Nima Mohammadi
Neural Networks
Prof. Babaali
Assignment #4

Assignment #4

Part 1)

I.

To determine the best matching unit (BMU) we iterate through all the nodes and calculate the Euclidean distance.

$D_1 = (0.5 - 0.3)^2 + (0.2 - 0.7)^2$	$= 0.08$
$D_2 = (0.5 - 0.6)^2 + (0.2 - 0.9)^2$	$= 0.5$
$D_3 = (0.5 - 0.1)^2 + (0.2 - 0.5)^2$	$= 0.25$
$D_4 = (0.5 - 0.4)^2 + (0.2 - 0.3)^2$	$= 0.02$
$D_5 = (0.5 - 0.8)^2 + (0.2 - 0.2)^2$	$= 0.09$

Based on the calculations above, node C_4, whose weight vector is closest to the input vector is tagged as the BMU.

II.

The weight vector for BMU is adjusted as below:

$$w_{ij}(new) = w_{ij}(old) + \alpha[xi - w_{ij}(old)]$$

$W_{14} = 0.4 + 0.2(0.5 - 0.4) = 0.42$

$W_{24} = 0.3 + 0.2(0.2 - 0.3) = 0.28$

III.

The weight vectors for the neighbors of C_4, which are C_5 and C_3, are updated as below:

$W_{15} = 0.8 + 0.2(0.5 - 0.8) = 0.74$

$W_{25} = 0.2 + 0.2(0.2 - 0.2) = 0.2$

$W_{13} = 0.1 + 0.2(0.5 - 0.1) = 0.18$

$W_{23} = 0.5 + 0.2(0.2 - 0.5) = 0.44$

IV.

$D_1 = (0.5 - 0.3)^2 + (0.5 - 0.7)^2$	$= 0.08$
$D_2 = (0.5 - 0.6)^2 + (0.5 - 0.9)^2$	$= 0.17$
$D_3 = (0.5 - 0.1)^2 + (0.5 - 0.5)^2$	$= 0.16$
$D_4 = (0.5 - 0.4)^2 + (0.5 - 0.3)^2$	$= 0.05$
$D_5 = (0.5 - 0.8)^2 + (0.5 - 0.2)^2$	$= 0.18$

Again the fourth node is the best matching unit.

$W_{14} = 0.4 + 0.1(0.5 - 0.4) = 0.41$

$W_{24} = 0.3 + 0.1(0.2 - 0.3) = 0.29$

$W_{15} = 0.8 + 0.1(0.5 - 0.8) = 0.77$

$W_{25} = 0.2 + 0.1(0.5 - 0.2) = 0.23$

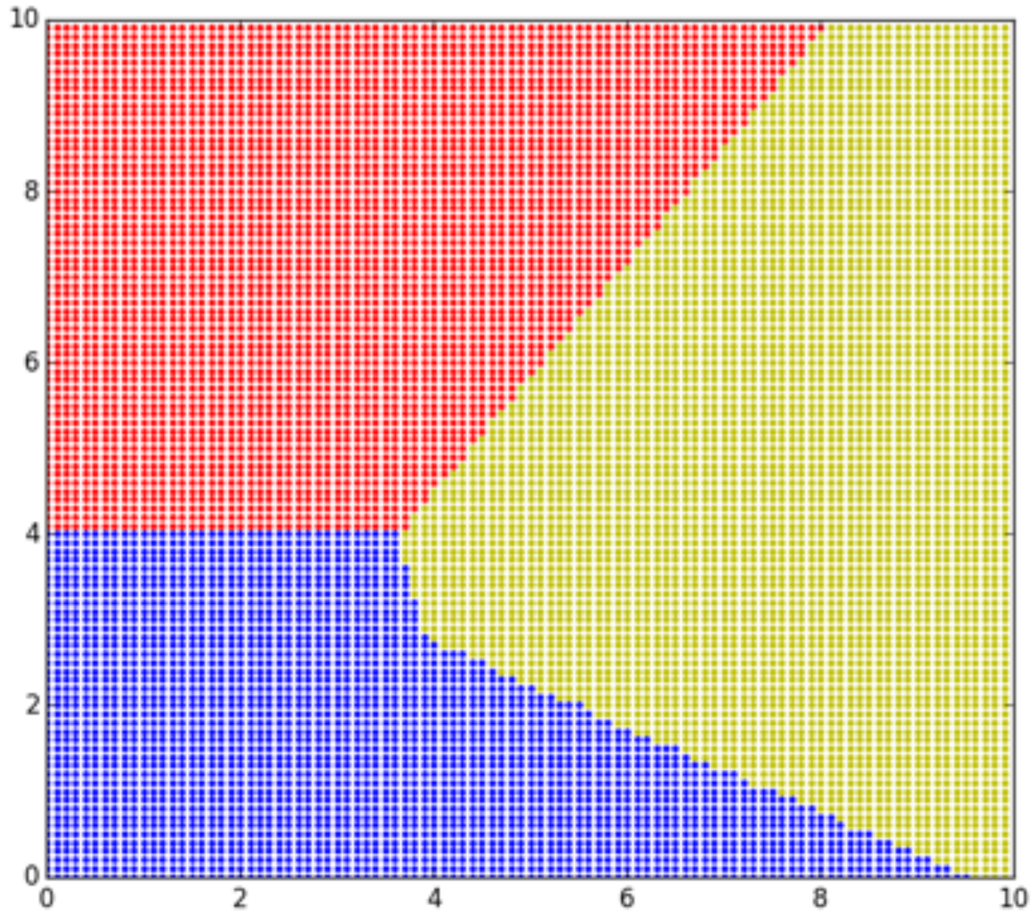
$W_{13} = 0.1 + 0.1(0.5 - 0.1) = 0.14$

$W_{23} = 0.5 + 0.1(0.5 - 0.5) = 0.5$

Part 2)

I.

We check each point within an interval in a subspace of the problem and associate it with a class by simply looking at it distance from the centroids.



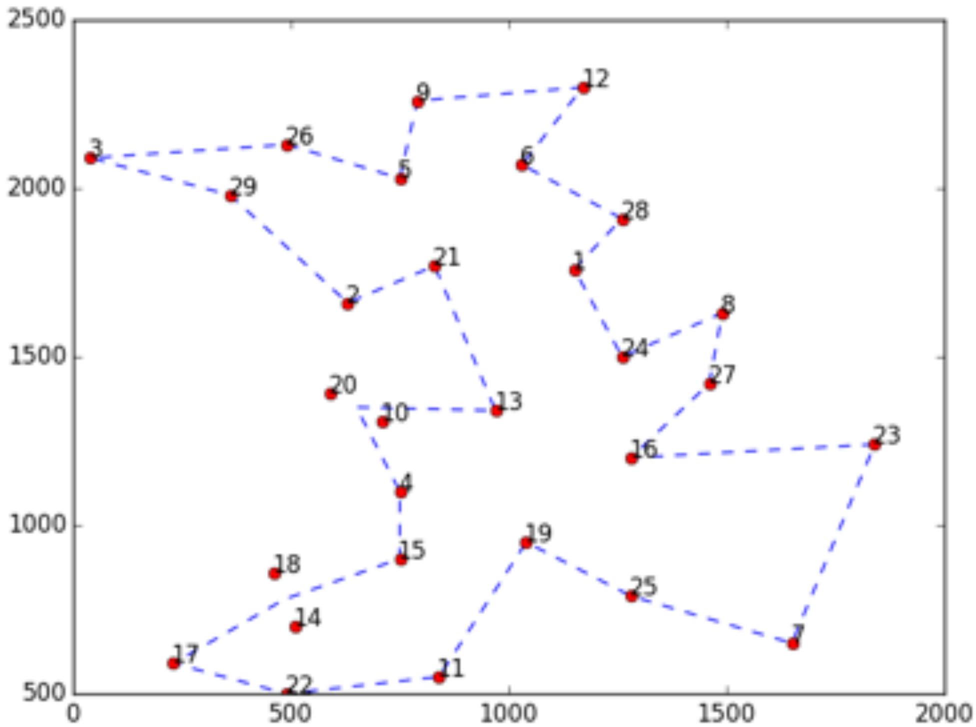
II.

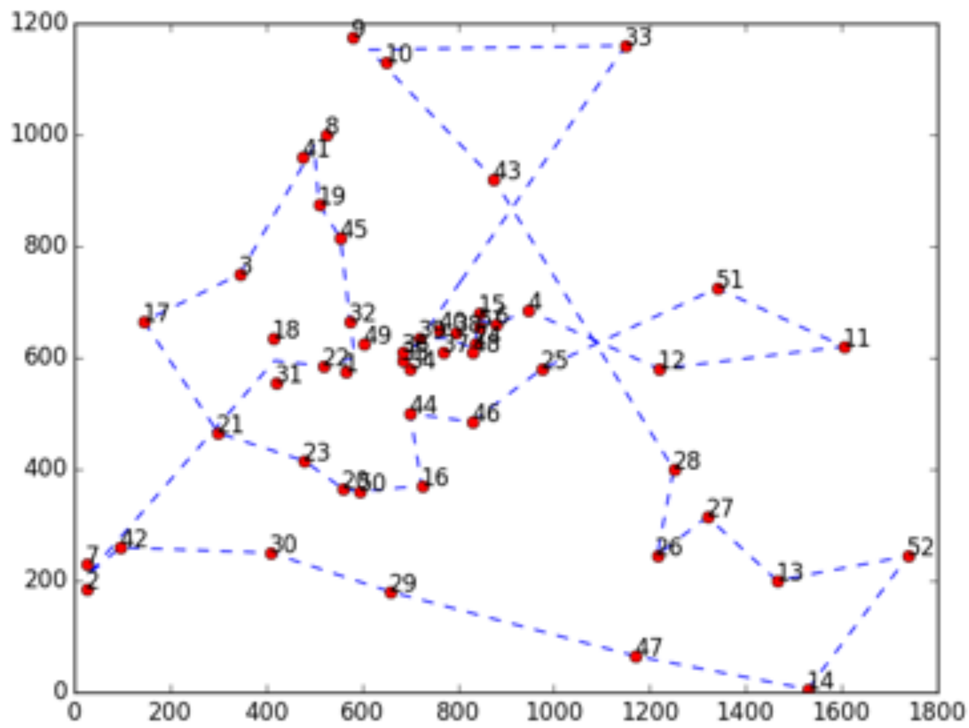
With more than one centroid representing each class comes higher number of centroids in general which corresponds to more computation and cost. However using more centroids for each class clearly preserves more discriminative information. And of course we would see a finer boundary between classes.

Part 3)

I.

The discriminant function used here the squared Euclidean distance between the input vector and the weight vectors.





The figures above are the resulting solutions for Bayg29 and Berlin52 by SOM. The length of their tour are 8893 and 9294.

Notice that the solutions are suboptimal and they do not necessarily pass through each and every city as some turning points are stuck between two cities. So these numbers should be taken with a grain of salt! The reported length in the assignment is irrelevant as Bayg29 is a weighted TSP problem, but we only deal with the classic TSP here.

We used two different learning rates in our network, one for the best matching unit and the other for the two neighbor units. Clearly the learning rate for neighbors assumes a lower value and decays faster.

II.

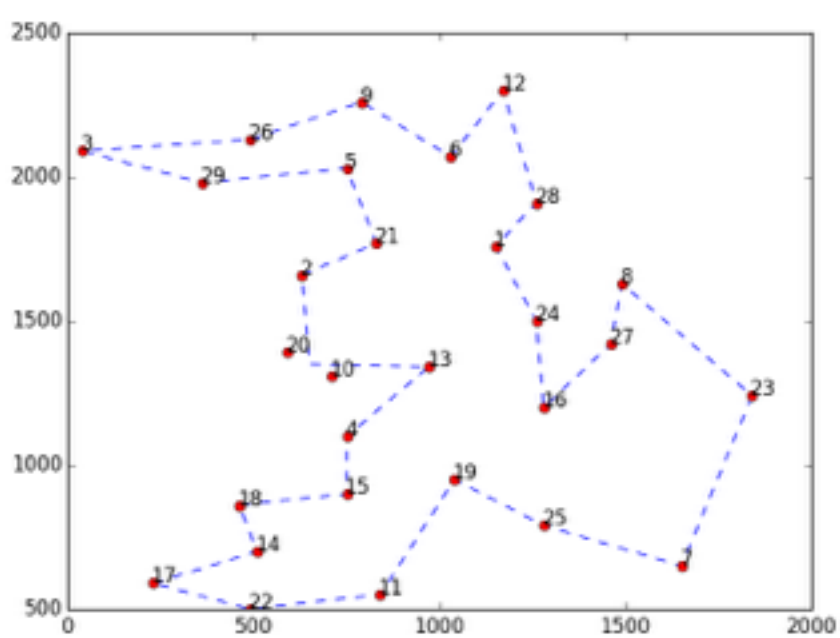
In this part we investigate the impact of lateral inhibition. We use a topological neighborhood that decays with distance.

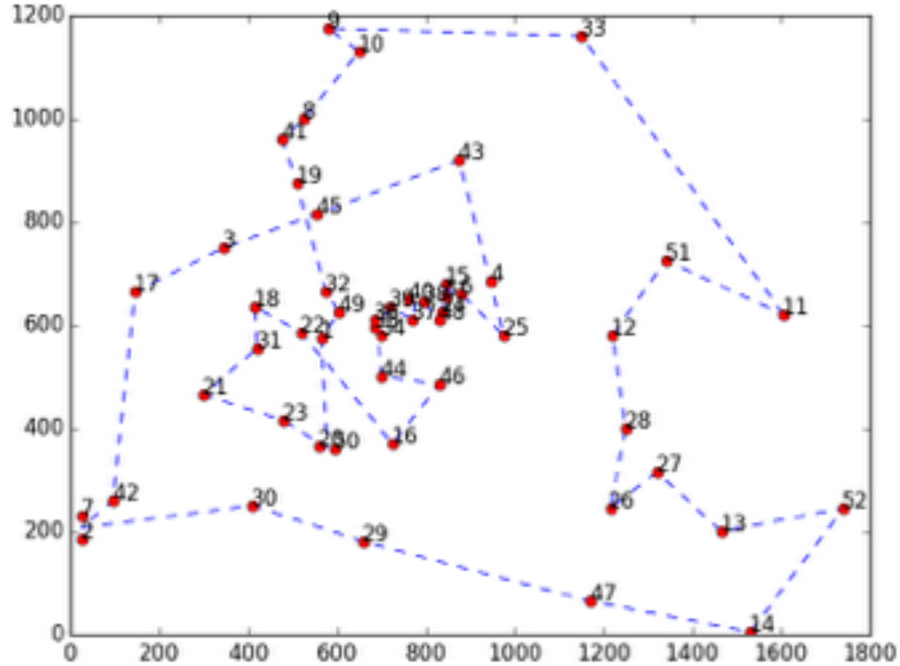
If S_{ij} is the lateral distance between neurons i and j on the linear placement of neurons, we take

$$T_{j,winner} = \exp(-S^2_{j,winner} / 2\partial^2)$$

as the topological neighborhood, where winner is the index of BMU. This equation is maximal at the winning neuron and is symmetrical about that neuron. It also decrease monotonically to zero as the distance increases. We also make sure that ∂ is decreased with time.

I tested this methodology for these two datasets with radius of 7 adjacent neurons. The results are depicted below:





Increasing the neighborhood radius, based on my observations, did not necessarily enhanced the performance. Though while for Bayg29 we got a slightly worse result, Berlin52 in fact had a better performance.

Increasing the number of iterations did not necessarily make matter better. Our decays is too fast and the performance depends more on the initial weighting than anything else.