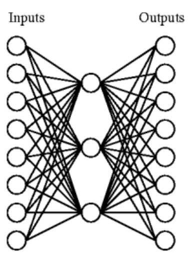
Artificial Neural Networks (ANNs)

Biologically inspired computational model:

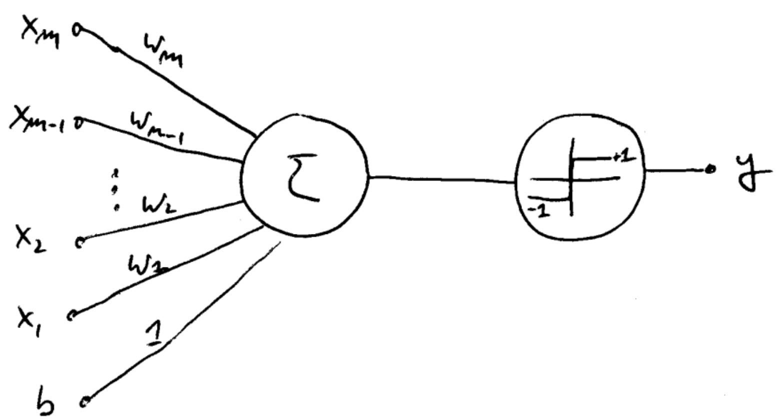
(1) <u>Simple</u> computational units (neurons).
(2) <u>Highly interconnected</u> - connectionist view
(3) Vast <u>parallel</u> computation, consider:
   - Human brain has ~10^{11} neurons
   - Slow computational units, switching time ~10^{-3} sec (compared to the computer >10^{-10} sec)
   - Yet, you can recognize a face in ~10^{-1} sec
   - This implies only about 100 sequential, computational neuron steps - this seems too low for something as complicated as recognizing a face
   - <u>Parallel processing</u>



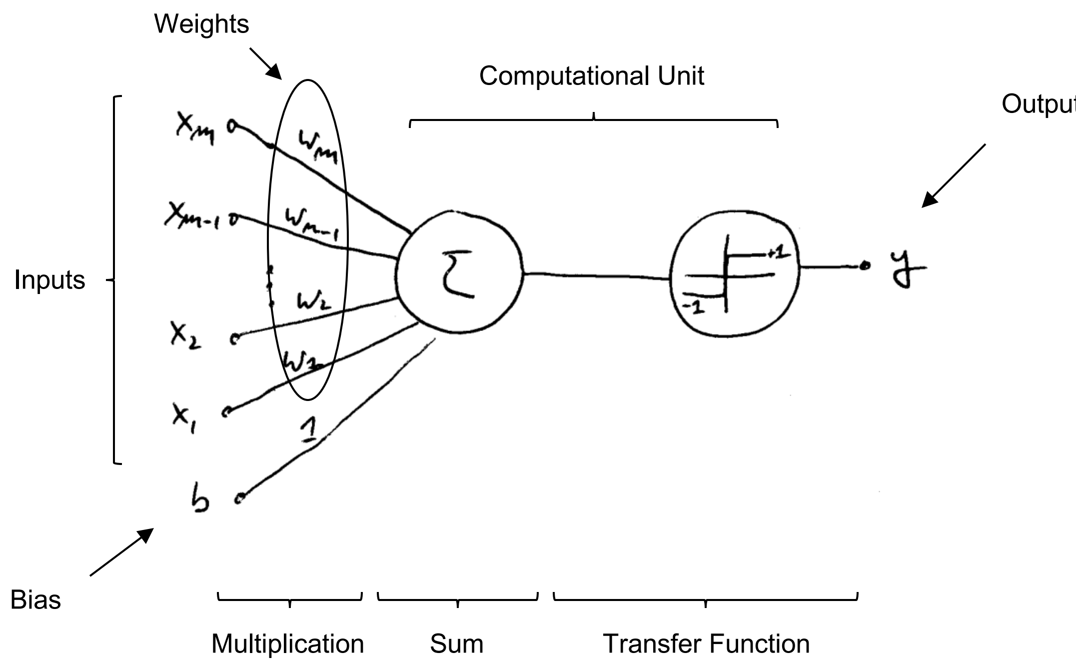
ANNs are naturally parallel - each neuron is a self-contained computational unit that depends only on its inputs.

The Perceptron

- A simple, single layered neural “network” - only has a single neuron.
- However, even this simple neural network is already powerful enough to perform classification tasks.



The Architecture



Transfer Function:

$$\operatorname{sgn}(k) = \begin{cases} +1 \text{ if } k \geq 0 \\ -1 \text{ otherwise} \end{cases}$$

Perceptron Computation: $$y = \operatorname{sgn}\left(b + \sum_{i=1}^{m} w_i x_i \right)$$

Note: $y \in \{+1, -1\}$ ← Binary Classification

Computation

A perceptron computes the value,

$$y = \operatorname{sgn}\left(b + \sum_{i=1}^{m} w_i x_i\right)$$

Ignoring the activation function sgn and setting *m* = 1, we obtain,

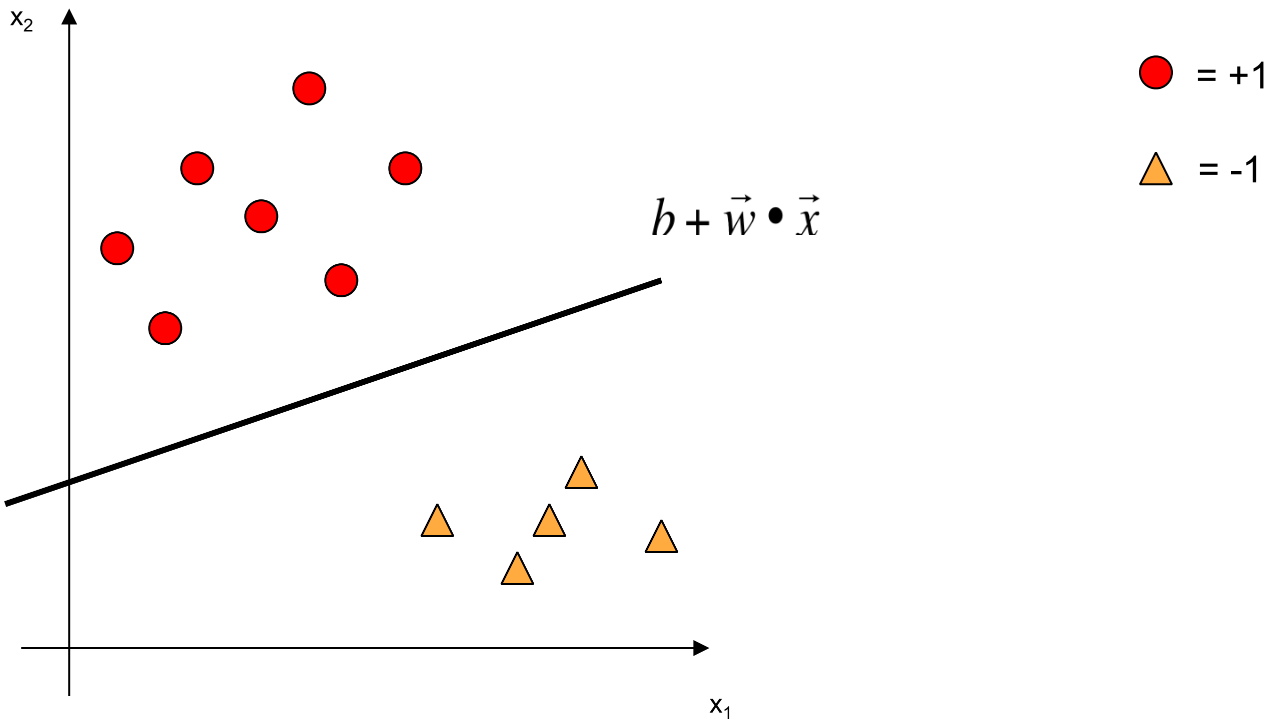
$$y' = b + w_1 x_1$$

But this is the equation of a <u>line</u> with slope *w* and offset *b*.

<u>Observation:</u> For the general case the perceptron computes a <u>hyperplane</u> in order to accomplish its classification task,

$$y' = b + \sum_{i=1}^{m} w_i x_i = b + \vec{w} \bullet \vec{x}$$

Classification



In order for the hyperplane to become a classifier we need to find *b* and *w* => learning!

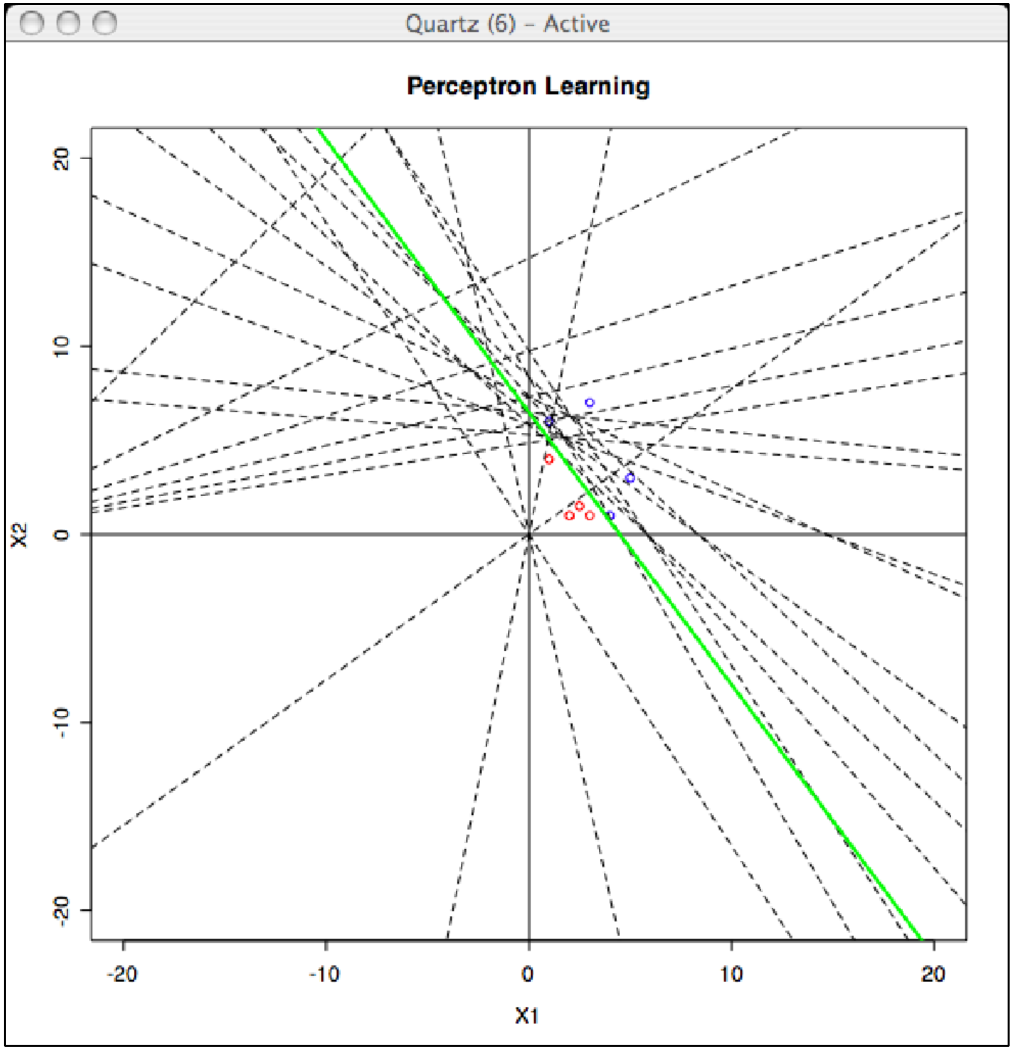
Learning Algorithm

Let $D = \{(\overline{x}_1, y_1), (\overline{x}_2, y_2), \ldots, (\overline{x}_n, y_n)\} \subset H \times \{-1, +1\}$
$\overline{w} \leftarrow \overline{0}$
$b \leftarrow 0$
$R \leftarrow \max_{1 \le i \le n} |\ \overline{x}_i\ |$
$\eta \leftarrow 0 < \eta < 1$
repeat
    for $i = 1$ to n
        if $sign(\overline{w} \bullet \overline{x}_i + b) \neq y_i$ then
            $\overline{w} \leftarrow \overline{w} + \eta y_i \overline{x}_i$
            $b \leftarrow b + \eta y_i R^2$
        end if
    end for
until no mistakes made in the for-loop
return $(\overline{w}, b)$

Note: learning is very different here compared to decision trees…here we have many passes over the data until the perceptron converges on a solution.

Demo

R perceptron demo



Observations

- The learned information is represented as weights and the bias ⇒ <u>sub-symbolic learning</u>
- In order to apply this learned information we need a neural network structure
- The learned information is not directly accessible to us ⇒ <u>non-transparent model</u>