Coursework 2

This coursework relates to the perceptron. In a perceptron there is an input vector of size d:

$$\mathbf{x} = (x_1, x_2, \ldots, x_d) \tag{1}$$

These are not restricted to discrete values. They feed forward to a MP neuron with value y using the usual thresholding function:

$$y = \begin{cases} 1 & \sum_i w_i x_i \geq \theta \\ -1 & \text{otherwise} \end{cases} \tag{2}$$

where w_is are weights, corresponding to synapse strenghts and you aren't really restricted in your choice of the two discrete values, one and -1 in this case. The idea is that you have a set of inputs:

$$X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \tag{3}$$

where, to be clear, each of these inputs is a d-vector, so the index on bold quantities labels data, not to be confused with the dimension index on unbolded quantities. Each input is labelled:

$$D = \{d_1, d_2, \ldots, d_n\} \tag{4}$$

where in this example the labels take two values, -1 and one corresponding to blue and red. The goal is to use the perceptron to map:

$$\mathbf{x} \to d \tag{5}$$

The perceptron rule is that we can do this by the update:

$$w_i \leftarrow w_i + \eta x_i (d - y) \tag{6}$$

and

$$\theta \leftarrow \theta + \eta (d - y) \tag{7}$$

In the `Coursework2` folder you will find a list of labelled points called `points.txt`. The goal is to programme a perceptron capable of classifying these points. Pick a modest η such as $\eta = 0.01$ and it should learn the task in tens of trials. I have supplied small programmmes, `load.py` and `load.jl`, to help load the data in python or julia.

If you have this working you can experiment with what happens if you change the MP neuron, or the perceptron rule, or mislabel some points.

`coms30127.github.io`