



THE UNIVERSITY  
OF BRITISH COLUMBIA



# Differentiable Decision Tree via "ReLU+Argmin" Reformulation

39th Conference on Neural Information Processing Systems (NeurIPS 2025)

🚩 Spotlight Paper

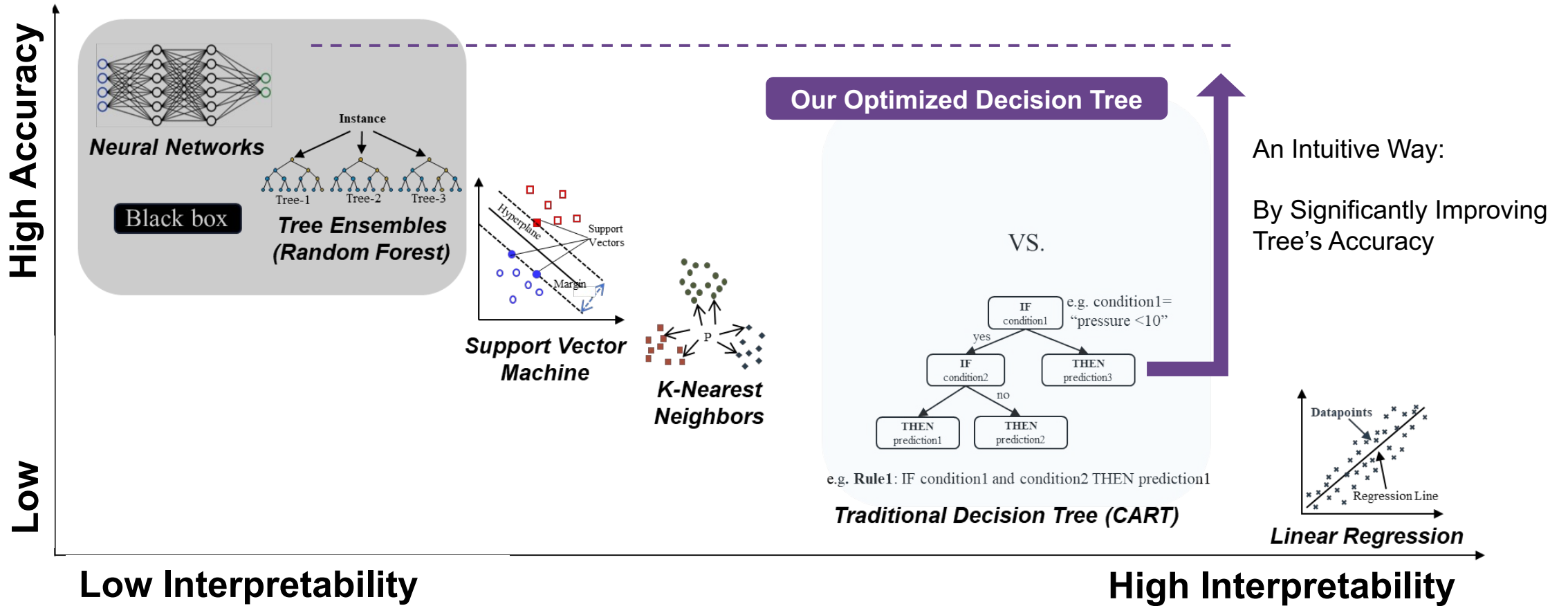
Presenter: Qiangqiang Mao

Authors: Mao Q., Ren J., Wang Y., Zou C., Zheng J. and Cao Y.

University of British Columbia, Vancouver, Canada

- 1 Introduction and Motivation**
  - ◆ Decision tree's two key issues that limit its broader applicability
- 2 Proposed “ReLU+Argmin”-based Differentiable Decision Tree**
  - ◆ Unconstrained oblique decision tree exact reformulation
  - ◆ Gradient-based entire tree optimization framework for RADDT
- 3 Numerical Experiments and Discussions**
  - ◆ Accuracy, model complexity, inference time and interpretability

## Decision Tree's Two Key Issues That Limit Its Broader Applicability: Lower testing accuracy and Non-differentiability



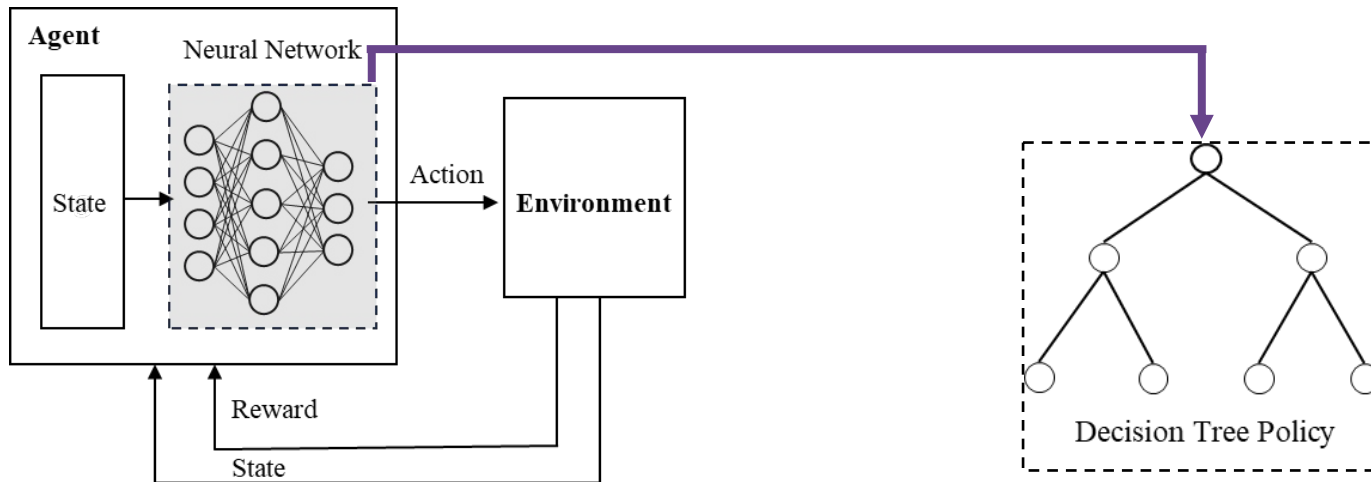
**First aim:** to improve tree's accuracy

## Decision Tree's Two Issues: Lower test accuracy and Non-differentiability

Non-differentiability: **two sources of hard decisions**

- Hard splits at **branch nodes** for **sample branching**.
- Unique decision path at **leaf node** for **sample prediction**.

Limitation in gradient-based optimization tasks.



➔ **Second aim:**  
differentiable tree

decision tree policy optimization in reinforcement learning.

## Improve Decision Tree's Accuracy Through Advanced Optimization

### Greedy Optimization

CART (Breiman et al., 1984)  
OC1 (Murthy et al., 1994)



### Mixed-integer Programming

OCT/ORT (Bertsimas and Dunn, 2017)  
Scalability Issue



### Other optimization

Gradient-based  
...  
Solvability Benefit

## Concerns on existing gradient-based trees

Straight-through estimator: **suboptimal**.



Approximate hard decisions (0 or 1) with probability (0, 1): **soft tree**.



➡ How can **high accuracy** be achieved in a differentiable tree **while** preserving the two sources of **hard decisions**?

- 1 Introduction and Motivation**
  - ◆ Decision tree's two key issues that limit its broader applicability
  
- 2 Proposed “ReLU+Argmin”-based Differentiable Decision Tree**
  - ◆ Unconstrained oblique decision tree exact reformulation
  - ◆ Gradient-based entire tree optimization framework for RADDT
  
- 3 Numerical Experiments and Discussions**
  - ◆ Accuracy, model complexity, inference time and interpretability

## Unconstrained Oblique Decision Tree Reformulation

### Basic mathematical notations

Dataset:  $\{\mathbf{x}_i, y_i\}_{i=1}^n$

Tree Depth:  $D$

Branch node:  $\mathbb{T}_B$

Leaf node:  $\mathbb{T}_L$

Weight :

$$\mathbf{a}_t \in \mathbb{R}^p \quad t \in \mathbb{T}_B$$

Threshold:

$$b_t \in \mathbb{R} \quad t \in \mathbb{T}_B$$

Tree split parameters

Leaf value:

$$\theta_t = \{\mathbf{k}_t \in \mathbb{R}^p, h_t \in \mathbb{R}\} \quad t \in \mathbb{T}_L \text{ for regression}$$

$$\theta_t = \{h_t \in \mathbb{R}^c\} \text{ for classification}$$

Leaf prediction parameter

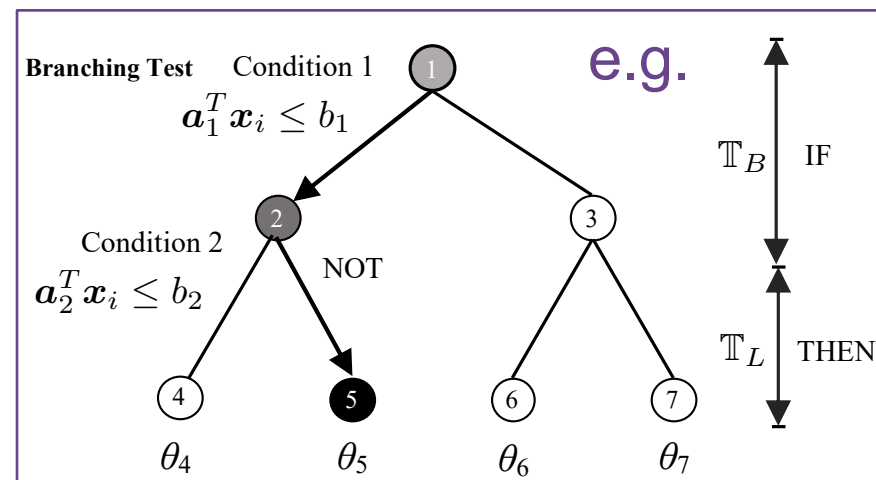
Final prediction:

$$\hat{y}_i = h_t$$

for tree with constant predictions

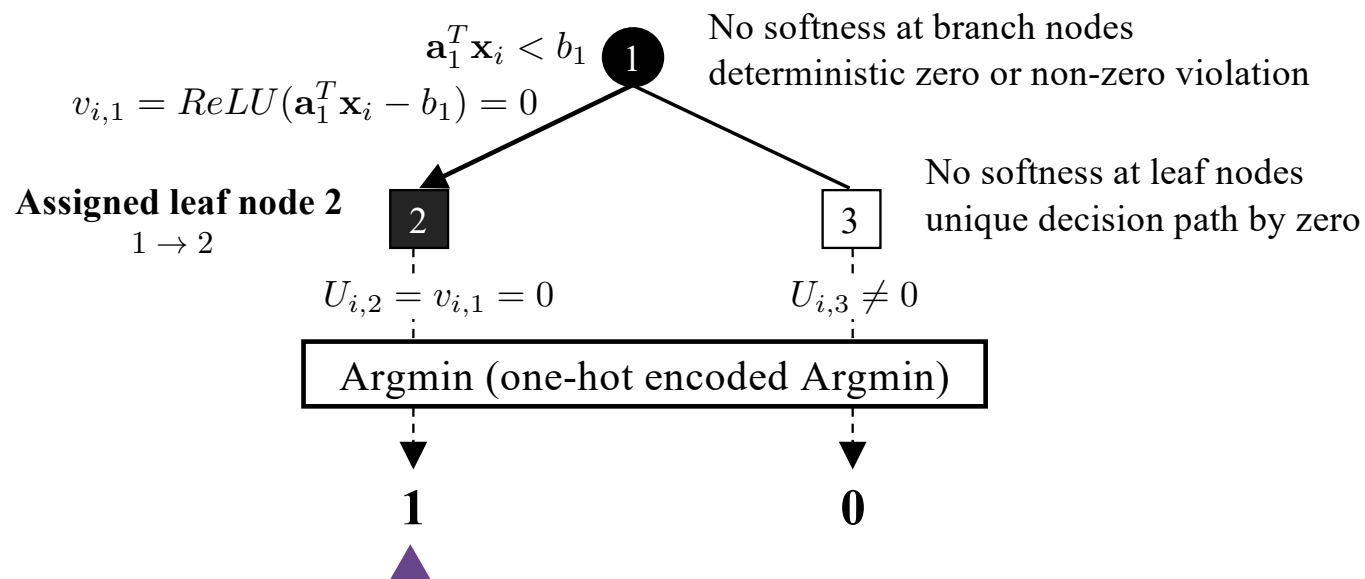
$$\hat{y}_i = \mathbf{k}_t^T \mathbf{x}_i + h_t$$

for tree with linear predictions



## Unconstrained Oblique Decision Tree Reformulation

Example of 1-depth tree (correct sample assignment  $1 \rightarrow 2$ )



◆ Violations of correctly-directed path

$$v_{i,1} = \text{ReLU}(\mathbf{a}_1^T \mathbf{x}_i - b_1) = 0$$

Correct-direction; No violation.

◆ Cumulative violations: only *one* zero-violation at the *unique* path

Path to node 2:  $U_{i,2} = 0$

Path to node 3:  $U_{i,3} \neq 0$

◆ Loss function:  $\mathcal{L}_i = (y_i - \theta_2)^2$

$$\mathcal{L}_i = \left[ \underline{1} \cdot (y_i - \theta_2)^2 + 0 \cdot (y_i - \theta_3)^2 \right]$$

Correctly Assigned

## Unconstrained Oblique Tree Exact Reformulation

ReLU-based hard splits  $\rightarrow$  left-right branching (zero or non-zero violation).

*No softness for sample branching*

❖ Correct direction: violation  $v_{i,j} = 0$

Cumulative violations across all ancestors  $\mathbb{A}_t = \mathbb{A}_t^l \cup \mathbb{A}_t^r$

$$U_{i,t} = \sum_{j \in \mathbb{A}_t^l} v_{i,j} + \sum_{j \in \mathbb{A}_t^r} v_{i,j}$$

❖ Unique decision path formulation using **Argmin** (*No softness for sample prediction*)

$$\mathbb{M}(U_{i,t}) = \mathbb{1}(t = (\text{Argmin}(\mathbf{U}_i) + \lceil T/2 \rceil))$$

$$\mathbf{U}_i = \{U_{i, \lceil T/2 \rceil + 1}, \dots, U_{i,T}\}$$



Only outputting **one** when  $U_{i,t} = 0$ ; otherwise 0.

**Unique tree path (correctly assigned)**

❖ Unconstrained optimization task

$$\mathcal{L} = \sum_{i=1}^n \sum_{t \in \mathbb{T}_L} \mathbb{M}(U_{i,t}) \ell(y_i, \hat{y}_i)$$

## Multi-run Warm Start Annealing Strategy For Scaling Softmin Operations

$$\mathcal{L} = \sum_{i=1}^n \sum_{t \in \mathbb{T}_L} \mathbb{M}(U_{i,t}) \ell(y_i, \hat{y}_i)$$

Involve Argmin operation  $\rightarrow$

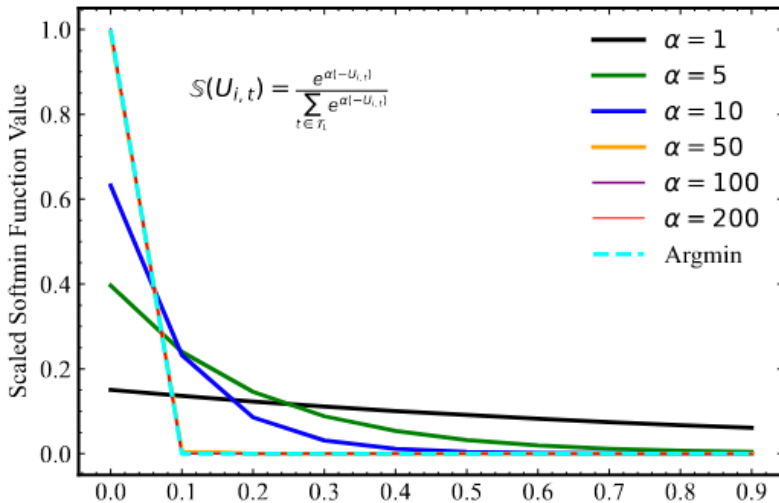
undefined gradient

### ◆ Scaled Softmin to approximate Argmin

$$\mathbb{S}(U_{i,t}) = \frac{e^{\alpha(-U_{i,t})}}{\sum_{t \in \mathbb{T}_L} e^{\alpha(-U_{i,t})}}$$

$\rightarrow$

$$\mathcal{L} = \sum_{i=1}^n \sum_{t \in \mathbb{T}_L} \mathbb{S}(U_{i,t}) \ell(y_i, \hat{y}_i)$$



By starting with a smaller  $\alpha$  and gradually increasing it

The solution from an optimization task with a smaller  $\alpha$  is used to warm-start the next task with a larger  $\alpha$ .

Trade-off between approximation accuracy and numerical stability

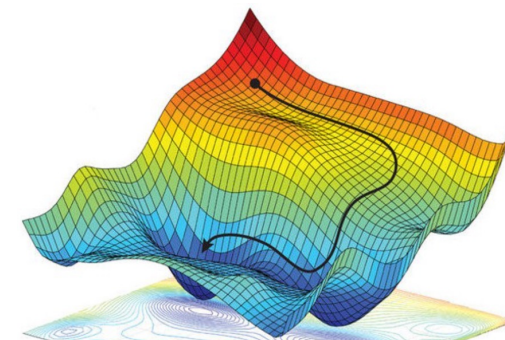
## Gradient-based Entire Tree Optimization Framework for RADDT

### Why to implement optimization within existing frameworks?

- Efficient gradient calculations  
Mature auto differentiation tools.
- Efficient parameter updates  
Powerful gradient-based **optimizers**, e.g. Adam optimizer.
- High-performance computing  
Distributed multi-GPU parallelization.

 TensorFlow

 PyTorch



- 1 Introduction and Motivation**
  - ◆ Decision tree's two key issues that limit its broader applicability
  
- 2 Proposed “ReLU+Argmin”-based Differentiable Decision Tree**
  - ◆ Unconstrained oblique decision tree exact reformulation
  - ◆ Gradient-based entire tree optimization framework for RADDT
  
- 3 Numerical Experiments and Discussions**
  - ◆ Accuracy, model complexity, inference time and interpretability

## 14 compared baselines across 8 groups

- (a) greedy: CART, HHCART (Wickramarachchi et al., 2015), RandCART (Blaser and Fryzlewicz, 2016), OC1 (Murthy et al., 1994)
- (b) non-greedy tree: TAO (Zharmagambetov and Carreira-Perpinan, 2020)
- (c) gradient-based trees using STE: GradTree (Marton et al., 2023), DGT (Karthikeyan et al., 2022), DTSemNet (Panda et al., 2024)
- (d) sigmoid-based soft tree: SoftDT (Frosst and Hinton, 2017)
- (e) relaxed MIP-based tree solved via implicit differentiation LatentTree (Zantedeschi et al., 2021)
- (f) Local search ORT-LS (Dunn, 2018)
- (g) soft tree variant using smooth-step function TEL (Hazimeh et al., 2020) (Tree ensemble)
- (h) other ensembles like random forest RF and XGBoost

Since it remains uncertain whether oblique trees can match tree ensemble's accuracy, we include RF and XGBoost as a commonly used and strong ensemble baselines, and TEL as the oblique ensemble counterpart.

## Evaluation on 4 groups of datasets

- (i) 17 medium-sized regression datasets (**primary focus**)
- (ii) 27 classification and 9 regression datasets used in the papers of certain baselines (for a **more credible comparison**)
- (iii) 4 real-world small datasets, and 3 synthetic datasets (for comparing **global optimality**)
- (iv) 7 **million-scale** datasets (for evaluating **scalability**).

We **mainly** evaluate **regression tasks**, but also include the same **classification** datasets used by certain baselines for a **more convincing comparison**.

## Test Accuracy Comparison

### Our tree with constant predictions RADDT

	Greedy Methods				Gradient-based <sup>1, 2</sup>		Local Search	Ours
	CART	RandCART	HHCART	OC1	GradTree	SoftDT	ORT-LS	RADDT
Testing Accuracy ( $R^2$ , %)	74.85	71.20	76.75	73.31	64.30	72.72	78.67	<b>82.39</b>
Tree Depth	10.24	8.12	8.29	8.12	10.29	10.29	6.24	7.29
Friedman Rank	4.65	5.88	3.65	5.06	7.00	4.76	3.24	<b>1.76</b>

7.54% higher than CART

3.72% higher than ORT-LS

<sup>1</sup> Other two gradient-based DGT and LatentTree are compared later with fixed depths due to scalability issues at depth 12.

<sup>2</sup> More results on their originally-used dataset are discussed in the paper, include GradTree, SoftDT, DGT, and DTSemNet.

### Our tree with linear predictions RADDT-Linear

	RF	XGBoost	TEL	RADDT	RADDT-Linear
Number of Trees	314.71	352.94	10	1	1
Test Accuracy ( $R^2$ , %)	82.62	83.51	83.87	82.39	<b>84.63</b>
Friedman Rank	2.88	<b>2.24</b>	3.29	4.35	<b>2.24</b>

- RADDT-Linear **outperforms RF by 2.01%**

Our aim is **NOT to claim** superiority over ensembles. Instead, we provide a reference showing that our tree can match ensemble's accuracy **with far fewer parameters**.

## Superior Testing Accuracy Analysis: From Training Optimality Perspective

Fixed-depth comparison of training, testing accuracy and training time on Group (i) datasets.

	D	Greedy Methods				Gradient-based Trees				Local Search	Ours
		CART	RankCART	HHCART	OC1	GradTree	SoftDT	DGT	LatenTree	ORT-LS	RADDT
Train ( $R^2$ ,%)	2	46.95	32.81	46.20	49.59	39.42	50.59	64.48	66.94	66.43	<b>71.78</b>
	4	61.16	54.09	62.65	63.21	53.29	56.83	75.83	72.25	79.52	<b>82.18</b>
	8	81.45	77.51	82.26	81.43	64.65	66.81	82.01	74.54	90.91	<b>90.93</b>
	12	93.45	93.06	94.74	93.02	66.86	73.03	/	/	97.46	96.36
Test ( $R^2$ ,%)	2	46.14	32.47	45.61	47.95	38.53	49.94	63.81	66.42	64.51	<b>70.07</b>
	4	58.92	52.69	61.30	60.22	51.45	56.32	75.16	71.03	75.06	<b>78.98</b>
	8	69.77	69.93	74.57	69.08	62.40	66.44	79.99	70.77	74.93	<b>77.89</b>
	12	68.28	64.13	68.37	65.57	63.81	72.45	/	/	68.25	<b>73.11</b>
Time (s)	2	0.03	0.70	4.43	3,216	31.32	22.24	2,102	1,624	457.21	542.22
	4	0.04	1.56	7.45	4,192	54.74	81.88	2,577	2,194	868.08	478.81
	8	0.07	5.08	12.52	4,803	298.92	1,209	4,049	2,381	9,336	602.69
	12	0.10	12.61	22.01	5,103	10,417	54,829	/	/	210,141	2,643

- Our RADDT outperforms the baseline CART by 24.83%, 21.02% and 9.48% in training for depths of 2, 4 and 8

⇒ showcase the efficacy of our tree optimization to achieve better training optimality.

High train accuracy → High test accuracy (if no overfitting)

## GPU Acceleration (Training Time), Model Complexity (Inference Time)

- Our method scales to 12-depth deep tree on **million-scale** datasets, where existing gradient-based trees fail.
- For a **similar accuracy**, our oblique tree may **NOT** require more parameters overall than orthogonal tree.
- Interpretability depends on **multiple factors**: tree depth, feature number, and the accuracy-simplicity trade-off.

- Inadequate regularization.
- A lack of theoretical analysis.
- Fail to consider the corner case where both  $a_j = 0$  and  $b_j = 0$ .
- More tuning on regularization parameters for tree ensembles.
- Apply our proposed optimization strategies to other gradient-based trees.
- Try different alternatives for scaled softmin operations, e.g. entmax function.

# Reference

- [1] Breiman, L., Friedman, J., Stone, C.J., Olshen, R.A., 1984. Classification and regression trees.
- [2] Murthy, S.K., Kasif, S., Salzberg, S., 1994. A System for Induction of Oblique Decision Trees.
- [3] Wickramarachchi, D.C., Robertson, B.L., Reale, M., Price, C.J., Brown, J., 2015. HHCART: An Oblique Decision Tree.
- [4] Blaser, R., Fryzlewicz, P., 2016. Random Rotation Ensembles. *Journal of Machine Learning Research* 17, 1–26.
- [5] Zharmagambetov, A., Carreira-Perpinan, M., 2020. Smaller, more accurate regression forests using tree alternating optimization.
- [6] Marton, S., Bartelt, C., Lüdtkke, S., 2023. Learning Axis-Aligned Decision Trees with Gradient Descent.
- [7] Karthikeyan, A., Jain, N., Natarajan, N., Jain, P., 2022. Learning Accurate Decision Trees with Bandit Feedback via Quantized Gradient Descent.
- [8] Panda, S.P., Genest, B., Easwaran, A., Suganthan, P.N., 2024. Vanilla Gradient Descent for Oblique Decision Trees.
- [9] Frosst, N., Hinton, G., 2017. Distilling a Neural Network Into a Soft Decision Tree.
- [10] Zantedeschi, V., Kusner, M., Niculae, V., 2021. Learning Binary Decision Trees by Argmin Differentiation.
- [11] Dunn, J.W., 2018. Optimal trees for prediction and prescription (Thesis).
- [12] Hazimeh, H., Ponomareva, N., Mol, P., Tan, Z., Mazumder, R., 2020. The Tree Ensemble Layer: Differentiability meets Conditional Computation.

Other references can be found in the paper of [“Differentiable decision tree via ‘Relu+Argmin’ reformulation”](#).



THE UNIVERSITY  
OF BRITISH COLUMBIA



**Thank you.**

**Differentiable Decision Tree via "ReLU+Argmin" Reformulation**

39th Conference on Neural Information Processing Systems (NeurIPS 2025)

🚩 Spotlight Paper

Presenter: Qiangqiang Mao

Authors: Mao Q., Ren J., Wang Y., Zou C., Zheng J. and Cao Y.

University of British Columbia, Vancouver, Canada