Introduction to Machine Learning

Supervised Learning Exercise (Part 1)

These exercises can be solved in any programming language of your choice. It is assumed you are proficient with programming. It will be helpful if the language has a library to plot graphics.

In the following exercises the objective is to program algorithms that, given examples and an expected output learn to mimic the behavior present in the data.

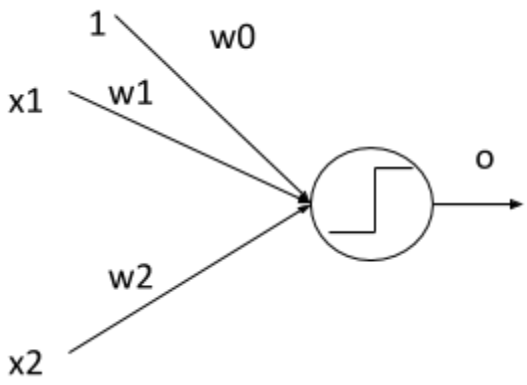

Fig. 1 Perceptron

1. The "network" in Fig. 1 represents a perceptron with two inputs that can also be described by the following equations:

$$o = f(w0 + x1\, w1 + x2\, w2)$$

$$f(x) = 1, \; se\, x > 0$$

$$f(x) = -1, \; se\, x <= 0$$

Choose one of the binary operations (AND or OR) and build a vector with all combination of two bit patterns: {{-1, -1}, {-1, 1}, {1, -1}, {1, 1}} and in another vector (called desired response, *d*) the corresponding result of the operation: OR {-1, 1, 1, 1} or AND {-1, -1, -1, 1}. Notice that 0,1 can also be used instead of -1,1 with the due adaptations.

First, initialize *w0*, *w1* and *w2* to small random values and, for each input pattern calculate the output (*o*).

Calculate the difference / error (*e*) between *o* and the desired response (*d*) for each output.

Add to the update term for w1 ($\Delta w1$) and w2 ($\Delta w2$) according to:

$$\Delta w0 \mathrel{+}= \alpha\,(d - o)$$
$$\Delta w1 \mathrel{+}= \alpha\, x1\,(d - o)$$
$$\Delta w2 \mathrel{+}= \alpha\, x2\,(d - o)$$

After all examples are presented (an epoch) update *w1* and *w2*

$$w0 \mathrel{+}= \Delta w0$$
$$w1 \mathrel{+}= \Delta w1$$
$$w2 \mathrel{+}= \Delta w2$$

so that in the next iteration the error will diminish.

How many epochs (iterations through the whole set) did it take to get all examples right? Vary alfa and measure the number of iterations to reach zero error with different values of alfa.

Repeat the experiment 30 times and present the average and standard deviation of the number of epochs it took to converge.

2. Try to learn the XOR function, did it succeed? why?

3. Repeat the process for a bit pattern with 10 inputs and generate the output (-1 or +1) according to a rule of your choice (try to have a dataset with a reasonably balanced number of elements of each class). Did it learn the correct result for all examples? (remember, 30 repetitions, display average and standard deviations of your measurements)

Introduce some errors in the output (flip a few classifications) and repeat the training procedure. Compare both results.

Not all rules of bit patterns are learnable with this rule and this network. Only networks with at least one hidden layer should be able to learn any rule made with bit patterns.