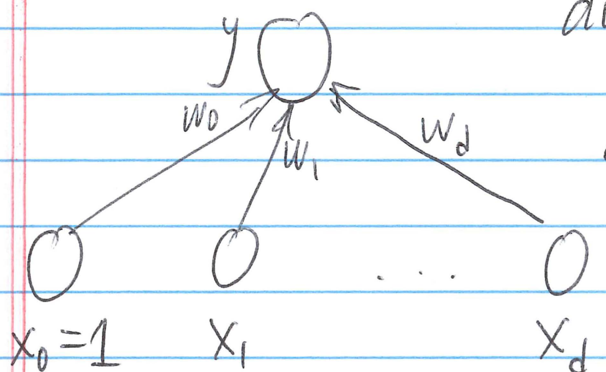
Chapter 11 Multilayer Perceptrons

Single perceptron:



Computes inner product between augmented input vector $\vec{x} = [1, x_1, \ldots, x_d]^T$ and weight vector $\vec{w} = [w_0, w_1, \ldots w_d]^T$

or

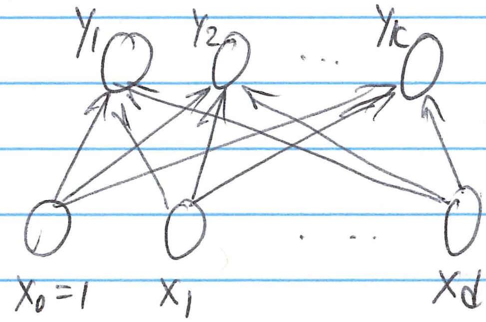
the result of that inner product passed through nonlinearity.

Key nonlinearities: Threshold $s(a) = \begin{cases} 1 & \text{if } a > 0 \\ 0 & \text{otherwise} \end{cases}$

sigmoid $s(a) = \dfrac{1}{1 + e^{-a}}$

With threshold nonlinearity, it separates input space with hyperplane perpendicular to $[w_1, w_2, \ldots, w_d]^T$, with offset determined by w_0.

Parallel perceptrons:

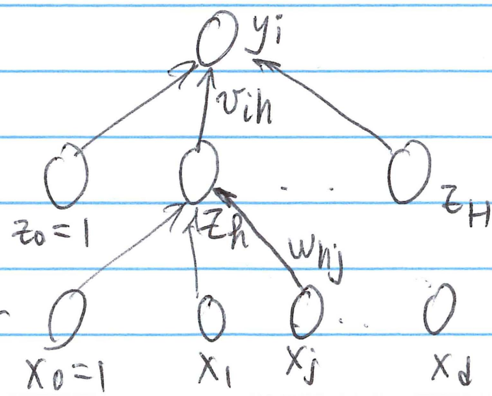


Weight vector becomes weight matrix.

Multilayer perceptrons:

H hidden units

More flexible than one layer



Training of multilayer perceptrons based on stochastic gradient descent: steps in direction of negative gradient of loss function, approximated from small number of samples or even one sample.

Key example: squared loss and sigmoid nonlinearity.
when r^t is the desired response } for training sample
y^t is output from MLP } indexed by t,

$$\Delta v_h = \eta (r^t - y^t) z_h^t$$

Note: uses intermediate computation at hidden unit.

$$\Delta w_{hj} = \eta (r^t - y^t) v_h z_h^t (1 - z_h^t) x_j^t$$

Called backpropagation because error $r^t - y^t$ is "backpropagated" from the output back to hidden layer and input layer.

One common use: Set response r^t to equal input x^t to train system to approximately reproduce input. Called an "autoencoder" and gives lower layers of MLP that work well for generic problems.