ARTIFICIAL NEURAL NETWORKS IN FRUIT SORTING

SORTAREA FRUCTELOR CU AJUTORUL REȚELELOR NEURONALE ARTIFICIALE

DIACONESCU RODICA, DUMITRU MARIANA
Technical University “Gh.Asachi’, Iaşi

Abstract. *The goal of this paper is to present the solution to a typical application of artificial neural networks problem classification. The classification problem and NeuroSolutions for Excel are described in detail. An application for fruits sorting is presented. Accurate results were obtained and were discussed.*

Rezumat. *Se defineşte problema clasificării cu ajutorul rețelelor neuronale artificiale. Se dezvoltă aplicația pentru sortarea fructelor, în Matlab şi Neurosolutions. Se descriu rezultatele obținute şi se compară cele două tehnici de lucru utilizate.*

INTRODUCTION

Neural networks have seen an explosion of interest over the last few years, and are being successfully applied across an extraordinary range of problem domains, in areas as diverse as finance, medicine, engineering, geology, horticulture, agriculture and physics. Indeed, anywhere that there are problems of prediction, classification or control, neural networks are being introduced. The technical literature comprises a large number of papers which deals with fruits or vegetables sorting using artificial neural networks. The most applications are solved in *Matlab*.

In this study, considering that *Microsoft Excel* is a software used on a very large scale with a high degree of recognition, the classification problem is described. Then, an application for fruits sorting is exemplified.

FUNDAMENTALS

- *Classification problem*

In *classification problems*, the purpose of the network is to assign each case to one of a number of classes (or, more generally, to estimate the probability of membership of the case in each class). The artificial neural networks must approximate a function in order to find the appropiate output for each set of inputs. The output is a class, thus they are discret.

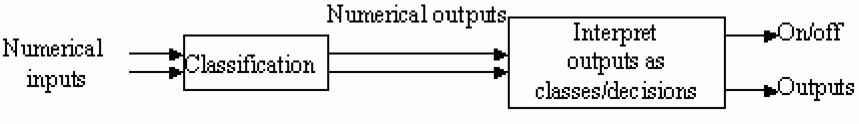


Figure 1 - Schematic representation of *classification problem*

Classification problems are evaluated by thresholding the outputs of the model.

Classification neural networks must translate the numeric activation level of the output neuron(s) to a nominal output variable. There are two very different approaches to assigning classifications. One of these, where it is the activation level of the output layer units, which are not radial units, that determines the class.

Two cases need to be distinguished: single output neuron versus multiple output neurons. Single output neurons are typically used for two-class problems, with a high output neuron level indicating one class and a low activation the other class. This configuration uses the Two-state conversion function.

Multiple output neurons are typically used for three or more class problems. One neuron is used for each class, and the highest activation neuron indicates the class. The neuron activation levels can be interpreted as confidence levels. This method is implemented by using the One-of-N conversion function for the output variable. You can optionally configure a multilayer perceptron to use two output neurons for a two-class output variable.

- *NeuroSolutions for Excel presentation*

NeuroSolutions for Excel allows we to access the power of *NeuroSolutions* as a neural network environment while working within a familiar spreadsheet environment.

We will train a neural network to classify objects. To accomplish this, we tag the columns that we want to use as inputs. Similarly, we tag the desired output columns.

The rows of data must also be tagged before a training process can be run. Rows can be tagged as *Training*, *Cross Validation*, *Testing*, or *Production*. Only the *Training* tag is required for running a training process. However, cross validation is a very useful tool for preventing over-training, so in most cases we will also want to tag a portion of our data as *Cross Validation*. The rows of data to use for testing the trained network (*Testing*) or producing the network output for new data (*Production*) can be tagged before or after the training process is run.

A pre-saved one hidden layer multilayer perceptron (*MLP*) will be opened within *NeuroSolutions* and used for modeling this data. When a training process is run, files are automatically created (using the tagged cross-sections) and loaded into the active *NeuroSolutions* network. The dialog box will be preset to use cross validation and train for 200 epochs.

A report has been generated within the worksheet named *Train1 Report*. This report summarizes the training results. The plot shows the learning curves for both the training and cross validation data sets. The table gives the epoch at which the training and cross validation mean-squared errors (*MSE*s) were minimum, the values of these minimum *MSE*s, and the values of the *MSE*s at the last epoch. The training *MSE* and cross validation *MSE* were collected for each epoch within the worksheet named *Train1 MSE*. The actual learning curves will vary with each run of the application since the weights are randomized at the beginning of a training run. The mean squared error was computed using the following formula:

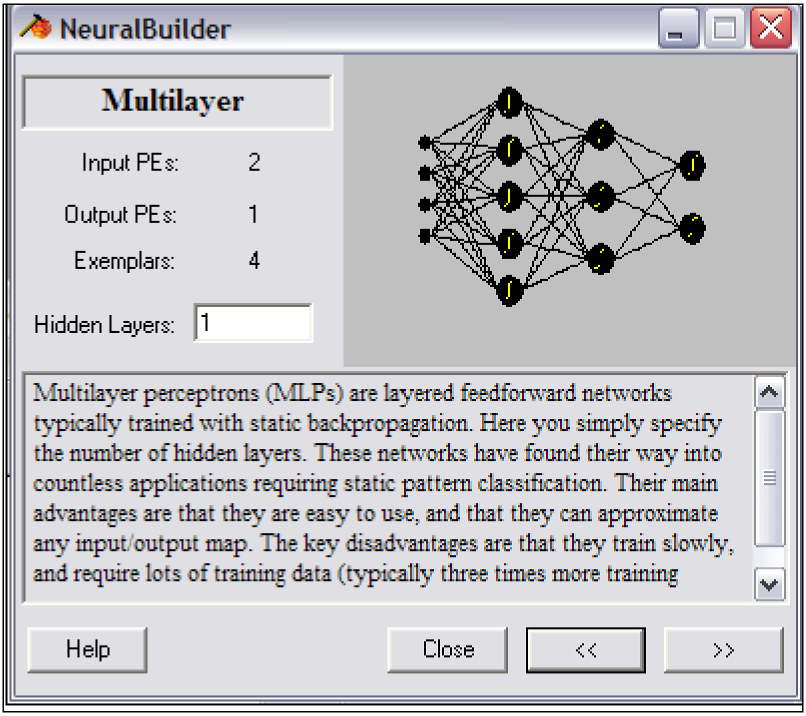
$$MSE = \frac{\sum_{j=1}^{P}\sum_{i=1}^{N}\left(d_{ij} - y_{ij}\right)^2}{N \cdot P}$$

where P is the number of output processing elements (in this case, $P=1$), N is the number of exemplars in the data det, y_{ij} is the network output for exemplar i at processing element j, and d_{ij} is the desired output for exemplar i at processing element j.

In this study we will determine the classification performance of the *Training* data set. This will tell us whether the neural network memorized the training data or truly learned to model the underlying relationship.

A classification report has been generated within the active worksheet. The first thing we will notice on this report is a *confusion matrix*. The *confusion matrix* summarizes the classification results in an easy to interpret format. This report also includes a table listing various performance measures. Notice the last measure gives the percentage of samples classified correctly for each class.

Again a classification report has been generated within the active worksheet. This report summarizes the classification results for the testing data set. We can see that we have developed a good model for this data. If we want to examine the actual network output, we can go to the data sheet created by this testing process.


NeuralBuilder

Multilayer

Input PEs: 2

Output PEs: 1

Exemplars: 4

Hidden Layers: 1

Multilayer perceptrons (MLPs) are layered feedforward networks typically trained with static backpropagation. Here you simply specify the number of hidden layers. These networks have found their way into countless applications requiring static pattern classification. Their main advantages are that they are easy to use, and that they can approximate any input/output map. The key disadvantages are that they train slowly, and require lots of training data (typically three times more training

Figure 2 - Neural Builder window where the *MLPs* are defined

Three ways in which we can create a neural network for use with *NeuroSolutions for Excel* exist. The primary and recommended method is to use the *NeuralBuilder* included with *NeuroSolutions*. A second method is to create the network manually by stamping the components onto the breadboard, connecting the components, and configuring their parameters by hand. Finally, networks built using the *NeuralExpert* (included with *NeuroSolutions*) can also be used.

The *NeuralBuilder* has a wizard-like interface which guides we step by step through the creation of a neural network. First, we choose the network model. There are several models to choose from ranging from a *Multilayer Perceptron* to a *Support Vector Machine*. Then, for the model selection, we select the network parameters (such as the # *of PEs, Step Sizes*, etc.). In this step, we will create a network which can be used to solve the *XOR problem*. The *Neural Model* panel will be displayed with *Multilayer Perceptron* selected by default.

RESULTS AND DISCUSSIONS

Problem statement

A produce dealer has a warehouse that stores a variety of fruits and vegetables. When fruit is brought to the warehouse, various types of fruit may be mixed together. The dealer wants a machine that will sort the fruit according to type. There is a conveyer belt on which the fruit is loaded. This conveyer passes through a set of sensors, which measure three properties of the fruit: *shape*, *texture* and *weight*.

The three sensor outputs will then be input to a neural network. The purpose of the network is to decide which kind of fruit is on the conveyor, so that the fruit can be directed to the correct storage bin. To make the problem even simpler, lets assume that there are only two kinds of fruit on the conveyor: apples and pears.

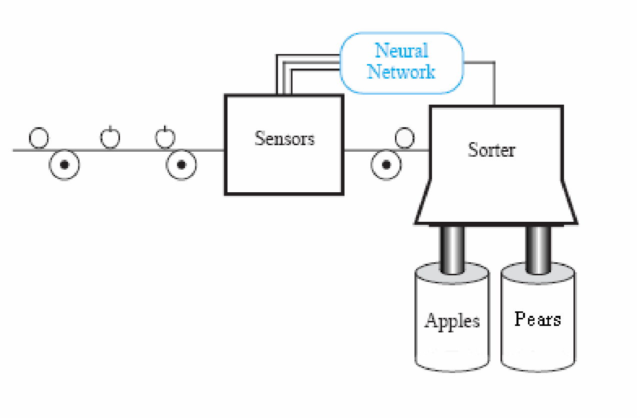


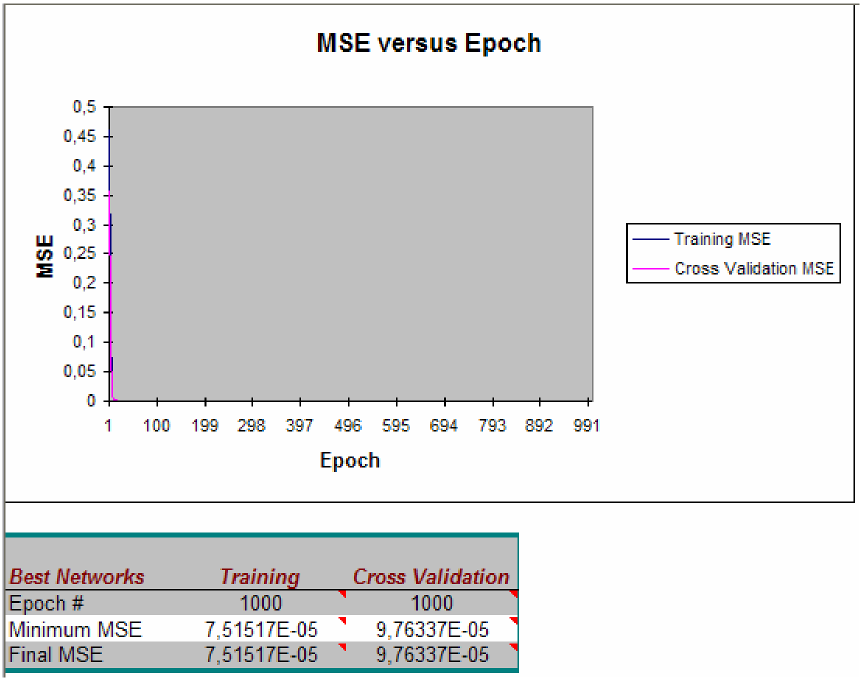
Figure 3 - Fruits sorting plant

We will train a neural network to classify fruits either apple or pear. There are 20 apples and 21 pears for each of two classes for a total of 41 fruits. The columns labeled *Diametru*, *Masa*, *Densitate*, will serve as inputs to the neural network and the columns labeled *Apples* and *Pears* will serve as the desired outputs.

The first 20 rows will be used for training and the next 10 rows will be used for cross validation. A pre-saved one hidden layer multilayer perceptron (MLP) is used for modeling this data. *Train1 Report* is presented in Figure 4.

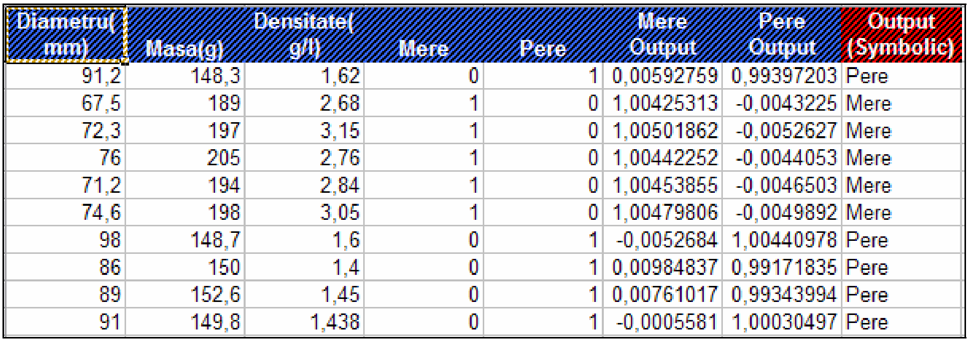
Then we will test the classification performance of data that the network has never seen. The last 10 rows of the data set have been tagged as *Testing* and will be used for this purpose.

Figures 5 and 6 contain *Test1 IO Data* and *Test1 Report*.



Best Networks	Training	Cross Validation
Epoch #	1000	1000
Minimum MSE	7,51517E-05	9,76337E-05
Final MSE	7,51517E-05	9,76337E-05

Figure 4 - Train1 Report

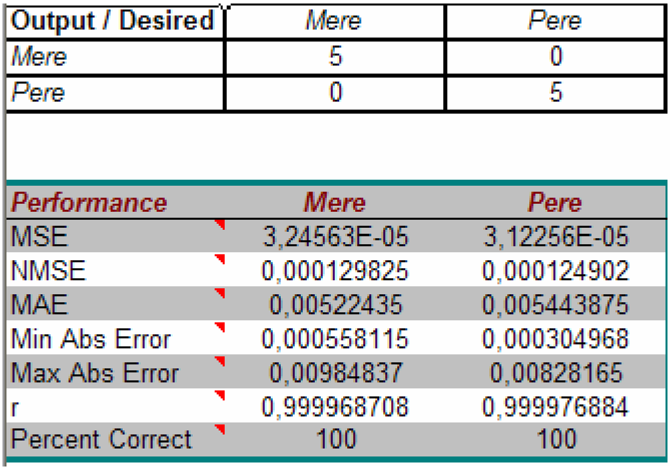


Diametru(mm)	Masa(g)	Densitate(g/l)	Mere	Pere	Mere Output	Pere Output	Output (Symbolic)
91,2	148,3	1,62	0	1	0,00592759	0,99397203	Pere
67,5	189	2,68	1	0	1,00425313	-0,0043225	Mere
72,3	197	3,15	1	0	1,00501862	-0,0052627	Mere
76	205	2,76	1	0	1,00442252	-0,0044053	Mere
71,2	194	2,84	1	0	1,00453855	-0,0046503	Mere
74,6	198	3,05	1	0	1,00479806	-0,0049892	Mere
98	148,7	1,6	0	1	-0,0052684	1,00440978	Pere
86	150	1,4	0	1	0,00984837	0,99171835	Pere
89	152,6	1,45	0	1	0,00761017	0,99343994	Pere
91	149,8	1,438	0	1	-0,0005581	1,00030497	Pere

Figure 5 - Test2 IOData

This report also includes the last measure gives the percentage of samples classified correctly for each class and it is equal with 100%. Thus, the classification problem solution is optimal.

Finally, we use the the neural network on a new set of data, for *production*. The verified fruits are correctly enclosed to the class that its belongs.



Output / Desired	Mere	Pere
Mere	5	0
Pere	0	5

Performance	Mere	Pere
MSE	3,24563E-05	3,12256E-05
NMSE	0,000129825	0,000124902
MAE	0,00522435	0,005443875
Min Abs Error	0,000558115	0,000304968
Max Abs Error	0,00984837	0,00828165
r	0,999968708	0,999976884
Percent Correct	100	100

Figure 6 - Test1 Report

CONCLUSIONS

In this study an approach for fruits sorting and its implementation in *NeuroSolutions for Excel* are presented.

A optimal classification even under fruits diammeters, mass and density is provided by use of an artificial neural network as a fruits classificator. The approach has proved to be robust with respect to caracteristiques variations and consequently highly applicable to the proposed field. It also can easily be applied to sorting systems of other vegetables or fruits.

REFERENCES

1. **Hagan M.T., Demuth H., Beale M., 1996 -** *Neural Network Design*. PWS Publishing, Boston.
2. **Haykin S., 1995 -** *Neural Networks: A Comprehensive Foundation*. 2nd ed. New York: Prentice-Hall.
3. **Munakata T., 1998** - *Fundamentals of the New Artificial Intelligence–Beyond Traditional Paradigms*. Berlin,Germany, Springer-Verlag.