УНИВЕРСИТЕТ ИТМО
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
июль–август 2022 Том 22 № 4 http://ntv.ifmo.ru/
SCIENTIFIC AND TECHNICAL JOURNAL OF INFORMATION TECHNOLOGIES, MECHANICS AND OPTICS
Jule–August 2022 Vol. 22 No 4 http://ntv.ifmo.ru/en/
ISSN 2226-1494 (print) ISSN 2500-0373 (online)
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК УНИВЕРСИТЕТ ИТМО
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

doi: 10.17586/2226-1494-2022-22-4-725-733

A multivariate binary decision tree classifier based on shallow neural network

Avazjon R. Marakhimov[1], Jabbarbergen K. Kudaybergenov[2], Kabul K. Khudaybergenov[3], Ulugbek R. Ohundadaev[4]✉

[1] Termez State University, Termez, 190111, Uzbekistan
[2] Tashkent University of Information Technologies Nukus branch named after Muhammad Al-Khwarizmi, Nukus, 230113, Uzbekistan
[3,4] National University of Uzbekistan, Tashkent, 100174, Uzbekistan

[1] termizdu@umail.uz, https://orcid.org/0000-0003-3735-6855
[2] kjabbarbergen@gmail.com, https://orcid.org/0000-0003-4494-6255
[3] kabul85@mail.ru, https://orcid.org/0000-0001-8143-625X
[4] ulugbek_1122@mail.ru✉, https://orcid.org/0000-0002-3240-6502

Abstract
In this paper, a novel decision tree classifier based on shallow neural networks with piecewise and nonlinear transformation activation functions are presented. A shallow neural network is recursively employed into linear and non-linear multivariate binary decision tree methods which generates splitting nodes and classifier nodes. Firstly, a linear multivariate binary decision tree with a shallow neural network is proposed which employs a rectified linear unit function. Secondly, there is presented a new activation function with non-linear property which has good generalization ability in learning process of neural networks. The presented method shows high generalization ability for linear and non-linear multivariate binary decision tree models which are called a Neural Network Decision Tree (NNDT). The proposed models with high generalization ability ensure the classification accuracy and performance. A novel split criterion of generating the nodes which focuses more on majority objects of classes on the current node is presented and employed in the new NNDT models. Furthermore, a shallow neural network based NNDT models are converted into a hyperplane based linear and non-linear multivariate decision trees which has high speed in the processing classification decisions. Numerical experiments on publicly available datasets have showed that the presented NNDT methods outperform the existing decision tree algorithms and other classifier methods.

Keywords
hierarchical classifier, neural networks, binary tree, multivariate decision tree, activation function

Acknowledgments
This research is supported by the Department of Algorithms and Programming technologies, National University of Uzbekistan, Uzbekistan.

For citation: Marakhimov A.R., Kudaybergenov J.K., Khudaybergenov K.K., Ohundadaev U.R. A multivariate binary decision tree classifier based on shallow neural network. *Scientific and Technical Journal of Information Technologies, Mechanics and Optics*, 2022, vol. 22, no. 4, pp. 725–733. doi: 10.17586/2226-1494-2022-22-4-725-733

УДК 517.95

Многомерный двоичный классификатор дерева решений на основе неглубокой нейронной сети

Авазжон Рахимович Марахимов[1], Жаббарберген Кадирбергенович Кудайбергенов[2], Кабул Кадирбергенович Худайбергенов[3], Улугбек Рахимжон угли Охундадаев[4]✉

[1] Термезский государственный университет, Термез, 190111, Узбекистан
[2] Нукусский филиал Ташкентского университета информационных технологий имени Мухаммад ал-Хоразмий, Нукус, 230113, Узбекистан
[3,4] Национальный университет Узбекистана, Ташкент, 100174, Узбекистан

[1] termizdu@umail.uz, https://orcid.org/0000-0003-3735-6855
[2] kjabbarbergen@gmail.com, https://orcid.org/0000-0003-4494-6255
[3] kabul85@mail.ru, https://orcid.org/0000-0001-8143-625X
[4] ulugbek_1122@mail.ru✉, https://orcid.org/0000-0002-3240-6502

Аннотация
Предложен новый классификатор дерева решений, основанный на неглубоких нейронных сетях с кусочными и нелинейными функциями активации преобразования. Данная сеть рекурсивно используется в методах линейного и нелинейного многомерного бинарного дерева решений, которые генерируют узлы разделения и классификатора. Представлено линейное многомерное бинарное дерево решений с неглубокой нейронной сетью, в которой использована выпрямленная линейная единичная функция. Описана новая функция активации с нелинейным свойством, с помощью которой в процессе обучения нейронных сетей получается оптимальная обобщающая способность. Рассмотренный метод продемонстрировал высокую способность к обобщению для моделей линейного и нелинейного многомерного бинарного дерева решений. Предложенные модели обеспечивают точность и производительность классификации. Представлен новый критерий разделения для генерации узлов, который может быть использован в новых моделях дерева решений нейронной сети для большинства классов объектов в текущем узле. Также эти модели могут быть преобразованы в линейные и нелинейные многомерные деревья решений на основе гиперплоскости, и имеют высокую скорость при обработке решений классификации. Численные эксперименты на общедоступных наборах данных показали, что представленные методы превосходят существующие алгоритмы дерева решений и другие методы классификации.

Ключевые слова
иерархический классификатор, нейронные сети, бинарное дерево, многомерное дерево решений, функция активации

Благодарности
Исследование выполнено при поддержке кафедры алгоритмов и технологий программирования Национального университета Узбекистана, Узбекистан.

Ссылка для цитирования: Марахимов А.Р., Кудайбергенов Ж.К., Худайбергенов К.К., Охундадаев У.Р. Многомерный двоичный классификатор дерева решений на основе неглубокой нейронной сети // Научно-технический вестник информационных технологий, механики и оптики. 2022. Т. 22, № 4. С. 725–733 (на англ. яз.). doi: 10.17586/2226-1494-2022-22-4-725-733

Introduction

The latest research works on artificial intelligence and machine learning have embraced prospering developments in many fields, such as regression [1, 2], classification [3], computer vision [4–6], natural language processing [7], image processing [8], speech information [9], and etc. Among the machine learning algorithms and methods which are used in solving real life problems, deep neural networks have showed unprecedented successes and precision due to the great flexibility and powerful generalization ability of learning and decision making [10]. However, deep neural networks also have shortages and some limitations in learning process [11–13].

In machine learning, decision tree methods have been applied in various practical fields, such as medical diagnosis [14], signal processing [15], classification problems [16], and they showed outperformance compared to other algorithms and methods. In general, decision tree models are considered as hierarchical classifiers according to its nature. Also, these methods are able to work with multi-class datasets as well as binary class classification tasks. A recent survey on decision tree models [17], which is widely studied the models and the related issues, show that there exists a lot of various types. In general, according to the different type of node generating or splitting, decision tree models can be categorized as follows:

1) univariate decision trees;
2) multivariate decision trees;
3) omnivariate decision trees.

Univariate decision tree algorithms work with a single attribute at each node and construct nodes and leaf nodes. Some of the algorithms, which work as a univariate decision tree model, are Iterative Dichotomiser 3 (ID3) [18], Classification And Regression Tree (CART) [19], and C4.5 (predecessor of ID3) [20] (which is splitting criterion), are based on information gain coefficient, Gini index, and gain ratio. These splitting criteria are considered as frequency heuristic for splitting decision trees. Later, a new node generating method is proposed in [21], which is considered to be better than information gain and Gini index criteria. However, due to simplicity and a good interpretability of these criteria, in univariate decision trees node splitting may not be appropriate when learning samples are numerically correlated. Moreover, univariate decision tree algorithms are considered to be greedy search, which negatively impacts time performance.

Multivariate decision tree algorithms employ multiple attributes or multiple features to generate a node and splitting that are associated with multiple attributes. Thus, multi attribute criterion generates linear or nonlinear split nodes in the trees. As an example, in [19] a linear multivariate decision tree algorithm, CART, was introduced. Also, a novel multivariate decision tree algorithm was introduced in [22]. This algorithm employs neural network to generate the nodes. Nonlinear multivariate decision tree models are considered sensitive to overfitting problems due to its highly complex node generating process.

Omnivariate decision tree algorithms are decision tree methods where a splitting process at each node can be in univariate, linear multivariate, and even with a combination of nonlinear multivariate techniques. This tree model is considered in [23] where a node splitting is automatically selected among univariate, linear, or nonlinear processes

depending on some statistical experiments. Later, in [24], authors proposed to use both linear and non-linear methods as an ensemble algorithm for node generating. Additionally, a hybrid Support Vector Machines [25, 26] based on decision tree model are proposed for classification problems which have both univariate and multivariate nodes.

Although there are different variants of decision tree models, linear multivariate type of decision tree models is much more preferable. Firstly, this type of model is not restricted to be orthogonal to attribute axis on each splitting hyperplane in nodes like univariate decision tree models. Secondly, linear multivariate type of decision tree models is less complex compared to other decision models and not prone to overfitting problems. Therefore, we focused on linear multivariate type of decision tree models.

In this work, our contribution includes the following. Firstly, a new linear multivariate decision tree model, Neural Network Decision Tree (NNDT) and its algorithm are presented which enhances the multivariate decision tree models. Secondly, we propose a new type of splitting node criterion with piecewise and nonlinear transformation functions. NNDT employs piecewise activation functions and joined activation function with nonlinear transformations. This helps to improve the generalization ability of the algorithm in classification tasks for multi-class datasets. Our model is based on the splitting nodes with hyperplanes in decision tree method which speeds up the classification tasks.

Related work review

A novel ensemble algorithm [22] based on multivariate decision tree and neural network is proposed which employs both binary tree and unsupervised node splitting algorithm. In this research work they have proposed two full binary tree classifiers constructed on multivariate decision trees. The first one was called randomly partitioned multivariate decision tree (MDT-1) and second one was an ensemble method, so called multivariate decision tree (MDT-2), which is based on a Principal Component Analysis (PCA) method.

In general, the ensemble algorithms, MDT-1 and MDT-2 (similar to each other), are considered. These algorithms construct a binary tree in a top-down fashion, constantly adding up child nodes and leaf nodes. Using recursive partition with hyperplanes, the process will continue until there is no node left to split. The algorithm stops generating nodes when all the nodes of a tree to be leaf nodes. The root node of the multivariate decision tree is the first one which separates the whole training data into two classes; nevertheless the data set contains labels more than 2. We can see all the processing steps of the algorithms as follows:

Step-1: Train a multivariate hyperplane for the root node which tries to split all the objects of an entire dataset into the left and right child nodes. After splitting, the first half hyperplane will contain mostly the objects which belong to the same class, and second half will contain all the other objects.

Step-2: Measure a splitting criterion. If the split criteria are satisfied for a next child node, process goes to generate the next node; otherwise, the node is processed as a leaf node with the majority class objects as the representatives of one class which contains mostly of the same class objects.

Step-3: Again, analyze the each obtained node that needs to be split and repeat from Step-1 unless there is any node left to split again.

Clearly, the two main problems in this ensemble algorithm are how to build and learn the multivariate hyperplane tree at each splitting node and measure the splitting condition. Further below, we will look at these two problems explaining how it works. Without losing generality, we consider a single splitting in the decision tree. The splitting procedure is conducted on a node with a dataset matrix $\mathbf{X} \in R^{d \times n}$, and the objects are given as class labels $\mathbf{Y} \in R^{1 \times n}$. Here, n is the number of the objects and d is the feature space dimension.

Split with simple hyperplane

In order to construct multivariate decision tree, both ensemble methods MDT-1 and MDT-2 use a simple linear hyperplane in the following form

$$wx - p = 0, \tag{1}$$

where w is the normal vector of the linear hyperplane and p is a bias parameter. Node splitting of the multivariate decision tree is constructed by separating the sample objects into the two child nodes in the following form:

$$\begin{cases} x \in \mathbf{X}_1, \; if \; wx > p, \\ x \in \mathbf{X}_2, \; if \; wx \le p, \end{cases}$$

where $\mathbf{X}_1$ and $\mathbf{X}_2$ are the sub datasets of objects belonging to left and right child nodes, respectively. Below, Fig. 1 shows an illustration of node splitting with a simple linear hyperplane in the multivariate decision tree.

It can be easily seen that the parameters in eq. (1) define a hyperplane. In both ensemble algorithms, MDT-1 and MDT-2, we can see that the hyperplanes on each node (except the terminal nodes) help to generate new knots.

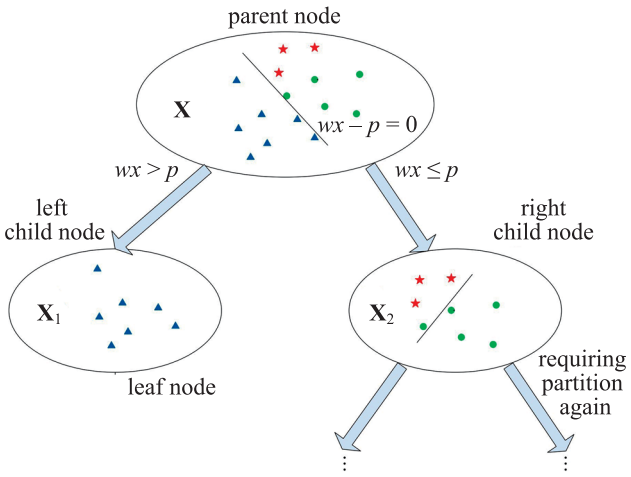

Fig. 1. Node generating with a simple linear hyperplane in the multivariate decision tree

In detail, the ensemble methods MDT-1 and MDT-2 learn the parameters of the hyperplanes in different ways. The first one, MDT-1, uses a vector $\mathbf{v} \in R^{d\times 1}$, such that the dimension is equal to the size of x. Elements of vector $\mathbf{v}$ lie in the interval [–1, 1]. Thus, the normal w in the hyperplane can be calculated as follows.

$$w = \frac{\mathbf{v}}{||\mathbf{v}||}.$$

However, the second method, MDT-2, uses PCA method to calculate a normal direction w. The main idea is to choose the largest principal component on the given matrix $\mathbf{X}$ to satisfy the normal direction.

Splitting criterion

For generated new nodes after each split, a split criterion is employed to determine whether there is need to continue split again. In general, split is continued if the split criterion holds for child node. Thus, split criterion is set in the following form:

$$P(S) = \max\left(\frac{s_i}{c} | i = l_1, \ldots, l_k\right) < \lambda,$$

where S is equal to $\mathbf{X}_1$ or $\mathbf{X}_2$; c is the number of the objects in dataset S; s_i is the number of the objects in the class i; and λ is a predefined threshold parameter in the interval $\left[\frac{1}{k}, 1\right)$. Specifically, if a newly generated child node does not meet the split criterion, it will be considered as a leaf node (terminal node) with a class label

$$l^* = \underset{i}{\operatorname{argmax}}\left\{\frac{s_i}{c_i} | i = l_1, \ldots, l_k\right\}.$$

Classification with ensemble methods

The generated multivariate decision tree can be used as a classification algorithm. The new object with its unknown class label is first come to the root node of the classification tree model, and then goes to child nodes directed by linear hyperplanes to which side the new object belongs of a half hyperplane (Fig. 2). Thus, a previously unseen object by tree model goes through the node of tree and finally reaches a leaf node where classification is made. Thus, learning hyperplanes on each node and generating a tree, the resulting ensemble method gives us a classifier method.

These algorithms, MDT-1 and MDT-2 (Fig. 2), employ $f(w, p)$ — linear hyperplane with parameters w and p.

The proposed method

In this section, we present a new type of multivariate binary decision tree model. The primary difference from above and other existing algorithms is that we propose a new criterion for splitting nodes with piecewise functions and nonlinear transformations.

In general, all the existing tree models construct tree nodes in a linear mode to reach the domain. The proposed algorithm constructs a binary decision tree which contains hyperplanes at each node that separates the objects of different class objects. To construct hyperplanes, we use feedforward neural network with one hidden layer, also called as shallow neural networks. The algorithm is shown below.

Algorithm #1. NNDT

Input: Training dataset matrix D with known class labels set L

Output: An ensemble algorithm which is a binary tree with neural network, the split nodes storing the parameters w of neural network and the leaf nodes storing the class label of objects

1. Initialize Ψ: = D and empty set $X = \varnothing$
2. Extract objects with the same class label from Ψ and denote them as $\mathbf{X}$
3. **while** the set Ψ is not empty **do**

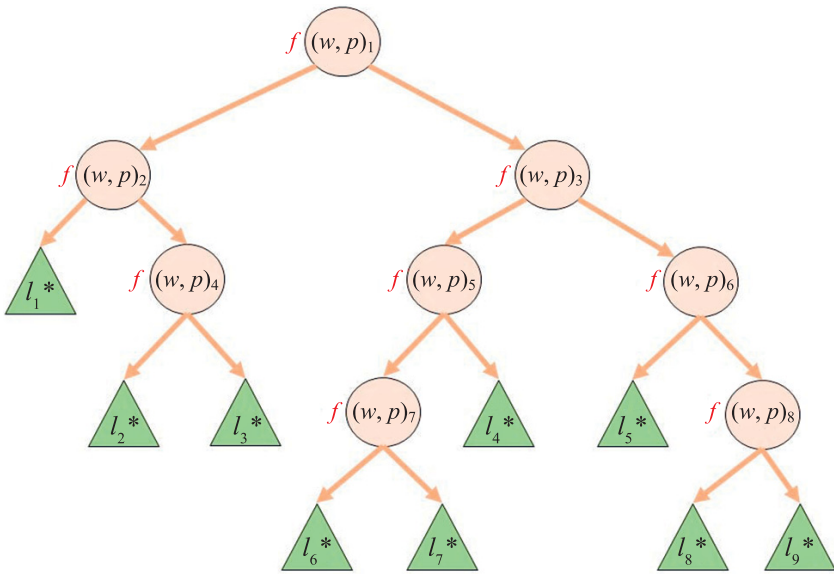

Fig. 2. Multivariate decision tree methods with linear hyperplane, MDT-1 and MDT-2

4. train a new neural network with a selected activation function where its hyperplane separates the objects of **X** and Ψ
5. **if** cross-validation condition holds **then**
6. create a leaf node and assign it to the majority class label of **X** exclude those objects from the set Ψ
7. **else**
8. create a child node with X_i from Ψ
9. **end if**
10. **end of while**

We can see the process of constructing decision tree of Algorithm #1 in Fig. 3.

In Fig. 3, the signs (+, –, ×) inside child nodes indicate labels of objects of the same class, l_1, l_k, and l_t — class labels which are assigned from terminal nodes.

Separating using activation functions

In general, MDT-1 and MDT-2 use unsupervised learning in order to construct tree model which leads to losing information when splitting dataset objects, and the resulting splitting structure is not enough clear. In MDT-1, constructing hyperplanes using randomly generating from the normal direction does not provide transparency. Moreover, using PCA, it's not the optimal solution for this problem relying on distribution of objects of multi class dataset. The shortcomings of the two methods lead us to consider another way to construct a new model for this problem.

Moreover, if we use neural network with activation functions and nonlinear transformation property, this leads to an increase in the generalization ability of the neural network.

In general, we propose to use shallow network with the following activation functions in hidden layers.

1. Linear and exponential (piece-wise linear unit) activation functions are given as follows:

$$f_{lrelu}(\mathbf{x}) = \begin{cases} x, \ if\, x > 0, \\ \alpha x, \ if\, x \le 0, \end{cases} \tag{2}$$

where x is input value for linear function ReLU $f_{lrelu}(\cdot)$.

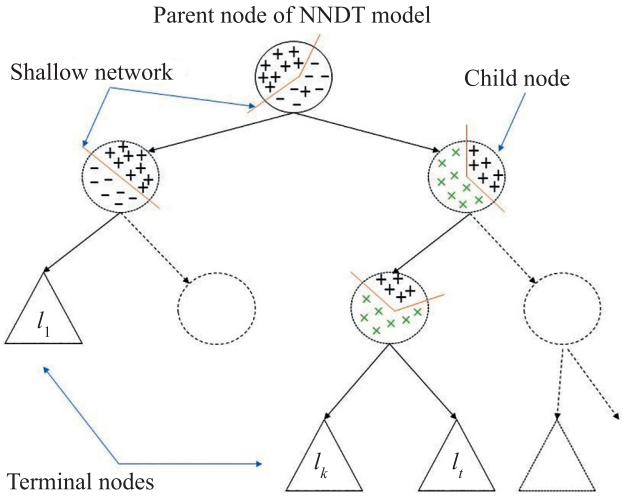

Fig. 3. NNDT model for constructing decision tree

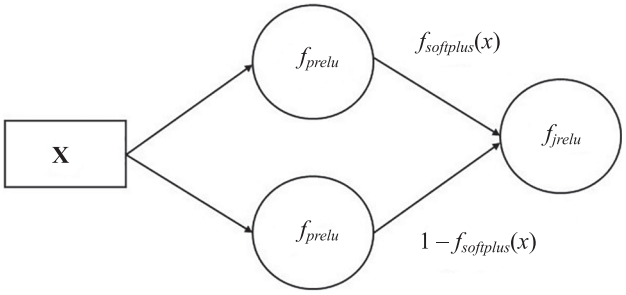

Fig. 4. Joined activation function

2. Exponential ReLU $f_{elu}(\cdot)$ is the activation function where α is a predefined parameter for controlling the angular coefficient.

$$f_{elu}(\mathbf{x}) = \begin{cases} x, & if\, x > 0, \\ \alpha(e^x - 1), & if\, x \le 0. \end{cases} \tag{3}$$

3. Joined activation function (Fig. 4) and its analytic form for this approach is given below:

$$f_{jrelu} = \beta f_{prelu}(\mathbf{x}) + (1 - \beta) f_{pelu}(\mathbf{x}), \tag{4}$$

where $\beta \in [0, 1]$ is a joining coefficient indicating the certain grouping of $f_{prelu}(\cdot)$ and $f_{pelu}(\cdot)$. The joining coefficient β is obtained from during the training process.

At each splitting node we can choose one of the activation functions in our neural network in decision tree. Rather than separating objects in the feature space with linear hyperplanes (MDT algorithms), we can use neural network which has ability of nonlinear transformations. Such activation functions are considered much more powerful when the classifying dataset has a very complex structure (high dimensional features space, multi-class datasets).

Numerical experiments

In this section, we provide the results of numerical experiments with proposed method and comparison analysis with other existing methods and algorithms. All experiments are tested on Python version 3.8 environment and Intel Core i7 10780H CPU, 2.20GHz processor with 16GB RAM memory. The datasets are downloaded from Machine Learning Repositories, they are publicly available online[1,2]. The datasets list is given in Table 1, it is used in numerical experiments.

Classification accuracy of NNDT models compared with the other methods and algorithms (C4.5, MDT-1, MDT-2 and CART) are given in Table 2. We can see that the proposed new model outcomes all the other models in accuracy. Moreover, if we use the proposed decision tree model with activation function which is given in (4), we can achieve more precise results. NNDT with (4) shows the

[1] UC Irvine Machine Learning Repository. Available at: https://archive.ics.uci.edu/ml/index.php (accessed: 01.07.2022).

[2] Machine Learning Repository. Available at: https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets (accessed: 01.07.2022).

Table 1. Information about datasets

Dataset	Number of		
	objects	attributes	classes
aloi	108000	128	1000
covtype	581012	54	7
mushrooms	8124	112	2
shuttle	43500	9	7
Wine	178	13	3
Spambase	4601	57	2
Hepatitis	155	19	2
Dry Bean Dataset	13611	17	7

highest accuracy compared to NNDT with (2) and NNDT with (3). This outperformance can be explained with a highly non-linearity of the activation function.

In the following Table 3, we can see comparison of training and classification time of the models. Below, we can see from Table 3 that most of the time C4.5 wins in the training process for nearly in all the datasets. Practically, this also affects the fact that univariate and multivariate decision tree models have different qualities for different cases. However, in Table 3, we can see that NNDT models show the fastest time in the classification among the compared with multivariate decision tree models. Moreover, our proposed models, especially NNDT with (4), show less time needed for classification compared with MDT-1 and MDT-2 multivariate decision models.

In all decision tree algorithms, the number of generated nodes plays important role as a main property. This property directly affects to the performance of the algorithm. In the following Table 4, we can see the number of splitting nodes generated in training process. The proposed model shows the least number of generated nodes, excluding C4.5 algorithm. However, NNDT model has greater accuracy compared to C4.5 (Table 2). Moreover, when we use the proposed decision tree model with activation function which is given in (4), NNDT algorithm generates less nodes. NNDT with (4) shows the less number of nodes compared to NNDT with (2) and NNDT with (3). Here, it is obtained with a highly non-linearity property of the activation function.

We provide numerical experiments of the proposed NNDT model with various activation functions and proposed joined activation function on Spambase, Hepatitis and mushrooms datasets. Firstly, we provide numerical experiments between piecewise linear activation function (ReLU), exponential activation function, and then with the proposed joined activation. First, we perform numerical experiments on the Spambase dataset. Secondly, we mainly focus on evaluating the performance effects of the joined activation function, and perform the comparison experiments with the other configurations on all datasets. In numerical experiments we employ our activation functions replacing $f_{relu}(\cdot)$ activation function in NNDT model which is given in Table 5. After configuration of NNDT models with the activation functions in Table 5, we train our models with the selected datasets. Five separate trials were carried out and the average value of the classification results was calculated.

Table 2. Classification accuracies, %

Dataset	CART	C4.5	MDT-1	MDT-2	NNDT with eq. (2)	NNDT with eq. (3)	NNDT with eq. (4)
aloi	75.5	5	48.02	77.8	98.2	99.1	**100**
covtype	92.8	42	85.6	91.06	92.5	93.6	**99.4**
mushrooms	99.5	99.7	100	100	100	100	**100**
shuttle	99.87	99.25	98.8	99.1	100	100	**100**
Wine	92	79	99.7	99.9	100	100	**100**
Spambase	65.5	45	92.5	98.9	99.5	99.4	**99.98**
Hepatitis	92.8	89.1	92.7	93.4	96.7	97.1	**99.1**
Dry Bean Dataset	62.2	49.4	89.78	91.2	98.8	98.9	**99.6**

Table 3. Training time comparison

Dataset	C4.5	MDT-1	MDT-2	NNDT with eq. (2)	NNDT with eq. (3)	NNDT with eq. (4)
aloi	$5.42 \cdot 10^{4}$	$4.27 \cdot 10^{3}$	$4.20 \cdot 10^{2}$	$3.10 \cdot 10^{2}$	$3.12 \cdot 10^{2}$	$3.91 \cdot 10^{2}$
covtype	$3.95 \cdot 10^{3}$	$1.25 \cdot 10^{4}$	$2.31 \cdot 10^{3}$	$0.91 \cdot 10^{3}$	$1.20 \cdot 10^{3}$	$1.91 \cdot 10^{3}$
mushrooms	$0.98 \cdot 10^{-1}$	$2.55 \cdot 10^{0}$	$7.25 \cdot 10^{-1}$	$1.21 \cdot 10^{-1}$	$1.22 \cdot 10^{-1}$	$1.25 \cdot 10^{-1}$
shuttle	$2.41 \cdot 10^{0}$	$4.40 \cdot 10^{0}$	$2.11 \cdot 10^{0}$	$1.23 \cdot 10^{0}$	$1.24 \cdot 10^{0}$	$2.00 \cdot 10^{0}$
Wine	$0.20 \cdot 10^{-1}$	$1.40 \cdot 10^{0}$	$2.25 \cdot 10^{-1}$	$0.55 \cdot 10^{-1}$	$0.55 \cdot 10^{-1}$	$0.91 \cdot 10^{-1}$
Spambase	$0.22 \cdot 10^{-1}$	$2.10 \cdot 10^{0}$	$4.25 \cdot 10^{-1}$	$0.48 \cdot 10^{-1}$	$0.47 \cdot 10^{-1}$	$0.95 \cdot 10^{-1}$
Hepatitis	$0.25 \cdot 10^{-1}$	$0.24 \cdot 10^{0}$	$0.39 \cdot 10^{-1}$	$0.28 \cdot 10^{-1}$	$0.31 \cdot 10^{-1}$	$0.35 \cdot 10^{-1}$
Dry Bean Dataset	$1.25 \cdot 10^{-1}$	$2.87 \cdot 10^{0}$	$3.20 \cdot 10^{-1}$	$0.81 \cdot 10^{-1}$	$0.79 \cdot 10^{-1}$	$1.25 \cdot 10^{-1}$

Table 4. Number of splitting nodes

Dataset	CART	C4.5	MDT-1	MDT-2	NNDT with eq. (2)	NNDT with eq. (3)	NNDT with eq. (4)
aloi	16492	78	60452	20598	19129	18561	11233
covtype	32152	86	162201	131305	7916	7920	7551
mushrooms	9	4	484	41	4	4	3
shuttle	15	58	904	527	13	13	11
Wine	21	59	102	76	15	14	8
Spambase	245	359	523	378	201	199	185
Hepatitis	32	45	41	35	15	13	9
Dry Bean Dataset	26	39	64	57	19	19	17

Table 5. Classification results of NNDT models on various activation functions.

Configuration of NNDT model	Classification Rates		
	Spambase dataset	Hepatitis dataset	Mushrooms dataset
$f_{relu}(\cdot)$	94.75	92.45	98.75
$f_{prelu}(\cdot)$	95.05	96.25	100
$f_{elu}(\cdot)$	99.40	97.40	100
$f_{jrelu}(\cdot)$ *with* $f_{prelu}(\cdot)/f_{pelu}(\cdot)$	99.98	99.15	100

Classification results of NNDT models between activation functions

First, we compare the results among basic activation functions and joined activation functions with configured NNDT models. We use basic activation functions, such as $f_{relu}(\cdot)$-non-trainable, $f_{lrelu}(\cdot)$-non-trainable and $f_{pelu}(\cdot)$-trainable, which are used mostly in the deep learning models. The numerical results in Table 6 with our proposed activation functions show superiority over non-trainable activation functions. The joined activation function shows better performance results than non-trainable activation functions on all datasets. Comparison on Spambase dataset shows that the joined activation function performs the best action and achieves an average improvement compared to other activation functions. We also determine that in the activation approach with trainable parameters, compared with training combination coefficients on the network layer, the performance of learning joining coefficients boosts on accuracy results

$$(f_{jrelu}(\cdot) \textit{ with } f_{prelu}(\cdot)/ f_{pelu}(\cdot) > f_{jrelu}(\cdot) \textit{ with } f_{lrelu}(\cdot)/ f_{elu}(\cdot))$$

enhancing the quality of trained hyper-parameters. Additionally, the performance achieved by the joined approach is almost consistently better than that achieved by the non-trainable approaches with constant coefficients on the same dataset. Overall, the trend of performance achieved by learning activation functions is almost

$$(f_{jrelu}(\cdot) \textit{ with } f_{prelu}(\cdot)/ f_{pelu}(\cdot) > f_{jrelu}(\cdot) \textit{ with } f_{lrelu}(\cdot)/ f_{elu}(\cdot)>.$$

Then we perform experiments to analyze the classification performance, and compare joined activation on other datasets. On every training epoch we can see the accuracy of every model and the proposed model (Fig. 5).

Five separate trials were carried out and average value of the classification results was calculated.

Table 6. Classification error with trainable and non-trainable activation functions of NNDT models. We run five separate trials and report average value of classification rates

Configuration of NNDT model	Classification Rates		
	Spambase dataset	Hepatitis dataset	Mushrooms dataset
$f_{jrelu}(\cdot)$ *with* $f_{lrelu}(\cdot)/f_{elu}(\cdot)$	99.98	99.15	100
$f_{jrelu}(\cdot)$ *with* $f_{prelu}(\cdot)/f_{pelu}(\cdot)$	99.25	99.10	100
$f_{jrelu}(\cdot)$ *with* $f_{lrelu}(\cdot)/f_{elu}(\cdot)$	99.10	98.50	100
$f_{jrelu}(\cdot)$ *with* $f_{prelu}(\cdot)/f_{pelu}(\cdot)$	99.10	98.20	100

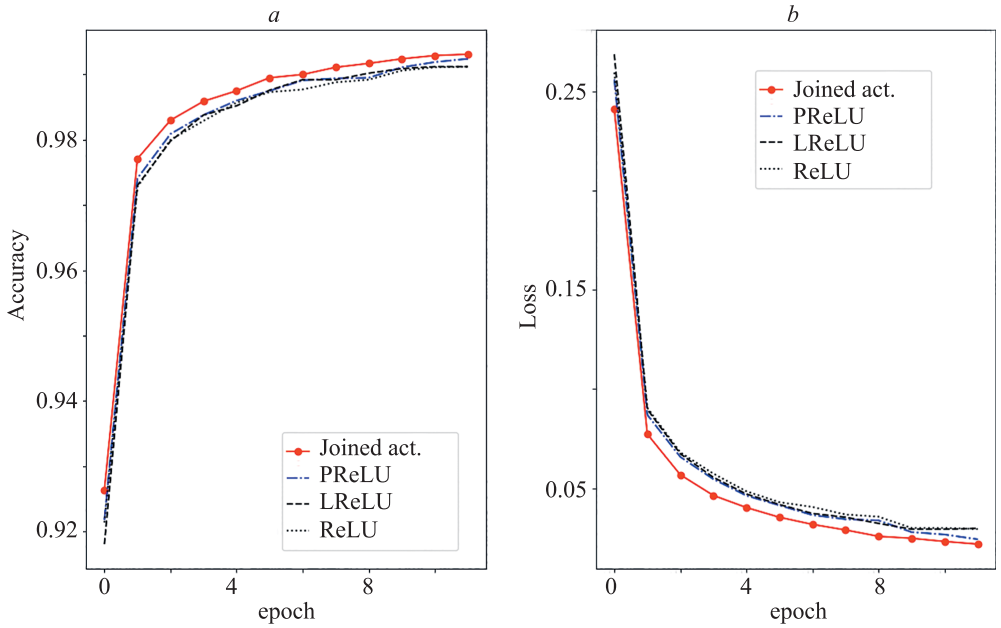

Fig. 5. Classification results of NNDT models between activation functions through epochs. Model accuracy (*a*), Model loss (*b*)

Conclusion

In this paper, we have proposed a novel ensemble algorithm based on multivariate decision tree and shallow neural network named as NNDT. NNDT generates nodes employing hyperplanes at each node, separating sample data objects. The proposed algorithm outperforms all the other decision tree algorithms. Moreover, NNDT algorithm provides high generalization ability, enhancing classification accuracy and performance.

In future researches, informative attribute selection (dimensionality reduction) methods are considered to generate nodes. Selecting informative attributes can lead to reduce the parameter sensitivity and can be obtained much higher generalization ability. The proposed NNDT decision tree model further can be employed to construct decision forest models.

References

1. Morala P., Cifuentes J.A., Lillo R.E., Ucar I. Towards a mathematical framework to inform neural network modelling via polynomial regression. *Neural Networks*, 2021, vol. 142, pp. 57–72. https://doi.org/10.1016/j.neunet.2021.04.036
2. Cao W., Mirjalili V., Raschka S. Rank consistent ordinal regression for neural networks with application to age estimation. *Pattern Recognition Letters*, 2020, vol. 140, pp. 325–331. https://doi.org/10.1016/j.patrec.2020.11.008
3. Messner E., Fediuk M., Swatek P., Scheidl S., Smolle-Jüttner F.M., Olschewski H., Pernkopf F. Multi-channel lung sound classification with convolutional recurrent neural networks. *Computers in Biology and Medicine*, 2020, vol. 122, pp. 103831. https://doi.org/10.1016/j.compbiomed.2020.103831
4. Youling L. A calibration method of computer vision system based on dual attention mechanism. *Image and Vision Computing*, 2020, vol. 103, pp. 104039. https://doi.org/10.1016/j.imavis.2020.104039
5. Palmerston J.B., Zhou Y., Chan H.M. Comparing biological and artificial vision systems: Network measures of functional connectivity. *Neuroscience Letters*, 2020, vol. 739, pp. 135407. https://doi.org/10.1016/j.neulet.2020.135407
6. Basha S.H.Sh., Dubey Sh.R., Pulabaigari V., Mukherjee S. Impact of fully connected layers on performance of convolutional neural networks for image classification. *Neurocomputing*, 2020, vol. 378, pp. 112–119. https://doi.org/10.1016/j.neucom.2019.10.008
7. Shuang K., Tan Y., Cai Zh., Sun Y. Natural language modeling with syntactic structure dependency. *Information Sciences*, 2020, vol. 523, pp. 220–233. https://doi.org/10.1016/j.ins.2020.03.022
8. Xu M. WITHDRAWN: Image processing system based on FPGA and convolutional neural network. *Microprocessors and Microsystems*, 2020, pp. 103379. https://doi.org/10.1016/j.micpro.2020.103379
9. Krizhevsky A., Sutskever I., Hinton G.E. Imagenet classification with deep convolutional neural networks. *Proc. of the 26th Annual*

Литература

1. Morala P., Cifuentes J.A., Lillo R.E., Ucar I. Towards a mathematical framework to inform neural network modelling via polynomial regression // Neural Networks. 2021. V. 142. P. 57–72. https://doi.org/10.1016/j.neunet.2021.04.036
2. Cao W., Mirjalili V., Raschka S. Rank consistent ordinal regression for neural networks with application to age estimation // Pattern Recognition Letters. 2020. V. 140. P. 325–331. https://doi.org/10.1016/j.patrec.2020.11.008
3. Messner E., Fediuk M., Swatek P., Scheidl S., Smolle-Jüttner F.M., Olschewski H., Pernkopf F. Multi-channel lung sound classification with convolutional recurrent neural networks // Computers in Biology and Medicine. 2020. V. 122. P. 103831. https://doi.org/10.1016/j.compbiomed.2020.103831
4. Youling L. A calibration method of computer vision system based on dual attention mechanism // Image and Vision Computing. 2020. V. 103. P. 104039. https://doi.org/10.1016/j.imavis.2020.104039
5. Palmerston J.B., Zhou Y., Chan H.M. Comparing biological and artificial vision systems: Network measures of functional connectivity // Neuroscience Letters. 2020. V. 739. P. 135407. https://doi.org/10.1016/j.neulet.2020.135407
6. Basha S.H.Sh., Dubey Sh.R., Pulabaigari V., Mukherjee S. Impact of fully connected layers on performance of convolutional neural networks for image classification // Neurocomputing. 2020. V. 378. P. 112–119. https://doi.org/10.1016/j.neucom.2019.10.008
7. Shuang K., Tan Y., Cai Zh., Sun Y. Natural language modeling with syntactic structure dependency // Information Sciences. 2020. V. 523. P. 220–233. https://doi.org/10.1016/j.ins.2020.03.022
8. Xu M. WITHDRAWN: Image processing system based on FPGA and convolutional neural network // Microprocessors and Microsystems. 2020. P. 103379. https://doi.org/10.1016/j.micpro.2020.103379
9. Krizhevsky A., Sutskever I., Hinton G.E. Imagenet classification with deep convolutional neural networks // Proc. of the 26th Annual

Conference on Neural Information Processing Systems (NIPS), 2012, pp. 1097–1105.
10. Goodfellow I., Bengio Y., Courville A. *Deep Learning*. MIT Press, 2016, 775 p.
11. Zhou Z.H., Feng J. Deep forest: Towards an alternative to deep neural networks. *Proc. of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 3553–3559. https://doi.org/10.24963/ijcai.2017/497
12. Frosst N., Hinton G. Distilling a neural network into a soft decision tree. *CEUR Workshop Proceedings*, 2018, vol. 2070.
13. Wan A., Dunlap L., Ho D., Yin J., Lee S., Jin H., Petryk S., Bargal S.A., Gonzalez J.E. NBDT: Neural-backed decision trees. *arXiv*, 2020, arXiv:2004.00221. https://doi.org/10.48550/arXiv.2004.00221
14. Pinto A., Pereira S., Rasteiro D.M., Silva C. Hierarchical brain tumour segmentation using extremely randomized trees. *Pattern Recognition*, 2018, vol. 82, pp. 105–117. https://doi.org/10.1016/j.patcog.2018.05.006
15. Vanli N.D., Sayin M.O., Mohaghegh N.M., Ozkan H., Kozat S.S. Nonlinear regression via incremental decision trees. *Pattern Recognition*, 2019, vol. 86, pp. 1–13. https://doi.org/10.1016/j.patcog.2018.08.014
16. Blanco-Justici A., Domingo-Ferrer J., Martínez S., Sánchez D. Machine learning explainability via microaggregation and shallow decision trees. *Knowledge-Based Systems*, 2020, vol. 194, pp. 105532. https://doi.org/10.1016/j.knosys.2020.105532
17. Kotsiantis S.B. Decision trees: a recent overview. *Artificial Intelligence Review*, 2013, vol. 39, no. 4, pp. 261–283. https://doi.org/10.1007/s10462-011-9272-4
18. Quinlan J.R. Induction of decision trees. *Machine Learning*, 1986, vol. 1, no. 1, pp. 81–106. https://doi.org/10.1023/A:1022643204877
19. Breiman L., Friedman J.H., Stone C.J., Olshen R.A. Classification and Regression Trees. Chapman & Hall/CRC, 1984.
20. Quinlan J.R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.
21. Chandra B., Paul Varghese P. Moving towards efficient decision tree construction. *Information Sciences*, 2009, vol. 179, no. 8, pp. 1059–1069. https://doi.org/10.1016/j.ins.2008.12.006
22. Wang F., Wang Q., Nie F., Yu W., Wang R. Efficient tree classifiers for large scale datasets. *Neurocomputing*, 2018, vol. 284, pp. 70–79. https://doi.org/10.1016/j.neucom.2017.12.061
23. Yildiz C., Alpaydin E. Omnivariate decision trees. *IEEE Transactions on Neural Networks*, 2001, vol. 12, no. 6, pp. 1539–1546. https://doi.org/10.1109/72.963795
24. Altinçay H. Decision trees using model ensemble-based nodes. *Pattern Recognition*, 2007, vol. 40, no. 12, pp. 3540–3551. https://doi.org/10.1016/j.patcog.2007.03.023
25. Kumar M.A., Gopal M. A hybrid SVM based decision tree. *Pattern Recognition*, 2010, vol. 43, no. 12, pp. 3977–3987. https://doi.org/10.1016/j.patcog.2010.06.010
26. Nie F., Zhu W., Li X. Decision Tree SVM: An extension of linear SVM for non-linear classification. *Neurocomputing*, 2020, vol. 401, pp. 153–159. https://doi.org/10.1016/j.neucom.2019.10.051

Conference on Neural Information Processing Systems (NIPS). 2012. P. 1097–1105.
10. Goodfellow I., Bengio Y., Courville A. Deep Learning. MIT Press, 2016. 775 p.
11. Zhou Z.H., Feng J. Deep forest: Towards an alternative to deep neural networks // Proc. of the 26th International Joint Conference on Artificial Intelligence (IJCAI). 2017. P. 3553–3559. https://doi.org/10.24963/ijcai.2017/497
12. Frosst N., Hinton G. Distilling a neural network into a soft decision tree // CEUR Workshop Proceedings. 2018. V. 2070.
13. Wan A., Dunlap L., Ho D., Yin J., Lee S., Jin H., Petryk S., Bargal S.A., Gonzalez J.E. NBDT: Neural-backed decision trees // arXiv. 2020. arXiv:2004.00221. https://doi.org/10.48550/arXiv.2004.00221
14. Pinto A., Pereira S., Rasteiro D.M., Silva C. Hierarchical brain tumour segmentation using extremely randomized trees // Pattern Recognition. 2018. V. 82. P. 105–117. https://doi.org/10.1016/j.patcog.2018.05.006
15. Vanli N.D., Sayin M.O., Mohaghegh N.M., Ozkan H., Kozat S.S. Nonlinear regression via incremental decision trees // Pattern Recognition. 2019. V. 86. P. 1–13. https://doi.org/10.1016/j.patcog.2018.08.014
16. Blanco-Justici A., Domingo-Ferrer J., Martínez S., Sánchez D. Machine learning explainability via microaggregation and shallow decision trees // Knowledge-Based Systems. 2020. V. 194. P. 105532. https://doi.org/10.1016/j.knosys.2020.105532
17. Kotsiantis S.B. Decision trees: a recent overview // Artificial Intelligence Review. 2013. V. 39. N 4. P. 261–283. https://doi.org/10.1007/s10462-011-9272-4
18. Quinlan J.R. Induction of decision trees // Machine Learning. 1986. V. 1. N 1. P. 81–106. https://doi.org/10.1023/A:1022643204877
19. Breiman L., Friedman J.H., Stone C.J., Olshen R.A. Classification and Regression Trees. Chapman & Hall/CRC, 1984.
20. Quinlan J.R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers Inc., 1993.
21. Chandra B., Paul Varghese P. Moving towards efficient decision tree construction // Information Sciences. 2009. V. 179. N 8. P. 1059–1069. https://doi.org/10.1016/j.ins.2008.12.006
22. Wang F., Wang Q., Nie F., Yu W., Wang R. Efficient tree classifiers for large scale datasets // Neurocomputing. 2018. V. 284. P. 70–79. https://doi.org/10.1016/j.neucom.2017.12.061
23. Yildiz C., Alpaydin E. Omnivariate decision trees // IEEE Transactions on Neural Networks. 2001. V. 12. N 6. P. 1539–1546. https://doi.org/10.1109/72.963795
24. Altinçay H. Decision trees using model ensemble-based nodes // Pattern Recognition. 2007. V. 40. N 12. P. 3540–3551. https://doi.org/10.1016/j.patcog.2007.03.023
25. Kumar M.A., Gopal M. A hybrid SVM based decision tree // Pattern Recognition. 2010. V. 43. N 12. P. 3977–3987. https://doi.org/10.1016/j.patcog.2010.06.010
26. Nie F., Zhu W., Li X. Decision Tree SVM: An extension of linear SVM for non-linear classification // Neurocomputing. 2020. V. 401. P. 153–159. https://doi.org/10.1016/j.neucom.2019.10.051

Authors

Avazjon R. Marakhimov — D. Sc., Professor, Rector, Termez State University, Termez, 190011, Uzbekistan, https://orcid.org/0000-0003-3735-6855, termizdu@umail.uz

Jabbarbergen K. Kudaybergenov — PhD, Lecturer, Tashkent University of Information Technologies Nukus branch named after Muhammad Al-Khwarizmi, Nukus, 230113, Uzbekistan, https://orcid.org/0000-0003-4494-6255, kjabbarbergen@gmail.com

Kabul K. Khudaybergenov — PhD, Lecturer, National University of Uzbekistan, Tashkent, 100174, Uzbekistan, https://orcid.org/0000-0001-8143-625X, kabul85@mail.ru

Ulugbek R. Ohundadaev — Basic Doctoral Student, National University of Uzbekistan, Tashkent, 100174, Uzbekistan, https://orcid.org/0000-0002-3240-6502, ulugbek_1122@mail.ru

Авторы

Марахимов Авазжон Рахимович — доктор технических наук, профессор, ректор, Термезский государственный университет, Термез, 190111, Узбекистан, https://orcid.org/0000-0003-3735-6855, termizdu@umail.uz

Кудайбергенов Жаббарберген Кадирбергенович – кандидат технических наук, преподаватель, Нукусский филиал Ташкентского университета информационных технологий имени Мухаммад ал-Хоразмий, Нукус, 230113, Узбекистан, https://orcid.org/0000-0003-4494-6255, kjabbarbergen@gmail.com

Худайбергенов Кабул Кадирбергенович — кандидат технических наук, преподаватель, Национальный университет Узбекистана, Ташкент, 100174, Узбекистан, https://orcid.org/0000-0001-8143-625X, kabul85@mail.ru

Охундадаев Улугбек Рахимжон угли — базовый докторант, Национальный университет Узбекистана, Ташкент, 100174, Узбекистан, https://orcid.org/0000-0002-3240-6502, ulugbek_1122@mail.ru

Received 11.04.2022
Approved after reviewing 06.06.2022
Accepted 14.07.2022

Статья поступила в редакцию 11.04.2022
Одобрена после рецензирования 06.06.2022
Принята к печати 14.07.2022