Classifying seeds type by using feedforward neural network

Zhuoheng Xie
School of Information Science
University of Pittsburgh
zhx21@pitt.edu

ABSTRACT

In this paper Feedforward neural network is implemented in Matlab to do a classification task where three types of seeds, Kama, Rosa and Canadian, are classified.

Keywords

Feedforward, neural network, Matlab, classification.

1. BACKGROUND

The feedforward neural network was the first and simplest type of artificial neural network devised. In this network, the information moves in only one direction, forward, from the input nodes, through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network, which is different from recurrent neural networks.[1]

In this paper, I used single-layer perceptron. Since it only consists of a single layer of output nodes, a single-layer perceptron network is considered as the simplest kind of neural network. The inputs are fed directly to the outputs via a series of weights. The sum of the products of the weights and the inputs is calculated in each node, and if the value is above some threshold (typically 0) the neuron fires and takes the activated value (typically 1); otherwise it takes the deactivated value (typically -1). In a feed forward network information always moves one direction; it never goes backwards. Figure 1 is a picture of the network.

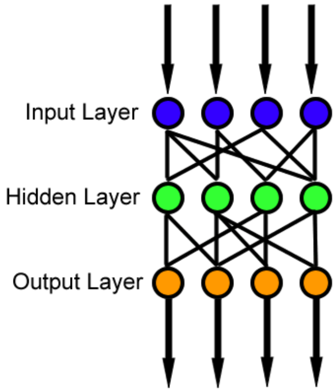


Figure 1. A single-layer feedforward neural network.

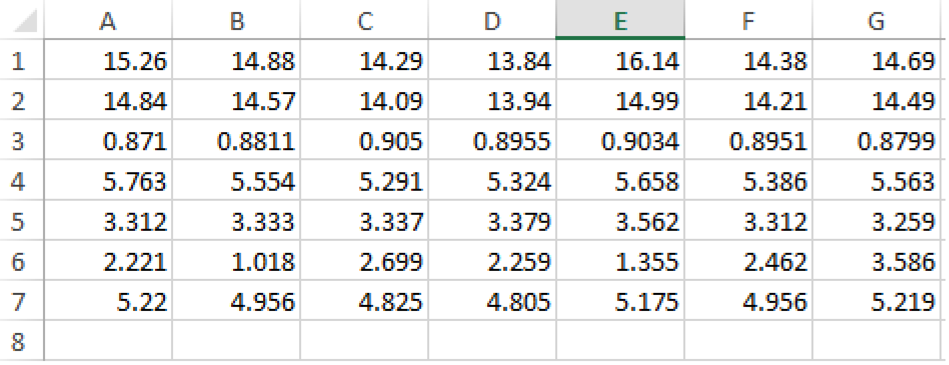
2. METHOD DESCRIPTION

Here, a single-layer feedforward network is created to do classification task. The network not only classifies the known seeds, but also can generalize to accurately classify seeds that were not used to design the solution.

2.1 Pre-processing

Before importing data into Matlab, I preprocessed origin dataset so that it can be used as two matrices, the input matrix X and the target matrix T.

There are 8 columns, 210 rows in origin dataset. The first seven columns are the attributes of the seed, and the eighth column is the type of the seed. First of all, I transposed the dataset in Matlab. Thus, a new dataset is formed by 8 rows and 210 columns. Second, I separated the dataset into two datasets. One contains the first seven rows which are seven attributes. And it is saved as "seeds_x.txt". Here is part of the snapshot of the excel file to explicate.

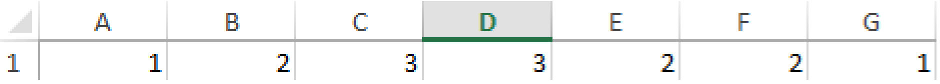


	A	B	C	D	E	F	G
1	15.26	14.88	14.29	13.84	16.14	14.38	14.69
2	14.84	14.57	14.09	13.94	14.99	14.21	14.49
3	0.871	0.8811	0.905	0.8955	0.9034	0.8951	0.8799
4	5.763	5.554	5.291	5.324	5.658	5.386	5.563
5	3.312	3.333	3.337	3.379	3.562	3.312	3.259
6	2.221	1.018	2.699	2.259	1.355	2.462	3.586
7	5.22	4.956	4.825	4.805	5.175	4.956	5.219
8							

Figure 2. Part of dataset seeds_x.

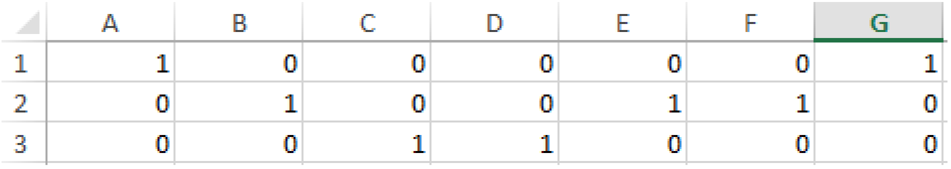
Seven rows stand for seven attributes. A, B, C, D, E, F and G are all instances with seven attibutes.

The other dataset contains the eighth row of the data which is the type of the seed. And it is saved as "seeds_t.txt". However, there is one row in "seeds_t.txt" so far. And three variables, 1, 2, and 3 are relatively stand for three type. In order to form target matrix, I generated three rows where first row stand for type 1, second for type 2 and third for type 3. If the type is true, the variable is 1. If it is wrong, the variable is 0. Here is a simple example to explicate.



	A	B	C	D	E	F	G
1	1	2	3	3	2	2	1

Figure 3. Data before adjusting, only one row in file. 1, 2 and 3 stand for each type.



	A	B	C	D	E	F	G
1	1	0	0	0	0	0	1
2	0	1	0	0	1	1	0
3	0	0	1	1	0	0	0

Figure 4. After adjusting, three rows to form target matrix.

As shown in the table above, Neuron A and Neuron G are Type 1. B, E and F are Type 2. C and D are Type 3.

Now, I have prepared two datasets which can be imported into Matlab as two matrices, input matrix x and target matrix t.

2.2 Codes in Matlab

The following is the code for total solution including transposing origin dataset, importing data, set up feedforward network, training and evaluation steps. The analysis of the result will be presented in Chapter 4.

```
>> seeds_data=importdata('seeds_data.txt')
%%import the origin dataset
>> seeds_data=seeds_data'
%%transpose the origin dataset
>> x=importdata('seeds_x.txt')
%%importdata to get input matrix
>> t=importdata('seeds_t.txt')
%%importdata to get target matrix
>> size(x)
%%view the size of inputs x, 7 rows, 210 columns
>> size(t)
%%view the size of target t, 3 rows, 210 columns
>>  net = feedforwardnet(10);
%%build the feedforward network with 10 neurons in single hidden layer. Also tried 1 to
30.
>> view(net)
%%view the network
>> [net,tr] = train(net,x,t);
%%train the network
>> plotperform(tr)
%%plot performance graph to see how network's performance improved during training,
to see the mean squared error (MSE) of the trained neural network
>> testX = x(:,tr.testInd);
%%generate test set
>> testT = t(:,tr.testInd);
%%generate test set
>> testY = net(testX);
%%test the network
```

```
>> testIndices = vec2ind(testY)
%%use vec2ind function to get the class indices as the position of the highest element in each output vector
>> plotconfusion(testT,testY)
%% plot confusion matrix to see the performance
>> [c,cm] = confusion(testT,testY);
>> fprintf('Percentage Correct Classification   : %f%%\n', 100*(1-c));
%% print the accuracy percentage
>> plotroc(testT,testY) %%plot ROC graph to see the performance
%% generate ROC graph
```

3. PROBLEM DESCRIPTION

3.1 Classification

Classification is popular in machine learning projects. It assigns items in a collection to target categories or classes. It is widely used in today's business. For example, banks can classify customers into high, medium or low credit levels based on their profiles, finance status and debt history. In this paper, the task is to classify three wheat seeds by given 7 attributes and 210 elements.

3.2 Dataset Information

The dataset[2] is from studies at the Institute of Agrophysics of the Polish Academy of Sciences in Lublin. They used a soft X-ray technique to get high quality visualization of the internal kernel structure of three different varieties of wheat: Kama, Rosa and Canadian, 70 elements each.[3]

To construct the data, 7 geometric parameters of wheat kernels were measured:

1. area A,
2. perimeter P,
3. compactness C = 4*pi*A/P^2,
4. length of kernel,
5. width of kernel,
6. asymmetry coefficient
7. length of kernel groove.

So the dataset can be very well used for the tasks of classification. In the single-layer feedforward neural network, every element is an input, and the type of the element is the output.

4. RESULTS

Three methods are used to evaluate the network's performance. Three methods are plotting performance graph, generating confusion matrix and ROC graph. Here we just take one trial when the number of neurons in hidden layer (N) is 12. Because each trail may get different result even N remains same. The comparisons of performances in different trials with different N will be discussed in Chapter 5.

4.1 Performance graph

Plotting performance graph can see how network's performance improved during training and see the mean squared error (MSE) of the trained neural network.[4] MSE masseurs the average of the squares of the "errors", that is, the difference between the estimator and what is estimated. In our case, the bias can be only 0 or 1 for each estimator because the outputs have only two values, 0 and 1. In Figure 5, the MSE hit the lowest point when the epoch is 5. And the MSE = 0.021156, which is a very good result.

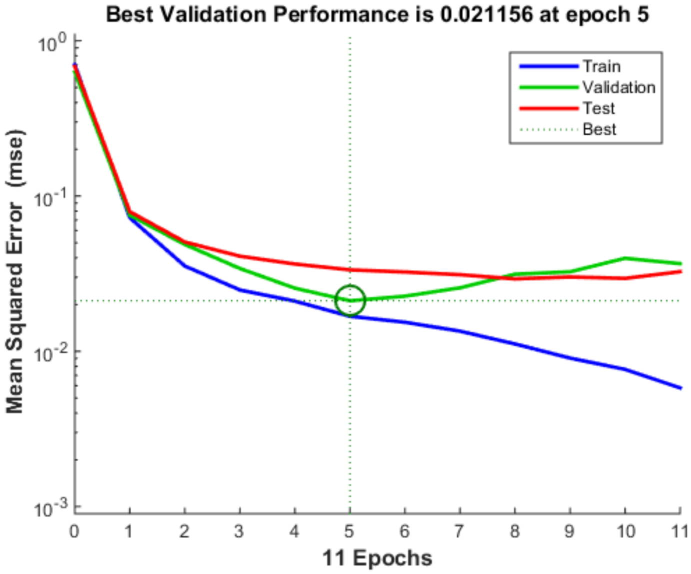
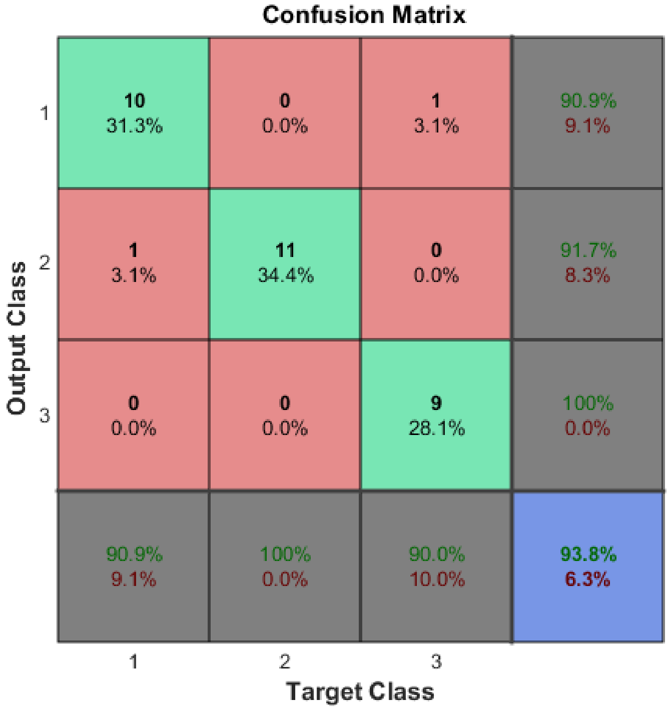


Figure 5. Performance graph with MSE (N=12).

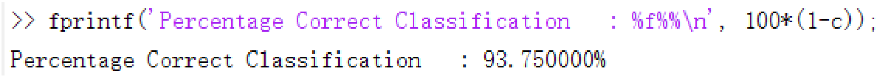
4.2 Confusion Matrix

A confusion matrix is a specific table layout that allows visualization of the performance of an algorithm. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. The name stems from the fact that it makes it easy to see if the system is confusing two classes.[5] In our case, we generate the accuracy from the Confusion Matrix where accuracy = 93.75%.



Output Class \ Target Class	1	2	3	
1	**10** 31.3%	**0** 0.0%	**1** 3.1%	90.9% 9.1%
2	**1** 3.1%	**11** 34.4%	**0** 0.0%	91.7% 8.3%
3	**0** 0.0%	**0** 0.0%	**9** 28.1%	100% 0.0%
	90.9% 9.1%	100% 0.0%	90.0% 10.0%	**93.8%** **6.3%**

Figure 6. Confusion Matrix (N=12).



```
>> fprintf('Percentage Correct Classification   : %f%%\n', 100*(1-c));
Percentage Correct Classification   : 93.750000%
```

Figure 7. Accuracy (N=12)

4.3 ROC

A receiver operating characteristic (ROC), or ROC curve, is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. The curve is created by plotting the true positive rate against the false positive rate at various threshold settings. The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test. Thus, in our test, the performance is very good because all the classes' curves are much above the 45-degree diagonal.

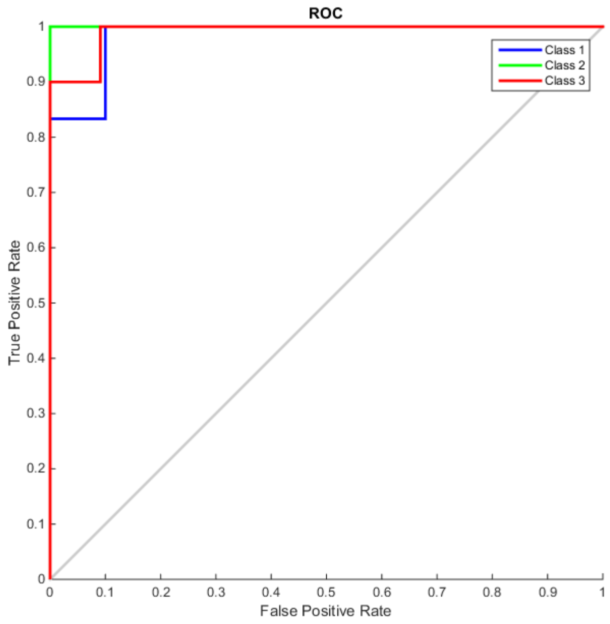


Figure 8. ROC graph (N=12)

5. CONCLUSION

In sum, the single-layer feedforward neural network works very well in this classification task. In many trials, the accuracy achieves above 96.875% and sometimes even get 100%. I also get some findings during trials.

First, when the number of neurons in hidden layer increases from 5 to 20 and even larger, the accuracy of test does not change very much. All of them can get 93.75% and 96.875%. When number of neurons in hidden layer is less than 5, decreasing from 5 to 1, the accuracy will fall quickly to 60% and lower. This may indicate that 5 neurons in hidden layer are enough to classify nodes in this case.

Also, when the number of neurons in hidden layer gets larger, the responding time we need to train the model increases. Changes are very slight when the number ranges from 1 to 20, mainly because all the responding time is very fast. However, when the number of neurons exceeds 30, I can feel the speed slow down intuitively. This reflect that the number of neurons in hidden layer is not the larger the better. We also need to consider efficiency and effectiveness when we train the network.

ACKNOWLEDGMENTS

Thanks to the dataset source from Institute of Agrophysics of the Polish Academy of Sciences in Lublin and also Matlab library tutorials.

REFERENCES

[1] Feedforward neural network
http://en.wikipedia.org/wiki/Feedforward_neural_network

[2] Dataset source
http://archive.ics.uci.edu/ml/datasets/seeds

[3] M. Charytanowicz, J. Niewczas, P. Kulczycki, P.A. Kowalski, S. Lukasik, S. Zak. *A Complete Gradient Clustering Algorithm for Features Analysis of X-ray Images.* Information Technologies in Biomedicine, Ewa Pietka, Jacek Kawa (eds.), Springer-Verlag, Berlin-Heidelberg, 2010, pp. 15-24.

[4] Mean squared error
http://en.wikipedia.org/wiki/Feedforward_neural_network

[5] Confusion matrix
http://en.wikipedia.org/wiki/Confusion_matrix

[6] Receiver operating characteristic
http://en.wikipedia.org/wiki/Receiver_operating_characteristic