Volume 17 | Issue 2 Article 7

3-7-2019

ϕ-Divergence Loss-Based Artificial Neural Network

R. L. Salamwade
Shivaji University, Kolhapur, India

D. M. Sakate
Shivaji University, Kolhapur, India, dms.stats@gmail.com

S. K. Mathur
Augusta University

ϕ-Divergence Loss-Based Artificial Neural Network

Cover Page Footnote
The first and second author wish to thank the Department of Science and Technology, New Delhi, India for financial support under DST-PURSE scheme.

Journal of Modern Applied Statistical Methods
November 2018, Vol. 17, No. 2, eP2646
doi: 10.22237/jmasm/1551966252

ϕ-Divergence Loss-Based Artificial Neural Network

R. L. Salamwade
Shivaji University
Kolhapur, India

D. M. Sakate
Shivaji University
Kolhapur, India

S. K. Mathur
Augusta University
Augusta, GA

Artificial Neural Networks (ANNs) can fit non-linear functions and recognize patterns better than several standard techniques. Performance of ANNs is measured by using loss functions. Phi-divergence estimator is generalization of maximum likelihood estimator and it possesses all its properties. A neural network is proposed which is trained using phi-divergence loss.

Keywords: Power divergence family, classification, back-propagation, loss function, mean square error

Introduction

Artificial neural networks (ANNs) are biologically-inspired models which are based on the neural structure of a human brain and on how human learns from experiences. ANNs are widely used for prediction, classification, pattern recognition, and optimization purposes. A typical ANN is a class of computational architecture made up of simple, multiple interconnected computational units, each of which performs a summation operation then applied to a nonlinear transfer function to produce output. It is widely used as a supervised learning technique for prediction. McCulloch and Pitts (1943) developed the first simple neuron model using electric circuits based on how a human brain actually learns. Hebb (1949) suggested the first learning rule which is known as Hebb's learning rule. Rosenblatt (1962) developed the first perceptron using the McCulloch-Pitts neuron. Minskey and Papert (1990) showed that perceptron could not solve the problem of non-linearly separable functions. The back-propagation algorithm for multilayer perceptrons (multilayer feed forward network) was proposed by Werbos (1972).

doi: 10.22237/jmasm/1551966252 | Accepted: May 8, 2018; Published: March 7, 2019.
Correspondence: D. M. Sakate, dms.stats@gmail.com

Rumelhart, Hinton, and Williams (1986) reinvented the back-propagation algorithm for multilayer neural networks.

ANN being a supervised learning technique assigns a network output for each input pattern in cognition with desired output. The learning process of the network requires identification of weights which is usually done by minimizing loss which measures the discrepancy between network and desired output. At every stage of the learning, weights are updated using the derivative of a loss function. Hence, the loss function plays a crucial role in the process of training a network. Usually, squared error loss is the first choice, and an ANN trained using this loss is called traditional ANN. Fallah et. al. (2009) suggested the use of negative log likelihood as a loss function if the distribution of the response variable is known. This notion followed from the famous maximum likelihood estimator (MLE) in the statistics literature.

MLE is a particular case of the minimum ϕ-divergence estimator. With the rise of the minimum ϕ-divergence estimator (Pardo, Pardo, & Pardo, 2005) in logistic regression as a good alternative to MLE when a sample is small, the options have increased. The idea in Fallah et al. (2009) is extended, and the proposal is to train an ANN for classification using ϕ-divergence loss.

Methodology

$\boldsymbol{\phi}$-Divergence

Let Y_i be a binomial random variable with parameters n_i and $\pi(\mathbf{x}_i)$, where $\mathbf{x}_i = (x_{i0}, x_{i1},\ldots, x_{ik})$ is the i^{th} row in the matrix of observations on k explanatory variables denoted by $\mathbf{X}$ with $x_{i0} = 1$, $i = 1,\ldots, I$. Denote $\pi_{i1} = \pi(\mathbf{x}_i)$, $\pi_{i2} = 1 - \pi(\mathbf{x}_i)$, $n_{i2} = n_i - n_{i1}$, and $N = \sum_{i=1}^{I} n_i$. Denote the actual and predicted probability vectors as $\mathbf{p}$ and $\boldsymbol{\pi}$, respectively, given by

$$\mathbf{p} = \left(\frac{n_{11}}{N}, \frac{n_{12}}{N}, \frac{n_{12}}{N}, \frac{n_{22}}{N}, \mathrm{K}, \frac{n_{I1}}{N}, \frac{n_{I2}}{N}, \right)^{\mathrm{T}}$$

$$\boldsymbol{\pi} = \left(\pi(\mathbf{x}_i)\frac{n_1}{N}, (1-\pi(\mathbf{x}_i))\frac{n_1}{N}, \mathrm{K}, \pi(\mathbf{x}_i)\frac{n_I}{N}, (1-\pi(\mathbf{x}_i))\frac{n_I}{N} \right)^{\mathrm{T}}$$

The Csiszár (1963) and Ali and Silvey (1966) ϕ-divergence measure between the two probability vectors $\mathbf{p}$ and $\boldsymbol{\pi}$ is given by

$$\mathrm{D}_{\phi}\left(\mathbf{p},\boldsymbol{\pi}\right) \equiv \sum_{j'=1}^{2}\sum_{i=1}^{I}\pi_{ij'}\frac{n_i}{N}\phi\left(\frac{\frac{n_{ij'}}{N}}{\pi_{ij'}\frac{n_i}{N}}\right); \quad \phi \in \Phi^{*} \tag{1}$$

where Φ^* is the class of all convex functions $\phi(t)$, $t > 0$, such that at $t = 1$, $\phi(1) = 0$, $\phi''(1) > 0$, and at $x = 0$, $0\phi(0/0) = 0$ and $0\phi(\mathbf{p}/0) = \log_{u\to\infty} \phi(u)/u$. For every $\phi \in \Phi^*$ that is differentiable at $t = 1$, the function

$$\psi\left(t\right) \equiv \phi\left(t\right) - \phi'\left(1\right)\left(t-1\right) \tag{2}$$

also belongs to Φ^*. Then, $\mathrm{D}_{\psi}(\mathbf{p}, \boldsymbol{\pi}) = \mathrm{D}_{\phi}(\mathbf{p}, \boldsymbol{\pi})$, and ψ has an additional property that $\psi'(1) = 0$. Consider the set Φ^* to be equivalent to the set

$$\Phi \equiv \Phi^{*} \cap \left\{\phi : \phi'\left(1\right) = 0\right\}$$

An important family of ϕ-divergences in statistical problems, which is introduced in Cressie and Read (1984), is the power divergence family:

$$\left.\begin{aligned} &\phi_{(\lambda)}\left(t\right) = \left(\lambda\left(\lambda+1\right)\right)^{-1}\left(t^{\lambda+1}-t\right); \quad \lambda \neq 0, \lambda \neq -1 \\ &\phi_{(0)}\left(t\right) = t\log\left(t\right) - t + 1; \\ &\phi_{(-1)}\left(t\right) = -\log\left(t\right) - t - 1; \end{aligned}\right\} \tag{3}$$

Minimizing ϕ-divergence for different values of λ yields different estimators. The minimum power divergence estimator for $\lambda = 0$ coincides with the MLE, for $\lambda = 1$ it corresponds with the minimum χ^2 estimator, for $\lambda = -1/2$ it is the Freeman-Tukey estimator, for $\lambda = -1$ it is the minimum modified likelihood estimator, for $\lambda = -2$ it is the minimum Neyman modified likelihood estimator, and for $\lambda = 2/3$ it is the Cressie-Read estimator. The main advantage of using ϕ-divergence loss is that different loss functions are obtained by changing the value of λ.

Computational Algorithm with ϕ-Divergence Loss

A commonly used neural network is the multilayer perceptron (MLP). An MLP consists of one input layer, one output layer, and one or more hidden layers. The input units pass their input to the units in the first hidden layer or directly to produce the output units. Each of the hidden layer units adds a bias to a weighted sum of its inputs and calculates an activation function of the result. Cybenko (1986) proved that a single hidden layer feed forward neural network with any continuous sigmoidal nonlinearity can approximate any continuous and multivariate function. Hence, single hidden layer MLP with any continuous sigmoidal nonlinearity is able to approximate any continuous and multivariate functions. To do this, usually a back-propagation algorithm is used which updates the weights using the delta method.

Back-propagation algorithms are used here to train both the proposed and traditional network. The objective is to approximate $\pi_{i1} = \pi(\mathbf{x}_i)$ for given inputs $\mathbf{x}_i$. The expression for π_{i1} with a single hidden layer feed forward network is given by

$$\pi_{i1} = \mathrm{f}_1\left(W_0 + \sum_{l=1}^{L} W_l \, \mathrm{f}_2\left(V_{l0} + \sum_{j=1}^{J} V_{lj} x_{ij} \right)\right) \tag{4}$$

where f_1 and f_2 are activation functions at the hidden and output layers, respectively. The proposal is to estimate the weights $\mathbf{W}$ and $\mathbf{V}$ by minimizing ϕ-divergence loss,

$$\mathrm{D}_\phi(\mathbf{p},\boldsymbol{\pi}) = \sum_{i=1}^{I} \pi_{i1} \frac{n_i}{N} \phi\left(\frac{\frac{n_{i1}}{N}}{\pi_{i1}\frac{n_i}{N}} \right) + \sum_{i=1}^{I} (1-\pi_{i1}) \frac{n_i}{N} \phi\left(\frac{\frac{n_{i2}}{N}}{(1-\pi_{i1})\frac{n_i}{N}} \right) \tag{5}$$

Subsequently, derive the formulae for updating weights. To update the input to hidden layer weights, differentiating eq. (5) with respect to $\mathbf{W}$ gives

$$\frac{\partial \mathrm{D}_\phi(\mathbf{p},\boldsymbol{\pi})}{\partial \mathbf{W}} = \frac{n_i}{N} \pi_{i1} (1-\pi_{i1}) \mathbf{O}^{\mathrm{T}} \mathrm{J}(\mathbf{W})$$

where $\mathbf{O}$ is the vector of hidden layer output and

$$\mathrm{J}(\mathbf{W}) = \phi\left(\frac{n_{i1}}{\mathrm{m}_i(\mathbf{W})} \right) - \frac{n_{i1}}{\mathrm{m}_i(\mathbf{W})} \phi'\left(\frac{n_{i1}}{\mathrm{m}_i(\mathbf{W})} \right) - \phi\left(\frac{n_i - n_{i1}}{n_i - \mathrm{m}_i(\mathbf{W})} \right) + \frac{n_i - n_{i1}}{n_i - \mathrm{m}_i(\mathbf{W})} \phi'\left(\frac{n_i - n_{i1}}{n_i - \mathrm{m}_i(\mathbf{W})} \right)$$

with $\mathrm{m}_i\left(\mathbf{W}\right)=n_i f_1\left(W_0+\sum_{l=1}^{L}W_l O_l\right), i=1,2,\ldots,I$.

Each output layer weight is updated using the increment

$$\nabla\mathbf{W}=-\eta\frac{\partial\,\mathrm{D}_\phi\left(\mathbf{p},\boldsymbol{\pi}\right)}{\partial\mathbf{W}}$$

where η is the learning parameter usually chosen by the user. The weight update formula for **W** at the (m + 1)th iteration is given by

$$\mathbf{W}^{m+1}=\mathbf{W}^{m}-\eta\nabla\mathbf{W}^{m} \tag{6}$$

Similarly, to update hidden to output layer weights, differentiating eq. (5) with respect to **V** gives

$$\frac{\partial\,\mathrm{D}_\phi\left(\mathbf{p},\boldsymbol{\pi}\right)}{\partial\mathbf{V}}=\frac{n_i}{N}\pi_{i1}\left(1-\pi_{i1}\right)\mathbf{W}^{m+1}\mathbf{O}\left(1-\mathbf{O}\right)^{\mathrm{T}}\mathbf{x}_i^{\mathrm{T}}\,\mathrm{J}\left(\mathbf{W}\right)$$

Each weight in the hidden layer is updated using the increment

$$\nabla\mathbf{V}=-\eta\frac{\partial\,\mathrm{D}_\phi\left(\mathbf{p},\boldsymbol{\pi}\right)}{\partial\mathbf{V}}$$

The weight update formula for **V** at the (m + 1)th iteration is given by

$$\mathbf{V}^{(m+1)}=\mathbf{V}^{m}-\eta\mathbf{V}^{m} \tag{7}$$

Using the above formulae for updating weights, the algorithm for neural network with ϕ-divergence loss is as follows:

Algorithm for Neural Network with ϕ-Divergence Loss

- Set error E to 0, initialize output layer weights **W** and hidden layer weights **V**.
- Choose any input vector x_{ij}, j = 0, 1,…, J and i = 1, 2,…, I and compute the hidden layer output

$$O_l = \mathrm{f}_2\left(V_{l0} + \sum_{j=1}^{J} V_{lj} x_{ij}\right), \quad l = 1, 2, \ldots, L$$

- Compute the output layer output $\pi_{i1} = \mathrm{f}_1\left(W_0 + \sum_{l-1}^{L} W_l O_l\right)$
- Update output layer weights **W** at the (m + 1)th step using eq. (6).
- Update hidden layer weights **V** at the (m + 1)th step using eq. (7).
- Compute the ϕ-divergence loss

$$E = E + \sum_{j'=1}^{2} \pi_{ij'} \frac{n_i}{N_i} \phi\left(\frac{n_{ij'}}{n_i \pi_{ij'}}\right)$$

- Repeat for all input patterns.
- Use appropriate stopping criteria.

Results

Simulation Study

The purpose of this simulation study is to compare the ANN based on ϕ-divergence loss for different values of λ with the traditional ANN. The data were generated from the model

$$\log \frac{\pi(\mathbf{x}_i)}{1 - \pi(\mathbf{x}_i)} = 0.5 - x_{i1} + 1.5x_{i2} + 2x_{i3} - 2x_{i4} \tag{8}$$

where the $\mathbf{x}_i$ are generated from U(0, 1) for distinct patterns, I = 50, 100, 200, and five different sample size combinations $\mathbf{n} = (n_1, \ldots, n_I)^T$ considered were

$\mathbf{n}^1$: all n_i are 5
$\mathbf{n}^2$: the n_i are taken from the set {5, 10}, each 50% of I
$\mathbf{n}^3$: the n_i are taken from the set {5, 15, 30, 40}, each 25% of I
$\mathbf{n}^4$: the n_i are taken from the set {5, 10, 15, 20}, each 25% of I
$\mathbf{n}^5$: all n_i are 10

A total of M = 1000 iterations were performed. For both the hidden layer and the output layer, a sigmoid activation function is used and the learning parameter η is set to 0.01. For training both the networks, a number of hidden nodes was obtained using four different formulae reported in Feng, Yu, and Kusiak (2006, p. 17):

$$H_1 = \frac{I_n + O_n}{2}, \quad H_2 = O_n\left(I_n + 1\right), \quad \frac{N}{10} - I_n - O_n \leq H_3 \leq \frac{N}{2} + I_n + O_n,$$
$$H_4 = I_n \log_2\left(\boldsymbol{\pi}\right)$$

where I_n = number of inputs, O_n = number of outputs, and N = total number of input patterns. Four different values of λ, $\lambda = -1/2, 0, 2/3$, and 1, were considered. Out of the total sample, 70% is used for training and the remaining 30% for testing. For judging which network performs better, two criteria were used for computing error in predictions. The first criterion is the most commonly used mean square error between the actual and predicted probabilities which is given below:

$$MSE = \frac{1}{M}\sum_{i=1}^{m}\left(p_i - \hat{p}_i\left(\mathbf{W},\mathbf{V}\right)\right)^2$$

where m = number of testing observations and $M = \sum_{i=1}^{m} n_i$. The second criterion is the mean absolute error between the actual and predicted probabilities as given below:

$$MAE = \frac{1}{M}\sum_{i=1}^{m}\left|p_i - \hat{p}_i\left(\mathbf{W},\mathbf{V}\right)\right|$$

For comparison purposes, a ratio of errors is reported in the Tables 1-4, defined as

$$\text{Ratio} = \frac{\text{Average of errors taken over iterations for traditional network}}{\text{Average of errors taken over iterations for network with } \phi\text{-divergence loss}}$$

From Tables 1-4, the ratio of errors is greater than 1, indicating the superiority of ANN with ϕ-divergence loss over traditional ANN. Comparing the ratio of errors across the tables for small and large sample sizes, the ratio of error is seen to be much larger for $\lambda = 2/3$ and $\lambda = -1/2$. Hence, our simulation study indicates that the ANN with ϕ-divergence loss outperforms traditional ANN.

Table 1. Ratio of errors for $\lambda = -1/2$

Hidden Node:		H_1		H_2		H_3		H_4	
n_i	I	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
n^1	50	1.0171	1.0360	1.0284	1.0391	1.0160	1.0337	1.0796	1.0629
	100	1.1088	1.0899	1.1154	1.0909	1.1268	1.0951	1.1421	1.0960
	200	1.1674	1.1195	1.1582	1.1151	1.2080	1.1295	1.1965	1.1201
n^2	50	1.0210	1.0291	1.0503	1.0403	1.0443	1.0415	1.1156	1.0755
	100	1.1762	1.1105	1.1607	1.1035	1.1811	1.1108	1.2193	1.1222
	200	1.2527	1.1475	1.2349	1.1378	1.2356	1.1260	1.2390	1.1255
n^3	50	1.0299	1.0228	1.0403	1.0273	1.0519	1.0325	1.0529	1.0351
	100	1.2391	1.1348	1.2608	1.1402	1.2440	1.1311	1.3188	1.1620
	200	1.3464	1.1787	1.3734	1.1867	1.4358	1.2109	1.5075	1.2382
n^4	50	1.0439	1.0328	1.0524	1.0367	1.0391	1.0316	1.0497	1.0370
	100	1.2366	1.1325	1.2086	1.1175	1.2389	1.1317	1.2757	1.1442
	200	1.3121	1.1672	1.3193	1.1664	1.3884	1.1931	1.4206	1.2040
n^5	50	1.0453	1.0368	1.0429	1.0353	1.0324	1.0317	1.0380	1.0305
	100	1.2043	1.1225	1.1826	1.1117	1.2058	1.1220	1.2289	1.1293
	200	1.2655	1.1540	1.2573	1.1466	1.3257	1.1766	1.3506	1.1881

Table 2. Ratio of errors for $\lambda = 0$

Hidden Node:		H_1		H_2		H_3		H_4	
n_i	I	MSE	MAE	MSE	MAE	MSE	MAE	MSE	MAE
n^1	50	1.0540	1.0464	1.0523	1.0450	1.0486	1.0433	1.0696	1.0479
	100	1.0810	1.0595	1.0979	1.0710	1.1010	1.0714	1.1272	1.0775
	200	1.1030	1.0688	1.1158	1.0769	1.1881	1.1114	1.1804	1.1059
n^2	50	1.0787	1.0540	1.0874	1.0557	1.0777	1.0513	1.0971	1.0579
	100	1.1054	1.0656	1.1281	1.0759	1.1357	1.0801	1.1618	1.0907
	200	1.1445	1.0817	1.1693	1.0949	1.2356	1.1260	1.2389	1.1255
n^3	50	1.1088	1.0673	1.0909	1.0574	1.0832	1.0531	1.1073	1.0637
	100	1.1508	1.0847	1.1701	1.0937	1.1830	1.0991	1.2219	1.1190
	200	1.2259	1.1186	1.2370	1.1248	1.3251	1.1615	1.3858	1.1869
n^4	50	1.0867	1.0533	1.0898	1.0561	1.0818	1.0540	1.0990	1.0570
	100	1.1398	1.0797	1.1392	1.0802	1.1629	1.0891	1.2066	1.1098
	200	1.1869	1.1003	1.2112	1.1120	1.2883	1.1453	1.3256	1.1605
n^5	50	1.0750	1.0500	1.0801	1.0519	1.0857	1.0555	1.0921	1.0560
	100	1.1050	1.0664	1.1226	1.0734	1.1234	1.0748	1.1588	1.0905
	200	1.1547	1.0893	1.1646	1.0937	1.2333	1.1273	1.2510	1.1330

Table 3. Ratio of errors for $\lambda = 2/3$

Hidden Node:		H_1		H_2		H_3		H_4	
n_i	l	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
$\mathbf{n}^1$	50	1.1128	1.0750	1.0982	1.0666	1.1063	1.0721	1.0978	1.0622
	100	1.0536	1.0558	1.0707	1.0631	1.0817	1.0672	1.1120	1.0741
	200	1.2050	1.1144	1.1883	1.1062	1.2116	1.1192	1.2140	1.1167
$\mathbf{n}^2$	50	1.2068	1.1183	1.1822	1.1037	1.1715	1.0977	1.1708	1.0906
	100	1.2794	1.1419	1.2494	1.1291	1.2794	1.1419	1.1501	1.0795
	200	1.0821	1.0507	1.3128	1.1554	1.3318	1.1671	1.3318	1.1650
$\mathbf{n}^3$	50	1.5783	1.2692	1.4116	1.2069	1.4896	1.2337	1.3699	1.1751
	100	1.7450	1.3315	1.5777	1.2634	1.5443	1.2500	1.2251	1.1112
	200	1.1951	1.0930	1.2300	1.1070	1.3330	1.1527	1.3675	1.1691
$\mathbf{n}^4$	50	1.3774	1.1885	1.2862	1.1490	1.3148	1.1670	1.2550	1.1257
	100	1.4767	1.2251	1.3673	1.1811	1.1449	1.0711	1.1842	1.0880
	200	1.1562	1.0738	1.2068	1.0986	1.2798	1.1299	1.3105	1.1453
$\mathbf{n}^5$	50	1.0453	1.0368	1.0429	1.0353	1.0324	1.0317	1.0380	1.0305
	100	1.3812	1.1946	1.3012	1.1586	1.2870	1.1524	1.1713	1.0877
	200	1.1351	1.0670	1.1623	1.0799	1.2130	1.1048	1.2353	1.1165

Table 4. Ratio of errors for $\lambda = 1$

Hidden Node:		H_1		H_2		H_3		H_4	
n_i	l	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**
$\mathbf{n}^1$	50	1.0999	1.0681	1.0993	1.0652	1.0971	1.0667	1.1069	1.0619
	100	1.0610	1.0314	1.0737	1.0376	1.0824	1.0465	1.0976	1.0534
	200	1.0778	1.0374	1.0904	1.0458	1.1408	1.0741	1.1487	1.0785
$\mathbf{n}^2$	50	1.1979	1.1065	1.1657	1.0873	1.1660	1.0922	1.1652	1.0879
	100	1.0964	1.0437	1.1088	1.0522	1.1128	1.0550	1.1342	1.0670
	200	1.1095	1.0491	1.1277	1.0593	1.1982	1.0961	1.2029	1.0980
$\mathbf{n}^3$	50	1.5293	1.2512	1.3829	1.1897	1.4827	1.2316	1.3585	1.1653
	100	1.1280	1.0625	1.1514	1.0746	1.1695	1.0818	1.2109	1.1007
	200	1.1763	1.0821	1.2156	1.1012	1.3027	1.1395	1.3338	1.1514
$\mathbf{n}^4$	50	1.3318	1.1639	1.2803	1.1415	1.3108	1.1592	1.2544	1.1230
	100	1.4436	1.2094	1.3638	1.1765	1.3660	1.1750	1.3871	1.1824
	200	1.1404	1.0639	1.1705	1.0782	1.2617	1.1202	1.2815	1.1299
$\mathbf{n}^5$	50	1.2575	1.1371	1.2238	1.1201	1.2351	1.1271	1.2028	1.1062
	100	1.3421	1.1714	1.2937	1.1503	1.2860	1.1473	1.2836	1.1436
	200	1.1168	1.0551	1.1450	1.0692	1.2056	1.0990	1.2329	1.1148

Table 5. Ratio of errors for compressive strength data

		Testing	
Hidden Nodes	**Training MSE**	**MSE**	**MAE**
H_1	2.2308	11.2313	3.4939
H_2	1.0690	6.7935	3.1505
H_3	1.2609	76.4282	8.0801
H_4	2.2917	13.6299	5.0307

Real Data Application

We considered a compressive strength data (Montgomery, Peck, & Vining, 2003, p. 479). The data consists of 10 loads (x) over the range 2500-4300 psi and a number of fasteners tested at those loads. The numbers of fasteners failing at each were recorded. Traditional and proposed neural networks were trained for different hidden nodes. The choice of λ in ϕ-divergence loss was fixed to $\lambda = 2/3$ based on the results obtained in our simulation study. The entire data set was divided into five equal parts each consisting of two observations. As such, eight observations were randomly chosen for training and the remaining two for testing. Five-fold cross-validation estimates of MSE and MAE for the test data and MSE for training data are obtained and their ratio for traditional ANN and ANN with ϕ-divergence loss is reported in Table 5.

From Table 5, the ratio of errors is greater than 1, indicating the superiority of ANN with ϕ-divergence loss over traditional ANN for all choices of hidden nodes.

Conclusion

Classification is a commonly-encountered problem in fields like the medical, biological, social sciences, etc. The interest of the researcher may be in the prediction of probabilities of the event of interest based on the observed covariates. In such situations, ϕ-divergence loss-based ANN is useful to build classifiers based on ANN when there are multiple observations on a response variable on individuals, which is a common situation in a controlled laboratory setting. An advantage of using ϕ-divergence loss-based ANN over traditional ANN is that it gives predictions with more accuracy. This idea can be extended to multiclass classification (multinomial setting), which may constitute the material for a new research paper.

The proposal is to train an ANN based on ϕ-divergence loss which is a kind of combination of data mining techniques and a statistical parametric procedure. This hybridization is useful as revealed in the simulation study. A computational algorithm to train the ANN based on ϕ-divergence loss is provided. The convergence of the algorithm is guaranteed because of the convexity of ϕ-divergence loss.

That training ANNs with different loss functions can be fruitful is demonstrated in this article. Our simulation study indicates the superiority of the ANN based on ϕ-divergence loss over traditional ANN and ANN with negative log likelihood loss ($\lambda = 0$ in ϕ-divergence loss) in the sense of error in prediction. MSE and MAE are used to compare the error in prediction of ANN when different loss functions are used. The ratio of MSEs and ratio of MAEs are greater than 1 for all combinations of number of trials and sample sizes, indicating the superiority of the proposed method.

The choice of λ leading to the choice of loss function is vital in determining the performance of ANN trained with ϕ-divergence loss. Not all values of λ provide considerable improvement over squared error loss function. In our simulation study, $\lambda = 2/3$ emerged as a better choice for all the sample sizes. This is not a surprising outcome as $\lambda = 2/3$ emerged as a good choice in Pardo et. al. (2005), Pardo and Pardo (2008), and Sakate and Kashid (2014). ANN is a flexible mathematical structure which can identify complex nonlinear relationships. The proposed model based on ϕ-divergence loss is useful and efficient in situations when the characteristics of the processes are difficult to describe using a set of linear equations. A new method to train ANN using ϕ-divergence loss is proposed which outperforms the traditional network with squared error loss.

References

Ali, S. M., & Silvey, S. D. (1966). A general class of coefficients of divergence of one distribution from another. *Journal of the Royal Statistical Society: Series B (Methodological), 26*(1), 131-142. doi: 10.1111/j.2517-6161.1966.tb00626.x

Cressie, N., & Read, T. R. C. (1984). Multinomial goodness of fit tests. *Journal of the Royal Statistical Society: Series B (Methodological), 46*(3), 440-464. doi: 10.1111/j.2517-6161.1984.tb01318.x

Csiszár, I. (1963). Eine informationstheoretische ungleichung und ihre anwendung auf den beweis der ergodizität von Markoffschen ketten [An information-theoretical inequality and its application to the proof of the ergodicity of Markov chains]. *A Magyar Tudományos Akadémia Matematikai Kutató Intézetének Közleményei, A Sorozat, 8*(1-2), 84-108.

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems, 2*(4), 303-314. doi: 10.1007/bf02551274

Fallah, N., Gu, H., Mohammad, K., Seyyedsalehi, S. A., Nourijelyani, K., & Eshraghian, M. R. (2009). Nonlinear Poisson regression using neural networks: A simulation study. *Neural Computing and Applications, 18*, 939-943. doi: 10.1007/s00521-009-0277-8

Feng, C.-X. J., Yu, Z.-G., & Kusiak, A. (2006). Selection and validation of predictive regression and neural network models based on designed experiments. *IIE Transactions, 38*(1), 13-23. doi: 10.1080/07408170500346378

Hebb, D. O. (1949). *The organization of behavior*. New York, NY: Wiley.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics, 5*(4),115-133. doi: 10.1007/bf02478259

Minskey, M. L., & Papert, S. A. (1990). *Perceptrons: An introduction to computational geometry* (Expanded ed.). Cambridge, MA: MIT Press.

Montgomery, D. C., Peck, E. A., & Vining, G. G. (2003). *Introduction to linear regression analysis* (3rd ed.). Singapore: John Wiley and Sons (Asia) Pte. Ltd.

Pardo, J. A., & Pardo M. C. (2008). Minimum φ-divergence estimator and φ-divergence statistics in generalized linear models with binary data. *Methodology and Computing in Applied Probability, 10*(3), 357-379. doi: 10.1007/s11009-007-9054-2

Pardo, J. A., Pardo, L., & Pardo, M. C. (2005). Minimum ϕ-divergence estimator in logistic regression models. *Statistical Papers, 47*(1), 91-108. doi: 10.1007/s00362-005-0274-7

Rosenblatt, E. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. Washington, D.C.: Spartan Books.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning internal representation by error propagation, In D. E. Rumelhart, J. L.McClelland, & The PDP Research Group (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (Vol. 1) (pp. 318-362). Cambridge, MA: MIT Press.

Sakate, D. M., & Kashid, D. N. (2014). Variable selection via penalized minimum φ-divergence estimation in logistic regression. *Journal of Applied Statistics, 41*(6), 1233-1246. doi: 10.1080/02664763.2013.864262

Werbos, P. (1972). *Beyond regression: New tools for prediction analysis in the behavioral sciences*. (Doctoral thesis). Harvard University, Cambridge, MA.