We can learn state-of-the-art axis-aligned **Decision Trees** with **Gradient Descent**!

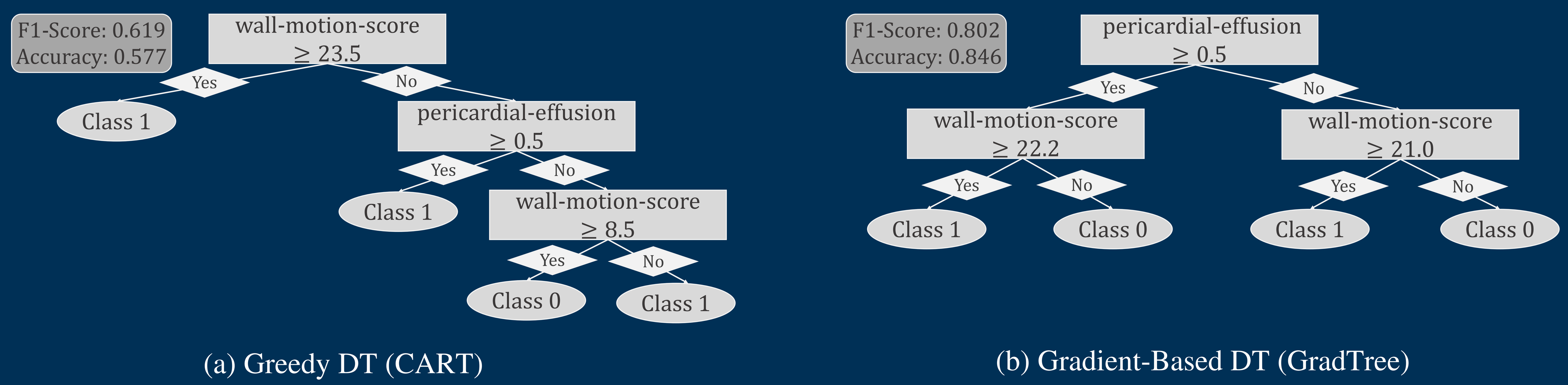


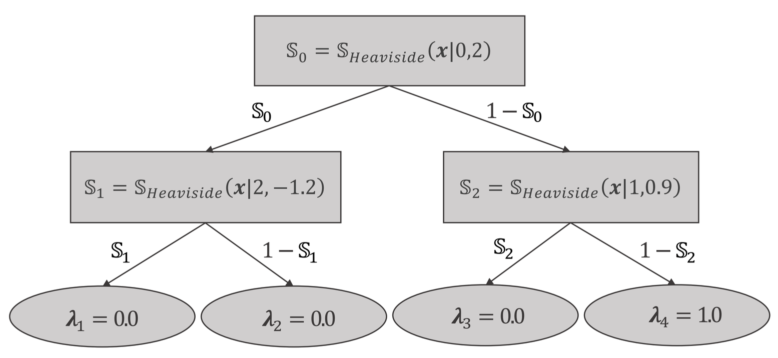
Figure 1: **Greedy vs. Gradient-Based DT.** Two DTs trained on the Echocardiogram dataset. The CART DT (left) makes only locally optimal splits, while GradTree (right) jointly optimizes all parameters, leading to significantly better performance.

The 38th Annual AAAI Conference on Artificial Intelligence

UNIVERSITY OF MANNHEIM

GradTree: Learning Axis-Aligned Decision Trees with Gradient Descent

Arithmetic Decision Tree Formulation



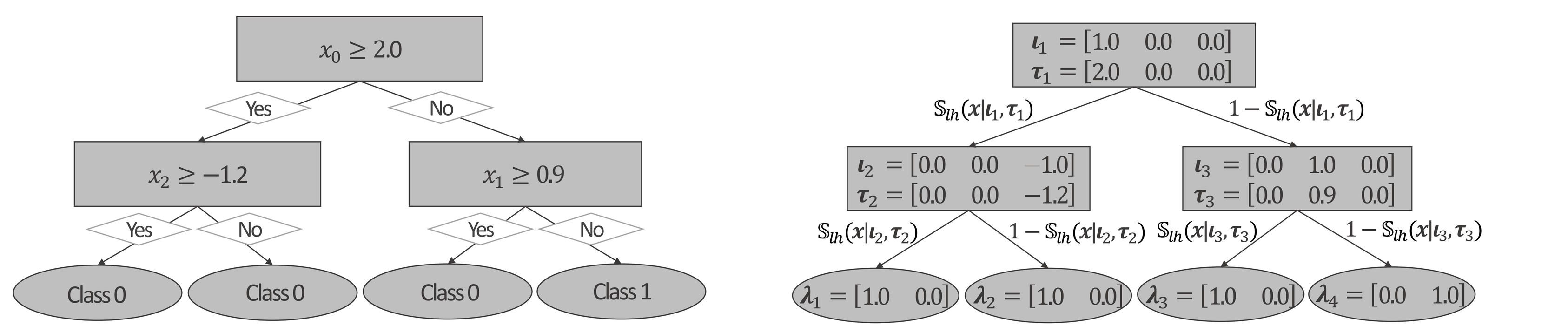
1. $\mathbb{S}_0 * \mathbb{S}_1 * \lambda_1$
2. $\mathbb{S}_0 * (1-\mathbb{S}_1) * \lambda_2$
3. $(1-\mathbb{S}_0) * \mathbb{S}_2 * \lambda_3$
4. $(1-\mathbb{S}_0) * (1-\mathbb{S}_2) * \lambda_4$

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

Dense DT Representation

We propose a dense representation relaxing split indices and thresholds

→ Allow reasonable optimization of parameters with gradient descent



(a) Vanilla DT Representation (b) Dense DT Representation

Straight-Through Operator for non-differentiable operations

(1) Hardmax to enforce one-hot encoded split index vectors → **univariate DTs**

(2) Discretization of the split function (round the sigmoid output) → **hard splits**

Algorithm 1: Tree Pass Function

function PASS($I, T, L, \boldsymbol{x}$)
  $I \leftarrow \text{entmax}(I)$
  $I \leftarrow I - c$ where $c = I - \text{hardmax}(I)$ ▷ ST operator
  $\hat{\boldsymbol{y}} \leftarrow [0]^c$
  for $l = 0, \dots, 2^d - 1$ **do**
    $p \leftarrow 1$
    for $j = 1, \dots, d$ **do**
      $\mathfrak{i} \leftarrow 2^{j-1} + \left\lfloor \frac{l}{2^{d-(j-1)}} \right\rfloor - 1$
      $\mathfrak{p} \leftarrow \left\lfloor \frac{l}{2^{d-j}} \right\rfloor \bmod 2$
      $s \leftarrow S\left(\sum_{i=0}^{n} T_{\mathfrak{i},i}\, I_{\mathfrak{i},i} - \sum_{i=0}^{n} x_i\, I_{\mathfrak{i},i}\right)$
      $s \leftarrow s - c$ where $c = s - \lfloor s \rceil$ ▷ ST operator
      $p \leftarrow p\,((1-\mathfrak{p})\, s + \mathfrak{p}\,(1-s))$
    end for
    $\hat{\boldsymbol{y}} \leftarrow \hat{\boldsymbol{y}} + L_l\, p$
  end for
  return $\sigma(\hat{\boldsymbol{y}})$ ▷ Softmax σ to get probability distribution
end function

GradTree in Action

```python
from GradTree import GradTree

params = {
        'depth': 5,

        'learning_rate_index': 0.01,
        'learning_rate_values': 0.01,
        'learning_rate_leaf': 0.005,

        'loss': 'crossentropy',
}

args = {
    'cat_idx': categorical_feature_indices,
    'objective': 'binary',
}

model_gradtree = GradTree(params=params, args=args)

model_gradtree.fit(X_train=X_train,
              y_train=y_train,
              X_val=X_valid,
              y_val=y_valid)

model_gradtree = model_gradtree.predict(X_test)
```

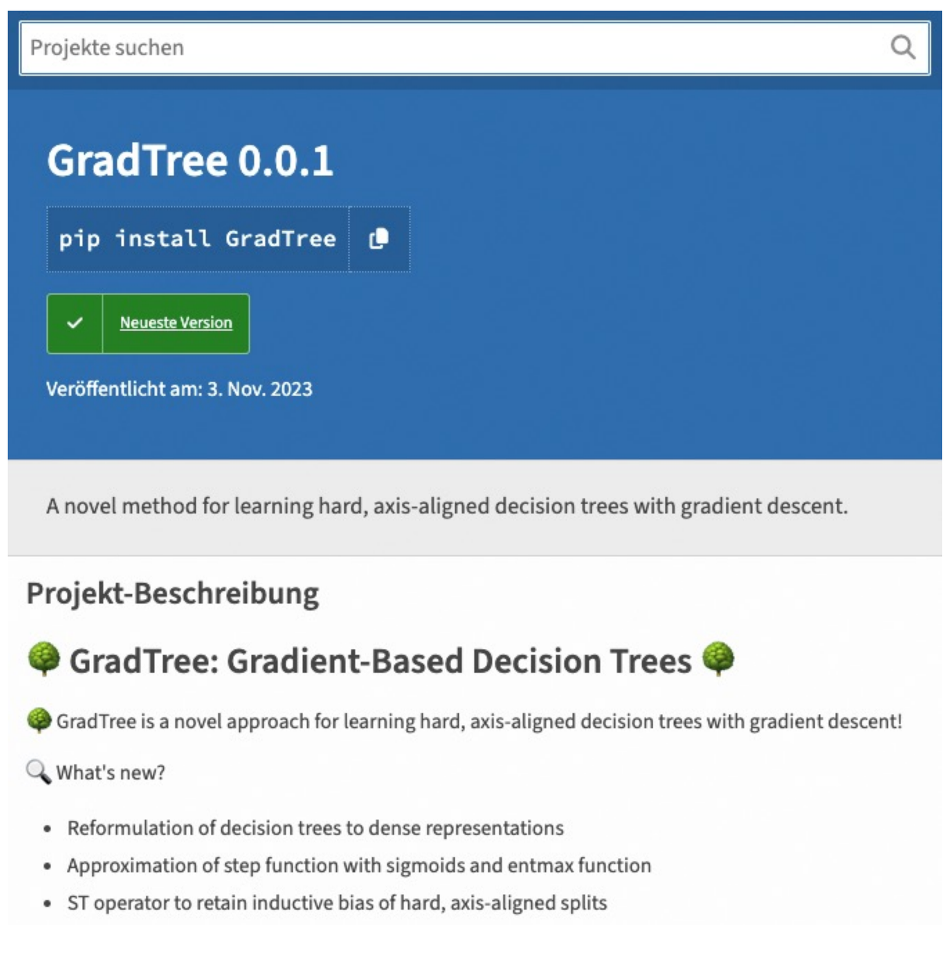
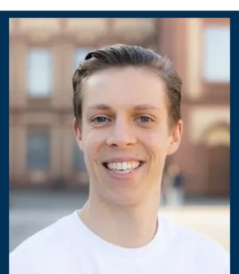


Table 1: **Binary Classification Performance.** We report macro F1-scores (mean ± stdev over 10 trials) on test data with optimized hyperparameters. The rank of each method is presented in brackets.

	Gradient-Based		Non-Greedy		Greedy
	GradTree (ours)	DNDT	GeneticTree	DL8.5 (Optimal)	CART
Blood Transfusion	**0.628 ± .036 (1)**	0.543 ± .051 (5)	0.575 ± .094 (4)	0.590 ± .034 (3)	0.613 ± .044 (2)
Banknote Authentication	**0.987 ± .007 (1)**	0.888 ± .013 (5)	0.922 ± .021 (4)	0.962 ± .011 (3)	0.982 ± .007 (2)
Titanic	**0.776 ± .025 (1)**	0.726 ± .049 (5)	0.730 ± .074 (4)	0.754 ± .031 (2)	0.738 ± .057 (3)
Raisins	0.840 ± .022 (4)	0.821 ± .033 (5)	**0.857 ± .021 (1)**	0.849 ± .027 (3)	0.852 ± .017 (2)
Rice	0.926 ± .007 (3)	0.919 ± .012 (5)	0.927 ± .005 (2)	0.925 ± .008 (4)	**0.927 ± .006 (1)**
Echocardiogram	**0.658 ± .113 (1)**	0.622 ± .114 (3)	0.628 ± .105 (2)	0.609 ± .112 (4)	0.555 ± .111 (5)
Wisconcin Breast Cancer	0.904 ± .022 (2)	**0.913 ± .032 (1)**	0.892 ± .028 (4)	0.896 ± .021 (3)	0.886 ± .025 (5)
Loan House	**0.714 ± .041 (1)**	0.694 ± .036 (2)	0.451 ± .086 (5)	0.607 ± .045 (4)	0.662 ± .034 (3)
Heart Failure	0.750 ± .070 (3)	0.754 ± .062 (2)	0.748 ± .068 (4)	0.692 ± .062 (5)	**0.775 ± .054 (1)**
Heart Disease	**0.779 ± .047 (1)**	$n > 12$	0.704 ± .059 (4)	0.722 ± .065 (2)	0.715 ± .062 (3)
Adult	0.743 ± .034 (2)	$n > 12$	0.464 ± .055 (4)	0.723 ± .011 (3)	**0.771 ± .011 (1)**
Bank Marketing	**0.640 ± .027 (1)**	$n > 12$	0.473 ± .002 (4)	0.502 ± .011 (3)	0.608 ± .018 (2)
Congressional Voting	**0.950 ± .021 (1)**	$n > 12$	0.942 ± .021 (2)	0.924 ± .043 (4)	0.933 ± .032 (3)
Absenteeism	**0.626 ± .047 (1)**	$n > 12$	0.432 ± .073 (4)	0.587 ± .047 (2)	0.564 ± .042 (3)
Hepatitis	0.608 ± .078 (2)	$n > 12$	0.446 ± .024 (4)	0.586 ± .083 (3)	**0.622 ± .078 (1)**
German	**0.592 ± .068 (1)**	$n > 12$	0.412 ± .006 (4)	0.556 ± .035 (3)	0.589 ± .065 (2)
Mushroom	**1.000 ± .001 (1)**	$n > 12$	0.984 ± .003 (4)	0.999 ± .001 (2)	0.999 ± .001 (3)
Credit Card	0.674 ± .014 (4)	$n > 12$	**0.685 ± .004 (1)**	0.679 ± .007 (3)	0.683 ± .010 (2)
Horse Colic	**0.842 ± .039 (1)**	$n > 12$	0.496 ± .169 (4)	0.708 ± .038 (3)	0.786 ± .062 (2)
Thyroid	0.905 ± .010 (2)	$n > 12$	0.605 ± .116 (4)	0.682 ± .018 (3)	**0.922 ± .011 (1)**
Cervical Cancer	**0.521 ± .043 (1)**	$n > 12$	0.514 ± .034 (2)	0.488 ± .027 (4)	0.506 ± .034 (3)
Spambase	0.903 ± .025 (2)	$n > 12$	0.863 ± .019 (3)	0.863 ± .011 (4)	**0.917 ± .011 (1)**
Mean Relative Diff. (MRD) ↓	**0.008 ± .012 (1)**	0.056 ± .051 (3)	0.211 ± .246 (5)	0.084 ± .090 (4)	0.035 ± .048 (2)
Mean Reciprocal Rank (MRR) ↑	**0.758 ± .306 (1)**	0.370 ± .268 (3)	0.365 ± .228 (4)	0.335 ± .090 (5)	0.556 ± .293 (2)

Sascha Marton
sascha.marton@uni-mannheim.de
University of Mannheim



Dr. Christian Bartelt
christian.bartelt@uni-mannheim.de
University of Mannheim

Jun.-Prof. Dr. Stefan Lüdtke
stefan.luedtke@uni-rostock.de
Institute for Visual and Analytic Computing

Prof. Dr. Heiner Stuckenschmidt
heiner.stuckenschmidt@uni-mannheim.de
University of Mannheim

Github

https://github.com/s-marton/GradTree

Follow-Up Work @ICLR'24

GRANDE: Gradient-Based Decision Tree Ensembles