Artificial Neural Networks
A Simple Self-Organizing Map Network in Python

Kevin Costa Scaccia
Federal University of São Paulo - UNIFESP

I. Implementation

The core of the model consists of the **SON_Network** class that encapsulates all the functionality of the model. Following is a description of its operation and the implemented methods. There is also an auxiliary **NeuronSON** class that represents a grid neuron

A. SON_Network Constructor

The constructor specifies the size of the neuron grid, the input data and the dimensionality of each example in the data. With this information, a neuron grid of size $(grid\_sizeXgrid\_size)$ is defined and allocated with their initial units.

B. NeuronSON Class

Objects of this class represent the grid neurons of the SOM Network and are intended to store the weights of each neuron as well as methods that measure the similarity (distance) of the current neuron to an example or another neighboring neuron.

C. Train method

This is the main method of the model and it is responsible for the unsupervised network training process. The method takes as input the parameters $epochs$, $learning\_rate$ and $sigma$, where $epochs$ is the number of times the model will pass over all examples and readjust its weights, $learning\_rate$ the step size of the neurons weight update, and $sigma$ is the weights update intensity coefficient of the neighborhood of the neuron elected winner. Small values of $sigma$ reverts to **winner-takes-all** learning.

In each epoch we iterate over all the examples in the model data, for each example we have to calculate and store their similarity (distance) to all neurons in the grid. To store these distances we use the two-dimensional matrix $distances$. After calculating the distances and elected a **winner neuron** whose distance to the example is the smallest, we start what is known as **Cooperative Phase**, where besides updating the weights of the neuron $winner$ (Synaptic Adaptation) and also your nearest neighbors in proportion to their distance on the grid with the neuron $winner$.

In addition to the training itself, this method maintains a $total\_error$ array containing the progression of the training error, ie. the average error of the entire network each epoch.

II. Experiments and Evaluation

The experiments were performed varying the number of epochs with 3 different datasets, the first randomly generated and the other two are classic datasets and used in the initial validation neural network models. Here we only show the results of training in the Generated Dataset and in the Breast Cancer Dataset, the document **SOM_Network.ipynb** follows with more information and experiments.

A. Generated Dataset

Experiments performed on a randomly generated dataset that has 4 sparse clusters. The image below shows the original data and neurons with randomly generated weights plotted on a two-dimensional space.

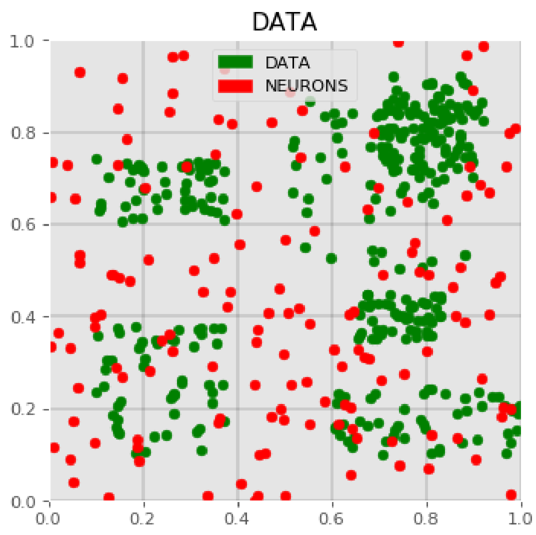

Fig. 1: Generated Data and Neurons (12x12 GRID)

Figures 2 and 3 show the training results in 20 and 200 epochs of a network with 12x12 neurons grid.

B. Sklearn Breast Cancer Dataset

This is a classic dataset used to classify cells into cancerigenic or not, whether it is a problem of some complexity

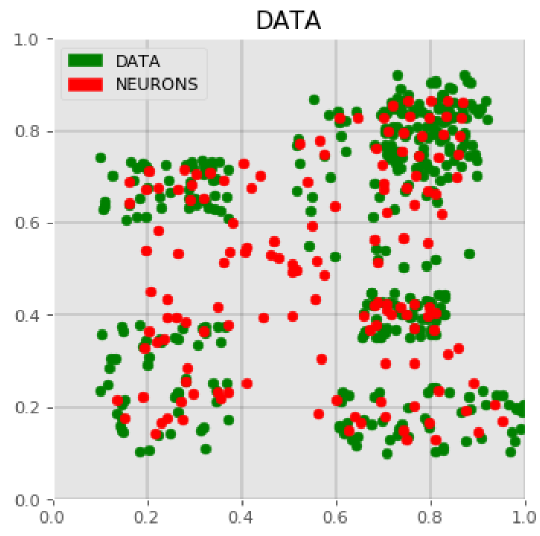

Fig. 2: Generated Data - 20 epochs (12x12 GRID)

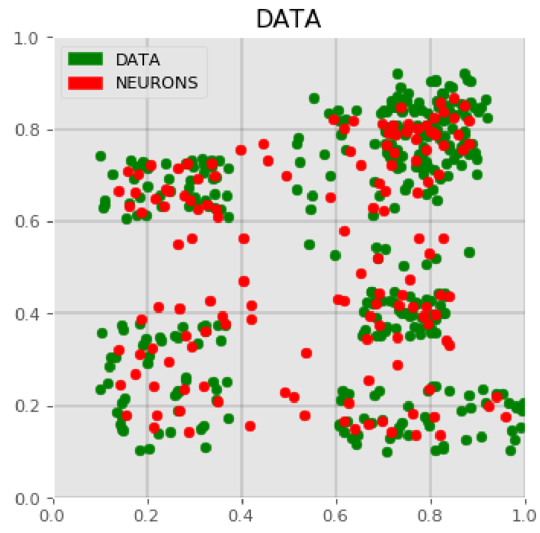

Fig. 3: Generated Data - 200 epochs (12x12 GRID)

and clearly nonlinearly separable. Each example consists of 30 features related to the cell, followed by its class.

From the figure 4 and 5 we can see that the model with a grid of 12x12 neurons learned and converged to get a distribution closer to original data distribution.

III. Conclusion

We conclude that the model has a tendency to represent the distribution of the data. The self-organizing map seems to be describing a mapping from a higher-dimensional input space to a two-dimensional space. Once trained, the main application of the model could be the classification of a vector from the input space by finding the node with the closest (smallest distance metric) weight vector to the input space vector.

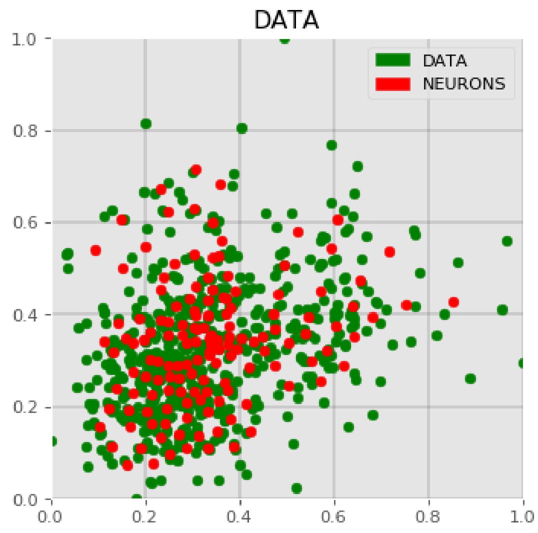

Fig. 4: Breast Cancer 100 epochs (12x12 GRID)

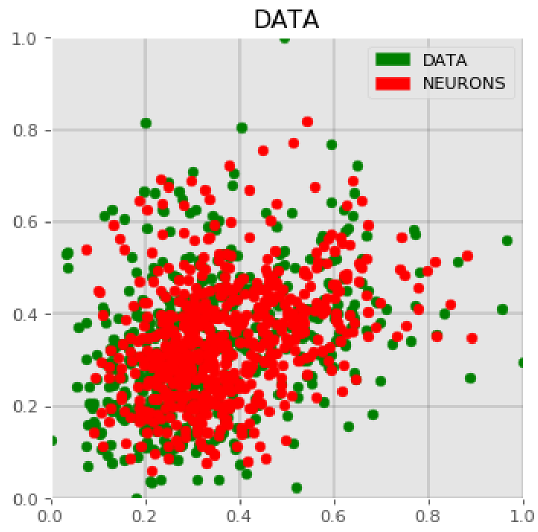

Fig. 5: Breast Cancer 100 epochs (24x24 GRID)