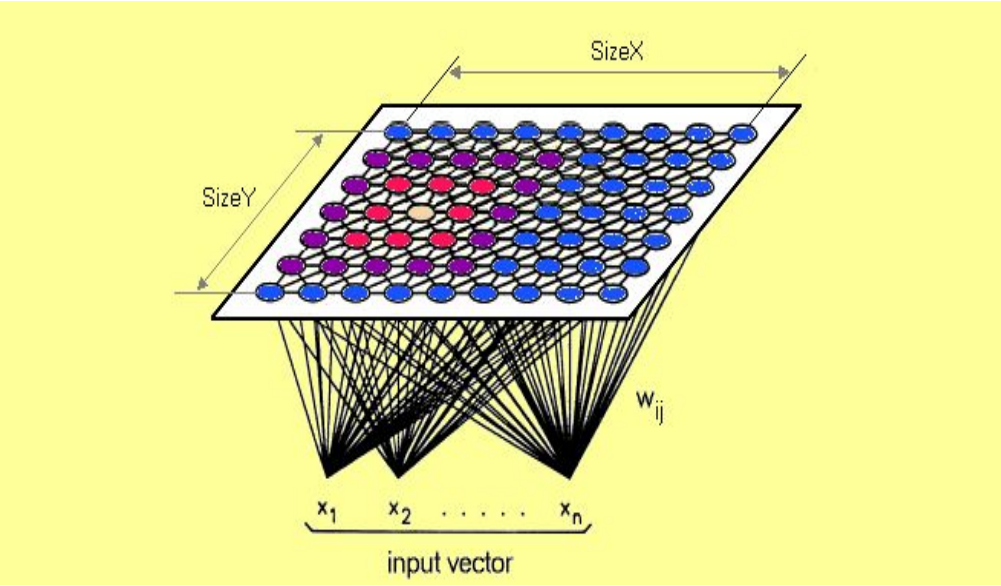
Roll No: MIT2019031
Name: Shrikant Temburwar

Self Organizing Maps

A self-organizing map (SOM) is a type of artificial neural network (ANN) that is trained using unsupervised learning to produce a low-dimensional (typically two-dimensional) discrete representation of the input space of training samples, called a map, and therefore it is a method of doing dimensionality reduction. Self-organizing maps differ from other artificial neural networks in that they apply competitive learning as opposed to error-correction learning (such as backward propagation with gradient descent), and in the sense that they use a neighbourhood function to preserve the topological properties of the input space.



Dimensionality reduction in SOM

SOM was introduced by Finnish professor Teuvo Kohonen in the 1980s, sometimes called the Kohonen Map.

When using a Kohonen (SOM) network there is always Two phases

- **Training phase**: Synaptic weights are changed to match inputs
- **Test phase**: Synaptic weights remain fixed. We test the network performance.

1) Training Phase

Each node is associated with an input pattern or a category of inputs through a weighing process change. Weights are gradually changed to more look like the input vector it has finally arrived at represent.

The Algorithm

1. The weights of each node are initialized.
2. A vector is chosen at random from the training data set.
3. Each node is examined to calculate which weights are closest to the input vector. The winning node is commonly known as the Best Matching Unit (BMU).
4. Then the neighbourhood of the BMU is calculated. The number of neighbours decreases over time.
5. The winning weight is rewarded by being more like the sample vector. Neighbours are also more like the sample vector. The closer a node is to the BMU, the more its weights will be altered, and the further away the neighbour is from the BMU, the less it will learn.
6. Repeat step 2 for N iterations.

2) Testing Phase

In the testing phase, the weights are fixed. With the weights that were obtained in the training phase, you can now test each vector. If the training was successful, the correct node should light up for the correct input vector In the testing phase, the Euclidean distance is calculated between the input vector and the weight vector. The node with the shortest distance is the winner.

Results:

Input vectors:
X1 = [0.1 0.8] , X2 = [0.5 -0.2] , X3 = [-0.8 -0.9] , X4 = [-0.06 0.9]

Output:

Neuron 5 respond to the input vector X1 = [0.1, 0.8]

Neuron 67 respond to the input vector X2 = [0.5, -0.2]

Neuron 90 respond to the input vector X3 = [-0.8, -0.9]

Neuron 4 respond to the input vector X4 = [-0.06, 0.9]