Neural Networks

First example:

$$f = \sigma(\underline{w}^T \underline{\phi} + b)$$

Logistic regression

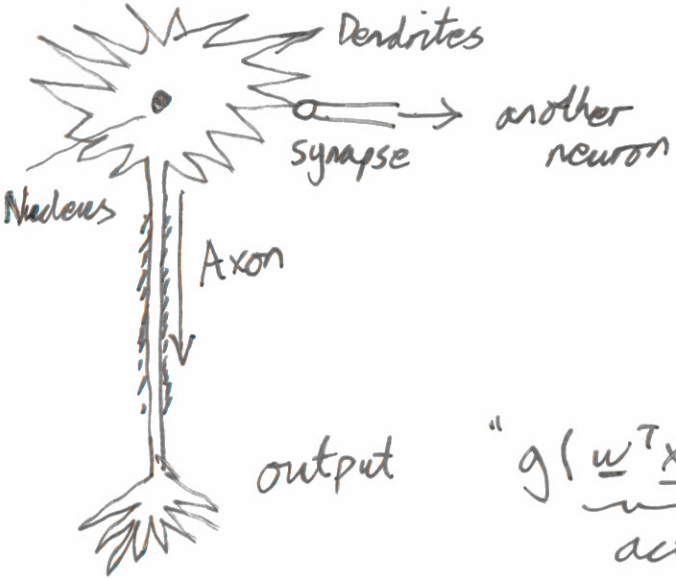
$$\phi_k = \sigma(\underline{v}^{(k)T}\underline{x} + b^{(k)})$$

"unit"
"neuron"

Fit $\{\{\underline{v}^{(k)}, b^{(k)}\}, \underline{w}, b\}$ with a gradient-based optimizer. Match f to training set, using some loss.

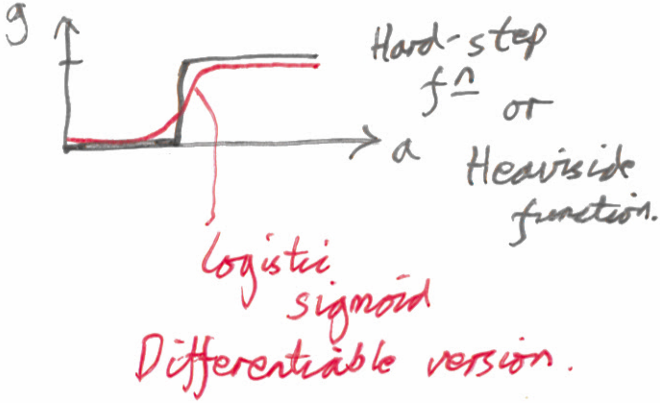
Why "Neural Network"? (non-examinable)

Neuron = Nerve cell



"Inputs" $\underline{x}$

output "$g(\underbrace{\underline{w}^T \underline{x}}_{\text{activation } a})$"

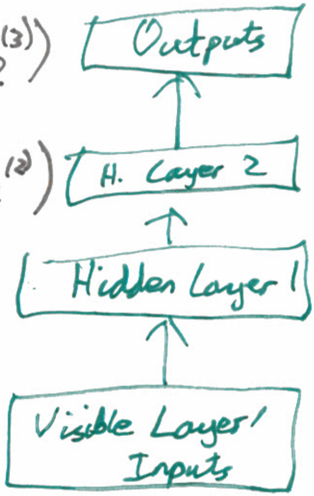


Feed-forward Neural Networks

$$\underline{f} = g^{(3)}(W^{(3)}\underline{h}^{(2)} + \underline{b}^{(3)})$$

$$\underline{h}^{(2)} = g^{(2)}(W^{(2)}\underline{h}^{(1)} + \underline{b}^{(2)})$$

$$\underline{h}^{(1)} = g^{(1)}(W^{(1)}\underline{x} + \underline{b}^{(1)})$$



- When f is a scalar, $W^{(3)}\underline{h}^{(2)} = \underline{w}^{(3)T}\underline{h}^{(2)}$

- Other architectures possible:
  "skip connections"
  parameterize the g's non-linearities

- Special layers for images/audio
  ConvNets ... and others.

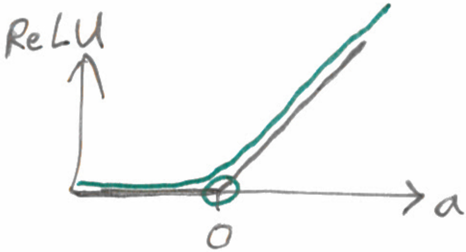
Non-linearities

These are what we called basis functions

Sigmoid σ : softly partitions space.

RBFs : is input near some point.

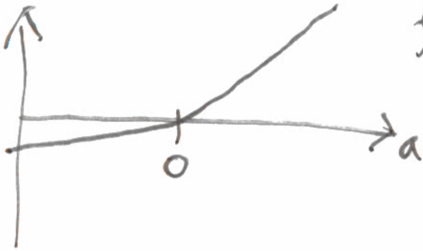
ReLU : Rectified Linear Units



Soft Plus
$= \log(1 + e^{a})$

$$\text{ReLU}(a) = \max(a, 0)$$

PReLU



$$f(a) = \begin{cases} a & a > 0 \\ sa & a \leq 0 \end{cases}$$

Parameter

Initialize the weights

Set initial weight matrix $W^{(l)}$

Must not set $W^{(l)}$ to be all zeros.

$\Rightarrow$ All hiddens extract same features

$\Rightarrow$ Weights stay the same.

$\rightarrow$ Randomly set each weight.