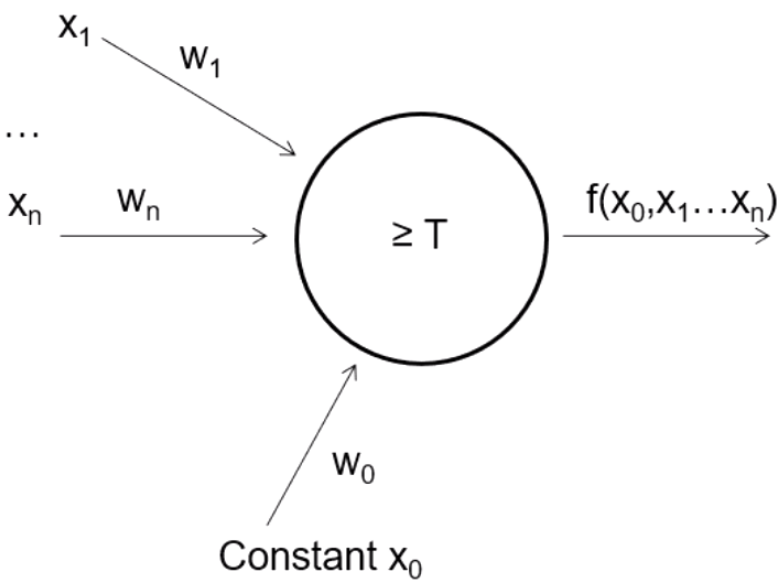
Perceptron Overview

- [[Artificial Neural Networks]] neutron
- inputs are now real numbers instead of just true or false
- each input has a certain weight

$$f(x_0 \ldots x_n): w_0x_0 + w_1x_1 + \ldots w_nx_n \geq T$$

- 

- perceptron networks allow non-linear separation

Perceptron Learning Algorithm

- $$f(x_0 \ldots x_n): w_0x_0 + w_1x_1 + \ldots w_nx_n \geq T$$
  f is our hypothesis!
- Variables:
  - d_j = desired output of perceptron for example j
    D = training set of example pairs $(\boldsymbol{x}_j, d_j)$
    $y_j(t)$ = actual output of perceptron for example j at time t
    r = learning rate
- Algorithm:

1. Initialize weights, typically to small values
2. For each example *j* in the training set D perform the following steps:
    1. $y_j(t) = (\mathbf{w}(t) * \mathbf{x}_j) \geq T$
    2. For all i=0…n: $w_i(t+1)=w_i(t) + r * (d_j-y_j(t))* x_{j,i}$

Weights stays the same if the output $y_j(t)$ is the same as the desired output d_j
Overall value needs to increase if actual output $y_j(t)$ is smaller than desired -> increase weights at positive inputs, decrease weights at negative inputs.
Overall value needs to decrease if actual output $y_j(t)$ is larger than desired -> decrease weights at positive inputs, increase weights at negative inputs.

- correct guess ==> weights stay the same
- bad guess ==>
    - output too small ==>
        - increase weight at positive inputs
        - decrease weight at negative inputs
    - output too big ==>
        - decrease weight at positive inputs
        - increase weight at negative inputs

- repeated until
    - average error below pre-defined threshold
    - or after pre-determined number of iterations

Example: Learning the OR function

Choice: $f(x_0, x_1, x_2, x_3) = w_0x_0 + w_1x_1 + w_2x_2 + w_3x_3 \geq 1$
$x_0=1$, ***r=0.5***, $w_i(0)=0$
Weight update formula from slide 13: $w_i(t+1)=w_i(t) + r * (d_j-y_j(t))* x_{j,i}$

t	Ex #	x_1	x_2	x_3	$w_0(t)$	$w_1(t)$	$w_2(t)$	$w_3(t)$	$y_{ex}(t)$	Correct (yes/no)
t=0	1	1	1	1						
t=1	2	1	1	0						
t=2	3	1	0	1						
t=3	4	1	0	0						

PLA Properties

- goes over set of examples
- compares each example's output with desired output
- weight changes based on correct output or not
- convergence towards solution guaranteed if training set is linearly separable
- may not find best solution, quality jumps around

- variants
  - batch learning
    - change weigths only after batch of training examples
    - less quality jumps
  - keep best seen solution in memory
  - include optimization criteria
    - maximizing distance of separation between both classes