.2). We further present an algorithm that allows us to deal with the non-differentiable nature of decision trees, while still allowing an efficient optimization using backpropagation with an adjusted gradient flow (Section 3.3).

[1]Available under: `https://archive.ics.uci.edu/ml/datasets/echocardiogram` (last accessed 16.08.2022)



(a) Greedy DT (CART)

(b) Gradient-Based DT (GDT)

Figure 1: **Greedy versus Gradient-Based Decision Tree.** Two DTs learned on the Echocardiogram dataset. The left tree is learned based on a standard greedy procedure using the CART algorithm, and the right tree is learned based on our gradient-based approach. While the CART DT only makes locally optimal splits, the gradient-based tree uses a global optimization, which results in a significantly higher performance.

We empirically evaluate GDTs on several real-world benchmark datasets (Section 4). GDTs outperform existing baselines on several benchmark datasets and achieves competitive results for decision tree learning. Furthermore, the resulting trees are less prone to overfitting and represent a more accurate quantification of uncertainty, since they comprise probability distributions at the leafs. A gradient-based optimization also provides more flexibility, since splits can be adjusted during the training. This allows an application of GDTs to dynamic scenarios, as for instance online learning.

2 Related Work

Starting with the publication of the first decision tree learning algorithm in 1963 (Morgan & Sonquist, 1963) the task of learning a decision tree from data remained well-researched until today. The most prominent and still frequently used algorithms, namely CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993) as an extension of ID3.0 (Quinlan, 1986), data back to 1980s and both follow a greedy procedure to learn a decision tree. Since then, many variations to those algorithms have been proposed, as for instance C5.0 (Kuhn et al., 2013) and GUIDE (Loh, 2002; 2009). C5.0 as an extension to C4.5 aims to reduce the runtime and size of the trees. Furthermore, GUIDE (Loh, 2002; 2009) was proposed by Loh as an improvement to existing greedy algorithms which is still under active development and mainly includes additional heuristics for feature selection and the determination of the split threshold. However, until today, none of these algorithms was able to consistently outperform CART (Breiman et al., 1984) and C4.5 (Quinlan, 1993) as shown by Zharmagambetov et al. (2021).

To overcome the issues of a greedy decision tree induction, many researchers focused on finding an efficient alternative. One solution to mitigate the impact of a greedy procedure are methods that look ahead during the induction (Sarkar et al., 1994). However, Murthy (1996) argue that those methods not only suffer from an enormous increase in the computational complexity, but also suffer from pathology, i.e., they frequently produce worse trees in terms of accuracy, tree size and depth. One explanation could be that lookahead trees, especially without regularization, are prone to overfitting.

Optimal decision trees try to optimize an objective (e.g., the purity) using approximate brute force search to find a globally optimal tree with a certain specification (Zharmagambetov et al., 2021). Aglin et al. (2020) and Lin et al. (2020) approximate a brute force search using a branch-and-bound algorithm to remove irrelevant parts from the search space. In contrast, OCT (Bertsimas & Dunn, 2017) defines the optimization as a mixed integer optimization (MIO) problem using binary variables, which is solved using a MIO solver. However, these approaches suffer from a very high computational complexity and are not able to consistently outperform greedy methods (Zharmagambetov et al., 2021).

In contrast to standard decision trees that make hard decision at each internal node, many approaches to hierarchical mixture of expert models (Jordan & Jacobs, 1994) have been proposed. They make soft decisions where each branch is associated with a probability (Irsoy et al., 2012; Frosst & Hinton, 2017). The resulting models do not comprise univariate, axis-aligned splits, but are oblique with

respect to the axis. These adjustments in the tree architecture allow the application of further optimization algorithms, as for instance the expectation–maximization (EM) algorithm for maximum likelihood estimation (Dempster et al., 1977; Jordan & Jacobs, 1994) or gradient descent (Irsoy et al., 2012; Frosst & Hinton, 2017). However, they are much more complex and more difficult to interpret than standard decision trees, since there are no hard decisions and multiple features are considered at each internal node.

Norouzi et al. (2015) proposed an approach for efficient non-greedy optimization of decision trees that overcomes the need of soft decisions to apply gradient-based algorithms. This includes a joint optimization of the split function at all levels and leaf parameters by minimizing a convex-concave upper bound on the tree's empirical loss. While this allows the use of hard splits, the approach is still limited to oblique trees only and therefore the splits of the resulting tree are significantly more complex and less interpretable. While GDTs, similar to Norouzi et al. (2015), used a gradient-based optimization and make hard splits at each internal node, they additionally provide a dense representation and training procedure that allows the use of univariate, axis-aligned splits at each internal node.Furthermore, GDTs directly optimize the loss function instead of an upper bound of the tree's loss.

Carreira-Perpinán & Tavallali (2018) proposed a tree altering optimization algorithm (TAO) that, given an input tree (its structure and the parameter values at its nodes), generates a new tree with the same or a smaller structure, but optimized parameter values. TAO uses alternating optimization (Bezdek & Hathaway, 2002), to deal with the non-differentiable nature of decision trees. While TAO achieved superior results in many tasks (Carreira-Perpinán & Tavallali, 2018; Zharmagambetov et al., 2021), the algorithm requires an existing DT as input and is not able to learn a decision tree purely from data. Therefore, TAO can be used as an optimization algorithm for decision trees learned with an arbitrary method, including the approach proposed in this paper.

Another way to learn decision trees in a non-greedy fashion is using evolutionary algorithms for decision tree induction. Evolutionary algorithms perform a robust global search in the space of candidate solutions based on the concept of survival of the fittest (Barros et al., 2011). This usually results in a better identification of feature interactions compared to a greedy, local search (Freitas, 2002). The resulting trees usually achieve high accuracies while keeping the tree comparatively simple (Papagelis & Kalles, 2001; Barros et al., 2011).

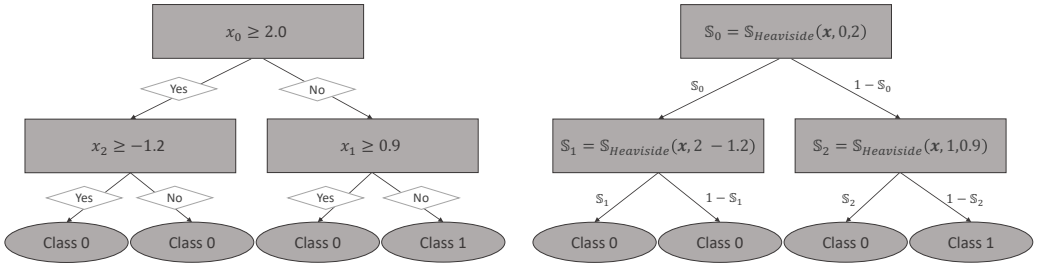
3 Gradient-Based Axis-Aligned Decision Trees

In this section, we propose gradient-based decision trees (GDTs) as a novel approach to learn decision trees. Therefore, we introduce a novel decision tree representation and a corresponding algorithm that allows learning univariate, axis-aligned decision trees with gradient descent. More specifically, we will use backpropagation with an adjusted gradient flow (Section 3.3) on a dense decision tree representation (Section 3.2) to adjust the model parameters during the training, as we will explain in Section 3.4.

3.1 Arithmetic Decision Tree Representation

In the following, we will introduce a notation for decision trees with respect to their parameters. We formulate decision trees as an arithmetic function based on addition and multiplication instead of a nested concatenation of rules. This formulation will be useful for describing the gradient-based learning in Sections 3.2-3.4. Note that we focus on learning fully-grown decision trees within this paper.

The parameters of a decision tree comprise one split threshold and one feature index for each internal node, which we denote as vectors $\boldsymbol{\tau} \in \mathbb{R}^{2^d-1}$ and $\boldsymbol{\iota} \in \mathbb{N}^{2^d-1}$ respectively, where $2^d - 1$ equals the number of internal nodes in a fully-grown decision tree.Additionally, each leaf node comprises a class membership in the case of a classification task or a value in the case of regression tasks, which we denote as the vector $\boldsymbol{\lambda} \in \mathcal{C}^{2^d}$, where $\mathcal{C}$ is the set of classes and 2^d equals the number of leaf nodes in a fully-grown decision trees.Figure 2 shows our tree representation in comparison to a common tree representation for an exemplary decision tree.



(a) Standard Decision Tree Representation (b) Adjusted Decision Tree Representation

Figure 2: **Decision Tree Representations.** Exemplary decision tree representations with depth 2 for a dataset comprising 3 variables and 2 classes. The decision tree at the right can be represented using the vectors $\boldsymbol{\iota} = [0, 2, 1]$, $\boldsymbol{\tau} = [2.0, -1.2, 0.9]$ and $\boldsymbol{\lambda} = [0, 0, 0, 1]$.

Formally, we can express a decision tree as a function $DT(\cdot|\boldsymbol{\tau}, \boldsymbol{\iota}, \boldsymbol{\lambda}) : \mathbb{R}^n \rightarrow \mathcal{C}$ with respect to its parameters:

$$DT(\boldsymbol{x}|\boldsymbol{\tau}, \boldsymbol{\iota}, \boldsymbol{\lambda}) = \sum_{l=0}^{2^d} \lambda_l \, \mathbb{L}(\boldsymbol{x}|l, \boldsymbol{\tau}, \boldsymbol{\iota}) \tag{1}$$

$\mathbb{L}$ is a function that indicates whether a sample $\boldsymbol{x} \in \mathbb{R}^n$ belongs to a leaf l.

The indicator function $\mathbb{L}$ can be defined as a multiplication of the split functions of the preceding internal nodes. We define a split function $\mathbb{S}$ as a Heaviside step function defined as:

$$\mathbb{S}_{\text{Heaviside}}(\boldsymbol{x}|\iota, \tau) = \begin{cases} 1, \text{if } x_\iota \geq \tau \\ 0, \text{otherwise} \end{cases} \tag{2}$$

where ι is the index of the feature considered at a certain split and τ is the corresponding split threshold.

Enumerating the internal nodes of a fully-grown tree with depth d in a breadth-first order, we can now define the indicator function $\mathbb{L}$ as:

$$\begin{aligned} \mathbb{L}(\boldsymbol{x}|l, \boldsymbol{\tau}, \boldsymbol{\iota}) = \prod_{j=1}^{d} & \left(1 - \mathfrak{p}(l, d, j)\right) \, \mathbb{S}(\boldsymbol{x}|\tau_{\mathfrak{i}(l,d,j)}, \iota_{\mathfrak{i}(l,d,j)}) \\ & + \mathfrak{p}(l, d, j) \, \left(1 - \mathbb{S}(\boldsymbol{x}|\tau_{\mathfrak{i}(l,d,j)}, \iota_{\mathfrak{i}(l,d,j)})\right) \end{aligned} \tag{3}$$

Here, $\mathfrak{i}$ is the index of the internal node preceding a specific leaf node l at a certain depth j. Further, $\mathfrak{p}$ defines whether the left branch ($\mathfrak{p} = 0$) or the right branch ($\mathfrak{p} = 1$) was taken at a certain internal node to reach a leaf node l. Both values, $\mathfrak{i}$ and $\mathfrak{p}$, are constant for a certain architecture and can be calculated straightforward (see Appendix A.1).

As becomes evident, decision trees involve non-differentiable operations (Equation 2), which precludes the application of the backpropagation algorithm for learning the parameters. More specifically, there are three major challenges we need to solve if we want to use backpropagation to efficiently learn a decision tree:

C1 The index ι for the selection of the considered feature in a certain split is defined as $\iota \in \mathbb{N}$. However, the index ι is a parameter of the decision tree and a standard optimization with gradient descent requires $\iota \in \mathbb{R}$.

C2 The split function $\mathbb{S}(\boldsymbol{x}, \iota, \tau)$ is a Heaviside step function with an undefined gradient for $x_\iota = \tau$ and 0 gradient elsewhere, which precludes an efficient optimization.

C3 The leaf node in a standard decision tree comprises only the class membership and therefore $\lambda \in \mathcal{C}$. To optimize the leaf node parameters, we need $\lambda \in \mathbb{R}$ to apply gradient descent and calculate an informative loss value as cost function.

The main difference of GDTs to a standard decision tree algorithm is that we use a dense representation for the internal node parameters ($\boldsymbol{\tau}$ and $\boldsymbol{\iota}$) to jointly optimize the split thresholds and the selection of the corresponding features for all internal nodes (Section 3.2). Furthermore, we adjust the gradient flow during the backpropagation to assure the use of hard and univariate decision trees during training and application (Section 3.3).

3.2 Dense Decision Tree Representation

In this subsection, we propose a differentiable representation of the feature indices $\boldsymbol{\iota}$ to allow a gradient-based optimization. Therefore, we extend the vector $\boldsymbol{\iota} \in \mathbb{R}^{2^d-1}$ to a matrix, where $I \in \mathbb{R}^{2^d-1} \times \mathbb{R}^n$. This is achieved by one-hot encoding the feature index as $\boldsymbol{\iota} \in \mathbb{R}^n$ for each internal node. This extension is inspired by the decision tree representation of Marton et al. (2022) used for predicting a decision tree as a numeric output of a neural network. This adjustment is necessary for the optimization process to account for the fact that feature indices are categorical instead of ordinal.

We further propose using a similar representation for the split thresholds as $T \in \mathbb{R}^{2^d-1} \times \mathbb{R}^n$ by storing one value for each feature as $\boldsymbol{\tau} \in \mathbb{R}^n$ instead of a single feature. This adjustment is designed to support the optimization procedure, since the split thresholds are not interchangeable for different features: A split threshold that is reasonable for one feature might not be reasonable for another feature. By storing one split threshold for each feature, we support exploration during the optimization.

Besides the previously mentioned advantages, using a dense decision tree representation allows the use of matrix multiplications for an efficient computation, as we will show in the following. Therefore, we reformulate the Heaviside split function in Equation 2 as follows:

$$\mathbb{S}_{\text{logistic}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) = \sigma\left(\sum_{i=0}^{n} \iota_i x_i - \sum_{i=0}^{n} \iota_i \tau_i\right) \tag{4}$$

$$\mathbb{S}_{\text{logistic_hard}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) = \lfloor \mathbb{S}_{\text{logistic}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) \rceil \tag{5}$$

where $\lfloor \cdot \rceil$ stands for rounding to the closest integer. For our case, where $\boldsymbol{\iota}$ is a one-hot encoded vector for the feature index, $\mathbb{S}_{\text{logistic_hard}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) = \mathbb{S}_{\text{Heaviside}}(\boldsymbol{x}, \iota, \tau)$ holds.

3.3 Backpropagation of Decision Tree Loss

While the dense decision tree representation introduced in the previous section emphasizes an efficient learning of axis-aligned decision trees, it does not solve the challenges **C1**-**C3**. Therefore, in this subsection, we will explain how we solve those challenges by adjusting the gradient flow of the backpropagation algorithm during the gradient descent optimization.

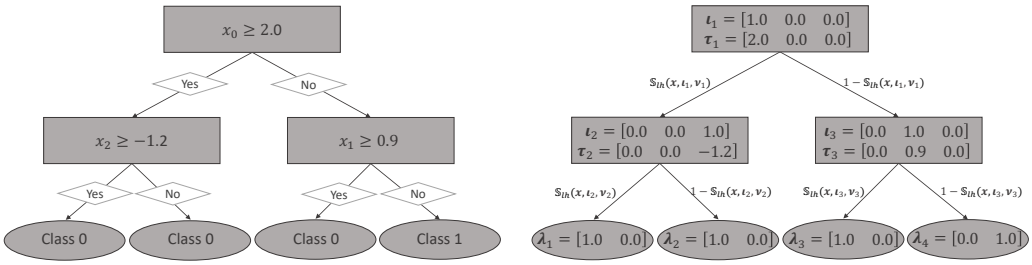
For the function value calculation in the forward pass, we need to assure that $\boldsymbol{\iota}$ is a one-hot encoded vector. This can be achieved by applying a hardmax function on the feature index vector of each internal node. The hardmax function is defined as:

$$\text{hardmax}(\boldsymbol{\iota}) = \begin{cases} 1, \text{if } \iota_i = \max(\boldsymbol{\iota}) \\ 0, \text{otherwise} \end{cases} \quad \text{for } i = 0, \ldots, |\boldsymbol{\iota}| \tag{6}$$

However, applying a hardmax is a non-differentiable operation, which precludes gradient computation. Therefore, we suggest applying the hardmax function as follows:

$$\begin{aligned} &\boldsymbol{\iota} \leftarrow \boldsymbol{\iota} - c^* \\ &\text{where } c^* = \boldsymbol{\iota} - \text{hardmax}(\boldsymbol{\iota}) \end{aligned} \tag{7}$$

We treat c^* as a constant which is not considered when calculating the gradients in the backward pass. Accordingly, we can optimize the parameters of $\boldsymbol{\iota}$ where $\boldsymbol{\iota} \in \mathbb{R}$ while still using univariate splits (**C1**).



(a) Standard Decision Tree Representation (b) Dense Decision Tree Representation

Figure 3: **Decision Tree Representations.** Exemplary decision tree representations with depth 2 for a dataset comprising 3 variables and 2 classes. Here, $\mathbb{S}_{\text{lh}}$ stands for $\mathbb{S}_{\text{logistic_hard}}$ (Equation 5)

A similar solution can be used for the split function in Equation 5:

$$\begin{aligned} &\mathbb{S}_{\text{logistic_hard}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) \leftarrow \mathbb{S}_{\text{logistic}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) - c^* \\ &\text{where } c^* = \mathbb{S}_{\text{logistic}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) - \lfloor \mathbb{S}_{\text{logistic}}(\boldsymbol{x}, \boldsymbol{\iota}, \boldsymbol{\tau}) \rceil \end{aligned} \tag{8}$$

where c^* is considered as a constant when calculating the gradients in the backward pass. This assures the use of hard splits for the function value calculation and simultaneously allows the application of the backpropagation algorithm to compute the gradients for $\boldsymbol{\iota}$ (**C2**). When applying Equation 8, utilizing the sigmoid logistic function as split function at the internal nodes (see Equation 4) utilizes the distance to the split threshold as additional information for the gradient calculation. If the feature considered at an internal node for a specific sample is close to the split threshold, this will result in smaller gradients compared to a sample that is more distant to the split threshold.

Furthermore, we need to adjust the leaf nodes of the decision tree to allow an efficient loss calculation (**C3**). Standard decision trees comprise the predicted class for each leaf node and are functions $DT : \mathbb{R}^n \rightarrow \mathcal{C}$. We use a probability distribution at each leaf node and therefore define decision trees as a function $DT : \mathbb{R}^n \rightarrow \mathbb{R}^c$. Accordingly, the parameters of the leaf nodes are defined as $L \in \mathbb{R}^{2^n} \times \mathbb{R}^c$ for the whole tree and $\boldsymbol{\lambda} \in \mathbb{R}^c$ for a specific leaf node. This adjustment allows the application of standard loss functions, as for instance the cross-entropy, during the optimization.

Figure 3 visualizes our dense tree representation in comparison to standard tree representation for an exemplary decision tree.

3.4 Training Procedure

In the previous subsections, we introduced the adjustments that are necessary to apply a gradient-based optimization for decision trees. We implement this optimization using a gradient descent algorithm. During the gradient descent optimization, we calculate the gradients using backpropagation (see Algorithm 2). We implement backpropagation using automatic differentiation based on the computation graph of the tree pass function, which is used to calculate the function values. The tree pass function is summarized in Algorithm 1 and utilizes the adjustments introduced in the previous sections.

Furthermore, our implementation optimizes the gradient descent algorithm by exploiting common stochastic gradient descent (SGD) techniques, including mini-batch calculation and momentum. Moreover, we implement an early stopping procedure that interrupts the training if the validation loss did not increase over a predefined number of epochs and restores the best parameters. To avoid bad initial parametrizations during the initialization, we additionally implement random restarts. Thereby, the best model is selected based on the validation loss.

Furthermore, we want to note that the sophisticated dense representation which is necessary during the training can be converted into an equivalent standard decision tree representation at each point in time.

Algorithm 1 Tree Pass of a Training Sample

function PASS$(I, T, L, \boldsymbol{x})$
  $c_1^* \leftarrow I_i - \text{hardmax}(I_i) \quad \text{for } i = 0, \ldots, |I|$ ▷ Excluded in the backward pass (Equation 7)
  $I \leftarrow I - c_1^*$ ▷ Excluded in the backward pass (Equation 7)
  $\hat{\boldsymbol{y}} \leftarrow [0]^c$
  for $l = 0, \ldots, 2^d$ **do**
    $p \leftarrow 1$
    for $j = 1, \ldots, d$ **do**
      $\mathfrak{i} \leftarrow 2^{j-1} + \left\lfloor \frac{l}{2^{d-(j-1)}} \right\rfloor - 1$ ▷ Equation 9
      $\mathfrak{p} \leftarrow \left\lfloor \frac{l}{2^{d-j}} \right\rfloor \bmod 2$ ▷ Equation 10
      $s \leftarrow \sigma\left(\sum_{i=0}^{n} T_{\mathfrak{i},i}\, I_{\mathfrak{i},i} - \sum_{i=0}^{n} x_i\, I_{\mathfrak{i},i}\right)$ ▷ Equation 4
      $c_2^* = s - \lfloor s \rceil$ ▷ Excluded in the backward pass (Equation 8)
      $s \leftarrow s - c_2^*$ ▷ Excluded in the backward pass (Equation 8)
      $p \leftarrow p\,((1 - \mathfrak{p})\, s + \mathfrak{p}\,(1 - s))$ ▷ Equation 3
    end for
    $\hat{\boldsymbol{y}} \leftarrow \hat{\boldsymbol{y}} + L_l\, p$ ▷ Equation 1
  end for
  return $\hat{\boldsymbol{y}}$
end function

4 Experimental Evaluation

The goal of our experiments is to evaluate the predictive performance of GDTs against existing approaches to learning decision trees. We focus on univariate, axis-aligned decision trees.

We compare GDTs to the following baseline algorithms:

- **CART**: We use the sklearn [2] implementation, which uses an optimized version of the CART algorithm. While CART usually only uses Gini as impurity measure, we also allow information gain/entropy as an option during our hyperparameter optimization. Since the impurity measure is also the relevant difference between CART and C4.5, we decided to not use both algorithms as a benchmark, but only the optimized CART algorithm which is also the stronger benchmark to compare against.
- **Evolutionary Decision Trees**: We use the GeneticTree[3] implementation for evolutionary decision trees that implements an efficient learning of decision trees using a genetic algorithm.

GDTs are implemented in Python using TensorFlow (Abadi et al., 2015)[4].

The experiments are conducted on several benchmark datasets, mainly from the UCI Machine Learning repository (Dua & Graff, 2017). The dataset specifications and source are listed in Table 5. We use a random $80\%/20\%$ split to train and test data for all datasets. Since GDTs requires a validation set for early stopping, we performed another $80\%/20\%$ split on the training data. However, this is only relevant for GDTs and the other approaches utilize the complete training data.

The hyperparameters were selected based on a random search independently for each dataset. Thereby, the number of trials was equal for all approaches. The final hyperparameters are listed in Tables 6-Table 8.

4.1 Results

GDTs are competitive to baseline DT learners First, we evaluated the performance of GDTs against the baseline approaches on the mentioned benchmark datasets in terms of the F1-Score.

[2]Available under `https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html#sklearn.tree.DecisionTreeClassifier`

[3]Available under: `https://github.com/pysiakk/GeneticTree` (last accessed on 17.08.2022)

[4]The code of our implementation is available to all reviewers in the supplementary material to assure anonymity. We will make it publically available here upon acceptance.

The results are shown in Table 1. In general, GDTs achieved competitive results to state-of-the-art approaches. For binary classification tasks, GDTs achieved a superior performance on $10/22$ datasets, while a greedy optimization with CART achieved the best performance on $10/22$ datasets and GeneticTree on $3/22$ datasets. For multi-class tasks, GDTs were only able to achieve the best performance on $3/14$ datasets, while a greedy optimization achieved a superior performance on the majority of datasets ($11/14$). However, as indicated by the mean scores, we can observe that the performance achieved by GDTs (0.841) was mostly very close to a greedy optimization (0.857) and significantly outperformed GeneticTree (0.751). In addition, we can observe that for several datasets a greedy optimization achieved very poor results and was consistently and significantly outperformed by a non-greedy optimization, as for instance on the *Wisconsin Diagnostic Breast Cancer*, *Heart Disease* and *Lymphography* dataset.

	GDT	Greedy (CART)	GeneticTree
Blood Transfusion	$\mathbf{0.753 \pm 0.029(1)}$	$0.743 \pm 0.042(2)$	$0.684 \pm 0.049(3)$
Banknote Authentication	$\mathbf{0.988 \pm 0.008(1)}$	$0.982 \pm 0.006(2)$	$0.928 \pm 0.024(3)$
Titanic	$\mathbf{0.808 \pm 0.031(1)}$	$0.802 \pm 0.033(2)$	$0.779 \pm 0.033(3)$
Raisins	$\mathbf{0.860 \pm 0.013(1)}$	$0.853 \pm 0.017(3)$	$0.857 \pm 0.021(2)$
Rice	$0.928 \pm 0.006(3)$	$\mathbf{0.928 \pm 0.004(1)}$	$0.928 \pm 0.005(2)$
Echocardiogram	$0.722 \pm 0.109(3)$	$\mathbf{0.730 \pm 0.086(1)}$	$0.723 \pm 0.079(2)$
Wisconsin Diagnostic Breast Cancer	$\mathbf{0.920 \pm 0.029(1)}$	$0.894 \pm 0.025(3)$	$0.901 \pm 0.024(2)$
Loan House	$0.769 \pm 0.030(2)$	$0.767 \pm 0.036(3)$	$\mathbf{0.769 \pm 0.033(1)}$
Heart Failure	$\mathbf{0.782 \pm 0.046(1)}$	$0.778 \pm 0.057(3)$	$0.779 \pm 0.062(2)$
Heart Disease	$0.792 \pm 0.042(2)$	$0.755 \pm 0.064(3)$	$\mathbf{0.803 \pm 0.043(1)}$
Adult	$0.837 \pm 0.005(2)$	$\mathbf{0.848 \pm 0.004(1)}$	$0.794 \pm 0.003(3)$
Bank Marketing	$0.859 \pm 0.011(2)$	$\mathbf{0.868 \pm 0.004(1)}$	$0.827 \pm 0.004(3)$
Cervical Cancer	$\mathbf{0.920 \pm 0.019(1)}$	$0.917 \pm 0.022(3)$	$0.920 \pm 0.019(2)$
Congressional Voting	$0.952 \pm 0.023(3)$	$\mathbf{0.960 \pm 0.017(1)}$	$0.960 \pm 0.017(2)$
Absenteeism	$\mathbf{0.690 \pm 0.040(1)}$	$0.688 \pm 0.031(2)$	$0.662 \pm 0.043(3)$
Hepatitis	$\mathbf{0.783 \pm 0.110(1)}$	$0.782 \pm 0.071(3)$	$0.782 \pm 0.071(2)$
German	$0.684 \pm 0.034(2)$	$\mathbf{0.699 \pm 0.026(1)}$	$0.636 \pm 0.038(3)$
Mushroom	$\mathbf{1.000 \pm 0.000(1)}$	$\mathbf{1.000 \pm 0.000(1)}$	$0.967 \pm 0.013(3)$
Credit Card	$0.797 \pm 0.004(2)$	$\mathbf{0.799 \pm 0.005(1)}$	$0.795 \pm 0.005(3)$
Horse Colic	$0.845 \pm 0.035(2)$	$0.840 \pm 0.034(3)$	$\mathbf{0.868 \pm 0.027(1)}$
Thyroid	$0.932 \pm 0.008(2)$	$\mathbf{0.963 \pm 0.003(1)}$	$0.831 \pm 0.031(3)$
Spambase	$0.894 \pm 0.012(2)$	$\mathbf{0.923 \pm 0.011(1)}$	$0.874 \pm 0.017(3)$
Mean	0.842	0.842	0.821
Iris	$\mathbf{0.944 \pm 0.047(1)}$	$0.930 \pm 0.032(2)$	$0.890 \pm 0.061(3)$
Balance Scale	$0.736 \pm 0.032(2)$	$\mathbf{0.773 \pm 0.022(1)}$	$0.713 \pm 0.049(3)$
Car	$0.851 \pm 0.038(2)$	$\mathbf{0.975 \pm 0.003(1)}$	$0.715 \pm 0.044(3)$
Glass	$0.678 \pm 0.074(2)$	$\mathbf{0.681 \pm 0.076(1)}$	$0.587 \pm 0.102(3)$
Contraceptive	$0.550 \pm 0.019(2)$	$\mathbf{0.559 \pm 0.025(1)}$	$0.503 \pm 0.048(3)$
Solar Flare	$0.793 \pm 0.027(2)$	$\mathbf{0.795 \pm 0.024(1)}$	$0.793 \pm 0.026(3)$
Wine	$\mathbf{0.923 \pm 0.041(1)}$	$0.914 \pm 0.032(2)$	$0.891 \pm 0.055(3)$
Zoo	$0.946 \pm 0.054(2)$	$\mathbf{0.970 \pm 0.050(1)}$	$0.850 \pm 0.100(3)$
Lymphography	$\mathbf{0.769 \pm 0.050(1)}$	$0.724 \pm 0.075(3)$	$0.740 \pm 0.071(2)$
Segment	$0.943 \pm 0.013(2)$	$\mathbf{0.966 \pm 0.005(1)}$	$0.770 \pm 0.049(3)$
Dermatology	$0.932 \pm 0.032(2)$	$\mathbf{0.965 \pm 0.023(1)}$	$0.894 \pm 0.102(3)$
Landsat	$0.848 \pm 0.005(2)$	$\mathbf{0.856 \pm 0.008(1)}$	$0.674 \pm 0.041(3)$
Annealing	$0.970 \pm 0.014(2)$	$\mathbf{0.973 \pm 0.011(1)}$	$0.821 \pm 0.091(3)$
Splice	$0.885 \pm 0.013(2)$	$\mathbf{0.918 \pm 0.006(1)}$	$0.667 \pm 0.066(3)$
Mean	0.841	0.857	0.751

Table 1: **Performance Comparison.** We report F1-scores (mean $\pm$ stdev over 10 trials). We also report the ranking of each approach in brackets. The top part comprises binary classification tasks and the bottom part multi-class datasets. The datasets are sorted by the number of features.

In general, it stands out, that GDTs mostly outperformed the baseline for binary classification datasets with a lower dimensionality ($n < 15$). We can explain this by the dense decision tree representation required for the gradient-based optimization. Using our representation, the difficulty of the optimization task increases with the number of features (more parameters at each internal node) and the number of classes (more parameters at each leaf node). This is not necessarily the case for greedy algorithms, where we always evaluate all potential splits at each node. Therefore, in future work, we aim to optimize the proposed dense representation, e.g., by using parameter sharing techniques.

GDTs are more robust to overfitting than a greedy optimization We can observe that GDTs were more robust and less prone to overfitting than a greedy optimization. We can measure this by the fact that the difference between mean training and test performance (see Table 1 and Table 4) was significantly smaller with GDTs (0.025 for binary and 0.037 for multi-class) opposed to CART (0.042 and 0.085). A global optimization with GeneticTree similarly was less prone to overfitting (0.016 and 0.020).

Probabilities at the GDTs leaf nodes provide calibrated uncertainties Another advantage of GDTs is that they provide probabilities as a direct measure of the model's confidence. We compared the probability-based confidence with the purity measure[5] of the baseline approaches in terms of the area under the receiver operator characteristic curve (ROC AUC). For binary classification tasks, GDTs achieved superior results on $18/22$ datasets, while CART achieves the best score on $4/22$ datasets and GeneticTree on $1/22$ datasets (see Table 3 for details).

GTDs can be learned efficiently for large and high-dimensional datasets Considering the runtime of the different approaches listed in Table 3, it stands out, that a greedy procedure was significantly faster than a global optimization. Nevertheless, for most datasets, learning a GDT took only between 15 and 30 seconds, with a maximum of 93 seconds for the *Adult* dataset.

GDTs do not rely on extensive hyperparameter optimization One advantage of Decision Trees over more sophisticated models, besides their interpretability, is that greedy approaches are not reliant on an extensive hyperparameter optimization to achieve reasonable results. This is also a very desirable property and therefore, we want to show that GDTs similarly are not reliant on an extensive hyperparameter optimization: We used the default parameter settings for each model and fixed the maximum depth to 5 for all datasets to assure a fair comparison[6]. It stands out that the performance gain when optimizing the hyperparameters was rather small for all approaches: The maximum average performance gain was for CART with an increase of only 0.007 (see Table 2 for details). When comparing the different approaches against each other, GDTs outperformed in $6/22$ cases, while CART achieved the best results on $10/22$ datasets and GeneticTree on $6/22$ datasets for binary classification tasks.

5 Conclusion and Future Work

In this paper, we proposed a method for learning univariate, axis-aligned decision trees based on a global optimization with gradient descent. We use backpropagation with an adjusted gradient flow to deal with the non-differentiable nature of decision trees and introduced a dense decision tree representation that allows an efficient optimization. Using a global optimization, GDTs are not prone to locally optimal solutions, as it is the case for standard, greedy decision tree induction methods like CART. We empirically showed that GDTs outperform existing baseline methods on several benchmark datasets. Additionally, GDTs provide calibrated uncertainties in terms of probability distributions at the leaf nodes, which also increases the interpretability of the model.

Furthermore, a gradient-based optimization provides more flexibility. It is straightforward to use a custom loss function that is well-suited to the specific application scenario. Another advantage of a gradient-based optimization is the possibility to relearn the threshold value as well as the split index. This allows the application of GDTs to dynamic environments, as for instance online learning scenarios, without adjustments.

Currently, GDTs are fully-grown. In future work, we want to apply pruning mechanisms to reduce the tree size. While we focused on standard decision trees to generate intrinsically interpretable models within this paper, GDTs can easily be extended to random forests as a performance-interpretability trade-off, which is subject to future work.

[5]Commonly, probabilities are obtained from the purity ratio of each class as a gateway to the model's confidence for vanilla DTs.

[6]For GeneticTree it was not possible to fix the maximum depth, however the trees just rarely result in a depth higher than 5 and therefore the results are not influenced significantly.

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

Gaël Aglin, Siegfried Nijssen, and Pierre Schaus. Learning optimal decision trees using caching branch-and-bound search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3146–3153, 2020.

Rodrigo Coelho Barros, Márcio Porto Basgalupp, Andre CPLF De Carvalho, and Alex A Freitas. A survey of evolutionary algorithms for decision-tree induction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(3):291–312, 2011.

Dimitris Bertsimas and Jack Dunn. Optimal classification trees. *Machine Learning*, 106(7):1039–1082, 2017.

James C Bezdek and Richard J Hathaway. Some notes on alternating optimization. In *AFSS international conference on fuzzy systems*, pp. 288–300. Springer, 2002.

Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, 1984. ISBN 0-534-98053-8. URL `http://lyle.smu.edu/~mhd/8331f06/cart.pdf`.

Miguel A Carreira-Perpinán and Pooya Tavallali. Alternating optimization of decision trees, with application to learning sparse oblique trees. *Advances in neural information processing systems*, 31, 2018.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1): 1–22, 1977.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Alex A Freitas. *Data mining and knowledge discovery with evolutionary algorithms*. Springer Science & Business Media, 2002.

Nicholas Frosst and Geoffrey Hinton. Distilling a neural network into a soft decision tree. *arXiv preprint arXiv:1711.09784*, 2017.

Ozan Irsoy, Olcay Taner Yıldız, and Ethem Alpaydın. Soft decision trees. In *Proceedings of the 21st international conference on pattern recognition (ICPR2012)*, pp. 1819–1822. IEEE, 2012.

Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214, 1994.

Max Kuhn, Kjell Johnson, et al. *Applied predictive modeling*, volume 26. Springer, 2013.

Hyafil Laurent and Ronald L Rivest. Constructing optimal binary decision trees is np-complete. *Information processing letters*, 5(1):15–17, 1976.

Jimmy Lin, Chudi Zhong, Diane Hu, Cynthia Rudin, and Margo Seltzer. Generalized and scalable optimal sparse decision trees. In *International Conference on Machine Learning*, pp. 6150–6160. PMLR, 2020.

Wei-Yin Loh. Regression tress with unbiased variable selection and interaction detection. *Statistica sinica*, pp. 361–386, 2002.

Wei-Yin Loh. Improving the precision of classification trees. *The Annals of Applied Statistics*, pp. 1710–1737, 2009.

Sascha Marton, Stefan Lüdtke, Christian Bartelt, Andrej Tschalzev, and Heiner Stuckenschmidt. Explaining neural networks without access to training data. *arXiv preprint arXiv:2206.04891*, 2022.

James N Morgan and John A Sonquist. Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, 58(302):415–434, 1963.

Kolluru Venkata Sreerama Murthy. *On growing better decision trees from data*. The Johns Hopkins University, 1996.

Mohammad Norouzi, Maxwell Collins, Matthew A Johnson, David J Fleet, and Pushmeet Kohli. Efficient non-greedy optimization of decision trees. *Advances in neural information processing systems*, 28, 2015.

Athanassios Papagelis and Dimitrios Kalles. Breeding decision trees using evolutionary techniques. In *ICML*, volume 1, pp. 393–400, 2001.

J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993. ISBN 1-55860-238-0. URL `http://portal.acm.org/citation.cfm?id=152181`.

UK Sarkar, PP Chakrabarti, Sujoy Ghose, and SC DeSarkar. Improving greedy algorithms by lookahead-search. *Journal of Algorithms*, 16(1):1–23, 1994.

Arman Zharmagambetov, Suryabhan Singh Hada, Magzhan Gabidolla, and Miguel A Carreira-Perpinán. Non-greedy algorithms for decision tree optimization: An experimental comparison. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021.

A Appendix

A.1 Calculations

The index $\mathfrak{i}$ of an internal node preceding a specific leaf node l at a certain depth j can be calculated as follows:

$$\mathfrak{i}(l, d, j) = 2^{j-1} + \left\lfloor \frac{l}{2^{d-(j-1)}} \right\rfloor - 1 \tag{9}$$

Additionally, for a certain leaf node l, $\mathfrak{p}$ defines whether the left branch ($\mathfrak{p} = 0$) or the right branch ($\mathfrak{p} = 1$) was taken at the internal node $\mathfrak{i}$. We can calculate $\mathfrak{p}$ as follows:

$$\mathfrak{p}(l, d, j) = \left\lfloor \frac{l}{2^{d-j}} \right\rfloor \bmod 2 \tag{10}$$

The calculation of the internal node index $\mathfrak{i}$ as well as the specification of the path position $\mathfrak{p}$ also involve non-differentiable operations. However, since we only focus on fully-grown trees, the resulting values are constant and can be calculated independently of the optimization, which does not preclude the application of a gradient-based optimization algorithm.

A.2 Gradient Descent Optimization

We use stochastic gradient descent (SGD) to minimize the loss function of GDTs, which is outlined in Algorithm 2. We use backpropagation to calculate the gradients in Line 11-13. Furthermore, our implementation optimizes Algorithm 2 by exploiting common SGD techniques, including mini-batch calculation and momentum. We also formulate Line 6-9 as a single matrix multiplication for efficiency reasons.

Algorithm 2 Gradient Descent Training for Decision Trees

function TRAINDT$(I, T, L, X, \boldsymbol{y}, n, c, d, \xi)$
  $I \sim \mathcal{U}\left(-\sqrt{\frac{6}{2^{2d-1}+n}}, \sqrt{\frac{6}{2^{2d-1}+n}}\right)$ ▷ Glorot uniform initialization
  $T \sim \mathcal{U}\left(-\sqrt{\frac{6}{2^{2d-1}+n}}, \sqrt{\frac{6}{2^{2d-1}+n}}\right)$ ▷ Glorot uniform initialization
  $L \sim \mathcal{U}\left(-\sqrt{\frac{6}{2^{2d}+c}}, \sqrt{\frac{6}{2^{2d}+c}}\right)$ ▷ Glorot uniform initialization
  for $i = 1, \ldots, \xi$ **do**
    $\hat{\boldsymbol{y}} \leftarrow \emptyset$
    for $j = 1, \ldots, |X|$ **do**
      $\hat{y}_j = \text{PASS}(I, T, L, X_j)$
      $\hat{\boldsymbol{y}} \leftarrow \hat{\boldsymbol{y}} \cup \hat{y}_j$
    end for
    $I \leftarrow I + \eta \frac{\partial}{\partial I} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ ▷ Calculate gradients with backpropagation
    $T \leftarrow T + \eta \frac{\partial}{\partial T} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ ▷ Calculate gradients with backpropagation
    $L \leftarrow L + \eta \frac{\partial}{\partial L} \mathcal{L}(\boldsymbol{y}, \hat{\boldsymbol{y}})$ ▷ Calculate gradients with backpropagation
  end for
end function

A.3 Additional Results

A.4 Datasets

The datasets along with their specifications and source are summarized in Table 5. For all datasets, we performed a standard preprocessing. We applied ordinal encoding to all non-numeric features so that they can be handled by decision trees. We further standardized all datasets to zero mean and unit variance. This step however is only necessary for our approach. While standard DT learning algorithms are independent of the feature scale, GDTs are sensitive to the feature scale due to the gradient updates.

A.5 Hyperparameters

In the following, we report the hyperparameters used for each approach. The hyperparameters were selected based on a random search over a predefined parameter range for each approach. All parameters that were considered are noted in the tables. The number of trials was equal for each approach. For GDTs, we did not optimize the batch size as well as the number of epochs, but used early stopping with a predefined patience. Further details can be found in the code, which we provided in the supplementary material. Furthermore, we used random restarts only for GDTs We made this decision based on the fact that CART is deterministic and therefore does not benefit from additional restarts and for GeneticTree, increasing the population size would achieve similar results. Therefore, both baseline approaches would suffer from the use of restarts, since less data can be used for the training.

	GDT	Greedy (CART)	GeneticTree
Blood Transfusion	$\mathbf{0.771 \pm 0.032(1)}$	$0.733 \pm 0.045(2)$	$0.671 \pm 0.036(3)$
Banknote Authentication	$\mathbf{0.977 \pm 0.005(1)}$	$0.971 \pm 0.010(2)$	$0.929 \pm 0.021(3)$
Titanic	$0.801 \pm 0.031(2)$	$\mathbf{0.801 \pm 0.030(1)}$	$0.777 \pm 0.035(3)$
Raisins	$0.849 \pm 0.016(2)$	$0.846 \pm 0.021(3)$	$\mathbf{0.859 \pm 0.022(1)}$
Rice	$\mathbf{0.928 \pm 0.005(1)}$	$0.924 \pm 0.008(3)$	$0.928 \pm 0.005(2)$
Echocardiogram	$\mathbf{0.732 \pm 0.093(1)}$	$0.717 \pm 0.093(3)$	$0.724 \pm 0.071(2)$
Wisconsin Diagnostic Breast Cancer	$0.900 \pm 0.028(2)$	$\mathbf{0.917 \pm 0.029(1)}$	$0.897 \pm 0.021(3)$
Loan House	$0.773 \pm 0.035(2)$	$0.747 \pm 0.042(3)$	$\mathbf{0.775 \pm 0.031(1)}$
Heart Failure	$0.771 \pm 0.055(2)$	$\mathbf{0.776 \pm 0.053(1)}$	$0.767 \pm 0.049(3)$
Heart Disease	$0.765 \pm 0.040(2)$	$0.726 \pm 0.051(3)$	$\mathbf{0.820 \pm 0.049(1)}$
Adult	$0.828 \pm 0.007(2)$	$\mathbf{0.842 \pm 0.005(1)}$	$0.793 \pm 0.004(3)$
Bank Marketing	$0.861 \pm 0.005(2)$	$\mathbf{0.862 \pm 0.004(1)}$	$0.827 \pm 0.004(3)$
Cervical Cancer	$0.918 \pm 0.018(2)$	$0.916 \pm 0.021(3)$	$\mathbf{0.920 \pm 0.019(1)}$
Congressional Voting	$0.948 \pm 0.020(2)$	$0.948 \pm 0.019(3)$	$\mathbf{0.960 \pm 0.017(1)}$
Absenteeism	$0.683 \pm 0.033(2)$	$\mathbf{0.696 \pm 0.027(1)}$	$0.678 \pm 0.042(3)$
Hepatitis	$\mathbf{0.820 \pm 0.057(1)}$	$0.777 \pm 0.089(2)$	$0.770 \pm 0.097(3)$
German	$0.693 \pm 0.030(2)$	$\mathbf{0.702 \pm 0.027(1)}$	$0.636 \pm 0.031(3)$
Mushroom	$\mathbf{0.996 \pm 0.006(1)}$	$0.979 \pm 0.002(2)$	$0.965 \pm 0.010(3)$
Credit Card	$0.796 \pm 0.005(2)$	$\mathbf{0.798 \pm 0.005(1)}$	$0.795 \pm 0.005(3)$
Horse Colic	$0.846 \pm 0.033(3)$	$0.849 \pm 0.055(2)$	$\mathbf{0.862 \pm 0.021(1)}$
Thyroid	$0.910 \pm 0.007(2)$	$\mathbf{0.937 \pm 0.005(1)}$	$0.840 \pm 0.040(3)$
Spambase	$0.875 \pm 0.021(2)$	$\mathbf{0.910 \pm 0.009(1)}$	$0.870 \pm 0.014(3)$
Mean	0.838	0.835	0.821
Iris	$0.930 \pm 0.038(2)$	$\mathbf{0.937 \pm 0.035(1)}$	$0.914 \pm 0.039(3)$
Balance Scale	$0.718 \pm 0.048(2)$	$\mathbf{0.736 \pm 0.016(1)}$	$0.700 \pm 0.033(3)$
Car	$0.779 \pm 0.018(2)$	$\mathbf{0.848 \pm 0.011(1)}$	$0.729 \pm 0.016(3)$
Glass	$0.520 \pm 0.124(3)$	$\mathbf{0.678 \pm 0.074(1)}$	$0.574 \pm 0.062(2)$
Contraceptive	$0.545 \pm 0.017(2)$	$\mathbf{0.553 \pm 0.022(1)}$	$0.488 \pm 0.056(3)$
Solar Flare	$0.793 \pm 0.026(2)$	$\mathbf{0.794 \pm 0.031(1)}$	$0.793 \pm 0.026(3)$
Wine	$0.905 \pm 0.030(2)$	$\mathbf{0.914 \pm 0.032(1)}$	$0.889 \pm 0.045(3)$
Zoo	$0.882 \pm 0.082(2)$	$\mathbf{0.921 \pm 0.090(1)}$	$0.828 \pm 0.105(3)$
Lymphography	$0.732 \pm 0.082(3)$	$\mathbf{0.746 \pm 0.083(1)}$	$0.732 \pm 0.095(2)$
Segment	$\mathbf{0.840 \pm 0.036(1)}$	$0.795 \pm 0.024(2)$	$0.731 \pm 0.074(3)$
Dermatology	$0.934 \pm 0.040(2)$	$\mathbf{0.949 \pm 0.023(1)}$	$0.865 \pm 0.115(3)$
Landsat	$0.806 \pm 0.014(2)$	$\mathbf{0.811 \pm 0.009(1)}$	$0.707 \pm 0.040(3)$
Annealing	$0.925 \pm 0.039(2)$	$\mathbf{0.973 \pm 0.011(1)}$	$0.803 \pm 0.078(3)$
Splice	$0.819 \pm 0.019(2)$	$\mathbf{0.900 \pm 0.008(1)}$	$0.703 \pm 0.084(3)$
Mean	0.795	0.825	0.747

Table 2: **Performance Comparison Default Hyperparameters.** We report the F1-score (mean $\pm$ stdev over 10 trials) on the test data. We also report the ranking of each approach for the corresponding dataset in brackets. The top part comprises binary classification tasks and the bottom part multi-class datasets. The datasets are sorted based on the number of features.

	F1-Score GDT	F1-Score Greedy (CART)	F1-Score GeneticTree	ROC AUC Score GDT	ROC AUC Score Greedy (CART)	ROC AUC Score GeneticTree	Runtime GDT	Runtime Greedy (CART)	Runtime GeneticTree
Blood Transfusion	$\mathbf{0.753 \pm 0.029(1)}$	$0.743 \pm 0.042(2)$	$0.684 \pm 0.049(3)$	$\mathbf{0.686 \pm 0.047(1)}$	$0.609 \pm 0.063(2)$	$0.522 \pm 0.044(3)$	$18.865 \pm 3.022(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$4.099 \pm 0.204(2)$
Banknote Authentication	$\mathbf{0.988 \pm 0.008(1)}$	$0.982 \pm 0.006(2)$	$0.928 \pm 0.024(3)$	$\mathbf{0.994 \pm 0.007(1)}$	$0.982 \pm 0.007(2)$	$0.928 \pm 0.026(3)$	$31.644 \pm 1.987(3)$	$\mathbf{0.003 \pm 0.000(1)}$	$1.245 \pm 0.388(2)$
Titanic	$\mathbf{0.808 \pm 0.031(1)}$	$0.802 \pm 0.033(2)$	$0.779 \pm 0.033(3)$	$\mathbf{0.839 \pm 0.039(1)}$	$0.786 \pm 0.037(2)$	$0.764 \pm 0.037(3)$	$15.400 \pm 5.987(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$7.635 \pm 2.017(2)$
Raisins	$\mathbf{0.860 \pm 0.013(1)}$	$0.853 \pm 0.017(3)$	$0.857 \pm 0.021(2)$	$\mathbf{0.902 \pm 0.018(1)}$	$0.853 \pm 0.017(3)$	$0.858 \pm 0.022(2)$	$29.521 \pm 2.588(3)$	$\mathbf{0.003 \pm 0.000(1)}$	$0.920 \pm 0.333(2)$
Rice	$\mathbf{0.928 \pm 0.006(1)}$	$\mathbf{0.928 \pm 0.004(1)}$	$\mathbf{0.928 \pm 0.005(1)}$	$\mathbf{0.956 \pm 0.008(1)}$	$0.925 \pm 0.004(3)$	$0.930 \pm 0.005(2)$	$29.096 \pm 7.326(3)$	$\mathbf{0.008 \pm 0.000(1)}$	$1.468 \pm 0.135(2)$
Echocardiogram	$0.722 \pm 0.109(3)$	$\mathbf{0.730 \pm 0.086(1)}$	$0.723 \pm 0.079(2)$	$0.567 \pm 0.072(2)$	$0.560 \pm 0.086(3)$	$\mathbf{0.603 \pm 0.110(1)}$	$37.131 \pm 6.030(3)$	$\mathbf{0.002 \pm 0.001(1)}$	$0.886 \pm 0.290(2)$
Wisconsin Diagnostic Breast Cancer	$\mathbf{0.920 \pm 0.029(1)}$	$0.894 \pm 0.025(3)$	$0.901 \pm 0.024(2)$	$\mathbf{0.939 \pm 0.035(1)}$	$0.891 \pm 0.030(3)$	$0.893 \pm 0.027(2)$	$18.677 \pm 2.042(3)$	$\mathbf{0.003 \pm 0.000(1)}$	$1.602 \pm 0.638(2)$
Loan House	$\mathbf{0.769 \pm 0.030(1)}$	$0.767 \pm 0.036(3)$	$\mathbf{0.769 \pm 0.033(1)}$	$\mathbf{0.695 \pm 0.033(1)}$	$0.688 \pm 0.031(3)$	$0.691 \pm 0.029(2)$	$60.400 \pm 21.409(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$1.710 \pm 0.496(2)$
Heart Failure	$\mathbf{0.782 \pm 0.046(1)}$	$0.778 \pm 0.057(3)$	$0.779 \pm 0.062(2)$	$\mathbf{0.805 \pm 0.048(1)}$	$0.757 \pm 0.068(2)$	$0.749 \pm 0.060(3)$	$14.280 \pm 2.299(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.058 \pm 0.002(2)$
Heart Disease	$0.792 \pm 0.042(2)$	$0.755 \pm 0.064(3)$	$\mathbf{0.803 \pm 0.043(1)}$	$\mathbf{0.857 \pm 0.050(1)}$	$0.760 \pm 0.066(3)$	$0.801 \pm 0.045(2)$	$7.415 \pm 1.755(2)$	$\mathbf{0.002 \pm 0.000(1)}$	$7.803 \pm 1.802(3)$
Adult	$0.837 \pm 0.005(2)$	$\mathbf{0.848 \pm 0.004(1)}$	$0.794 \pm 0.003(3)$	$\mathbf{0.892 \pm 0.004(1)}$	$0.766 \pm 0.008(2)$	$0.679 \pm 0.015(3)$	$93.248 \pm 21.621(3)$	$\mathbf{0.073 \pm 0.001(1)}$	$35.054 \pm 2.311(2)$
Bank Marketing	$0.859 \pm 0.011(2)$	$\mathbf{0.868 \pm 0.004(1)}$	$0.827 \pm 0.004(3)$	$\mathbf{0.684 \pm 0.010(1)}$	$0.604 \pm 0.009(2)$	$0.500 \pm 0.000(3)$	$44.546 \pm 12.009(3)$	$\mathbf{0.077 \pm 0.001(1)}$	$5.856 \pm 0.233(2)$
Cervical Cancer	$\mathbf{0.920 \pm 0.019(1)}$	$0.917 \pm 0.022(3)$	$\mathbf{0.920 \pm 0.019(1)}$	$0.497 \pm 0.067(3)$	$\mathbf{0.522 \pm 0.036(1)}$	$0.500 \pm 0.000(2)$	$6.356 \pm 1.419(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.169 \pm 0.011(2)$
Congressional Voting	$0.952 \pm 0.023(3)$	$\mathbf{0.960 \pm 0.017(1)}$	$\mathbf{0.960 \pm 0.017(1)}$	$\mathbf{0.974 \pm 0.016(1)}$	$0.961 \pm 0.019(2)$	$0.961 \pm 0.019(3)$	$84.285 \pm 22.070(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.118 \pm 0.043(2)$
Absenteeism	$\mathbf{0.690 \pm 0.040(1)}$	$0.688 \pm 0.031(2)$	$0.662 \pm 0.043(3)$	$\mathbf{0.673 \pm 0.057(1)}$	$0.657 \pm 0.039(2)$	$0.625 \pm 0.037(3)$	$28.713 \pm 2.102(3)$	$\mathbf{0.003 \pm 0.001(1)}$	$6.584 \pm 1.259(2)$
Hepatitis	$\mathbf{0.783 \pm 0.110(1)}$	$0.782 \pm 0.071(3)$	$0.782 \pm 0.071(2)$	$0.661 \pm 0.134(2)$	$\mathbf{0.670 \pm 0.122(1)}$	$0.600 \pm 0.065(3)$	$76.987 \pm 51.828(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.366 \pm 0.182(2)$
German	$0.684 \pm 0.034(2)$	$\mathbf{0.699 \pm 0.026(1)}$	$0.636 \pm 0.038(3)$	$\mathbf{0.711 \pm 0.031(1)}$	$0.621 \pm 0.037(2)$	$0.544 \pm 0.034(3)$	$11.402 \pm 1.679(3)$	$\mathbf{0.003 \pm 0.000(1)}$	$6.735 \pm 1.377(2)$
Mushroom	$\mathbf{1.000 \pm 0.000(1)}$	$\mathbf{1.000 \pm 0.000(1)}$	$0.967 \pm 0.013(3)$	$\mathbf{1.000 \pm 0.000(1)}$	$\mathbf{1.000 \pm 0.000(1)}$	$0.967 \pm 0.012(3)$	$23.520 \pm 6.543(2)$	$\mathbf{0.008 \pm 0.000(1)}$	$31.528 \pm 3.875(3)$
Credit Card	$0.797 \pm 0.004(2)$	$\mathbf{0.799 \pm 0.005(1)}$	$0.795 \pm 0.005(3)$	$\mathbf{0.752 \pm 0.008(1)}$	$0.653 \pm 0.005(2)$	$0.646 \pm 0.011(3)$	$31.306 \pm 8.460(3)$	$\mathbf{0.154 \pm 0.001(1)}$	$2.049 \pm 0.031(2)$
Horse Colic	$0.845 \pm 0.035(2)$	$0.840 \pm 0.034(3)$	$\mathbf{0.868 \pm 0.027(1)}$	$\mathbf{0.876 \pm 0.043(1)}$	$0.816 \pm 0.037(3)$	$0.849 \pm 0.026(2)$	$32.767 \pm 14.357(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$6.675 \pm 2.057(2)$
Thyroid	$0.932 \pm 0.008(2)$	$\mathbf{0.963 \pm 0.003(1)}$	$0.831 \pm 0.031(3)$	$\mathbf{0.961 \pm 0.011(1)}$	$0.955 \pm 0.002(2)$	$0.735 \pm 0.051(3)$	$64.529 \pm 8.808(2)$	$\mathbf{0.025 \pm 0.001(1)}$	$65.636 \pm 23.280(3)$
Spambase	$0.894 \pm 0.012(2)$	$\mathbf{0.923 \pm 0.011(1)}$	$0.874 \pm 0.017(3)$	$0.899 \pm 0.011(2)$	$\mathbf{0.916 \pm 0.010(1)}$	$0.864 \pm 0.016(3)$	$9.259 \pm 2.478(2)$	$\mathbf{0.035 \pm 0.002(1)}$	$9.736 \pm 2.623(3)$
Mean	0.842	0.842	0.821	0.810	0.771	0.737	34.970	0.019	8.997
Iris	$\mathbf{0.944 \pm 0.047(1)}$	$0.930 \pm 0.032(2)$	$0.890 \pm 0.061(3)$	$\mathbf{0.958 \pm 0.035(1)}$	$0.948 \pm 0.025(2)$	$0.919 \pm 0.042(3)$	$8.690 \pm 1.852(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$1.058 \pm 0.549(2)$
Balance Scale	$0.736 \pm 0.032(2)$	$\mathbf{0.773 \pm 0.022(1)}$	$0.713 \pm 0.049(3)$	$0.714 \pm 0.023(2)$	$\mathbf{0.727 \pm 0.017(1)}$	$0.699 \pm 0.032(3)$	$28.378 \pm 2.797(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$6.936 \pm 2.387(2)$
Car	$0.851 \pm 0.038(2)$	$\mathbf{0.975 \pm 0.003(1)}$	$0.715 \pm 0.044(3)$	$0.790 \pm 0.070(2)$	$\mathbf{0.963 \pm 0.010(1)}$	$0.598 \pm 0.034(3)$	$27.025 \pm 1.527(3)$	$\mathbf{0.002 \pm 0.001(1)}$	$6.495 \pm 2.365(2)$
Glass	$0.678 \pm 0.074(2)$	$\mathbf{0.681 \pm 0.076(1)}$	$0.587 \pm 0.102(3)$	$0.688 \pm 0.132(2)$	$\mathbf{0.697 \pm 0.142(1)}$	$0.606 \pm 0.078(3)$	$10.726 \pm 1.573(3)$	$\mathbf{0.002 \pm 0.001(1)}$	$1.885 \pm 0.632(2)$
Contraceptive	$0.550 \pm 0.019(2)$	$\mathbf{0.559 \pm 0.025(1)}$	$0.503 \pm 0.048(3)$	$0.650 \pm 0.014(2)$	$\mathbf{0.655 \pm 0.021(1)}$	$0.622 \pm 0.028(3)$	$31.651 \pm 9.000(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$4.648 \pm 1.387(2)$
Solar Flare	$0.793 \pm 0.027(2)$	$\mathbf{0.795 \pm 0.024(1)}$	$0.793 \pm 0.026(3)$	$0.500 \pm 0.000(2)$	$\mathbf{0.501 \pm 0.003(1)}$	$0.500 \pm 0.000(3)$	$8.562 \pm 2.597(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.236 \pm 0.006(2)$
Wine	$\mathbf{0.923 \pm 0.041(1)}$	$0.914 \pm 0.032(2)$	$0.891 \pm 0.055(3)$	$\mathbf{0.943 \pm 0.028(1)}$	$0.934 \pm 0.023(2)$	$0.927 \pm 0.036(3)$	$25.826 \pm 1.352(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.325 \pm 0.004(2)$
Zoo	$0.946 \pm 0.054(2)$	$\mathbf{0.970 \pm 0.050(1)}$	$0.850 \pm 0.100(3)$	$0.586 \pm 0.172(2)$	$\mathbf{0.600 \pm 0.200(1)}$	$0.569 \pm 0.143(3)$	$100.411 \pm 64.047(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$4.269 \pm 0.856(2)$
Lymphography	$\mathbf{0.769 \pm 0.050(1)}$	$0.724 \pm 0.075(3)$	$0.740 \pm 0.071(2)$	$\mathbf{0.514 \pm 0.043(1)}$	$0.508 \pm 0.023(3)$	$0.511 \pm 0.033(2)$	$21.139 \pm 7.552(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$0.628 \pm 0.444(2)$
Segment	$0.943 \pm 0.013(2)$	$\mathbf{0.966 \pm 0.005(1)}$	$0.770 \pm 0.049(3)$	$0.966 \pm 0.008(2)$	$\mathbf{0.980 \pm 0.003(1)}$	$0.874 \pm 0.024(3)$	$32.325 \pm 2.352(3)$	$\mathbf{0.014 \pm 0.001(1)}$	$11.776 \pm 2.568(2)$
Dermatology	$0.932 \pm 0.032(2)$	$\mathbf{0.965 \pm 0.023(1)}$	$0.894 \pm 0.102(3)$	$0.955 \pm 0.027(2)$	$\mathbf{0.964 \pm 0.026(1)}$	$0.928 \pm 0.054(3)$	$12.458 \pm 4.322(3)$	$\mathbf{0.002 \pm 0.001(1)}$	$6.980 \pm 1.868(2)$
Landsat	$0.848 \pm 0.005(2)$	$\mathbf{0.856 \pm 0.008(1)}$	$0.674 \pm 0.041(3)$	$0.894 \pm 0.005(2)$	$\mathbf{0.900 \pm 0.006(1)}$	$0.788 \pm 0.022(3)$	$62.950 \pm 12.204(3)$	$\mathbf{0.046 \pm 0.001(1)}$	$2.241 \pm 0.176(2)$
Annealing	$0.970 \pm 0.014(2)$	$\mathbf{0.973 \pm 0.011(1)}$	$0.821 \pm 0.091(3)$	$0.823 \pm 0.165(2)$	$\mathbf{0.831 \pm 0.169(1)}$	$0.640 \pm 0.101(3)$	$9.437 \pm 1.858(3)$	$\mathbf{0.002 \pm 0.000(1)}$	$2.997 \pm 0.888(2)$
Splice	$0.885 \pm 0.013(2)$	$\mathbf{0.918 \pm 0.006(1)}$	$0.667 \pm 0.066(3)$	$0.915 \pm 0.010(2)$	$\mathbf{0.936 \pm 0.006(1)}$	$0.759 \pm 0.048(3)$	$17.678 \pm 3.045(3)$	$\mathbf{0.017 \pm 0.000(1)}$	$5.608 \pm 2.111(2)$
Mean	0.841	0.857	0.751	0.778	0.796	0.710	28.375	0.007	4.006

Table 3: **Test Performance and Runtime Comparison.** We report F1-score and ROC AUC score (mean $\pm$ stdev over 10 trials) on the test data, along with the runtime without restarts. We also report the ranking of each approach for the corresponding dataset in brackets. The top part comprises binary classification tasks and the bottom part multi-class datasets. The datasets are sorted based on the number of features.

	GDT	Greedy (CART)	GeneticTree
Blood Transfusion	$0.767 \pm 0.038(2)$	$\mathbf{0.798 \pm 0.015(1)}$	$0.682 \pm 0.048(3)$
Banknote Authentication	$\mathbf{0.998 \pm 0.002(1)}$	$0.998 \pm 0.001(2)$	$0.937 \pm 0.017(3)$
Titanic	$\mathbf{0.823 \pm 0.014(1)}$	$0.821 \pm 0.008(2)$	$0.790 \pm 0.015(3)$
Raisins	$\mathbf{0.880 \pm 0.008(1)}$	$0.866 \pm 0.004(2)$	$0.863 \pm 0.004(3)$
Rice	$0.927 \pm 0.003(2)$	$\mathbf{0.929 \pm 0.002(1)}$	$0.922 \pm 0.002(3)$
Echocardiogram	$0.822 \pm 0.024(3)$	$\mathbf{0.829 \pm 0.050(1)}$	$0.827 \pm 0.028(2)$
Wisconsin Diagnostic Breast Cancer	$\mathbf{0.958 \pm 0.008(1)}$	$0.920 \pm 0.006(2)$	$0.916 \pm 0.011(3)$
Loan House	$0.785 \pm 0.025(2)$	$\mathbf{0.800 \pm 0.009(1)}$	$0.779 \pm 0.014(3)$
Heart Failure	$0.838 \pm 0.028(2)$	$\mathbf{0.878 \pm 0.017(1)}$	$0.827 \pm 0.019(3)$
Heart Disease	$0.830 \pm 0.031(3)$	$\mathbf{0.891 \pm 0.007(1)}$	$0.845 \pm 0.011(2)$
Adult	$0.838 \pm 0.004(2)$	$\mathbf{0.861 \pm 0.002(1)}$	$0.793 \pm 0.004(3)$
Bank Marketing	$0.861 \pm 0.010(2)$	$\mathbf{0.882 \pm 0.001(1)}$	$0.828 \pm 0.001(3)$
Cervical Cancer	$0.902 \pm 0.011(2)$	$\mathbf{0.922 \pm 0.010(1)}$	$0.901 \pm 0.005(3)$
Congressional Voting	$\mathbf{0.964 \pm 0.012(1)}$	$0.956 \pm 0.004(2)$	$0.956 \pm 0.004(3)$
Absenteeism	$\mathbf{0.830 \pm 0.021(1)}$	$0.806 \pm 0.015(2)$	$0.677 \pm 0.023(3)$
Hepatitis	$0.813 \pm 0.050(3)$	$\mathbf{0.862 \pm 0.018(1)}$	$0.838 \pm 0.022(2)$
German	$0.754 \pm 0.026(2)$	$\mathbf{0.777 \pm 0.018(1)}$	$0.686 \pm 0.022(3)$
Mushroom	$\mathbf{1.000 \pm 0.000(1)}$	$1.000 \pm 0.000(2)$	$0.968 \pm 0.012(3)$
Credit Card	$0.795 \pm 0.004(2)$	$\mathbf{0.801 \pm 0.002(1)}$	$0.794 \pm 0.002(3)$
Horse Colic	$0.871 \pm 0.019(2)$	$\mathbf{0.897 \pm 0.010(1)}$	$0.867 \pm 0.015(3)$
Thyroid	$0.942 \pm 0.005(2)$	$\mathbf{0.987 \pm 0.001(1)}$	$0.833 \pm 0.034(3)$
Spambase	$0.883 \pm 0.009(2)$	$\mathbf{0.966 \pm 0.002(1)}$	$0.874 \pm 0.017(3)$
Mean	$0.867(2)$	$\mathbf{0.884(1)}$	$0.837(3)$
Iris	$\mathbf{0.964 \pm 0.016(1)}$	$0.964 \pm 0.009(2)$	$0.939 \pm 0.013(3)$
Balance Scale	$0.815 \pm 0.024(2)$	$\mathbf{1.000 \pm 0.000(1)}$	$0.778 \pm 0.017(3)$
Car	$0.868 \pm 0.037(2)$	$\mathbf{0.995 \pm 0.001(1)}$	$0.720 \pm 0.049(3)$
Glass	$0.773 \pm 0.037(2)$	$\mathbf{0.995 \pm 0.006(1)}$	$0.593 \pm 0.038(3)$
Contraceptive	$0.551 \pm 0.012(2)$	$\mathbf{0.578 \pm 0.008(1)}$	$0.509 \pm 0.032(3)$
Solar Flare	$0.770 \pm 0.012(2)$	$\mathbf{0.825 \pm 0.007(1)}$	$0.766 \pm 0.006(3)$
Wine	$0.990 \pm 0.011(2)$	$\mathbf{1.000 \pm 0.000(1)}$	$0.947 \pm 0.015(3)$
Zoo	$0.978 \pm 0.011(2)$	$\mathbf{1.000 \pm 0.000(1)}$	$0.880 \pm 0.064(3)$
Lymphography	$0.905 \pm 0.038(2)$	$\mathbf{0.993 \pm 0.012(1)}$	$0.816 \pm 0.044(3)$
Segment	$0.958 \pm 0.007(2)$	$\mathbf{0.997 \pm 0.002(1)}$	$0.763 \pm 0.046(3)$
Dermatology	$0.974 \pm 0.019(2)$	$\mathbf{0.988 \pm 0.003(1)}$	$0.900 \pm 0.064(3)$
Landsat	$0.882 \pm 0.007(2)$	$\mathbf{0.909 \pm 0.002(1)}$	$0.676 \pm 0.044(3)$
Annealing	$0.981 \pm 0.005(2)$	$\mathbf{0.987 \pm 0.004(1)}$	$0.842 \pm 0.062(3)$
Splice	$0.888 \pm 0.016(2)$	$\mathbf{0.957 \pm 0.003(1)}$	$0.670 \pm 0.068(3)$
Mean	$0.878(2)$	$\mathbf{0.942(1)}$	$0.771(3)$

Table 4: **Train Performance Comparison.** We report F1-score (mean $\pm$ stdev over 10 trials) on the training data. We also report the ranking of each approach for the corresponding dataset in brackets. The top part comprises binary classification tasks and the bottom part multi-class datasets. The datasets are sorted based on the number of features.

Dataset Name	Number of Features	Number of Samples	Samples by Class	Number of Classes	Link
Blood Transfusion	4	748	178 / 570	2	https://archive.ics.uci.edu/ml/datasets/Blood+Transfusion+Service+Center
Banknote Authentication	4	1372	610 / 762	2	https://archive.ics.uci.edu/ml/datasets/banknote+authentication
Titanic	7	891	342 / 549	2	https://www.kaggle.com/c/titanic
Raisin	7	900	450 /450	2	https://archive.ics.uci.edu/ml/datasets/Raisin+Dataset
Rice	7	3810	1630 / 2180	2	https://archive.ics.uci.edu/ml/datasets/Rice+%28Cammeo+and+Osmancik%29
Echocardiogram	8	132	107 / 25	2	https://archive.ics.uci.edu/ml/datasets/echocardiogram
Wisconsin Diagnostic Breast Cancer	10	569	212 / 357	2	https://archive.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+(diagnostic)
Loan House	11	614	422 / 192	2	https://www.kaggle.com/code/sazid28/home-loan-prediction/data
Heart Failure	12	299	96 /203	2	https://archive.ics.uci.edu/ml/datasets/Heart+failure+clinical+records
Heart Disease	13	303	164 / 139	2	https://archive.ics.uci.edu/ml/datasets/heart+disease
Adult	14	32561	7841 / 24720	2	https://archive.ics.uci.edu/ml/datasets/adult
Bank Marketing	14	45211	5289 / 39922	2	https://archive.ics.uci.edu/ml/datasets/bank+marketing
Cervical Cancer	15	858	55 / 803	2	https://archive.ics.uci.edu/ml/datasets/Cervical+cancer+%28Risk+Factors%29
Congressional Voting	16	435	267 / 168	2	https://archive.ics.uci.edu/ml/datasets/congressional+voting+records
Absenteeism	18	740	279 / 461	2	https://archive.ics.uci.edu/ml/datasets/Absenteeism+at+work
Hepatitis	19	155	32 / 123	2	https://archive.ics.uci.edu/ml/datasets/hepatitis
German	20	1000	300 / 700	2	https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)
Mushrooms	22	8124	4208 /3916	2	https://archive.ics.uci.edu/ml/datasets/mushroom
Credit Card	23	30000	23364 / 6636	2	https://archive.ics.uci.edu/ml/datasets/default+of+credit+card+clients
Horse Colic	27	368	232 / 136	2	https://archive.ics.uci.edu/ml/datasets/Horse+Colic
Thyroid	29	9172	2401 / 6771	2	https://archive.ics.uci.edu/ml/datasets/thyroid+disease
Spambase	57	4601	1813 / 2788	2	https://archive.ics.uci.edu/ml/datasets/spambase
Iris	4	150	50 / 50 / 50	3	https://archive.ics.uci.edu/ml/datasets/iris
Balance Scale	4	625	49 / 288 / 288	3	https://archive.ics.uci.edu/ml/datasets/balance+scale
Car	6	1728	384 / 69 / 1210 / 65	4	https://archive.ics.uci.edu/ml/datasets/car+evaluation
Glass	9	214	70 / 76 / 17 / 13 / 9 / 29	6	https://archive.ics.uci.edu/ml/datasets/glass+identification
Contraceptive	9	1473	629 / 333 / 511	3	https://archive.ics.uci.edu/ml/datasets/Contraceptive+Method+Choice
Solar Flare	10	1389	1171 / 141 / 40 / 20 / 9 / 4 / 3 / 1	8	http://archive.ics.uci.edu/ml/datasets/solar+flare
Wine	12	178	59 / 71 / 48	3	https://archive.ics.uci.edu/ml/datasets/wine
Zoo	16	101	41 / 20 / 5 / 13 / 4 / 8 /10	7	https://archive.ics.uci.edu/ml/datasets/zoo
Lymphography	18	148	2 / 81 /61 /4	4	https://archive.ics.uci.edu/ml/datasets/Lymphography
Segment	19	2310	330 / 330 / 330 / 330 / 330 / 330 / 330	7	https://archive.ics.uci.edu/ml/datasets/image+segmentation
Dermatology	34	366	112 / 61 / 72 / 49 / 52 / 20	6	https://archive.ics.uci.edu/ml/datasets/dermatology
Landsat	36	6435	1533 / 703 / 1358 / 626 / 707 /1508	6	https://archive.ics.uci.edu/ml/datasets/Statlog+(Landsat+Satellite)
Annealing	38	798	8 / 88 / 608 / 60 / 34	5	https://archive.ics.uci.edu/ml/datasets/Annealing
Splice	60	3190	767 / 768 / 1655	3	https://archive.ics.uci.edu/ml/datasets/Molecular+Biology+(Splice-junction+Gene+Sequences)

Table 5: **Dataset Specifications.**

Dataset Name	depth	lr_index	lr_values	lr_leaf	optimizer	batch_size	epochs	restarts	early_stopping_epochs
Blood Transfusion	7	0.1000	0.1000	0.0050	adam	512	10000	10	200
Banknote Authentication	11	0.0100	0.0100	0.0500	adam	512	10000	10	200
Titanic	7	0.0100	0.0500	0.0050	adam	512	10000	10	200
Raisins	11	0.0500	0.0010	0.0100	adam	512	10000	10	200
Rice	9	0.0100	0.0005	0.0005	adam	512	10000	10	200
Echocardiogram	11	0.0050	0.1000	0.0010	adam	512	10000	10	200
Wisconsin Diagnostic Breast Cancer	9	0.0001	0.0005	0.0050	adam	512	10000	10	200
Loan House	9	0.0500	0.0500	0.0005	adam	512	10000	10	200
Heart Failure	9	0.0050	0.1000	0.0050	adam	512	10000	10	200
Heart Disease	5	0.0500	0.0005	0.0100	adam	512	10000	10	200
Adult	11	0.0500	0.0100	0.0050	adam	512	10000	10	200
Bank Marketing	3	0.1000	0.0001	0.0010	adam	512	10000	10	200
Cervical Cancer	3	0.0500	0.0001	0.0500	adam	512	10000	10	200
Congressional Voting	11	0.1000	0.0010	0.0005	adam	512	10000	10	200
Absenteeism	11	0.0500	0.0010	0.0100	adam	512	10000	10	200
Hepatitis	3	0.0001	0.1000	0.0001	adam	512	10000	10	200
German	7	0.0010	0.0100	0.0050	adam	512	10000	10	200
Mushroom	9	0.0050	0.0100	0.0500	adam	512	10000	10	200
Credit Card	7	0.0010	0.0100	0.0010	adam	512	10000	10	200
Horse Colic	5	0.0500	0.0005	0.0005	adam	512	10000	10	200
Thyroid	11	0.0050	0.0100	0.0050	adam	512	10000	10	200
Spambase	3	0.0100	0.0100	0.0100	adam	512	10000	10	200
Iris	7	0.0500	0.1000	0.0500	adam	512	10000	10	200
Balance Scale	11	0.0005	0.0050	0.0050	adam	512	10000	10	200
Car	11	0.0100	0.0100	0.0500	adam	512	10000	10	200
Glass	9	0.1000	0.0010	0.1000	adam	512	10000	10	200
Contraceptive	9	0.1000	0.0050	0.0010	adam	512	10000	10	200
Solar Flare	7	0.0001	0.0050	0.1000	adam	512	10000	10	200
Wine	11	0.0500	0.1000	0.0500	adam	512	10000	10	200
Zoo	7	0.0100	0.0005	0.0010	adam	512	10000	10	200
Lymphography	9	0.0100	0.0005	0.0010	adam	512	10000	10	200
Segment	11	0.1000	0.0500	0.0500	adam	512	10000	10	200
Dermatology	9	0.0100	0.0100	0.0500	adam	512	10000	10	200
Landsat	11	0.0100	0.0100	0.0500	adam	512	10000	10	200
Annealing	7	0.0500	0.1000	0.0500	adam	512	10000	10	200
Splice	9	0.0500	0.0010	0.1000	adam	512	10000	10	200

Table 6: **GDT Hyperparameters**

Dataset Name	max_depth	criterion	max_features	min_impurity_decrease	min_impurity_split	min_samples_leaf	min_samples_split	ccp_alpha
Blood Transfusion	5	entropy	None	0.0	None	1	10	0.0
Banknote Authentication	7	gini	None	0.0	None	1	10	0.0
Titanic	3	entropy	None	0.0	None	5	5	0.0
Raisins	3	entropy	None	0.0	None	50	2	0.0
Rice	3	gini	None	0.0	None	5	2	0.0
Echocardiogram	3	gini	None	0.0	None	1	50	0.0
Wisconsin Diagnostic Breast Cancer	5	gini	None	0.0	None	50	50	0.0
Loan House	3	gini	None	0.0	None	1	50	0.0
Heart Failure	3	gini	None	0.0	None	5	2	0.0
Heart Disease	7	gini	None	0.0	None	5	2	0.0
Adult	11	gini	None	0.0	None	10	50	0.0
Bank Marketing	11	entropy	None	0.0	None	5	50	0.0
Cervical Cancer	5	entropy	None	0.0	None	5	2	0.0
Congressional Voting	3	gini	None	0.0	None	50	2	0.0
Absenteeism	7	gini	None	0.0	None	1	10	0.0
Hepatitis	3	gini	None	0.0	None	1	50	0.0
German	5	entropy	None	0.0	None	5	2	0.0
Mushroom	9	gini	None	0.0	None	1	2	0.0
Credit Card	5	entropy	None	0.0	None	10	50	0.0
Horse Colic	5	entropy	None	0.0	None	10	2	0.0
Thyroid	11	gini	None	0.0	None	1	10	0.0
Spambase	11	entropy	None	0.0	None	1	10	0.0
Iris	3	gini	None	0.0	None	1	50	0.0
Balance Scale	11	gini	None	0.0	None	1	2	0.0
Car	11	entropy	None	0.0	None	1	2	0.0
Glass	11	gini	None	0.0	None	1	2	0.0
Contraceptive	5	entropy	None	0.0	None	10	50	0.0
Solar Flare	9	gini	None	0.0	None	1	5	0.0
Wine	5	gini	None	0.0	None	1	5	0.0
Zoo	7	gini	None	0.0	None	1	2	0.0
Lymphography	9	entropy	None	0.0	None	1	2	0.0
Segment	11	entropy	None	0.0	None	1	2	0.0
Dermatology	7	entropy	None	0.0	None	1	5	0.0
Landsat	9	entropy	None	0.0	None	10	2	0.0
Annealing	5	gini	None	0.0	None	1	2	0.0
Splice	9	gini	None	0.0	None	5	2	0.0

Table 7: **CART Hyperparameters**

Dataset Name	n_thresholds	n_trees	max_iter	cross_prob	mutation_prob
Blood Transfusion	10	250	100	1.0	0.5
Banknote Authentication	10	50	500	0.4	0.9
Titanic	10	500	250	0.6	0.8
Raisins	10	100	50	1.0	0.6
Rice	10	100	250	0.4	0.3
Echocardiogram	10	500	50	0.0	0.7
Wisconsin Diagnostic Breast Cancer	10	250	500	0.4	0.3
Loan House	10	100	500	1.0	0.5
Heart Failure	10	100	10	0.4	0.7
Heart Disease	10	500	250	1.0	0.0
Adult	10	250	100	0.2	0.7
Bank Marketing	10	50	50	0.4	0.1
Cervical Cancer	10	50	50	0.4	0.1
Congressional Voting	10	50	50	0.4	0.1
Absenteeism	10	500	250	0.2	0.5
Hepatitis	10	100	100	0.2	0.8
German	10	500	250	0.2	0.9
Mushroom	10	500	250	0.8	0.6
Credit Card	10	50	10	0.6	0.8
Horse Colic	10	500	250	0.6	0.8
Thyroid	10	500	250	1.0	0.5
Spambase	10	100	500	0.8	0.9
Iris	10	100	250	1.0	0.1
Balance Scale	10	250	500	0.8	0.4
Car	10	250	500	0.6	0.4
Glass	10	250	500	0.2	0.7
Contraceptive	10	250	250	0.4	0.2
Solar Flare	10	50	50	0.4	0.1
Wine	10	100	100	0.6	0.3
Zoo	10	500	250	0.2	0.8
Lymphography	10	50	500	0.4	0.9
Segment	10	500	500	0.2	0.7
Dermatology	10	500	500	0.2	0.8
Landsat	10	50	250	0.2	0.8
Annealing	10	250	500	0.2	0.7
Splice	10	100	500	0.6	0.8

Table 8: **GeneticTree Hyperparameters**