

**ELEC320 – Neural Network**

Department of Electrical Engineering & Electronics

(2020 - 2021)

**Assignment\_Report**

- Multi-Layer Perceptrons-

**Task 1.**

**1)**

Initial weights

Input

A. **Neuron 1:**

Induced local field

Output

B. **Neuron 2:**

Induced local field

Output

C. **Neuron 3:**

Induced local field

Output

**2)** Desired output and run backpropagation

A. **Neuron 3:**

The local gradient for Neuron 3

The first thing we need is the error at this Neuron

The second factor to calculate is the derivative

1

Just need to substitute the values

B. **Neuron 2:**

The local gradient for Neuron 2

The first factor to calculate is the derivative

Just need to substitute the values

C. **Neuron 1:**

The local gradient for Neuron 1

The first factor to calculate is the derivative

Just need to substitute the values

D. **Derivative cost function w.r.t weights**

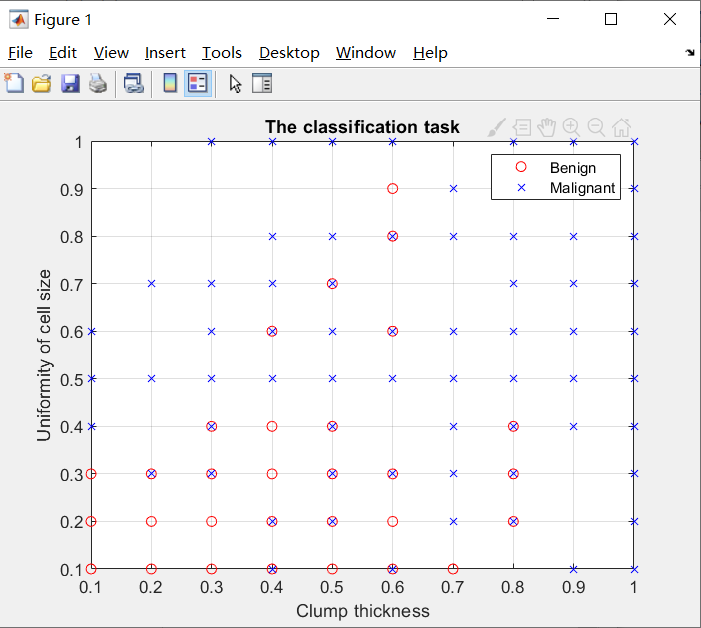
First consider

**3)** Updated weights with learning rate **η** = 0.1

**4)** By running the network forward with the updated weight, the weight will return . Compared with the previous output , from which it can be seen that the output of the update weights is closer to the desired output of , as expected from the training mode, for the network to be well trained.

**Task 2.**

**1) Perform the classification task**



**Figure 2.1:** The Classification task

In the cancer data set, x represents the feature vector of 9 dimensions, and t is the expected output of the feature vector, which is the target category, divided into benign and malignant. Figure 2.1 is an attempt to classify benign and malignant with the first two dimensions of x. The classification is done by multiplying the first two matrices of x by the benign and malignant matrices respectively to obtain the benign for the red circle and the malignant for the blue cross. Through classification, it can be found that it is not easy to see the difference based on the first two functions. In the range of benign features, there are also malignant.

**2) Calculate the number of data points, among which how many are benign and malignant**



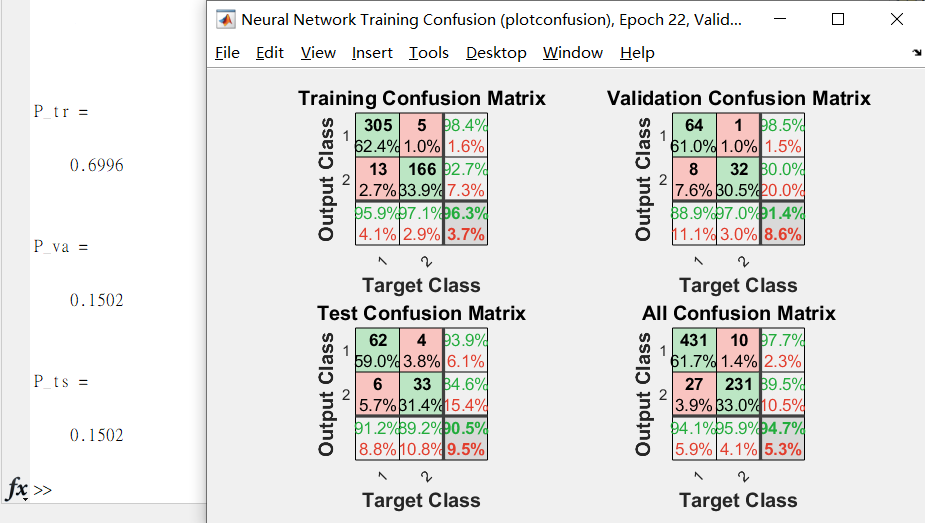
**Figure 2.2:** The number of the total data point, Benign and Malignant

Figure 2.2 is obtained by using the ‘size ()’ function to obtain the size of the benign and malignant matrices in Figure 2.1, and then multiplying the rows and columns. There are 458 data points for benign and 241 data points for malignant. The sum of benign and malignant is the total number of data points in the data set, which is 699.

**3) The purpose of the training set, the validation set and the test set**

The training set is used to debug the neural network, through training and fitting the model and determining the parameters, usually accounting for 60-70% of the data, in order to reflect the complexity and diversity of the model. The validation set is used to view the training effect, and to evaluate the performance of the model when adjusting the hyperparameters of the model. The test set is used to test the actual learning ability of the network, through unbiased evaluation of the final model fitting and performance in the training set. In addition, the division of the three data sets is to prevent the model from overfitting.

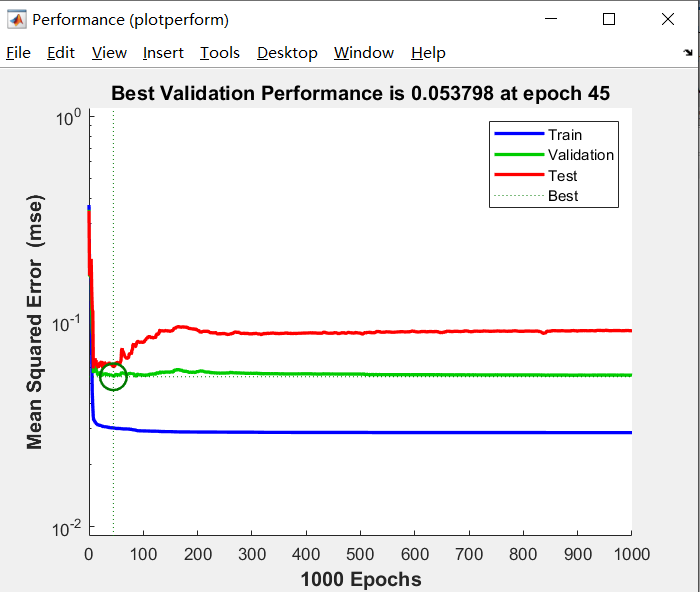
**4) Create and train a network with N=5**



**Figure 2.4:** The percentage of data points in training set, validation set and test set

Figure 2.4 shows the percentage of data points obtained by dividing the amount of data used in tr.trainInd, tr.valInd and tr.testInd by the amount of total data point using the ‘numel ()’ function for the training set, validation set and test set respectively. The percentage of data points in the training set is 69.96%. The percentage of data points in the validation set is 15.02%. The percentage of data points in the test set is 15.02%. In addition, these percentages will not change as N changes.

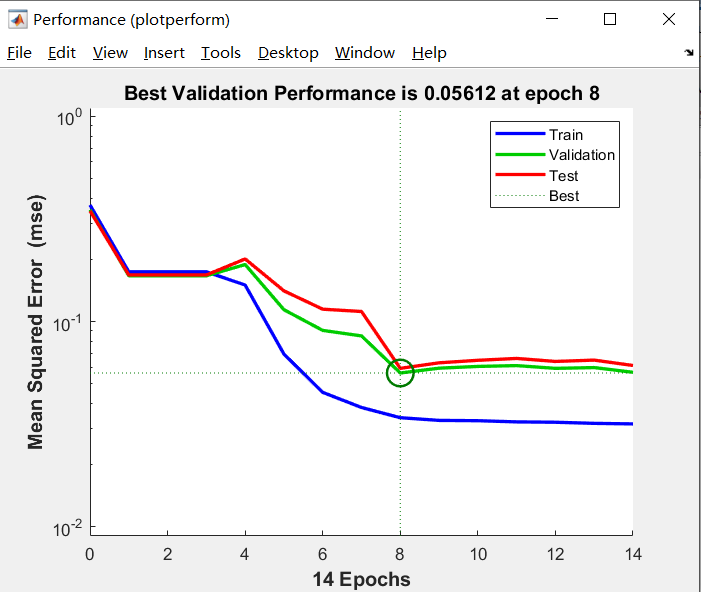
**5) Use the mean square error as a cost function to train the network**



**Figure 2.5:** The Performance of training set, validation set and test set

Figure 2.5 uses the performance function to get when at epoch = 45, the best verification performance is 0.053798. The mean square error reflects the accuracy of the data, and is the degree of deviation between the data and their true value. The default check value for the validation sample is 6. The training is terminated when the error of the validation sample no longer decreases for 6 consecutive iterations, and the check for the validation sample becomes 1000 by setting ‘net.trainParam.max\_fail = 1000’. After continuous "max\_fail" epochs, if the network cannot improve its performance, stop training due to the trained network being faulty and overfitting.

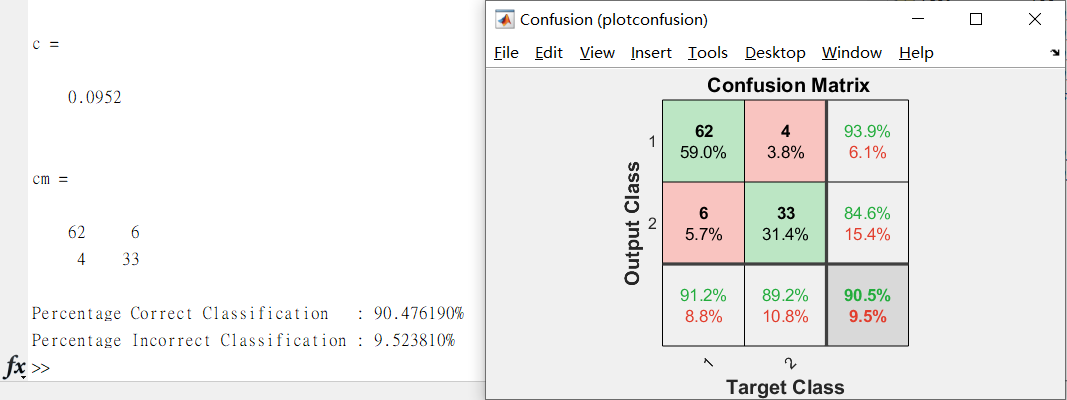
**6)** **Repeat step 5), but do not call the command ‘net.trainParam.max\_fail = 10000;’**



**Figure 2.6:** The percentage of data points in training set, validation set and test set

Figure 2.6 removes the command "net.trainParam.max\_fail = 10000;" to get when at epoch 8, the best verification performance is 0.05612. Generally, with the training of the network, the error of the verification sample has basically no longer reduced or even increased, and the training of the network will be stopped, otherwise it may overfit.

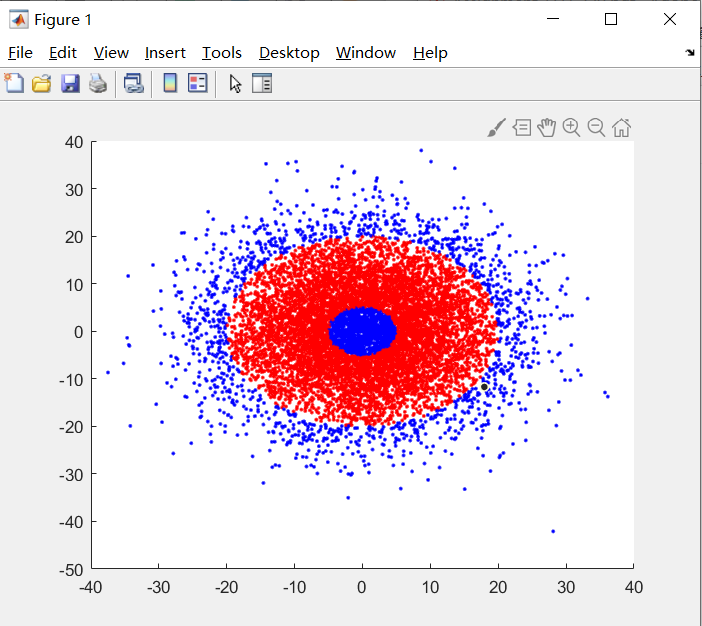
**7)** **Plot the confusion matrices for the trained neural network**



**Figure 2.7:** The Confusion matrix of the test set

The confusion matrix is used to classify the correctness of the results and is a standard to measure the adaptability of the neural network to the data. It can be seen from Figure 2.7 that the percentage of correct classification is 90.48 %, and the percentage of incorrect classification is 9.52 %. The target class are benign and malignant, thereby there are also two outputs. In the figure, the green square is the correct classification, and the red square is the incorrect classification. If the network has learned to classify correctly, the percentage in the red square should be very small, indicating that there is very little misclassification. The percentage in the red square in the figure is very small, indicating that there are few misclassifications, indicating that the network has learned to classify correctly.

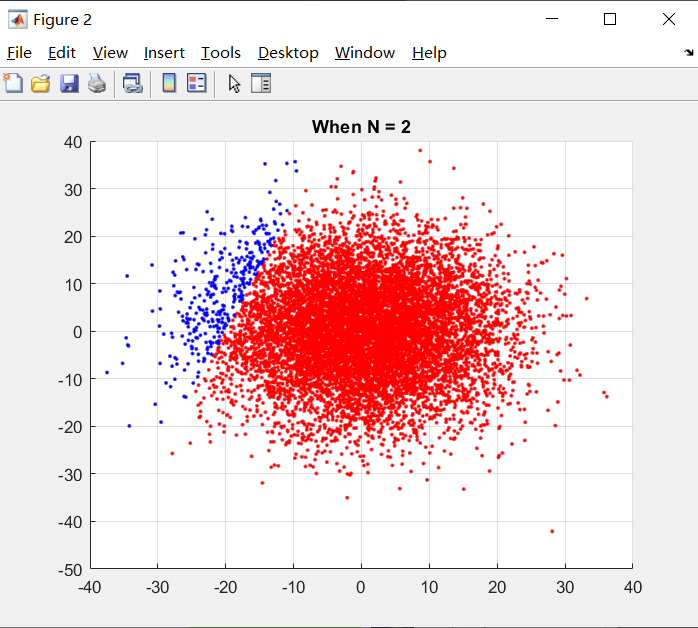
**8)** **Evaluate a comprehensive data set**



**Figure 2.8:** The data set

Figure 2.8 shows the best results for classification.

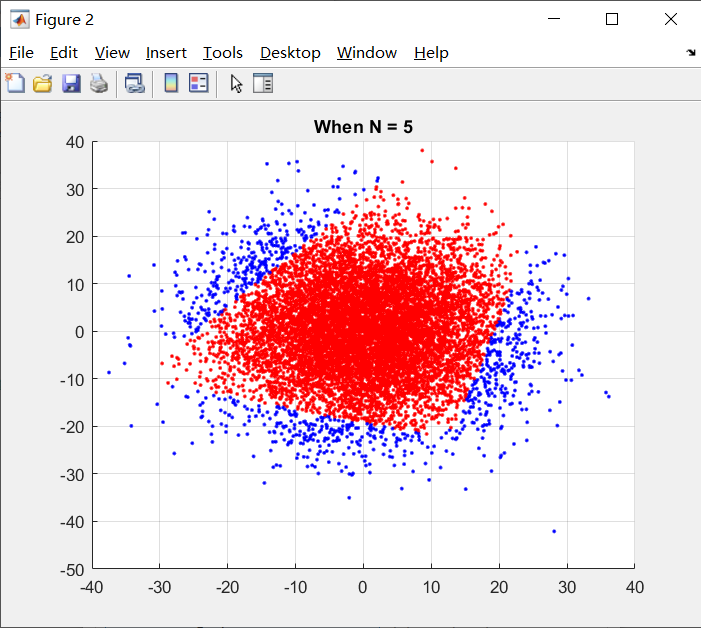
**9)** **Calculate the output of the neural network for the points in the dataset**



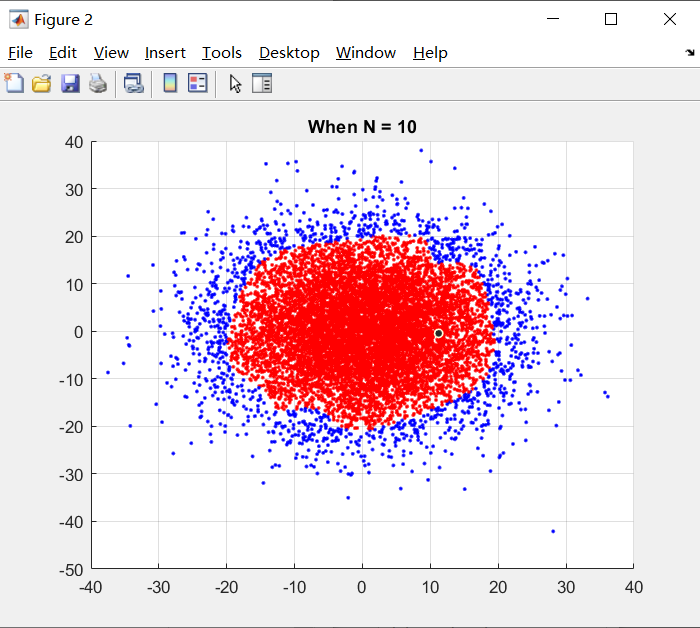
**Figure 2.9:** When N=2, use neural network to classify all points in the data set

Figure 2.9 shows all points in the data set classified by neural network. This classification has only been trained twice and cannot achieve the effect of Figure 2.8.

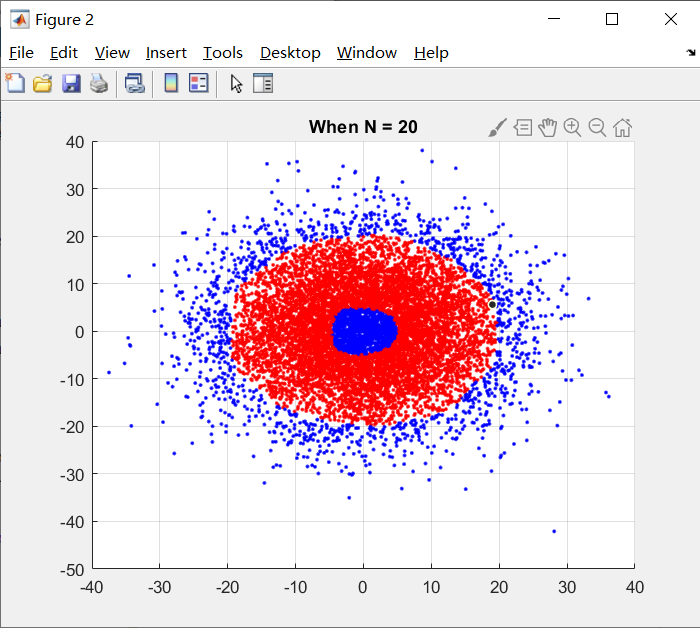
**10)** **Re-run the code with N=5, N=10, N=20**



**Figure 2.10.1:** When N=5, use neural network to classify all points in the data set



**Figure 2.10.2:** When N=10, use neural network to classify all points in the data set



**Figure 2.10.3:** When N=20, use neural network to classify all points in the data set

Through the above three pictures, it can be found that the more times the neural network is used for training, the more the result of data classification conforms to Figure 2.8.

**Appendix-Coding**

