EE5904 Neural Networks

Homework 3

Liu Xingyu

A0116430W

National University of Singapore

Electrical and Computer Engineering

14 Mar 2020

Q1. Function Approximation with RBFN (10 Marks)

Consider using RBFN to approximate the following function:

$y = 1.2\sin(\pi x) - \cos(2.4\pi x)$, for $x \in [-1,1]$

The training set are corrupted by random noise as follows:

$$y(i) = 1.2\sin(\pi x(i)) - \cos\big(2.4\pi x(i)\big) + 0.3n(i)$$

a). Exact interpolation method

Assume RBD is Gaussian function with standard deviation of 0.1. The performance of the RBFN using the test set is as shown in Figure 1a. We can see that the resulting RBFN using exact interpolation is over-fitting in the range of -1 to 1.

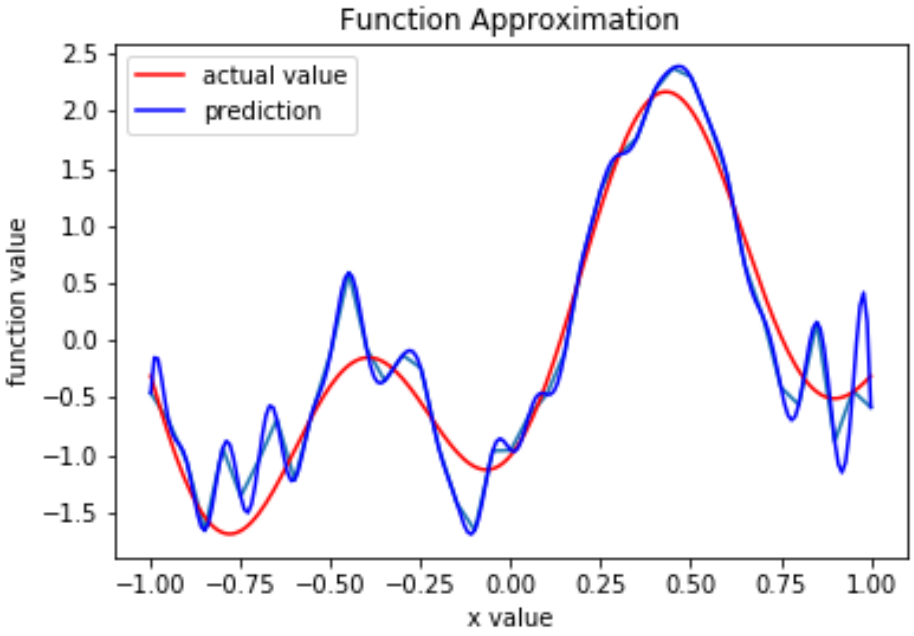


Figure 1a

b). Fixed Centers Selected at Random

15 centers among the sampling points are randomly selected. The performance of the RBFN using the test set is shown in Figure 1b. We can see that the resulting RBFN using the 15 fixed randomly selected centers is much more smoother comparing to using exact interpolation.

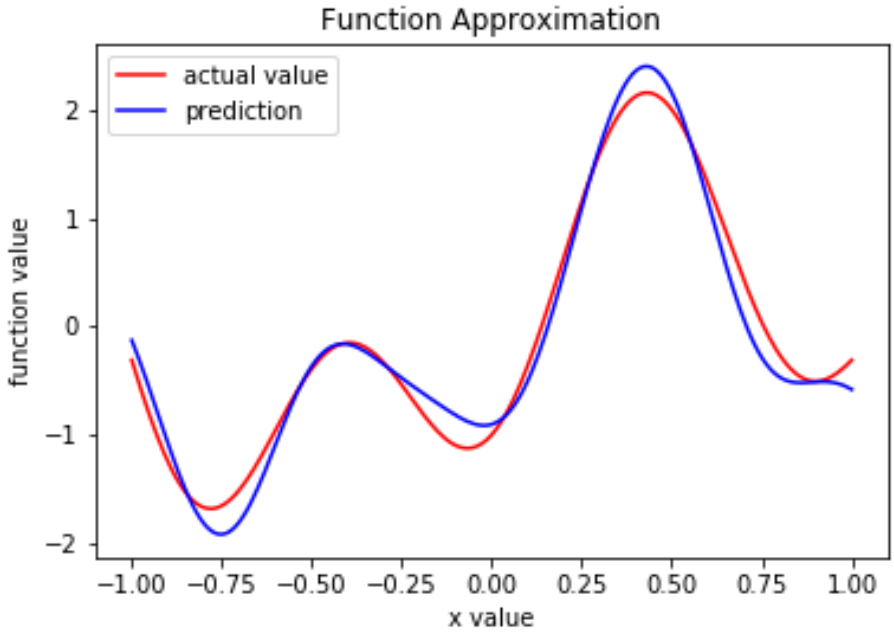


Figure 1b

c). Fixed centers RBFN with regularization

Using the same certers and widths determined in part a) and apply the different regularization factor 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5. The performance of the RBFN with different regularization factor is shown in Figure 1c. We can see that when increasing the regularization factor, the resulting plotting is becoming more flat and hence more under-fitting with larger regularization factor.

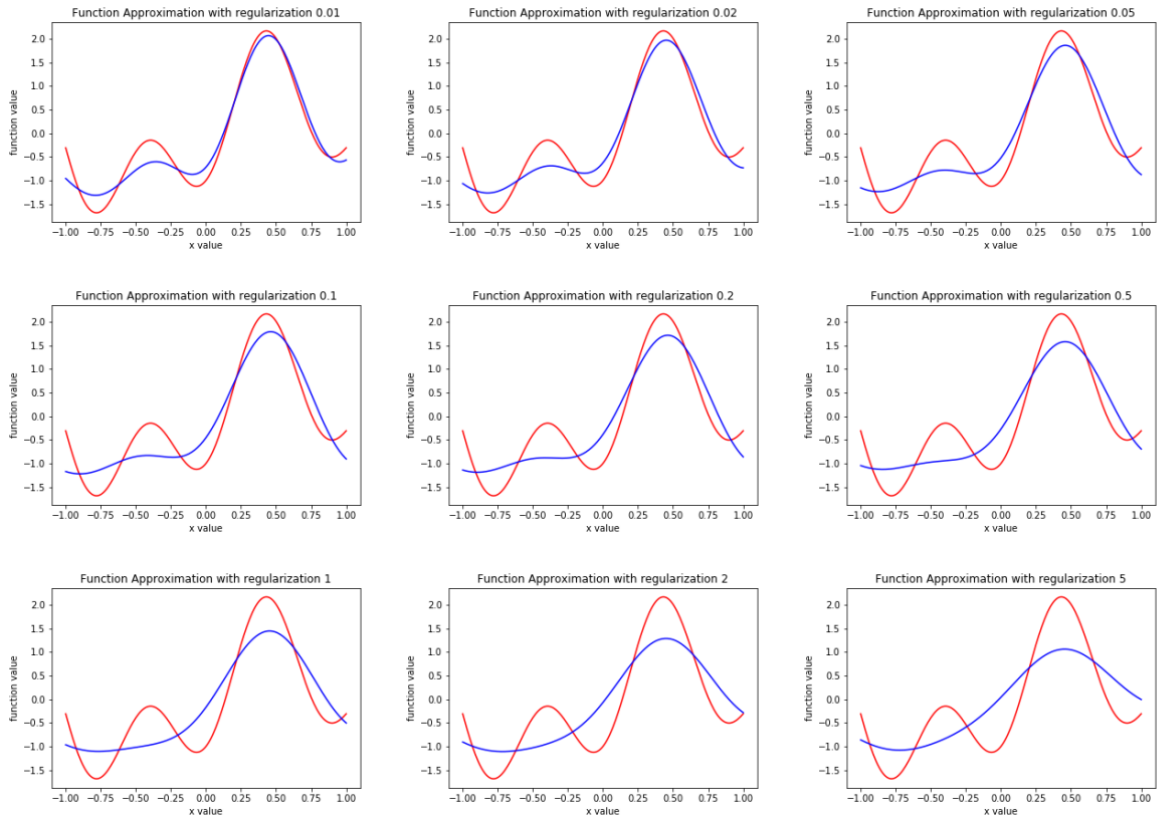


Figure 1c

Q2. Handwritten Digits Classification using RBFN (20 Marks)

a). Exact Interpolation Method

Assume the RBF is Gaussian function with standard deviation of 100. First evaluate the performance of the RBFN without regularization as shown in Figure 2a1 and then evaluate the performance of the RBFN with regularization factor 0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 20, 50 as shown in Figure 2a2. We can see that the training accuracy without regularization is always 1 regardless threshold. The testing accuracy firstly increase and reach the maximum of 1 when threshold is about 0.5 and then decrease when threshold increases from 0 to 1.

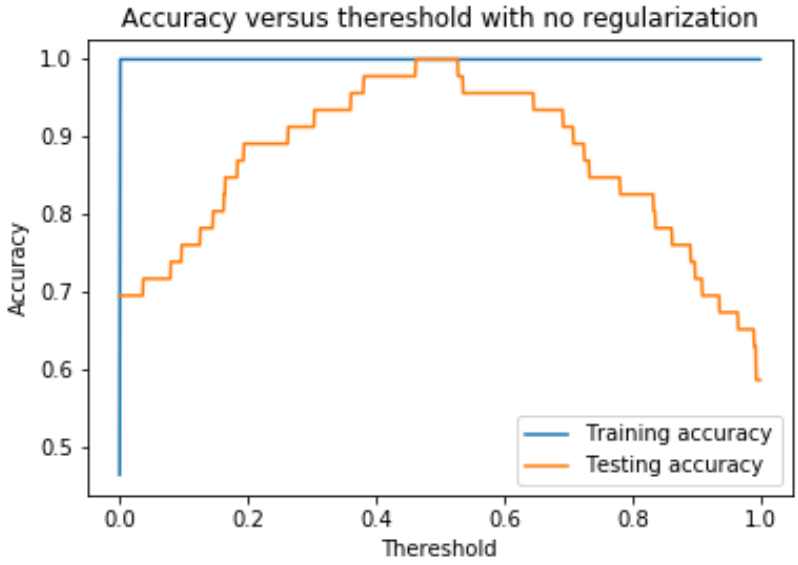
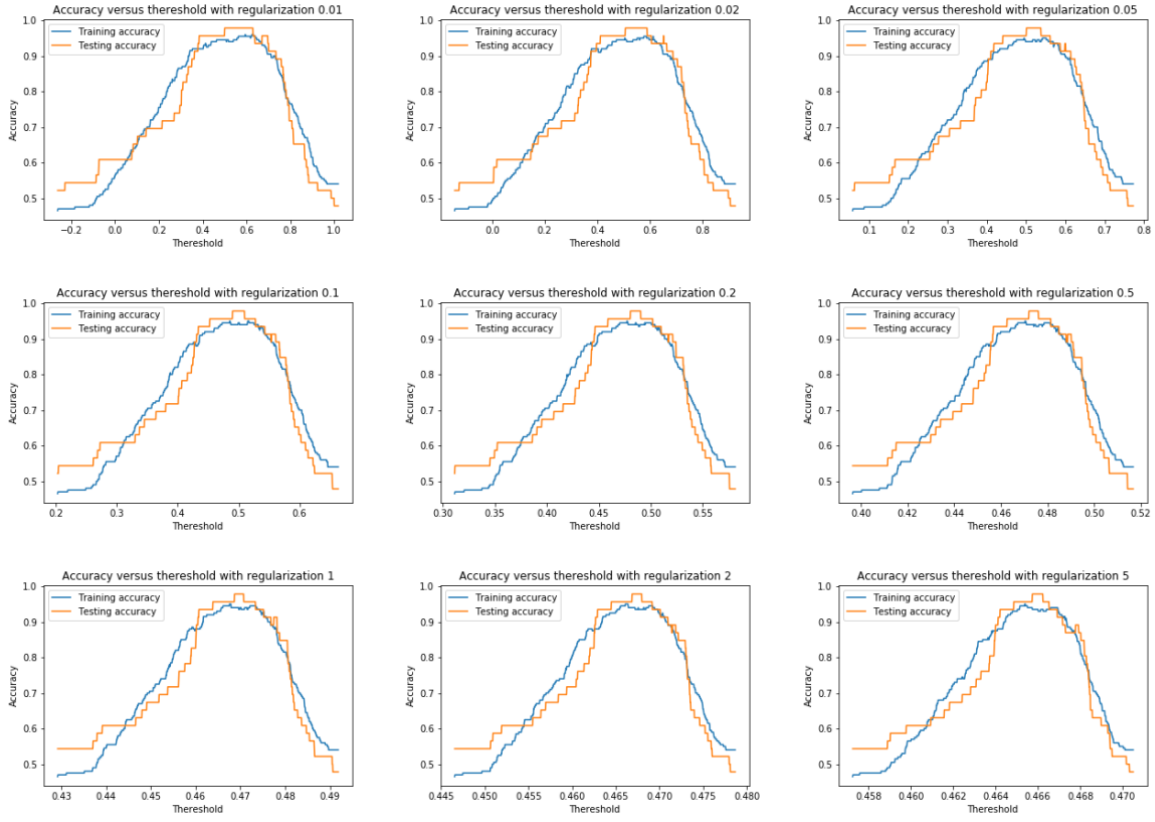


Figure 2a2

After applying the regularization, the training accuracy becomes more smooth and closed to the testing accuracy and it cannot reach to accuracy of 1.

As the regularization factor increases, the threshold that when both training accuracy and testing accuracy decrease from about 0.5 to 0.46.



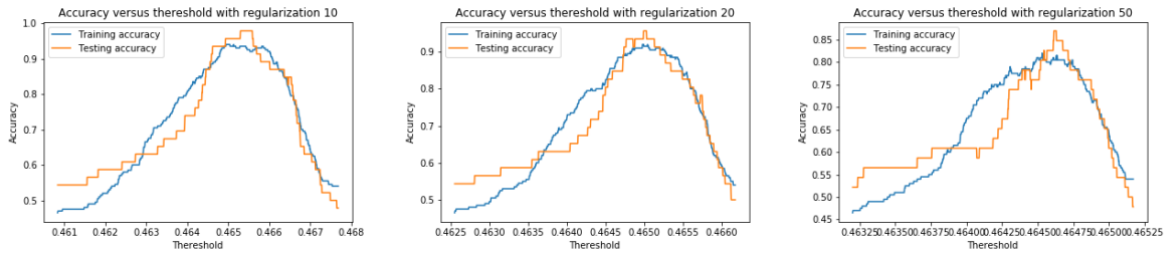


Figure 2a2

b). Fixed Centers Selected at Random using different width from 0.1 to 10000. First, set the width as the maximum distance between the selected centers and the performance of RFBN is shown as Figure 2b1. For maximum distance between the selected centers, the calculated width is 14.622.

Output of code:

```
The dismax selected is 14.622061773349627
```

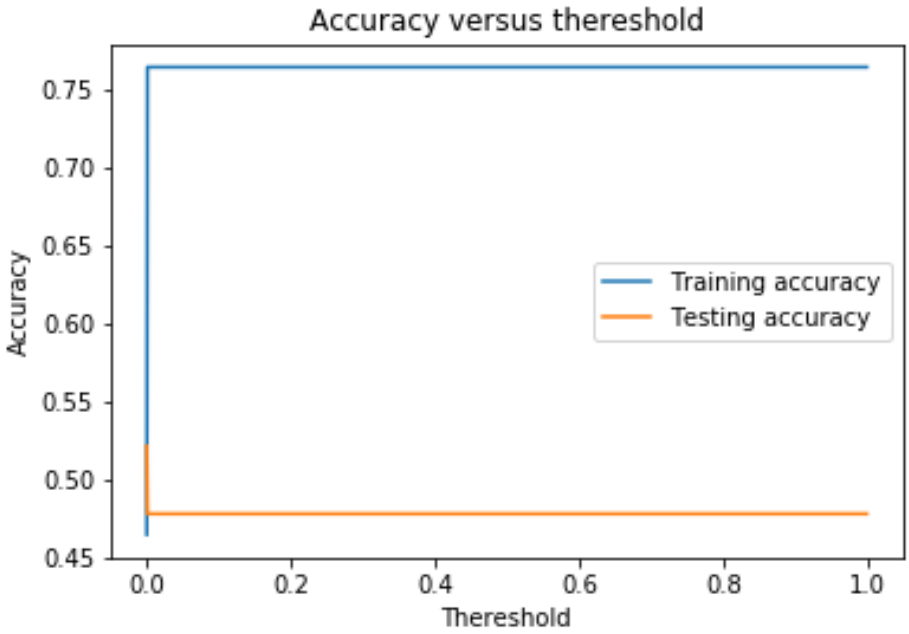


Figure 2b1

Then vary the width from 0.1 to 10000. The performance of RFBN is shown as Figure 2b2. We can see for width from 0.1 to 10, the accuracy for training and testing are all the same regardless threshold, 0.76 for training accuracy and 0.48 for testing accuracy. Starting from width of 50, the accuracy of training and testing will vary among different threshold. The training accuracy and testing accuracy can achieve about 0.99 with threshold of 0.2 to 0.7 as width is larger than 100.

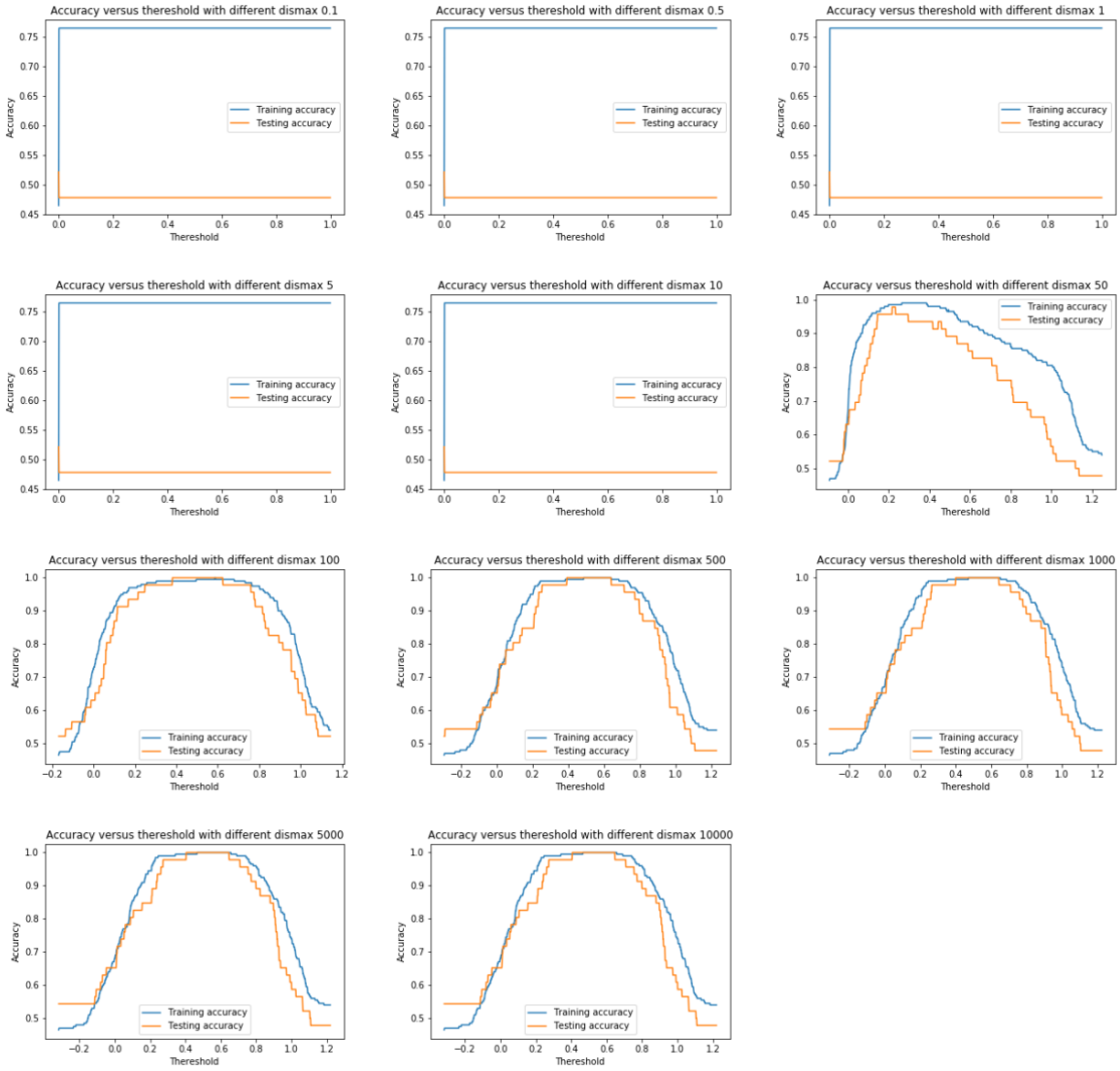


Figure 2b2

c). K-Mean Clustering

By using the K-Mean Clustering to determine the 2 centers. The width is selected as 100. The performance of the RFBN is shown in Figure 2c1. The testing accuracy can achieve maximum 0.85 with threshold 0.5.

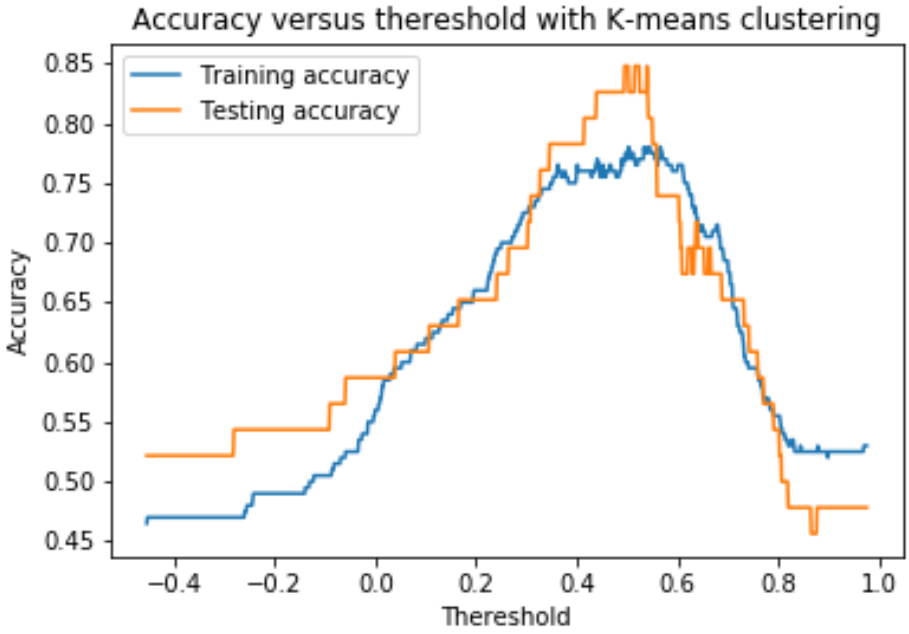


Figure 2c1

The obtained centers are shown as images below. Comparing to the mean of training images of class 0 and 3 is shown below Figure 2c2 as well. We can see the obtained centers using K-Mean clustering exactly match the mean of training images. The result is quite good.

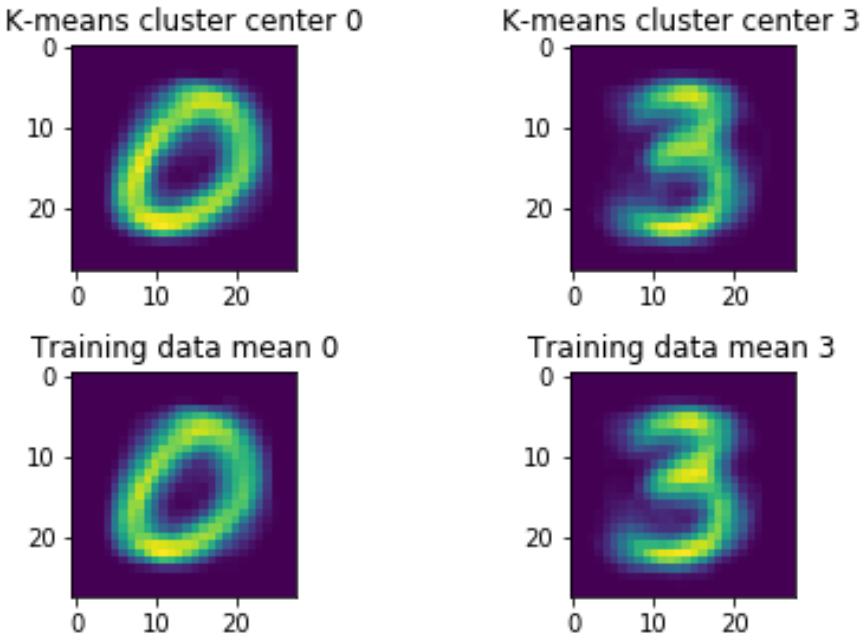


Figure 2c2

Q3. Self-Organizing Map (SOM) (20 Marks)

a). Implement a SOM that maps a 1-dimensional output layer of 25 neurons to a “heart curve”. And every topological adjacent neuron is connected by lines. The resulting map is shown in Figure 3a.

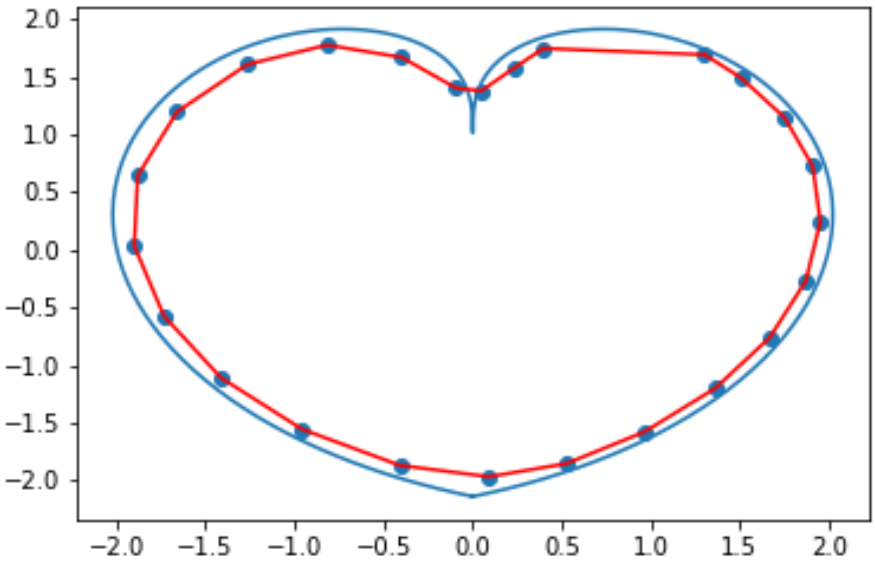


Figure 3a

b). Implement a SOM that maps a 2-dimensional output layer of 25 (5x5) neurons to a “square”. And every topological adjacent neuron is connected by lines. The resulting map is shown in Figure 3b.

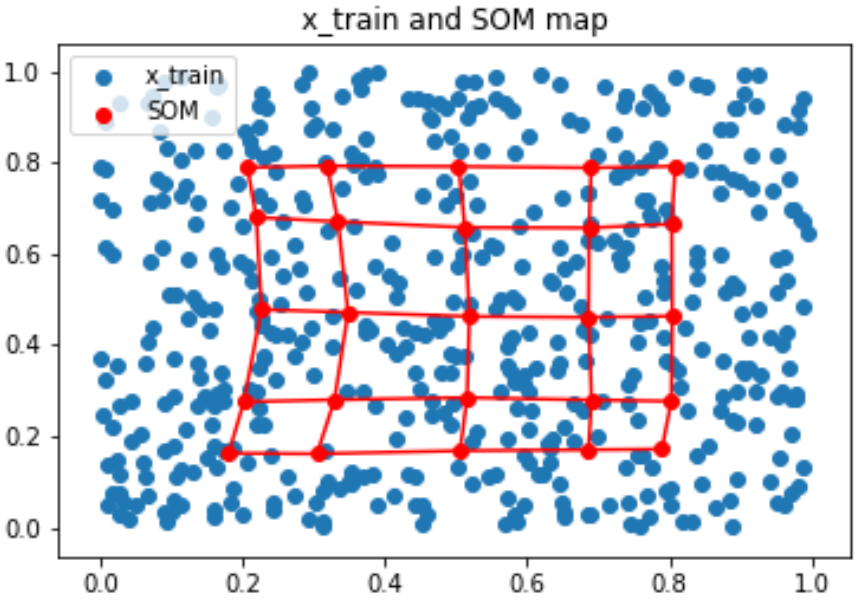


Figure 3b

c-1). Print out the semantic map of the trained SOM and visualize the trained weights of each output neuron on a 10x10 map. As my matric no is A0116430W, so the last two different digits 3 and 0 will be ignored. The map is shown in Figure 3c. We can see that after 1000 iterations, the same class of hand written digits are arranged in the same group.

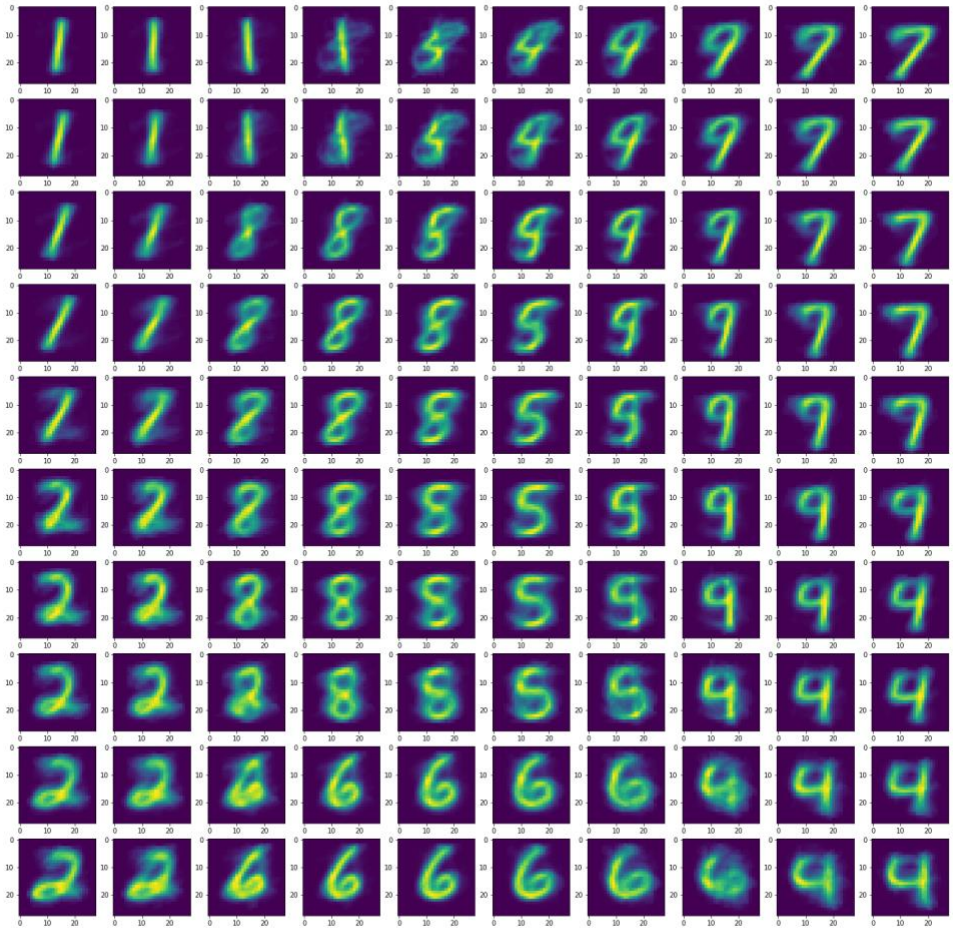


Figure 3c

c-2). Apply the trained SOM to the test images. The classification accuracy on the test images is 0.7598

Output of code:

```
The accuracy of SOM on testing data is: 0.7598039215686274
```