Supplementary materials for ”NCART: Neural Classification and Regression Tree for Tabular Data”

Jiaqi Luo[a], Shixin Xu[a,*]

[a] *Data Science Research Center, Duke Kunshan University, No.8 Duke Avenue, Kunshan, 215000, Jiangsu Province, China*

1. Sparse Function

We denote the d–probability simplex (the set of vectors representing probability distributions over d choices) by $\Delta^d := \{\mathbf{p} \in \mathbb{R}^d : \mathbf{p} \geq \mathbf{0}, \|\mathbf{p}\|_1 = 1\}$.

sparsemax is an alternative to softmax which tends to yield sparse probability distributions:

$$\text{sparsemax}(\mathbf{z}) := \text{argmin}_{\mathbf{p}\in\Delta^d} \|\mathbf{p}-\mathbf{z}\|^2.$$

entmax is a more general sparse function that has the following expression:

$$\alpha - \text{entmax}(\mathbf{z}) := \text{argmin}_{\mathbf{p}\in\Delta^d} \mathbf{p}^\mathsf{T}\mathbf{z} + H^T_\alpha(\mathbf{p}).$$

*Corresponding author

Email addresses: `jiaqi.luo@dukekunshan.edu.cn` (Jiaqi Luo), `shixin.xu@dukekunshan.edu.cn` (Shixin Xu)

Here, $H_\alpha^T(\mathbf{p})$ is called $Tsallis\alpha - entropie$ which is defined as:

$$H_\alpha^T(\mathbf{p}) = \begin{cases} \frac{1}{\alpha(\alpha-1)} \sum_{j=1}^{d} (p_j - p_j^\alpha) & \alpha \neq 1 \\ -\sum_{j=1}^{d} p_j \log p_j & \alpha = 1, \end{cases}$$

where p_j is the j_{th} component of $\mathbf{p}$. It's worth noting that $1 - entmax$ is equivalent to the $softmax$ function, and $2 - entmax$ is equal to the $sparsemax$ function.

2. Datasets Description

Table. 1 are the datasets used in this paper, the column #Tar. means the number of distinct target values in the label, and the column Objective gives a brief introduction to the objectives of each dataset. The selection of data follows the following principles: 1. Diversity of feature types; 2. Diversity of feature dimensions; 3. Diversity of sample numbers.

3. Optimization of hyperparameters

Table. 2 lists the search range of hyperparameters, which refers to the survey [1] and some adjustments are made on this basis to try to avoid GPU memory overflow and to lower the risk of overfitting on small-scale datasets. We implement the NCART model [1] using PyTorch and employ the official open-source

[1]The code is available at https://github.com/Luojiaqimath/NCART

Dataset	#Samp.	#Num.	#Cat.	#Tar.	#Maj.	Data id	Objective
			Binary Classification				
diabetes	768	8	0	2	0.65	37	Patient's diabetes status recognition
credit-g	1000	20	13	2	0.70	31	People's credit risks prediction
qsar-biodeg	1055	41	0	2	0.66	1494	Analyze molecule biodegradation and structure
scene	2407	299	5	2	0.82	312	Scene recognition
ozone-level	2534	72	0	2	0.94	1487	Ozone level alarm forecasting
delta_ailerons	7129	5	0	2	0.53	803	Aileron control prediction for aircraft
MagicTelescop	13376	10	0	2	0.50	44125	Gamma signal recognition
jannis	57580	54	0	2	0.50	44131	For challenging machine learning competitions
road-safety	111762	32	3	2	0.50	44161	Personal injury road accidents recognition
Higgs	940160	24	0	2	0.50	44129	Higgs boson signals recognition
			Multiclass Classification				
autoUniv-au7	700	12	4	3	0.35	1553	An advanced dataset generator for classification
plants-margin	1600	64	0	100	0.01	1491	Plant leaf classification
waveform	5000	40	0	3	0.34	60	Waves prediction
gas-drift	13910	128	0	6	0.21	1476	Gas sensor drift prediction
EMNIST	131600	784	0	47	0.02	41039	Handwritten character digits recognition
			Regression				
analcatdata	4052	7	5	10	-	44055	For analyzing categorical data
kin8nm	8192	8	0	8188	-	189	Predict the dynamics of a robot arm
superconduct	21263	79	0	3007	-	44148	The critical temperature prediction
house_sales	21613	17	2	4028	-	44066	House sale prices prediction
year	515345	90	0	89	-	44027	Song release year prediction

Table 1: Details of datasets (sorted by dataset size in each task type). #Samp. = #Sample; #Num. = #Numerical features; #Cat. = #Categorical features; #Tar. = #Target; #Maj. = percentage of the majority class. #Target in regression task means the number of unique values.

implementations for other models [2] [3] [4] [5] [6] [7] [8] [9] [10].

[2]XGBoost: https://xgboost.readthedocs.io/en/stable/
[3]CatBoost: https://catboost.ai/
[4]LightGBM: https://lightgbm.readthedocs.io/en/latest/
[5]NODE: https://github.com/Qwicen/node
[6]TabNet: https://github.com/dreamquark-ai/tabnet
[7]SAINT: https://github.com/somepago/saint
[8]FT-Transformer & ResNet: https://github.com/Yura52/rtdl
[9]TabCaps: https://github.com/WhatAShot/TabCaps
[10]RLN: https://github.com/irashavitt/regularization_learning_networks

HyperParameters	Range	HyperParameters	Range
		XGBoost	
num_boost_round	1000	*early_stopping_rounds*	10
max_depth	LogUniformInt [2, 10]	*alpha*	LogUniform [1e-8, 0.1]
lambda	LogUniform [0.5, 2]	*eta*	LogUniform [0.05, 0.3]
		CatBoost	
iterations	1000	*od_wait*	10
max_depth	LogUniformInt [2, 10]	*l2_leaf_reg*	LogUniform [0.1, 2]
learning_rate	LogUniform [0.05, 0.3]		
		LightGBM	
iterations	1000	*early_stopping_round*	10
num_leaves	LogUniformInt [8, 64]	$lambda\_l_1$	LogUniform [1e-8, 0.1]
$lambda\_l_2$	LogUniform [1e-8, 0.1]	*learning_rate*	LogUniform [0.05, 0.3]
		NODE	
num_layers	[2, 4, 8]	*total_tree_count*	[128, 256]
tree_depth	[4, 6, 8]	*tree_output_dim*	[2, 3]
		TabNet	
n_d	LogUniform [8, 16]	*n_steps*	UniformInt [1, 6]
gamma	Uniform [1, 2]	*n_independent*	UniformInt [1, 5]
n_shared	UniformInt [1, 5]	*momentum*	LogUniform [0.01, 0.4]
mask_type	[sparsemax, entmax]		
		SAINT	
dim	[16, 32, 64]	*depth*	[1, 2, 3, 6]
heads	[2, 4, 8]	*dropout*	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
		FT-Transformer	
num_blocks	UniformInt [1, 6]	*num_tokens*	[8, 16, 24, 32, 64, 128, 256, 512]
dropout_att	[0, 0.1, 0.2, 0.3, 0.4, 0.5]	*dropout_ffn*	[0, 0.1, 0.2, 0.3, 0.4, 0.5]
dropout_res	[0, 0.1, 0.2, 0.3, 0.4, 0.5]		
		TabCaps	
learning_rate	UniformInt [0.02, 0.1]	*num_senior_capsules*	UniformInt[1, 5]
primary_capsule_size	UniformInt[64, 128]	*num_primary_capsules*	UniformInt[4, 32]
senior_capsule_size	UniformInt[4, 32]	*num_learnable_words*	UniformInt[16, 64]
		RLN	
layers	UniformInt [2, 6]	*theta*	UniformInt [-12, -8]
norm	[1, 2]		
		ResNet	
n_blocks	UniformInt [1, 10]	*d_hidden*	[32, 64, 128, 256, 512]
dropout	[0, 0.1, 0.2, 0.3, 0.4, 0.5]		
		NCART	
N in Eq. (9)	[8, 16, 32, 64]	d in Eq. (4)	UniformInt [2, 10]
L in Eq. (12)	[2, 4]	h in Eq. (5)	[sparsemax, entmax]

Table 2: Hyperparameters space.

4. Inference Time Figures

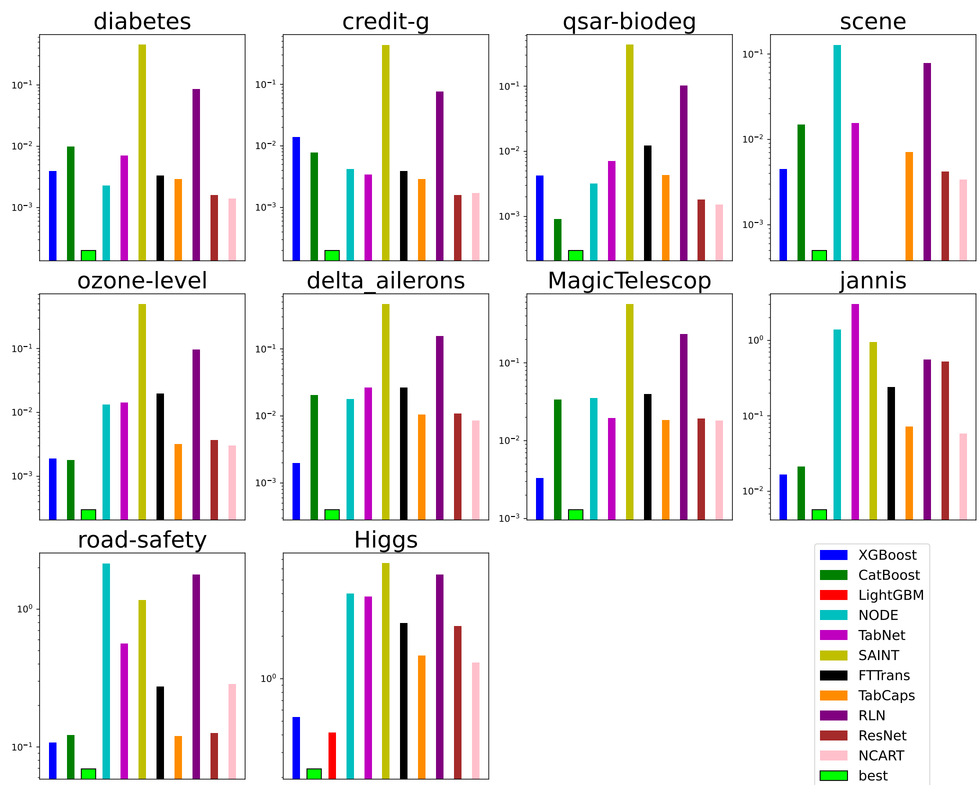


Figure 1: Average inference time(s) of a five cross-validation with the best hyperparameters for binary classification tasks. Subfigures are sorted by dataset size. The value on the y-axis represents the inference time. Missing areas indicate that the model has GPU memory overflow.

5. Training Time

The detailed training time(s) of five cross-validation with the best hyperparameters are shown in Table 3.

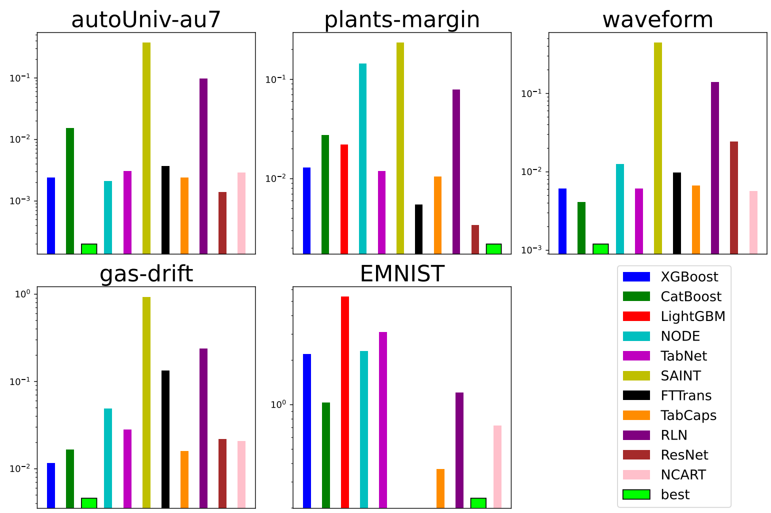


Figure 2: Average inference time(s) of a five cross-validation with the best hyperparameters for multi-class classification tasks. Subfigures are sorted by dataset size. The value on the y-axis represents the inference time. Missing areas indicate that the model has GPU memory overflow.

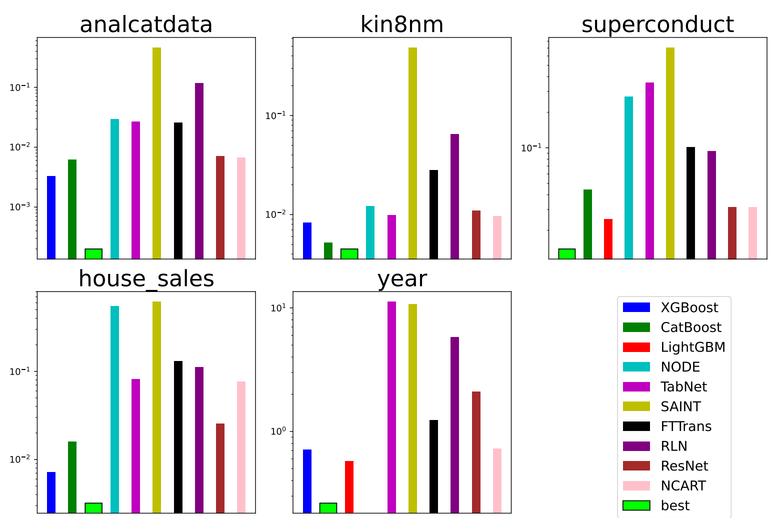


Figure 3: Average inference time(s) of a five cross-validation with the best hyperparameters for regression tasks. Subfigures are sorted by dataset size. The value on the y-axis represents the inference time. Missing areas indicate that the model's running time exceeds the time limit.

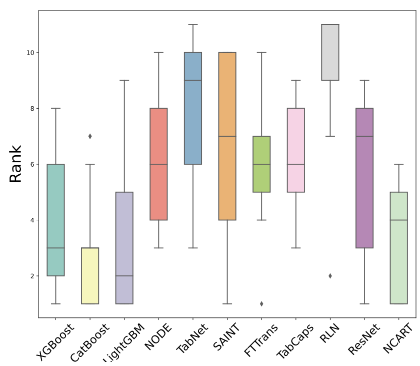
Dataset	XGBoost	CatBoost	LightGBM	NODE	TabNet	SAINT	FTTrans	TabCaps	RLN	ResNet	NCART
					Binary Classification						
diabetes	**0.08**	1.08	0.11	32.85	0.13	156.10	1.09	0.74	24.22	1.99	5.79
credit-g	0.18	4.19	0.16	84.10	**0.07**	73.99	6.28	0.39	16.08	2.44	7.18
qsar-biodeg	**0.13**	3.21	0.18	20.84	0.14	199.24	29.71	2.02	46.89	1.39	13.34
scene	0.44	3.48	**0.21**	193.95	14.73	*OOM*	*OOM*	2.13	34.73	1.63	13.16
ozone-level	**0.11**	1.91	0.24	211.53	2.15	298.27	70.45	1.30	4.98	3.23	8.26
delta_ailerons	**0.06**	2.95	0.17	346.49	50.27	89.17	91.79	1.81	5.11	10.97	15.22
MagicTelescop	**0.29**	10.61	0.38	369.31	31.19	180.62	18.59	22.92	59.22	20.43	19.19
jannis	1.31	18.75	**0.86**	500.89	684.32	169.47	155.88	37.78	83.88	48.70	36.45
road-safety	3.87	5.85	**3.82**	1000.29	1306.03	443.30	805.73	135.45	25.47	192.86	132.75
Higgs	**6.63**	9.63	10.73	954.56	8209.51	1162.72	2914.61	1022.71	599.21	1015.46	1424.64
					Multiclass Classification						
autoUniv-au7	0.20	1.01	0.18	30.11	**0.06**	136.29	7.06	0.33	41.37	1.11	4.13
plants-margin	6.53	10.05	6.66	405.50	81.59	103.81	39.45	27.11	4.60	**3.94**	4.85
waveform	1.19	2.30	**0.45**	28.52	9.57	129.35	4.24	2.38	46.18	4.34	9.48
gas-drift	**2.10**	6.68	6.81	1145.76	41.46	455.32	131.10	8.73	33.51	14.81	35.15
EMNIST	251.56	207.63	249.05	2649.74	1536.03	*OOM*	*OOM*	304.60	**98.74**	99.62	288.04
Average Rank	**1.93**	3.93	2.20	9.53	6.73	10.13	8.20	4.80	7.13	5.07	6.47
Best/Worst	**7/0**	0/0	4/0	0/4	2/3	0/9	0/3	0/0	1/0	1/0	0/0
Total time	**274.67**	289.33	280.00	7974.43	11967.27	≫4420.14	≫4276.10	1570.42	1164.22	1422.93	1991.12
					Regression						
analcatdata	0.31	1.10	**0.15**	97.77	7.66	42.17	9.39	-	30.32	6.41	11.49
kin8nm	1.69	4.13	**1.43**	210.32	108.63	323.10	29.08	-	39.24	15.71	14.98
superconduct	**2.46**	13.37	16.24	858.54	132.95	569.11	279.68	-	81.50	28.59	44.02
house_sales	0.81	5.07	**0.64**	1744.48	132.61	387.82	70.98	-	169.44	22.45	36.68
year	14.22	**8.25**	12.92	*TimeOut*	3036.32	3916.41	1805.23	-	120.02	1594.27	690.45
Average Rank	2.0	2.4	**1.6**	9.6	8.0	8.6	6.8	-	6.2	4.8	5.0
Best/Worst	1/0	1/0	**3/0**	0/3	0/1	0/1	0/0	-	0/0	0/0	0/0
Total time	**19.48**	31.91	31.37	≫2911.10	3580.22	5238.61	2686.11	-	401.14	1667.43	789.83

Table 3: Average training time(s) of a five cross-validation with the best hyperparameters. The **bold** indicates the top result; - indicates that the model can not be applied to the task type; *OOM* represents there exists GPU overflow; *TimeOut* means the running time exceeds the time limit.

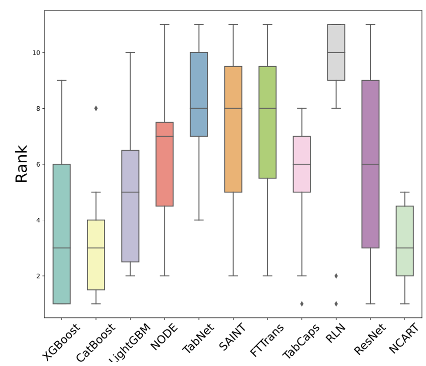
6. Performance Figures

References

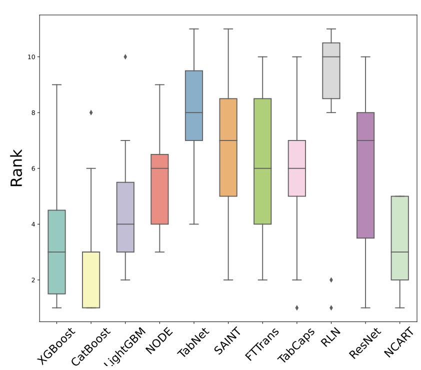
[1] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, G. Kasneci, Deep neural networks and tabular data: A survey, IEEE Transactions on Neural Networks and Learning Systems (2022).



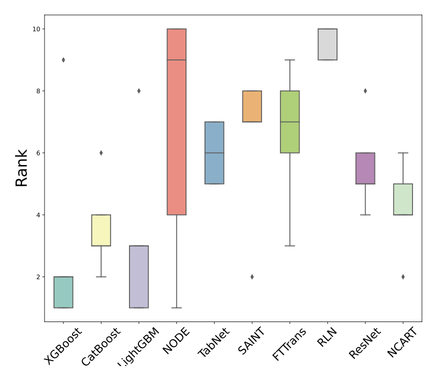
(a) AUC rank



(b) F1-score rank



(c) Accuracy rank



(d) MSE rank

Figure 4: Rank values of different models on 20 datasets. Fig. (a), (b) and (c) are for classification tasks and Fig. (c) is for regression tasks.