The Influence of the Risk Functional in Data Classification with MLPs

Luís M. Silva[1], Mark Embrechts[2], Jorge M. Santos[1,4] and J. Marques de Sá[1,3]

[1] INEB -Instituto de Engenharia Biomédica, Divisão de Sinal e Imagem
Campus FEUP, Rua Dr. Roberto Frias, 4200 - 465 Porto - Portugal
[2] Rensselaer Polytechnic Institute, Troy, New York, USA
[3] Faculdade de Engenharia da Universidade do Porto, Porto, Portugal
[4] Instituto Superior de Engenharia do Porto, Porto, Portugal
{lmsilva,jmsa}@fe.up.pt, embrem@rpi.edu, jms@isep.ipp.pt

Abstract. We investigate the capability of multilayer perceptrons using specific risk functionals attaining the minimum probability of error (optimal performance) achievable by the class of mappings implemented by the multilayer perceptron (MLP). For that purpose we have carried out a large set of experiments using different risk functionals and datasets. The experiments were rigorously controlled so that any performance difference could only be attributed to the different risk functional being used. Statistical analysis was also conducted in a careful way. From the several conclusions that can be drawn from our experimental results it is worth to emphasize that a risk functional based on a specially tuned exponentially weighted distance attained the best performance in a large variety of datasets. As to the issue of attaining the minimum probability of error we also carried out classification experiments using non-MLP classifiers that implement complex mappings and are known to provide the best results until this date. These experiments have provided evidence that at least in many cases, by using an adequate risk functional, it will be possible to reach the optimal performance.

1 Introduction

We consider a classification problem with a set of classes $\Omega = \{\omega\}$ and a parametric machine (parameter set $W = \{w\}$) such as the multilayer perceptron (MLP) performing a mapping $Y = \varphi(X) = \varphi_w(X)$ from the input variable X into the output variable Y. The machine is trained by some algorithm in order to minimize a risk functional on a parameter set $W = \{w\}$ of the function class $\Phi = \{\varphi_w\}$ implemented by the classifier, attempting to approximate some target variable T. The risk (written here only for the continuous case) is often an expected distance between T and Y,

$$R_\Phi = \sum_\Omega P(\omega) \int_{X,T} L(t,y) dF(t,x|\omega) \quad \text{with } y = \varphi_w(x) \tag{1}$$

where the so-called cost or loss function $L(\cdot)$ can be chosen in various ways. For the popular mean square error (MSE), $L(t, y) = (t - y)^2$ is a quadratic distance; for cross-entropy and two-class problems with $Y \in [0, 1]$ and $T \in \{0, 1\}$, $L(t, y) = t \ln y + (1 - t) \ln(1 - y)$, is a special case of the Kullback-Leibler divergence; etc. Recently, a generalized exponential loss function enjoying the property of being able to emulate the behavior of a whole family of loss functions, by single parameter tuning, has also been proposed [1]. It is also possible to choose the risk as a functional of the error probability density function (pdf), whenever it exists, $f(e) \equiv f_E(e)$ with $E = T - Y$, particularly the Shannon's entropy functional,

$$R_{\Phi} = -\int_E \ln f(e) dF(e), \tag{2}$$

or Rényi's entropy functional

$$R_{\Phi} = \frac{1}{1 - \alpha} \ln \int_E [f(e)]^{\alpha - 1} dF(e). \tag{3}$$

Note that $E \equiv E(\varphi)$. This approach corresponds to the Minimum of Error Entropy (MEE) principle [2–4]. These are more sophisticated risk functionals in the sense that they reflect the whole pdf of $T - Y$, instead of a single distance between T and Y. The main problem in data classification is the possibility of attaining the minimum probability of error, $\min_W Pe_{\Phi}$, afforded by the machine architecture for some w^*, the so-called *optimal solution*. For instance, if hypothetically a certain $\min_W R_{\Phi}$ does not lead to $\min_W Pe_{\Phi}$, one has to conclude that a risk functional is being used which fails to adequately take into account the whole complexity available in the family set Φ. One should then turn to another risk functional. The practical problem is that usually $\min_W Pe_{\Phi}$ is unknown and some risk functionals - equivalently, some learning algorithms may exhibit a better behavior than competing ones in some datasets and worse behavior in other datasets. (In fact, this must necessarily happen, taking into account the well-known results of the no-free-lunch theorems (see e.g. [5]).) In order to acquire experimental evidence on how different risk functionals behave in different datasets, we performed an extensive comparison of MLP classification results involving several types of risk functionals and datasets. Since we use the same type of classifier (MLP with the same architecture) and the same training and test sets the only influence in the results is from the loss functions. The experiments were conducted in a rigorously controlled way. The results were thoroughly evaluated.

2 The Datasets

We restricted ourselves to artificial and real-world two-class datasets. As artificial datasets we used checkerboard datasets such as the one shown in Figure 1. Checkerboard datasets are complex, controllable and unbalanced datasets. We used two different configurations: 2×2 and 4×4 checkerboards. For each one of

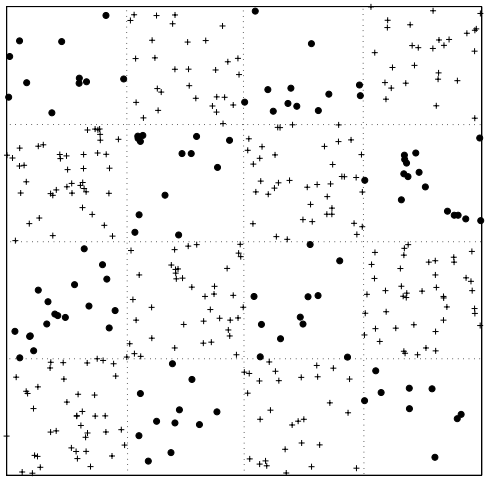

Fig. 1. An example of the 4×4 checkerboard dataset with 400 points (100 elements in the minority class: dots). Dotted lines are for visualization purpose only.

the configurations we built three datasets with different numbers of elements (points) but with a common characteristic: a fixed number of elements belonging to the minority class. The percentage of elements of this minority class is 50, 25 and 10% of the total number of elements. In Table 1 we show the different characteristics of the checkerboard datasets. CB2×2(200, 50) means "checkerboard 2×2 dataset with a total of 200 cases, 50% of them of the minority class"; likewise for the other checkerboard datasets.

Table 1. The artificial checkerboard datasets.

Data set	number of elements	number of elements per class
CB2×2(200,50)	200	100-100
CB2×2(400,25)	400	300-100
CB2×2(1000,10)	1000	900-100
CB4×4(200,50)	200	100-100
CB4×4(400,25)	400	300-100
CB4×4(1000,10)	1000	900-100

We used several real-world datasets summarized in Table 2. All datasets can be found in the UCI repository [6]. All these datasets differ a lot among them specially in what concerns the number of features and their topology. As a matter of fact and regarding this last aspect, the Wilks' lambda, a $[0, 1]$-separability measure related to the within-class over between-class sum of squares, varies -

as shown in Table 2 - from such a low value as 0.23 (Wdbc) indicating a quite good separability to such a high value as 0.86 (Liver) corresponding to a large class overlap.

Table 2. The 2-class real-world datasets.

Data set	number of elements	number of features	number of el. per class	Wilks' lambda
Cleveland Heart Disease 2	297	13	160-137	0.47
Diabetes	768	8	500-268	0.70
Ionosphere	351	34	225-126	0.38
Liver	345	6	200-145	0.86
Sonar	208	60	111-97	0.38
Wdbc	569	30	357-212	0.23

3 MLP Architectures

In all experiments with a given dataset we used the same MLP architectures, with as many inputs as the number of features, one hidden layer and a single output. The number of hidden neurons, n_h, was chosen in order to assure a not too complex network with acceptable generalization. For that purpose we used some criteria as guidelines and performed some preliminary experiments. As criteria we took into account the minimum number of lines needed to separate the checkerboard classes and the well-known rule of thumb $n_h = w/\epsilon$ (based on a formula given in [7]), where w is the number of weights and the expected error rate. The number of hidden neurons is given in Table 3. All neurons used the

Table 3. Number of hidden neurons, n_h.

Data set	n_h	Data set	n_h
Checkerboard 2×2	4	Ionosphere	4
Checkerboard 4×4	16	Sonar (10)*	10
Cleveland Heart Disease 2	2	Sonar (20)*	20
Diabetes	9	Liver	8
Wdbc	3		

(*)Same dataset but used in two n_h-different series of experiments

hyperbolic tangent as activation function. As risk functionals we used:

1. Mean square error (MSE):

$$R_\Phi = \sum_{\Omega} P(\omega) \int_{X,T} (t-y)^2 dF(t,x|\omega) \tag{4}$$

2. Cross-entropy (CE):

$$R_{\Phi} = \sum_{\Omega} P(\omega) \int_{X,T} (t \ln y + (1-t) \ln(1-y)) dF(t, x|\omega) \tag{5}$$

3. Generalized exponential (EXP):

$$R_{\Phi} = \sum_{\Omega} P(\omega) \int_{X,T} \tau e^{(t-y)^2/\tau} dF(t, x|\omega) \tag{6}$$

4. Shannon's entropy of the errors (HS):

$$R_{\Phi} = -\int_{E} \ln f(e) dF(e) \tag{7}$$

5. Quadratic Rényi's entropy of the errors (HR2):

$$R_{\Phi} = -\ln \int_{E} [f(e)] dF(e) \tag{8}$$

All these risks depend on the family of functions $\Phi = \{\varphi_w\}$ implemented by the MLP, where $W = \{w\}$ is the set of MLP weights.

4 The Experiments

As learning algorithm we used backpropagation (BP) of the errors. The practical implementations of this algorithm for all the risk functionals are described in the cited references. Note that for MEE the only available learning algorithm is BP; therefore, we had to use it for a fair comparison among all risk functionals. Regularization was performed by early stopping according to the same criterion, as follows: for each data set 10 runs were performed in order to determine the optimal number of epochs, Ep, (as well as the optimal smoothing parameter h for HS and HR2) for each method. The optimal Ep (h) were chosen as those values achieving the mean test error over the 10 runs. Initial values for h were selected using the formula given in [8]. The inputs were all pre-processed in order to standardize them to zero mean and unit variance. The support of the output target variable was $\{-1, 1\}$. In all experiments we used the 2-fold cross validation method. In this method in each run half of the data set is randomly chosen for training and the other half for testing. The datasets are next used with reverse roles (the original training set becomes the test set and vice-versa). Each experiment consisted of 20 runs of the algorithm. For this purpose twenty different random splits of the data sets were generated and stored. The *same* twenty different random splits were used as inputs for all MLPs with different risk functionals. This guaranteed that no differences in the results were due to different splits of the data sets. After the 20 runs the mean and standard deviation of the following performance measures were computed:

AUC: The area in percentage under the Receiver Operating Characteristic (ROC) curve, which measures the trade-off between sensitivity and specificity in two-class decision tables (for details, see e.g. [9]). The higher the area the better is the decision rule.

BCR: The balanced correct defined as $50\times TN/(TN+FP)+50\times TP/(FN+TP)$ in percentage (T=true; F=false; P=positive; N=negative).

COR: The classification correct rate in percentage.

The first two measures are based on the resulting 2×2 decision table, considering as "abnormal" class the one with lesser cases. They are specially suitable for unbalanced datasets, as the artificial checkerboard datasets, where an optimistically high COR could arise from a too high sensitivity or specificity. AUC and BCR give an adequate picture in those situations.

5 The Results

All results were ranked and subject to "multiple comparison among groups" (post-hoc one-way anova tests) statistical tests, using Tukey's least significant difference criterion when the test probability was less that the specified significance level (0.05), i.e., when the test was significant (rejecting the null hypothesis of equal means), and the more strict Tukey's honestly significant difference criterion, otherwise. Based on the statistical tests we were able to decide whether or not a functional that performed better was indeed significantly better (at that significance level) than others with worst performance. The results obtained for the 2×2 and 4×4 checkerboard datasets are shown in Table 4. In all tables average values are followed by standard deviations between parentheses. In all the tables the best average results are in bold; the statistically significant best are underlined. For the 2×2 checkerboard datasets the EXP and HS functionals performed better in general than all other functionals (both in average and variance). The sum of the ranks for the BCR performance index disclosed the following order from best to worst: EXP, HS, HR2, CE and MSE (*exaequo*). For the 4×4 checkerboard datasets the CE and EXP functionals performed better than all other functionals. The sum of the ranks for the BCR performance index disclosed the following order from best to worst: CE, EXP, MSE, HS and HR2 (*exaequo*). Table 5 shows the results for real-world datasets. These datasets are more challenging in terms of number of features, but less challenging in terms of class unbalance and class topology. For these datasets we chose to only look to the COR performance index. The sum of the ranks for the COR performance index disclosed the following order from best to worst: CE, EXP, HS, MSE, HR2. In general, taking into account the sum of ranks for the COR performance index for all datasets, the best functional was EXP.

Table 4. Results for 2×2 and 4×4 checkerboard datasets. Significantly best results underlined.

Dataset	CE	EXP	MSE	HS	HR2
CB2×2(200,50)					
AUC	97.58 (1.26)	97.89 (2.85)	**98.41** (1.08)	84.85 (17.07)	97.13 (4.51)
BCR	91.25 (2.50)	**92.94** (4.30)	92.87 (2.48)	86.96 (3.17)	92.48 (3.17)
CB2×2(400,25)					
AUC	98.21 (2.34)	98.89 (1.64)	96.41 (8.30)	**99.03** (1.32)	91.88 (10.82)
BCR	92.87 (2.35)	93.29 (3.61)	92.47 (4.73)	**94.36** (1.80)	90.70 (5.45)
CB2×2(1000,10)					
AUC	97.97 (2.77)	**98.94** (2.50)	62.72 (15.92)	95.07 (7.07)	96.15 (6.22)
BCR	83.40 (3.69)	**94.14** (5.16)	76.97 (6.51)	90.22 (5.66)	91.80 (4.87)
CB4×4(200,50)					
AUC	**85.01** (3.13)	83.89 (3.39)	80.89 (4.96)	77.32 (8.42)	74.39 (6.61)
BCR	**79.40** (3.20)	78.54 (3.57)	77.94 (4.39)	73.58 (4.96)	75.59 (5.45)
CB4×4(400,25)					
AUC	**89.95** (1.65)	84.11 (6.21)	76.96 (6.88)	71.41 (5.56)	70.63 (6.78)
BCR	**82.98** (1.78)	80.21 (3.05)	76.56 (3.96)	70.93 (4.03)	71.20 (3.69)
CB4×4(1000,10)					
AUC	**91.28** (3.06)	89.72 (4.13)	70.94 (5.37)	68.15 (3.23)	75.44 (5.45)
BCR	80.49 (1.98)	**81.47** (3.64)	70.77 (3.63)	67.98 (3.95)	73.04 (2.71)

Table 5. Results for real-world datasets. Significantly best results underlined.

Dataset	CE	EXP	MSE	HS	HR2
Clev. Heart Dis. 2					
COR	**83.33** (1.07)	81.72 (1.29)	82.42 (1.08)	82.72 (1.11)	81.77 (1.81)
Diabetes					
COR	**76.82** (0.77)	76.76 (0.79)	76.58 (0.88)	76.66 (0.85)	75.84 (0.78)
Ionosphere					
COR	88.04 (1.46)	88.37 (1.88)	87.81 (1.21)	87.71 (1.37)	**88.50** (1.33)
Liver					
COR	69.04 (2.01)	69.80 (1.27)	68.52 (1.97)	69.08 (1.86)	**70.32** (1.54)
Sonar (10)					
COR	77.93 (2.86)	**78.75** (2.84)	77.91 (2.96)	77.76 (2.57)	75.50 (4.35)
Sonar (20)					
COR	78.70 (2.60)	78.82 (2.92)	78.82 (2.51)	**79.18** (2.50)	77.43 (3.01)
Wdbc					
COR	**97.44** (0.55)	97.21 (0.68)	97.39 (0.67)	97.36 (0.66)	96.89 (0.42)

6 How Far from Optimality

We have of course no way to know the $\min_W Pe_\Phi$ value for the real-world datasets. However, we may find it instructive to compare the MLP results with the best results achievable by a battery of more sophisticated classifiers that do not depend on a stochastic iterative process as BP. For that purpose the same datasets were submitted to the following classifying algorithms:

1. PLS - Partial Least Squares algorithm described in [10, 11] with five latent variables.
2. K-PLS - Kernel Partial Least Squares algorithm described in [10, 11] with five latent variables.
3. LS-SVM - Least Squares Support Vector Machine algorithm, described in [12–14].
4. SVMR - The classical Support Vector Machine algorithm applied in regression mode, described in [15–17].
5. SVMc - The classical Support Vector Machine algorithm, described in [15–17].

For the K-PLS and SVM algorithms the Gaussian kernel was used. Table 6 allows comparing the best MLP BCR result with the best BCR result obtained with the above algorithms (table header "Other") for the checkerboard datasets (with the algorithm number between brackets). Table 7 does the same for the real-world datasets and for COR.

Table 6. MLP versus "Other" (checkerboard datasets). Significantly best results underlined.

Dataset	MLP	Other
CB2×2(200,50)	**92.94** (4.30)	92.43 (2.22) (2)
CB2×2(400,25)	**94.36** (1.80)	92.44 (1.88) (2)
CB2×2(1000,10)	**94.14** (5.16)	92.55 (2.42) (5)
CB4×4(200,50)	79.40 (3.20)	**83.31** (3.02) (4)
CB4×4(400,25)	82.98 (1.78)	**85.24** (1.71) (2)
CB4×4(1000,10)	81.47 (3.64)	**82.89** (2.64) (3)

7 Conclusions

In the checkerboard experiments the EXP and CE functionals were the ones with better achievement, with high values of AUC and BCR even in presence of severe data unbalance. The HS functional performed very well in the 2×2 cases but poorly in the 4×4. The best functionals for the real-world datasets were CE and EXP. HS also behaved quite well achieving in general better performance

Table 7. MLP versus "Other" (real-world datasets). Significantly best results underlined.

Dataset	MLP	Other
Clev. Heart Dis. 2	**83.33** (1.06)	83.31 (0.97) (5)
Diabetes	76.82 (0.77)	**76.97** (0.87) (1)
Ionosphere	88.50 (1.33)	**95.00** (0.62) (3)
Liver	70.32 (1.54)	**70.84** (1.63) (3)
Sonar (10)	78.75 (2.84)	**85.07** (1.93) (3)
Sonar (20)	79.18 (2.50)	**85.07** (1.93) (3)
Wdbc	**97.44** (0.55)	97.28 (0.45) (3)

than MSE. The bottom line conclusions drawn from our experiments are as follows: EXP is a very performing functional in a large variety of datasets, which can be explained for its capability of emulating a large class of functionals [1]; EXP, CE and HS seem definitely well performing algorithms only superseded in rare cases by MSE or HR2; the entropic functionals have sometimes a remarkable performance (e.g., for CB2×2(400,25), Ionosphere, Liver, Sonar20), although the characterization of such situations is for now unclear. When comparing the best classification results attained with MLPs with those attained with (generally considered) more sophisticated algorithms we conclude from our experiments that often MLPs will compete and outperform those ones. This may come as a surprise for those who perhaps thought that MLPs was an outdated approach. On the other hand, if we regard the best performance obtained as a sort of upper bound of the unknown $\min_W Pe_\Phi$, then our results provide some evidence that an MLP with a suitable risk functional will often be able to reach $\min_W Pe_\Phi$. (We say "upper bound" because the non-MLP algorithms correspond to more complex Φ-function families.) The suitability of algorithms to datasets is a tricky subject, given the many factors that come into play and the many points of view one may have on the matter. As to the former one may for instance ask whether or not it is fair to compare all algorithms with the same architecture and features; it may happen that one algorithm will have better performance in more complex architectures than other competing ones, or that it will cope better with redundant features than competing ones. As to the latter, one could consider being interested in looking to other performance indexes than the ones we have used. Nevertheless, the suitability of algorithms to datasets is an interesting and useful subject and a central topic in metalearning issues. The existing evaluation and comparison studies carried out with care and with a statistically sound background are scarce. One of the reasons of such scarcity is also that the experiments involved are much time consuming and demand a variety of resources that are not often available. The present paper is just one contribution to this topic that we intend to widen in the future.

Acknowledgments This work was supported by the Portuguese FCT-Fundação para a Ciência e a Tecnologia (project POSC/EIA/56918/2004). First author is also supported by FCT's grant SFRH/BD/16916/2004.

References

1. Silva, L.M., Alexandre, L.A., Marques de Sá, J.: Data classification with multilayer perceptrons using a generalized error function. Neural Networks (2008) doi:10.1016/j.neunet.2008.04,004.
2. Erdogmus, D., Príncipe, J.C.: Generalized information potential criterion for adaptive system training. IEEE Transactions on Neural Networks **13**(5) (2002) 1035–1044
3. Erdogmus, D., Príncipe, J.C.: An Error-Entropy Minimization Algorithm for Supervised Training of Nonlinear Adaptive Systems. IEEE Transactions on Signal Processing **50**(7) (2002) 1780–1786
4. Santos, J.M., Alexandre, L.A., Marques de Sá, J.: The Error Entropy Minimization Algorithm for Neural Network Classification. In: Int. Conf. on Recent Advances in Soft Computing, Nottingham, United Kingdom (2004)
5. Duda, R., Hart, P., Stork, D.: Pattern Classification. Wiley-Interscience (2001)
6. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI repository of machine learning databases. (1998)
7. Baum, E.B., Haussler, D.: What size net gives valid generalization? Neural Computation **1**(1) (1990) 151–160
8. Santos, J.M., Marques de Sá, J., Alexandre, L.A.: Neural Networks Trained with the EEM Algorithm: Tuning the Smoothing Parameter. WSEAS Transactions on Systems **4**(4) (2005) 295–300
9. Marques de Sá, J.
10. Embrechts, M.J., Szymanski, B., Sternickel, M.: Introduction to Scientific Data Mining: Direct Kernel Methods and Applications. In: Chapter 10 in Computationally Intelligent Hybrid Systems. Wiley Interscience (2004) 317–363
11. Bennett, K.P., Embrechts, M.J.: An Optimization Perspective on Kernel Partial Least Squares Regression. In: Chapter 11 in Advances in Learning Theory: Methods, Models and Applications. NATO Science Series, Series III: Computer and System Sciences - Vol. 190, IOS Press (2003) 227–249
12. Suykens, J.A.K., Vandewalle, J.: Least squares support vector machine classifiers. Neural Processing Letters **9**(3) (1999) 293–300
13. Suykens, J.A.K., Gestel, T.V., Brabanter, J.D., Moor, B.D., Vandewalle, J.: Least Squares Support Vector Machines. World Scientific Pub Co (2003)
14. Keerthi, S.S., Shevade, S.K.: SMO algorithm for least squares SVM formulations. Neural Computation **15** (2003) 487–507
15. Cristianini, N., Shawe-Taylor, J.: Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press, Cambridge, MA (2000)
16. Vapnik, V.: Statistical Learning Theory. Wiley-Interscience (1998)
17. Schölkopf, B., Smola, A.J.: Learning with Kernels. MIT Press, Cambridge, MA (2002)