Available online at http://ijdea.srbiau.ac.ir

Int. J. Data Envelopment Analysis (ISSN 2345-458X)

Vol. 10, No. 4, Year 2022 Article ID IJDEA-00422, Pages 43-56
Research Article

International Journal of Data Envelopment Analysis

Science and Research Branch (IAU)

A DEA-neural network approach to solve binary classification problems

S. Kashanifar[1], M. Farahnak[*] [2]

[1]Department of industrial engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran

[2]Department of industrial engineering, Gazvin Branch, Islamic Azad University, Gazvin, Iran

Received 12 April 2022, Accepted 24 October 2022

Abstract

In this paper, we propose a new hybrid neural network including Data Envelopment Analysis(DEA) and Radial Basis Function Network (RBFN) for binary classification problems. In the supervised learning phase of the neural network, the additive model is used to learn the classification function and Gaussian Radial Basis Function (GRBF) is used in the unsupervised learning phase of the neural network. Compared with the existing RBFN-DEA model for solving classification problems, the proposed model has low CPU time and can be applied to solve classification problems with negative data.

Keywords: Data Envelopment Analysis, Binary classification, Radial Basis Function, Linear programming problem.

[*] Corresponding author: Email: mona_farahnak@yahoo.com

1. Introduction

Classification, a branch of artificial intelligence, is a scientific discipline in Machine Learning [1]. Classification normally refers to a supervised procedure. Classification is a procedure that learns to classify new instances based on learning from a training set of instances that have been properly labeled with the correct classes. All binary classification algorithms learn a classification function of the form $f: R^n \to \{0,1\}$. This function is then applied to new instances and its value represents the class to which the instance is classified, so, classification algorithms differ in the form of learning function. The common classification algorithms include Fisher Linear Discriminant [2,3], k-Nearest Neighbors [4-7], Decision Trees [8], Neural Networks [9], Naive Bayes [10,11], Support Vector Machine (SVM) [12], AdaBoost [13].

Another algorithm for solving classification problems is Data Envelopment Analysis [14]. DEA developed by Charnes et al. is a nonparametric methodology for evaluating the performance of a group of Decision Making Units (DMUs) which use multiple inputs to produce multiple outputs. DEA successfully divides them into two categories; efficient DMUs and inefficient DMUs, so DEA can be used to solve binary classification problems [15].

The DEA models so far used to solve binary classification problems are CCR and BCC models. For solving classification problems with the BCC model, DMUs must have a monotonicity property and inputs should be non-negative. Some classification problems include negative data does not satisfy in monotonicity property, in this case, DEA can not apply for solving this problems itself, so Pendharkar (2011) for solving these drawbacks used radial basic function neural network (RBFN) and proposed a hybrid RBFN-DEA neural network for solving binary classification problems [16]. For more information abaut DEA-NN model, see [17-25].

In this paper, we combine additive the model in DEA with RBFN and introduce a new model for solving binary classification problems.

The proposed model has a lower CPU-time and more accuracy with respect to the RBFN-DEA model, furthermore our model can be applied to solving linear separable classification problems with negative data, in this problem, we do not need to apply RBFN to generate positive data. This paper is divided into five sections. In the next section preliminary information is introduced to facilitate later discussions. In section III, we present the proposed model and describe its properties. In section IV, illustrative examples are discussed. In section V, we present the results of these research.

2. Background

In this section, we introduce the related definitions and the additive model for later discussion in the next section.

Definition1: Monotonicity property:

If a DMU has a higher value of attributes then it belongs to a certain class with a higher probability and vice versa [26].

Theorem 1. Cover's Theorem:

"A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space" [26, 27].

Definition 2: Basis Functions and Feature Space:

Let $X = \{x_1, \dots, x_n\}$ be a set of n vectors in m-dimensional space, each of which is assigned to be one of two classes, A and B. Define a function $\phi(x)$ as $\phi(x) = [\phi_1(x), \dots, \varphi_r(x)]$, that $\phi_i(x): R^m \to R^r (i = 1, \dots, m)$ is called basic functions and the space spanned by a set of basis functions $\{\phi_i\}_{i=1}^r$ is called feature space.

A dichotomy $\{A, B\}$ of X is said to be ϕ-separable if there exists $w \in R^r$ such that

$$\begin{Bmatrix} w^t\phi(x) > 0 & if\ x \in \text{A} \\ w^t\phi(x) < 0 & if\ x \in \text{B} \end{Bmatrix}$$

The separating surface is given by $w^t\phi(x) = 0$.

In this paper, we use the following basic function called radial basic function [19]:

$$\phi_i(x) = \exp^{\frac{-\|x-\mu_i\|}{2\sigma^2}} \qquad i = 1,\dots,r$$

Where $\sigma = \frac{d_{max}}{2r}$, that d_{max} is the maximum distance between chosen $\mu_i(i = 1,\dots,r)$ and maximum of r can be equal to 5. $\mu_i\ (i = 1,\dots,r)$ can be initialized randomly by vectors in X or determined using cluster analysis approaches [27, 28].

Definition 3: Radial Basis Function Networks (RBFNs):

RBFNs have three layers. The hidden layer applies a nonlinear transformation from the input space to the hidden space. The hidden units use radial basis functions. The output layer applies a linear transformation from the hidden space to the output space [29, 30]. RBFNs have two parts for learning, In part I, from the input layer to the hidden layer use unsupervised learning and in part II, from the hidden layer to the output layer use supervised learning. RBFNs can be used for pattern classification [31], function approximation and control [32, 33].

Definition 4: Additive model in DEA:

Suppose that there are n Decision Making Units (DMU_s) to be evaluated in terms of m inputs and s outputs. Let $x_{ij}\ (i = 1,\dots,m)$ and $y_{rj}\ (r = 1,\dots,s)$ be the input and output values of $DMU_j\ (j = 1,\dots,n)$. There are several types of additive models, from which we select the following form in terms of $DMU_p\ (p = 1,\dots,n)$: (1)

$$Max \quad z = \sum_{i=1}^{m} S_i^+ + \sum_{r=1}^{s} S_r^-$$
$$S.t. \quad \sum_{j=1}^{n} \lambda_j x_{ij} + S_i^+ = x_{ip}, i = 1,\dots,m$$
$$\sum_{j=1}^{n} \lambda_j y_{rj} - S_r^- = y_{rp}, r = 1,\dots,s$$
$$\sum_{j=1}^{n} \lambda = 1$$
$$\lambda \geq 0$$
$$S_i^+ \geq 0 \quad i = 1,\dots,m$$
$$S_r^- \geq 0 \quad r = 1,\dots,s$$

(x_p, y_p) is evaluated DMU [14].

3. RBFN-ADD neural network

The RBFN-DEA model of Pendharkar (2011) motivates us to propose a new model to solve binary classification problems [16]. The proposed RBFN-ADD model has two parts, in part I, input data using Gaussian Radial Basis Function (GRBF)are transferred to high-dimensional space (feature space) that in this space can be linear separable with high probability by cover's theorem [26, 27]. Using GRBF, negative data convert to non-negative data in the feature space [16].

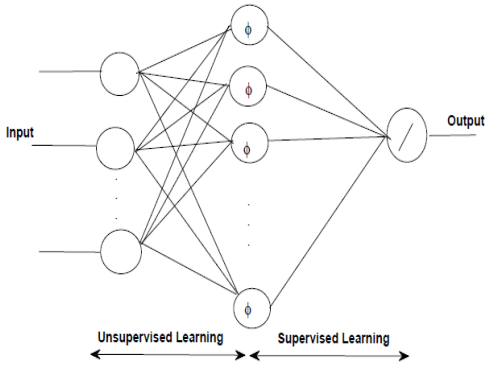

Fig.1: RBFNs structure.

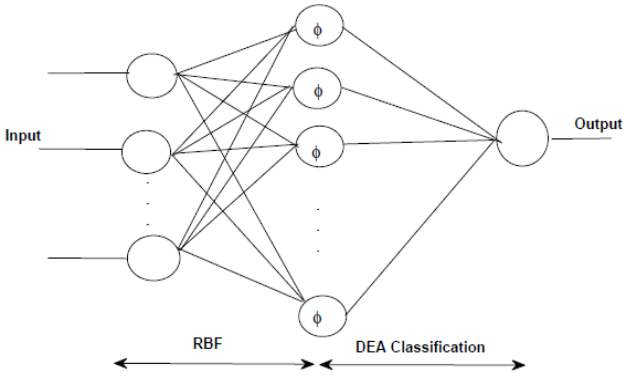

Fig.2:RBFN-ADDmodel structure.

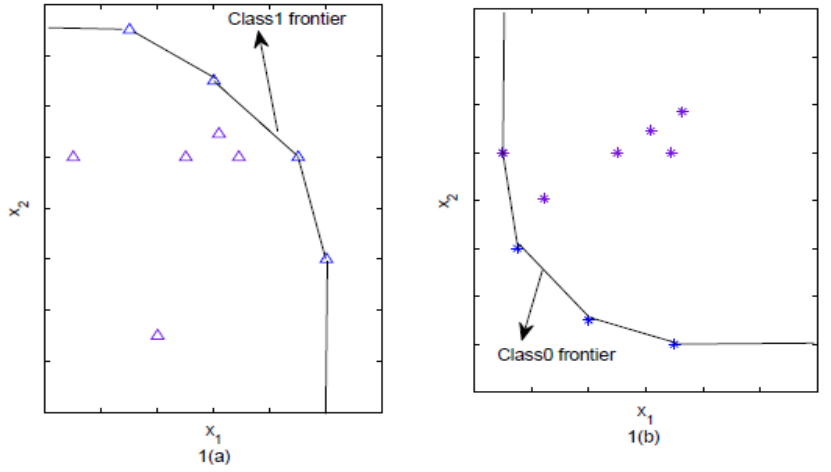

Fig.3: 3(a) Behavior of linear programming (3) in the training part of RBFN-ADD model to generate class 1 frontier. 3(b) Behavior of linear programming (2) in training part of RBFN-ADD model to generate class 0 frontier.

In the DEA part, we can use additive model. We apply the following models to develop a classification hyperplane for class 0 and class1:

$$Max \;\; z = \sum_{i=1}^{m} S_i^- \qquad (2)$$
$$s.t. \;\; \sum_{j=1}^{n} \lambda_j x_{ij} + S_i^- = x_{ip}, i = 1,\dots,m$$
$$\sum_{j=1}^{n} \lambda_j = 1$$
$$\lambda_j \geq 0\,, \;\; j = 1,\dots,n$$
$$S_i^- \geq 0 \;\; , i = 1,\dots,m$$

$$Max \;\; z = \sum_{i=1}^{m} S_i^+ \qquad (3)$$
$$s.t. \;\; \sum_{j=1}^{n} \lambda_j x_{ij} - S_i^+ = x_{ip} \;, i = 1,\dots,m$$
$$\sum_{j=1}^{n} \lambda_j = 1$$
$$\lambda_j \geq 0 \quad j = 1,\dots,n$$
$$S_i^+ \geq 0 \;\; i = 1,\dots,m$$

where $x_j \in R^m \; (j = 1,\dots,n)$ are training data.Suppose the training data set consists of *n* DMUs that k DMUs belong to class 0 and the rest belongs to class 1. We use model (2) to generate the class 0 frontier and model (3) to generate the class 1 frontier in the training part of our model. Since data have a monotonicity property, suppose data far from the origin belongs to class 0 and data close to the origin belongs to class 1. To determine which class is closer to theory gin in feature space, the Euclidean norm of the average vector in each class can be calculated in the

procedure for solving classification problems with minimum error for identifying test data belonging to class0 is the below form:

- For training data use only the cases from class 0.
- Using the linear programming (2) determine the efficient set of cases from the class 0, A^*, so determine the class 0 frontier.
- Take a unit from the test data, training data units from the efficient set A^* and solve the linear programming (4).
- If model (4) has a feasible solution then test data belongs to class 0 otherwise it belongs to class 1.

Figure 3(b) shows the classification hyperplane obtained by members of the efficient setA^*.

$$
\begin{aligned}
Min \quad & z_0 = \sum_{i=1}^{m} S_i^- && (4)\\
s.t. \quad & \sum_{j\in A^*} \lambda_j x_{ij} + S_i^- = x_i^{test},\ i = 1,\dots,\mathrm{m}\\
& \sum_{j\in A^*} \lambda_j = 1\\
& \lambda_j \geq 0 \quad j \in A^*\\
& S_i^- \geq 0 \quad i = 1,\dots,\mathrm{m}
\end{aligned}
$$

4. Numerical examples

In order to demonstrate the effectiveness and efficiency of the proposed model, in this section, we discuss the results obtained from the proposed model and RBFN-DEA model through two examples. Criteria NT0, NT1 and NNA RBFN-DEA model is found in Pendharkar (2011).

Example 1. In this example, we compare the performance of the additive model and RBFN-DEA model [16] with negative and nonnegative values. We generate our training and test data sets using three normal distributions with means of -1, -5 and -8. The standard deviations for distribution with means -5 and -8 are equal to one and with mean -1 equal to 2. The examples that were generated from normal distributions with means of -5 and -8 were labeled as belonging to class 1, and the examples that were generated from a normal distribution with a mean of -1 were labeled as belonging to class 0.

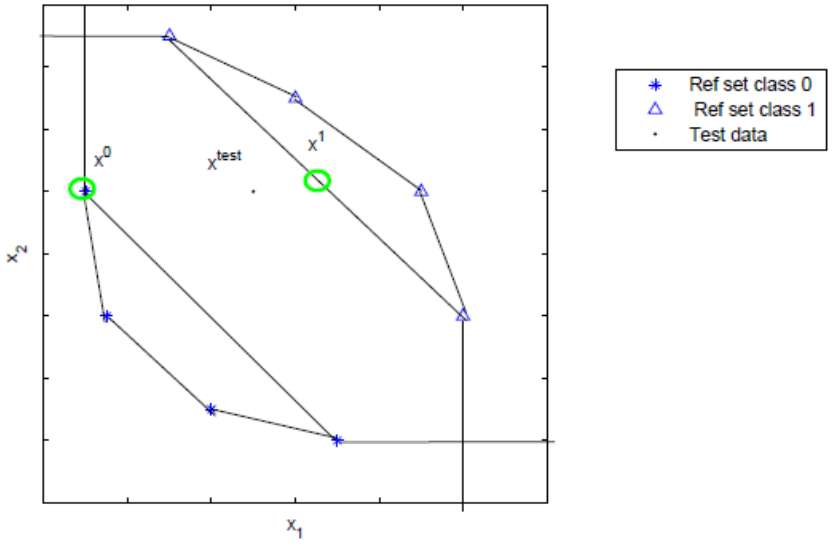

Fig.4: Behavior of linear programming programs (4) and (5).

Table.1: A training data set and Testing data set in example 1.

No	Class	Training data set		Testing data set	
		Attribute1	Attribute2	Attribute1	Attribute2
1	1	-4.7103	-4.3597	-4.5005	-4.2237
2	1	-5.3769	-7.0505	-3.7048	-5.4585
3	1	-3.2888	-5.9843	-5.4779	-5.1087
4	1	-4.0685	-4.7478	-4.1814	-5.4877
5	1	-6.4098	-5.708	-5.254	-5.4015
6	1	-4.488	-5.5647	-5.8325	-2.8829
7	1	-3.7755	-5.4872	-7.0353	-5.4414
8	1	-4.3283	-4.9362	-5.4744	-4.1605
9	1	-4.6692	-5.1205	-6.2172	-2.6892
10	1	-3.4518	-6.2368	-4.8427	-4.8349
11	0	-1.4164	3.144	-1.8805	0.3128
12	0	-1.9457	-0.3928	0.6626	0.9091
13	0	-1.0674	-0.1396	-0.7842	-1.2974
14	0	-2.0278	-1.9569	-2.6745	-2.4704
15	0	-0.6361	0.2279	-2.0132	-1.3974
16	0	-4.4317	-1.1481	-0.2282	-0.7301
17	0	1.4957	-2.0156	-1.0938	-4.537
18	0	-2.1321	0.844	0.3055	-1.5289
19	0	-4.0836	-1.2838	-0.642	-0.9577
20	0	-1.0343	-0.0087	1.0751	-0.283
21	1	-9.5943	-7.8686	-6.2367	-8.7376
22	1	-9.8711	-9.1471	-5.9822	-8.567
23	1	-9.0726	-8.6674	-9.2913	-6.8093
24	1	-8.1561	-7.7639	-9.5852	-7.8546
25	1	-6.6442	-7.898	-9.4142	-8.5579
26	1	-9.4063	-9.1122	-7.6845	-9.5352
27	1	-8.7654	-9.1373	-11.3198	-8.6381
28	1	-7.293	-9.2545	-6.4947	-8.405
29	1	-9.0533	-9.2209	-7.1127	-7.721
30	1	-7.387	-8.7511	-6.1414	-7.7317

Table.2: Training data set and Testing data set after use of RBF in example 1.

No	Class	Training data set					Testing data set				
		Attribute1	Attribute 2	Attribute3	Attribute4	Attribute5	Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5
1	1	0.3087	0.667	0.2653	0.7351	0.6019	0.1485	0.6639	0.4778	0.713	0.5441
2	1	0.0948	0.3134	0.566	0.336	0.93	0.1397	0.5716	0.4246	0.8013	0.6511
3	1	0.2406	0.5734	0.2555	0.4918	0.6276	0.2602	0.4747	0.3092	0.5684	0.7026
4	1	0.3195	0.6885	0.2464	0.6875	0.5963	0.1721	0.5383	0.3848	0.7396	0.6923
5	1	0.1175	0.3538	0.5469	0.4583	0.8594	0.2551	0.4644	0.3056	0.5935	0.7372
6	1	0.2207	0.5513	0.3436	0.5614	0.7297	0.1608	0.6155	0.4124	0.4829	0.3928
7	1	0.2651	0.616	0.2712	0.5743	0.645	0.4379	0.3007	0.1716	0.3521	0.7302
8	1	0.2841	0.6425	0.2807	0.6581	0.6431	0.2087	0.5673	0.3786	0.572	0.5663
9	1	0.2463	0.5884	0.3246	0.6248	0.6958	0.1694	0.5718	0.3734	0.4243	0.3679

10	1	0.2117	0.5298	0.2843	0.4591	0.6674	0.1958	0.5639	0.3889	0.6645	0.644
11	0	0.7524	0.4386	0.0014	0.4342	0.0095	0.0075	0.9185	0.913	0.4977	0.063
12	0	0.9365	0.9263	0.0181	0.8458	0.0865	0.0011	0.6972	0.8608	0.3882	0.0208
13	0	0.99	0.8764	0.0104	0.7226	0.0578	0.0085	0.957	0.9961	0.7363	0.0994
14	0	0.811	1	0.0412	0.8641	0.1707	0.0384	0.9551	0.8307	0.8223	0.249
15	0	1	0.811	0.0067	0.6424	0.0412	0.018	1	0.9418	0.734	0.1436
16	0	0.6011	0.818	0.0671	0.9957	0.2069	0.0047	0.9004	0.9936	0.6437	0.0651
17	0	0.7415	0.6786	0.0061	0.3721	0.0429	0.0299	0.734	0.6865	1	0.3135
18	0	0.9215	0.7825	0.0095	0.7708	0.0487	0.0047	0.8557	0.9652	0.7275	0.0762
19	0	0.6424	0.8641	0.0647	1	0.2079	0.0067	0.9418	1	0.6865	0.0823
20	0	0.9933	0.8613	0.0095	0.711	0.0536	0.0016	0.7323	0.9063	0.5174	0.0326
21	1	0.0105	0.0562	0.9359	0.1	0.762	0.4738	0.1258	0.0704	0.2796	0.9973
22	1	0.0045	0.0292	0.9792	0.051	0.6881	0.4389	0.1436	0.0823	0.3135	1
23	1	0.0092	0.052	0.9905	0.0838	0.8166	0.8061	0.0928	0.0428	0.1235	0.6665
24	1	0.0233	0.108	0.9126	0.1606	0.9306	0.9006	0.0571	0.0251	0.0905	0.6772
25	1	0.0412	0.1707	0.7899	0.2079	1	0.9002	0.0467	0.0204	0.0848	0.7115
26	1	0.0059	0.0369	0.9957	0.0609	0.7526	0.6668	0.0582	0.0284	0.1384	0.8951
27	1	0.0082	0.0484	0.9972	0.0735	0.8282	1	0.018	0.0067	0.0299	0.4389
28	1	0.0151	0.0798	0.9077	0.0997	0.9318	0.5094	0.1354	0.0748	0.2793	0.9917
29	1	0.0067	0.0412	1	0.0647	0.7899	0.5852	0.1485	0.0795	0.2618	0.944
30	1	0.0194	0.0965	0.9107	0.1247	0.9608	0.4499	0.1916	0.1108	0.3565	0.9793

Table.3: comparing the performance of the proposed model and RFN-DEA model in terms of NT0, NT1 and NNA in the example1.

No	Class	RBFN-ADD model			RBFN-DEA model		
		NT0	NT1	NNA	NTIIEM	NTIEM	NNA
1	1	1	0	1	1	0	1
2	1	1	1	1	1	0	1
3	1	1	1	1	1	0	1
4	1	1	1	1	1	0	0
5	1	1	1	1	1	0	0
6	1	1	0	0	1	0	1
7	1	1	1	1	1	0	1
8	1	1	0	0	1	1	1
9	1	1	0	1	1	1	1
10	1	1	1	1	1	0	1
11	0	0	0	0	1	0	0
12	0	0	0	0	1	0	0
13	0	0	0	0	1	0	1
14	0	1	0	1	1	0	0
15	0	0	0	0	1	0	1
16	0	0	0	0	1	0	1
17	0	1	0	0	1	0	1
18	0	0	0	0	1	0	0
19	0	0	0	0	1	0	1
20	0	0	0	0	1	0	0
21	1	1	1	1	1	0	0
22	1	1	1	1	1	0	0

23	1	1	1	1	1	0	0
24	1	1	1	1	1	0	1
25	1	1	1	1	1	0	0
26	1	1	1	1	1	0	0
27	1	1	1	1	1	0	1
28	1	1	1	1	1	0	1
29	1	1	1	1	1	0	1
30	1	1	1	1	1	0	1
-	Result	93%	86%	90%	66%	40%	60%

Table4.: Comparing the performance of the proposed model and the Pendharkar model in terms of CPU time in example 1.

Model	CPU time
RBFN-ADD	1.031250
RBFN-DEA	6.625000

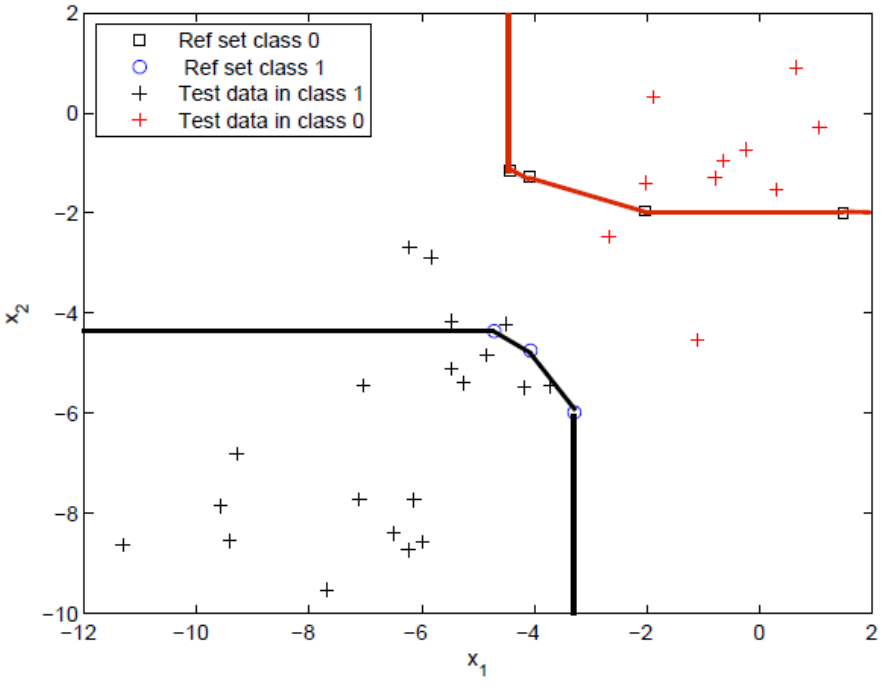

Fig.5: Behavior of NT0 and NT1 in example 1.

Example 2. In this example, we compare the performance of the proposed model and the RBFN-DEA model [16] using data converted by RBF. We generate our training and test data sets using three normal distributions with means of 1, 0 and -1. The standard deviations for all the distributions are considered equal to one. The examples that were generated from normal distributions with means of 1 and -1 were labeled as belonging to class 1, and the examples that were generated from a normal distribution with a mean of 0 were labeled as belonging to class 0.

Table.5: Training data set and Testing data set in example 2.

No	Class	Training data set		Testing data set	
		Attribute1	Attribute2	Attribute1	Attribute2
1	1	-0.3346	0.8579	3.584	1.2426
2	1	1.2393	1.3487	0.7811	0.3243
3	1	1.0545	1.8025	1.1598	3.3752
4	1	2.0268	1.6545	-0.26	-0.9992
5	1	1.4598	2.4093	2.0197	2.594
6	1	2.6394	1.5504	-0.9164	2.0155
7	1	-0.5144	0.9438	1.2827	0.6769
8	1	0.8262	1.3032	0.0963	1.5636
9	1	0.6107	0.7013	1.4808	2.352
10	1	1.5579	0.0616	-0.9528	1.845
11	0	0.6399	-1.3433	-0.1572	1.6539
12	0	1.0121	-1.2212	-1.0263	-0.0353
13	0	-0.6382	-0.4795	0.4599	0.3926
14	0	-0.2699	-0.5724	-0.5598	-0.2012
15	0	0.9175	-0.0485	0.8145	-1.3405
16	0	0.7324	1.421	-0.5583	-0.2888
17	0	-0.9144	1.1108	-1.6351	-0.0835
18	0	0.1638	0.6545	0.6964	-0.5615
19	0	-0.1301	-0.3713	0.3668	0.1142
20	0	0.0309	-0.971	0.2383	-0.3099
21	1	-1.9996	-3.442	-0.3388	-0.3671
22	1	-1.6587	-1.5397	-1.3173	-2.3894
23	1	-0.4759	-1.4906	1.1673	-1.3443
24	1	0.6849	0.5113	-0.6708	-0.5433
25	1	-1.3515	-1.5649	-1.1455	0.5424
26	1	-1.8397	-1.7603	-0.4769	-1.7101
27	1	0.0011	-3.1554	-1.3849	-1.535
28	1	0.179	-2.2486	-0.1344	-1.5353
29	1	-1.3078	-1.953	-1.6754	-0.3577
30	1	-0.5804	-1.0938	-0.1527	-2.832

Table.6: Training data set and Testing data set after use of RBF in example 2.

No	Class	Training data set			Testing data set		
		Attribute1	Attribute2	Attribute3	Attribute1	Attribute2	Attribute3
1	1	0.1149	0.9068	0.3568	0.0022	0.0001	0.0187
2	1	0.0142	0.3173	0.6506	0.5174	0.1485	0.3938
3	1	0.0081	0.3448	0.448	0.0067	0.0019	0.0008
4	1	0.0027	0.1122	0.5096	0.7938	0.4201	0.5041
5	1	0.0015	0.1669	0.2593	0.0101	0.0015	0.0058
6	1	0.0011	0.0435	0.4369	0.2016	0.2094	0.007
7	1	0.1102	0.9551	0.2893	0.2662	0.0557	0.272
8	1	0.0246	0.4725	0.6018	0.3243	0.163	0.0473
9	1	0.0726	0.5435	0.7266	0.0336	0.007	0.0126
10	1	0.0498	0.1714	1	0.2518	0.2646	0.0095
11	0	0.3612	0.1271	0.5023	0.3183	0.1915	0.0329

12	0	0.2353	0.1068	0.6218	0.9251	0.8882	0.1258
13	0	0.5271	0.5289	0.2863	0.6425	0.2308	0.3271
14	0	0.4822	0.4519	0.4005	1	0.6896	0.2559
15	0	0.1228	0.3169	0.9019	0.3633	0.09	0.9612
16	0	0.0224	0.5033	0.5388	0.9976	0.6827	0.2725
17	0	0.097	1	0.1714	0.6896	1	0.0498
18	0	0.1117	0.7152	0.5706	0.5812	0.1654	0.7671
19	0	0.3865	0.5028	0.476	0.7376	0.2765	0.415
20	0	0.5097	0.2785	0.4357	0.8137	0.3226	0.5411
21	1	0.5173	0.0047	0.0023	0.976	0.5715	0.3592
22	1	0.9307	0.1568	0.0426	0.182	0.1788	0.0994
23	1	0.8013	0.1824	0.2018	0.2559	0.0498	1
24	1	0.0858	0.4901	0.79	0.9597	0.6959	0.2788
25	1	0.9634	0.1658	0.0661	0.7523	0.8183	0.059
26	1	0.9247	0.1081	0.0264	0.4841	0.2817	0.406
27	1	0.4619	0.0095	0.044	0.4577	0.5019	0.1248
28	1	0.5701	0.0473	0.1704	0.5364	0.2503	0.577
29	1	1	0.097	0.0498	0.6682	0.9759	0.0563
30	1	0.7336	0.2965	0.2359	0.1053	0.0451	0.2846

Table.7: Comparing the performance of the proposed model and RBFN-DEA model in terms of NT0, NT1 and NNA in example 2.

No	Class	RBFN-ADD model			RBFN-DEA model		
		NT0	NT1	NNA	NTIIEM	NTIEM	NNA
1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1
4	1	0	0	0	0	0	0
5	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1
9	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1
11	0	1	1	1	1	1	1
12	0	1	0	0	1	0	0
13	0	1	1	1	1	1	1
14	0	0	0	0	0	0	0
15	0	1	0	0	1	0	1
16	0	0	0	0	0	0	0
17	0	1	0	0	1	0	1
18	0	0	0	0	0	0	0
19	0	1	0	0	1	0	0
20	0	0	0	0	0	0	0
21	1	0	0	0	0	0	0
22	1	1	1	1	1	1	1
23	1	1	0	0	1	0	0

24	1	0	0	0	0	0	0
25	1	1	0	0	1	0	0
26	1	1	1	1	1	1	1
27	1	1	1	1	1	1	1
28	1	0	0	0	0	0	0
29	1	1	0	0	1	0	0
30	1	1	1	1	1	1	1
-	Result	66%	70%	70%	66%	70%	60%

Table.8: Comparing the performance of the proposed model and the Pendharkar model in terms of CPU time in example 2.

Model	CPU time
RBFN-ADD	1.125000
RBFN-DEA	4.171875

5. Conclusion

In this paper, we have presented a novel RBFN-ADD neural network for solving binary classification problems. We use the additive model in the DEA section for the RBFN-ADD model and proposed new criteria for NNA in the RBFN-ADD model. We compare the performance of the proposed model and the RBFN-DEA model in terms of accuracy and CPU time. Numerical results show RBFN-ADD model has well performance with respect to RBFN-DEA model, furthermore proposed model can solve binary classification problems with negative value.

References

[1] Mitchell, T., (1997). "Pattern classification and Scene Analysis", *McG raw Hill*.

[2] Fisher,R., (1963). "The Use of Multiple Measurements in Taxonomic Problems", *Annals of Eugenics*; 7: 179—188.

[3] Morrison, D.F., (1990). "Multivariate Statistical Methods", *McGraw-Hill*.

[4] Bremner,D., E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin and G. Toussaint, (2005). "Output-sensitive algorithms for computing nearest-neighbor decision boundaries", *Discrete and Computational Geometry*; 33 (4): 593–604.

[5] Terrell,D. G. and D. W. Scott,(1992). "Variable kernel density estimation", *Annals of Statistics*; 20: 1236-65.

[6] Mills,P., (2011). "Efficient statistical classification of satellite measurements". *International Journal of Remote Sensing.*

[7] Hall, P., B. U. Park and R. J. Samworth, (2008). "Choice of neighbor order in nearest-neighbor classification", *Annals of Statistics*; 36: 2135–52.

[8] Quinlan, J. R., (1986). "Induction to decision tree, Machine Learning", *Kluwer Academic Press*; 81-106.

[9] Krogh, A., J. Hertz and R. G. Palmer, (1991). "Introduction to the theory of neural computation", *Perseus* .

[10] Kantardzic, M(2003). Data Mining - Concepts, Models, Methods, and Algorithms, IEEE Press, Wiley-Interscience.

[11] Han,J. and M. Kamber, (2006). "Data Mining: Concepts and Techniques", *Elsevier*.

[12] Press,W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, (2007). "Section 16.5. Support Vector Machines, Numerical Recipes: The Art of Scientific Computing (3rd ed.)", *New York: Cambridge University Press*.

[13] Freund, Y., R. E. Schapire, (1999). "A short intro duction to boosting", *Journal of Japanese Society for Artificial Intelligence*; 5(14): 771–80.

[14] Cooper,W. W., L. M. Seiford and K. Tone, (2007). "Introduction to data envelopment analysis and its uses: with DEA-solver software and references", *Springer*.

[15] Charnes, A., W. W. Cooper and E. Rhodes, (1978). "Measuring the efficiency of decision making units",*European Journal of Operational Research* ; 2(6):429–44.

[16] Pendharkar, P. C., (2011). "A hybrid radial basis function and data envelopment analysis neural network for classification", *Computers and Operations Research*; 38: 256–66.

[17] Wu, D. D., Yang, Z., & Liang, L. (2006). Using DEA-neural network approach to evaluate branch efficiency of a large Canadian bank. *Expert systems with applications*, *31*(1), 108-115.

[18] Aslani, G., MOUMENI, M. S., Malek, A., & Ghorbani, F. (2009). Bank efficiency evaluation using a neural network-DEA method.

[19] Lotfi, F. H., Jahanshahloo, G. R., Givehchi, S., & Vaez-Ghasemi, M.

(2013). Using DEA-neural network approach to solve binary classification problems. *Journal of Data Envelopment Analysis and Decision Science, 2013*, 1-12.

[20] Karamali, L., Memariani, A., Jahanshahloo, G. R., & Rostamy-malkhalifeh, M. (2013). Benchmarking by an Integrated Data Envelopment Analysis-Artificial Neural Network Algorithm. *J. Basic. Appl. Sci. Res*, *3*(6), 892-898.

[21] Hanafizadeh, P., Khedmatgozar, H. R., Emrouznejad, A., & Derakhshan, M. (2014). Neural network DEA for measuring the efficiency of mutual funds. *International journal of applied decision sciences*, *7*(3), 255-269.

[22] Hanafizadeh, P., Khedmatgozar, H. R., Emrouznejad, A., & Derakhshan, M. (2014). Neural network DEA for measuring the efficiency of mutual funds. *International journal of applied decision sciences*, *7*(3), 255-269.

[23] Shokrollahpour, E., Hosseinzadeh Lotfi, F., & Zandieh, M. (2016). An integrated data envelopment analysis–artificial neural network approach for benchmarking of bank branches. *Journal of Industrial Engineering International*, *12*, 137-143.

[24] Rostamy-Malkhalifeh, M., Amiri, M., & Mehrkam, M. (2021). Predicting financial statement fraud using fuzzy neural networks. *Advances in Mathematical Finance and Applications*, *6*(1), 137-145.

[25] Kordnoori, S., Mostafaei, H., Rostamy-Malkhalifeh, M., Ostadrahimi, M., & Banihashemi, S. A. (2021). Diagnosis of Heart Disease Using Feature Selection Methods Based on Recurrent Fuzzy Neural Networks. *IPTEK The Journal for Technology and Science*, *32*(2), 64-73.

[26] Haykin, S., (2009). "Neural Networks and Learning Machines Third Edition",*Upper Saddle River, New Jersey: Pearson Education*.

[27] Cover, T. M., (1965). "Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recognition", *IEEE Transactions on Electronic Computers*; 14:326-34.

[28] Buhmann, M. D., (2003). "Radial Basis Functions: Theory and Implementations", *Cambridge University Press*.

[29] Bianchini,M., P. Frasconi and M. Gori, (1995). "Learning without local minimal in radial basis functions networks", *IEEE Transactions in Neural Networks*; 6 (3): 749–55.

[30] Chen,S., C. F. N. Cowan and P. M. Grant,(1991). "Orthogonal least squares learning algorithm for radial basis function networks", *IEEE Transactions on Neural Networks*; 2(2): 302–9.

[31] Bors, A. G., G. Gabbouj, (1994). "Minimal topology for a radial basis function neural network for pattern classification", *Digital Signal Processing: a reviwe journal*; 4(3): 173-88.

[32] Yingwei, L., N. Sundararajan, P. Saratchandran, (1997). "A sequential learning scheme for function approximation by using minimal radialbasis function neural networks", *Neural Comput*; 9(2): 461-78.

[33] Sanner, R. M., J. J. E. Slotine, (1994). "Gaussian networks for direct adaptive control", *IEEE trans. In Neural Network*; 3(6): 837-63.