Journal of Data Envelopment Analysis and Decision Science 2013 (2013) 1-12

ISPACS
International Scientific Publications and Consulting Services

Available online at www.ispacs.com/dea
Volume: 2013, Year 2013 Article ID: dea-00002, 12 Pages
doi:10.5899/2013/dea-00002
Research Article

DEA&DS
Data Envelopment Analysis and Decision Science

Using DEA-neural network approach to solve binary classification problems

Farhad Hosseinzadeh Lotfi[1*], Gholam Reza Jahanshahloo[1], Shadi Givehchi[1], Mohsen Vaez-Ghasemi[1]

(1) Mathematics Department, Science and Research Branch, Islamic Azad University, Tehran, Iran.

Abstract

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

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

1 Introduction

Classification, a branch of artificial intelligence, is a scientific discipline in Machine Learning [17]. 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 [9], [18], k-Nearest Neighbors [3], [23], [16], [11], Decision Trees [21], Neural networks [15], Naive Bayes [14], [12], Support Vector Machine (SVM) [20], AdaBoost [10].

Another algorithm for solving classification problem is Data Envelopment Analysis [7]. DEA developed by Charnes et al. a nonparametric methodology for evaluating the performance of a group of Decision Making Units (DMUs) which use multiple inputs to produce multiple outputs [5]. DEA successfully divides them into two categories, efficient DMUs and inefficient DMUs, so DEA can be used to solve binary classification problems.

* Corresponding Author. Email address: farhad@hosseinzadeh.ir

The DEA models so far are used to solve binary classification problem are CCR and BCC models. For solving classification problems with BCC model, DMUs must have monotonicity property and inputs should be non-negative. Some of classification problems include negative data or data do not satisfy in monotonicity property, in this case, DEA can not apply for solving this problems itself, so pendharkar [19] for solving these drawbacks used radial basic function neural network (RBFN) and proposed a hybrid RBFN-DEA neural network for solving binary classification problems. In this paper, we combine additve model in DEA with RBFN and introduce a new model for solving binary classification problems. Proposed model has lower CPU time and more accuracy respect to RBFN-DEA model, furthermore our model can be applied for solving linear separable classification problem with negative data, in this problem, we do not need to apply RBFN for generate positive data. This paper is divided into five sections. In 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, adiitive model for later discussion in next section.

Definition1: Monotonicity property:

If a DMU has higher value of attributes then it belongs to a certain class with the higher probability and vice versa [19].

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" [13], [8].

Definition 2: Basis Functions and Feature Space:

Let $X = \{x_1, \ldots, x_n\}$ be a set of n vector in m-dimensional space, each of which is assigned to be one of two classes, A and B. Define a function $\phi(x)$ as $\phi(x) = [\phi_1(x), \ldots, \varphi_r(x)]$, that $\phi_i(x)$: $R^m \to R$ $(i = 1, \ldots, m)$ 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

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

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

In this paper we use the following basic function that called radial basic function [4]:

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

Where $\sigma = \frac{d_{max}}{2r}$, that d_{max} is the maximum distance between chosen μ_i $(i = 1, \ldots, r)$ and maximum of r can be equal 5. μ_i $(i = 1, \ldots, r)$ can be initialized randomly by vectors in X or determine using cluster analysis approaches [19], [13].

Definition 3: Radial Basis Function Networks (RBFNs):

RBFNs have three layer. 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 [1], [6]. RBFNs have two part for learning, In part I, from input layer to hidden layer use unsupervised learning and in part II, from hidden layer to output layer use

supervised learning. RBFNs can be used for pattern classification [2], function approximation [24] and control [22].

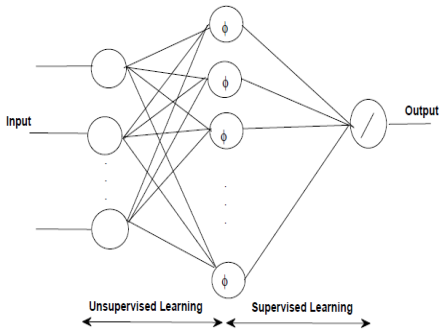

Fig 1: RBFNs structure.

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 $(p = 1,\dots,n)$:

$$Max \quad z = \sum_{i=1}^{m} S_i^+ + \sum_{r=1}^{s} S_r^-$$

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

(x_p, y_p) is evaluated DMU [7].

3 RBFN-ADD neural network

The RBFN-DEA model of Pendharkar [19] motivate us to propose a new model to solve binary classification problems. The proposed RBFN-ADD model has two part, in part I, input data using Gaussian Radial Basis Function (GRBF) are transfered to high-dimensional space (feature space) that in this space can be linear separable with high probability by cover's theorem [13], [8]. Using GRBF, negative data convert to non-negative data in the feature space [19].

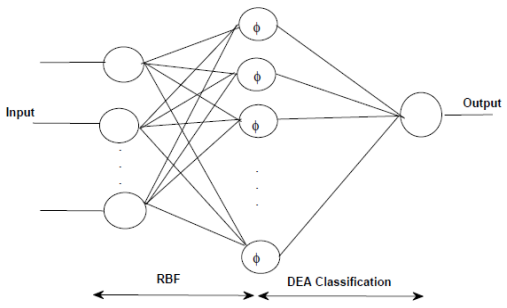

Fig 2: RBFN-ADD model structure.

In DEA part, we use additive model. We apply the following models to develop classification hyperplane for class 0 and class 1:

$$Max \;\; z = \sum_{i=1}^{m} S_i^-$$

$$\begin{aligned} s.t. \quad & \sum_{j=1}^{n} \lambda_j x_{ij} + S_i^- = x_{ip} \;\; i = 1,\ldots,m \\ & \sum_{j=1}^{n} \lambda_j = 1 \\ & \lambda_j \geq 0 \quad j = 1,\ldots,n \\ & S_i^- \geq 0 \;\; i = 1,\ldots,m \end{aligned} \tag{2}$$

$$Max \;\; z = \sum_{i=1}^{m} S_i^+$$

$$\begin{aligned} s.t. \quad & \sum_{j=1}^{n} \lambda_j x_{ij} - S_i^+ = x_{ip} \;\; i = 1,\ldots,m \\ & \sum_{j=1}^{n} \lambda_j = 1 \\ & \lambda_j \geq 0 \quad j = 1,\ldots,n \\ & S_i^+ \geq 0 \;\; i = 1,\ldots,m \end{aligned} \tag{3}$$

where $x_j \in R^m \; (j = 1,\ldots,n)$ are training data. Suppose the training data set consist of n DMUs that k DMUs belongs to class 0 and the rest belongs to class 1. We use model (2) to generate class 0 frontier and model (3) to generate class 1 frontier in training part of our model. Since data have monotonicity property so suppose data far from origin belongs to class0 and data close to origin belongs to class1. To determine which class is closer to the origin in feature space, Euclidean norm of average vector in each class can be calculated.

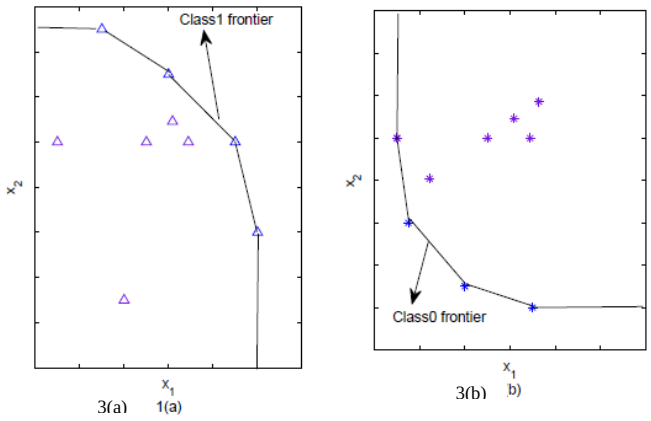

Fig 3:

3(a) Behavior of linear programming (3) in 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.

Procedure for solving classification problems with minimum error for identify test data belong to class 0 is the below form:

- For training data use only the cases from the class 0.

- Using the linear programming (2) determine the efficient set of cases from the class 0, A^*, so determine 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 set A^*.

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

For facility in later discussion, the above procedure is called NT0.

Proposition 1.

Linear programming (4) has feasible solution if and only if x^{test} belongs to class 0.

Proof:

If model (4) has solution then exist (λ^*, s^{-^*}) such that $\sum_{j\in A^*} \lambda_j^* x_j + S^{-^*} = x^{test}$, since $S^{-^*} \geq 0$ then $\sum_{j\in A^*} \lambda_j^* x_j \leq x^{test}$. According to the convexity condition, $\sum_{j\in A^*} \lambda_j^* x_j \in class\ 0$, so using monotonicity property we have $x^{test} \in class\ 0$.

Now we suppose $x^{test} \in class\ 0$, thus $x^{test} \in \partial\ class\ 0$ or $x^{test} \in int\ class\ 0$. Clearly, from the condition of convexity class 0 and monotonicity property the linear programming (4) has feasible solution $(\lambda', s^{-'})$.

Procedure for solving classification problems with minimum error for identify test data belong to class 1 is the below form:

- For training data use only the cases from the class 1.
- Using the linear programming (3) determine the efficient set of cases from the class 1, B^*, so determine class 1 frontier.
- Take a unit from the test data, training data units from the efficient set B^* and solve the linear programming (5).
- If model (5) has a feasible solution then test data belongs to class 1 otherwise it belongs to class 0.

Figure 3(a) shows the classification hyperplane obtained by members of the efficient set B^*.

$$
\begin{aligned}
&Min \quad z_1 = \textstyle\sum_{i=1}^{m} S_i^+ \\
&s.t. \quad \textstyle\sum_{j\in B^*} \lambda_j x_{ij} - S_i^+ = x_i^{test} \;\; i = 1,\ldots,\mathrm{m} \\
&\qquad \textstyle\sum_{j\in B^*} \lambda_j = 1 \\
&\qquad \lambda_j \geq 0 \quad j \in B^* \\
&\qquad S_i^+ \geq 0 \quad i = 1,\ldots,\mathrm{m}
\end{aligned} \tag{5}
$$

for facility in later discussion, the above procedure is called NT1.

Proposition 2.

Linear programming (5) has feasible solution if and only if x^{test} belongs to class 1.

Each of the models NT0 and NT1 can be applied in the supervised learning phase of neural network. If both NT0 and NT1 simultaneously are chosen then, for most classification problems, there will be cases in test data set that will overlap and may be classified both class 0 and class 1, in this case we use Nearest Nighbor approach (NNA) to determine the class of a case in the overlapping region [19], [3], [23], [16], [11]. For this, we proposed a criteria for selection appropriate class:

Suppose that x^{test} is a case in overlapping region, define $x^0 = x^{test} - s^{-^*}$, $x^1 = x^{test} + s^{+^*}$ that s^{+^*} and s^{-^*} are optimal solution of (4) and (5) respectively. So x^0 and x^1 are projection of x^{test} under norm1 over frontier of convex hulls of A^* and B^* respectively, thus using NNA we proposed the following criteria:

$$\begin{cases} x^{test}\ belongs\ to\ class\ 0 & if\ \ z_1^* > z_0^* \\ x^{test}\ belongs\ to\ class\ 1 & if\ \ z_1^* < z_0^* \end{cases}$$

where z_0^* and z_1^* are optimal objective values of (4) and (5) respectively.

if $z_1^* = z_0^*$ then x^{test} randomly assigned to a class.

Remark 1.

If in testing part NT0 and NT1 for x^{test}, models (4) and (5) simultaneusely have not feasible solution then test data randomly assigned to a class.

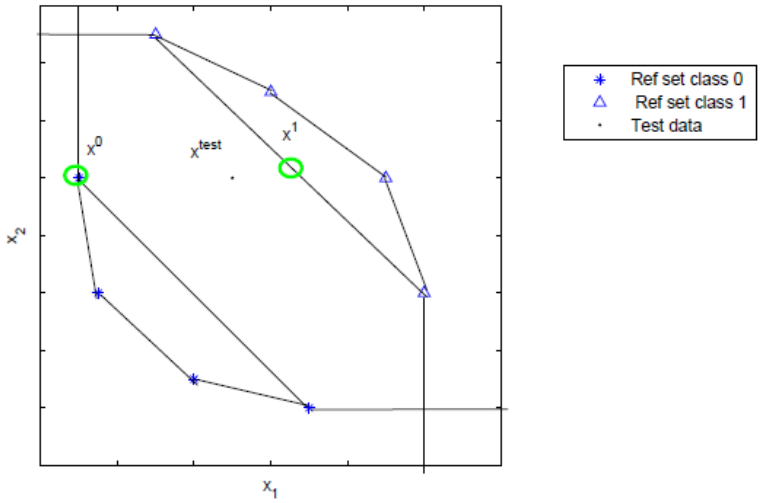

Fig 4: Behavior of linear programming programs (4) and (5).

4 Numerical examples

In order to demonstrate the effectiveness and efficiency of the proposed model, in this section, we discuss the results obtained of proposed model and RBFN-DEA model through two examples. Criteria NTIIEM, NTIEM and NNA RBFN-DEA model is found in Pendharkar [19].

Example 1. In this example, we compare the performance of additive model and RBFN-DEA model [19] with negative and nonnegative value. 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 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 normal distribution with a mean of -1 were labeled as belonging to class 0.

Table 1: 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 3: Comparing the performance of proposed model and RBFN-DEA model in terms of NT0, NT1 and NNA in example 1.

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%

Table 4: Camparing The performance of proposed model and pendharkar model in terms of CPU time in example 1.

Model	CPU time
RBFN-ADD	1.031250
RBFN-DEA	6.625000

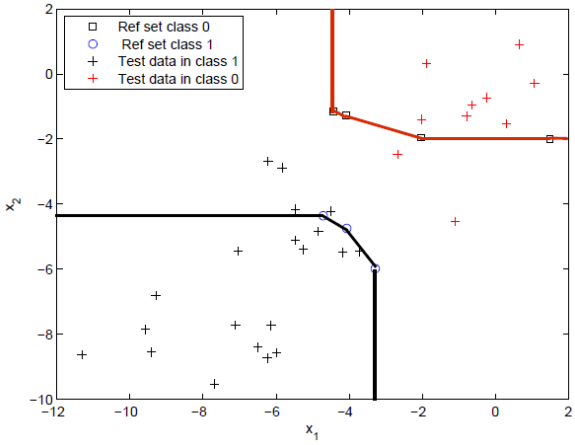

Fig 5: Behavior of NT0 and NT1 in example 1.

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

No	Class	Training data set					Testing data set				
		Attribute 1	Attribute 2	Attribute 3	Attribute 4	Attribute 5	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

Example 2. In this example we compare the performance of proposed model and RBFN-DEA model [19] 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 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 7: Camparing the performance of proposed model and RBFN-DEA model in terms of NT0, NT1 and NNA in example2.

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: Camparing the performance of proposed model and pendharkar model in terms of CPU time in example 2.

Model	CPU time
RBFN-ADD	1.125000
RBFN-DEA	4.171875

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

5 Conclusion

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

References

[1] M. Bianchini, P. Frasconi and M. Gori, {Learning without local minimal in radial basis functions networks}, IEEE Transactions in Neural Networks, 6 (3) (1995) 749-55.
http://dx.doi.org/10.1109/72.377979

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

[3] D. Bremner, E. Demaine, J. Erickson, J. Iacono, S. Langerman, P. Morin and G. Toussaint, {Output-sensitive algorithms for computing nearest-neighbor decision boundaries}, Discrete and Computational Geometry, 33 (4) (2005) 593-604.
http://dx.doi.org/10.1007/s00454-004-1152-0

[4] M. D. Buhmann, {Radial Basis Functions: Theory and Implementations}, Cambridge University Press. (2003).
http://dx.doi.org/10.1017/CBO9780511543241

[5] A. Charnes, W. W. Cooper and E. Rhodes, {Measuring the efficiency of decision making units}, European Journal of Operational Research, 2 (6) (1978) 429-44.
http://dx.doi.org/10.1016/0377-2217(78)90138-8

[6] S. Chen, C. F. N. Cowan, P. M. Grant, {Orthogonal least squares learning algorithm for radial basis function networks}, IEEE Transactions on Neural Networks, 2 (2) (1991) 302-9.
http://dx.doi.org/10.1109/72.80341

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

[8] T.M. Cover, {Geometrical and Statistical properties of systems of linear inequalities with applications in pattern recognition}, IEEE Transactions on Electronic Computers , 14 (1965) 326-34.
http://dx.doi.org/10.1109/PGEC.1965.264137

[9] R. Fisher, {The Use of Multiple Measurements in Taxonomic Problems}, Annals of Eugenics, 7 (1963) 179-188.
http://dx.doi.org/10.1111/j.1469-1809.1936.tb02137.x

[10] Y. Freund, R. E. Schapire, {A short intro duction to boosting}, Journal of Japanese Society for Artificial Intelligence, 5 (14) (1999) 771-80.

[11] P. Hall, B. U. Park and R. J. Samworth, {Choice of neighbor order in nearest-neighbor classification}, Annals of Statistics, 36 (2008) 2135-52.
http://dx.doi.org/10.1214/07-AOS537

[12] J. Han, M. Kamber, {Data Mining: Concepts and Techniques}, Elsevier, (2006).

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

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

[15] A. Krogh, J. Hertz and R. G. Palmer, {Introduction to the theory of neural computation}, Perseus, (1991).

[16] P. Mills, {Efficient statistical classification of satellite measurements}, International Journal of Remote Sensing, (2011).

[17] T. Mitchell, {Pattern classification and Scene Analysis}, McG raw Hill, (1997).

[18] D. F. Morrison, {Multivariate Statistical Methods}, McG raw-Hill, (1990).

[19] P. C. Pendharkar, {A hybrid radial basis function and data envelopment analysis neural network for classification}, Computers and Operations Research, 38 (2011) 256-66.
http://dx.doi.org/10.1016/j.cor.2010.05.001

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

[21] J. R. Quinlan, {Induction to decision tree, Machine Learning}, Kluwer Academic Press, (1986) 81-106.

[22] R. M. Sanner, J. J. E. Slotine, {Gaussian networks for direct adaptive control}, IEEE trans. In Neural Network, 3 (6) (1994) 837-63.
http://dx.doi.org/10.1109/72.165588

[23] D. G. Terrell, D. W. Scott, {Variable kernel density estimation}, Annals of Statistics, 20 (1992) 1236-65.
http://dx.doi.org/10.1214/aos/1176348768

[24] L. Yingwei, N. Sundararajan, P. Saratchandran, {A sequential learning scheme for function approximation by using minimal radial basis function neural networks}, Neural Comput, 9 (2) (1997) 461-78.