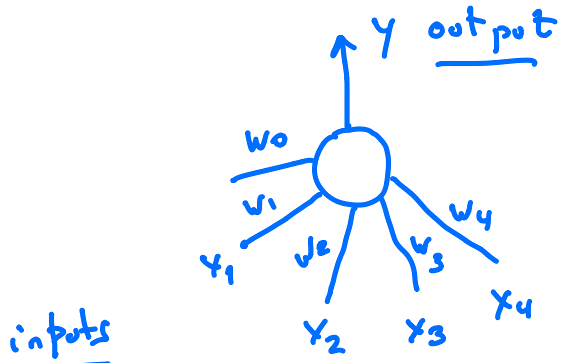
W08 - Neural Networks. Learning as Inference

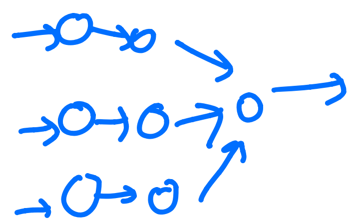
MacKay's lectures 15, 16. Chapters 39, 41, 42.

Feedforward Networks

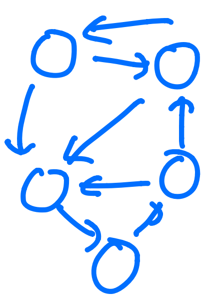
A single neuron



Feedforward network



Feedback network



Elements of the network

- inputs $x_1, \ldots, x_I$
- output y = activity $y >$ threshold, neuron fires
- weights (parameters) $w_0, w_1, \ldots, w_I$

This is not a probabilistic process

Given the inputs, how does the neuron produce y?

i) neuron adds all weights

$$a = w_0 + \sum_{i=1}^{I} w_i x_i \quad = \text{activation}$$

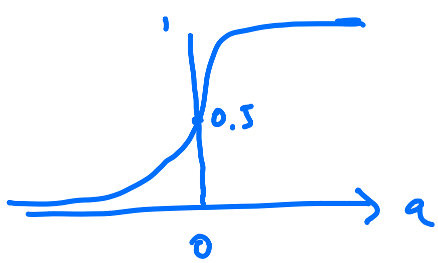
$\uparrow$ bias

ii) the activity of the neuron y is a function of the activation a, $y(a)$

$y(a)$ is generally a "threshold" function

Several commonly used forms for the activity $y(a)$ are:

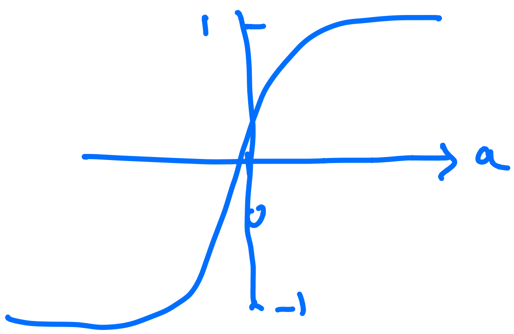
. The linear logistic function



$$y(a) = \frac{1}{1+e^{-a}} = \frac{1}{1+e^{-\bar{w}\bar{x}}}$$

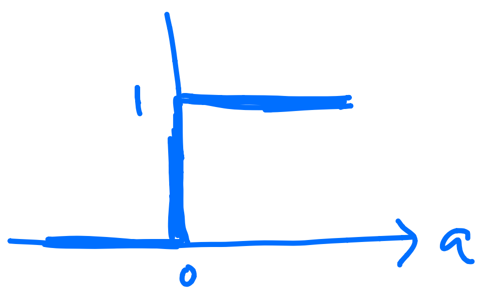
$$\bar{w}\bar{x} = \sum_i w_i x_i + w_0$$
$$= \sum_{i=0}^{I} w_i x_i, \quad x_0 = 1$$

· the sigmoidal function



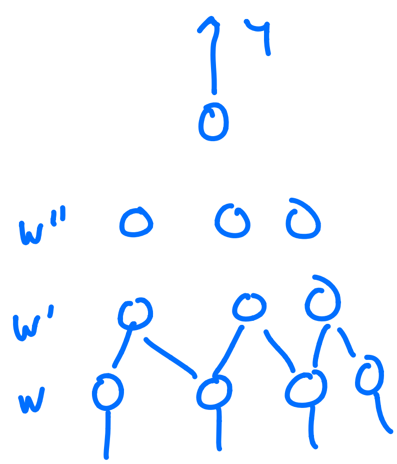
$$y(a) = \tanh(a) = \frac{e^{a} - e^{-a}}{e^{a} + e^{-a}}$$

. The step function



$$y(a) = \begin{cases} 1 & a > 0 \\ 0 & a \leq 0 \end{cases}$$

Recap Neural Network



i) architecture

w_i

ii) activity $a = \sum_{i=0}^{I} w_i x_i$

$= \bar{w} \bar{x}$

iii) the activity rule $y(a) = \frac{1}{1+e^{-a}} = \frac{1}{1+e^{-\bar{w}\bar{x}}}$

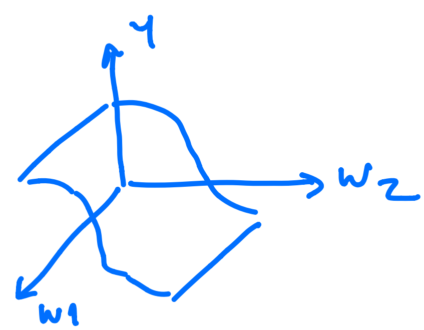
the activity $y(a)$ can be interpreted as:

The probability according to the neuron (weights) that the inputs ($\bar{x}$) deserves a response

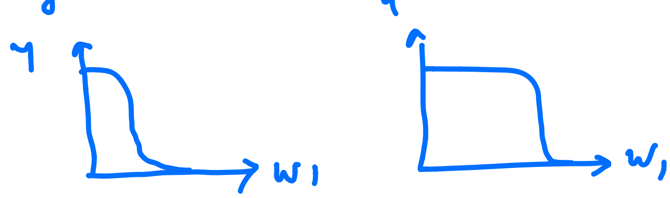
$y \simeq 1$ response

$y \simeq 0$ no response

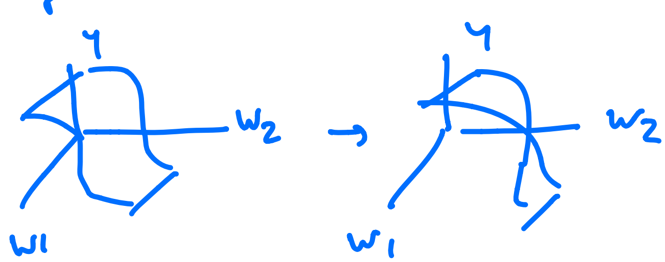
The Space of weights



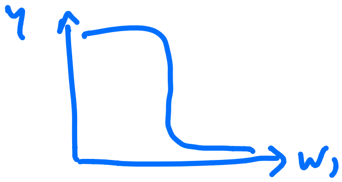
+ change in bias w_0

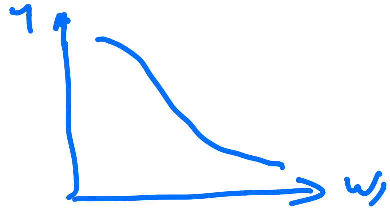


+ change in w_1/w_2 (a twist)



+ scale of weight





Contour plots

What can a single neuron learn?

to be a classifier!

Idea of supervised learning is:

given a number of examples of

input vectors $\bar{x}^{(1)} \dots \bar{x}^{(N)}$

and their outputs $t^{(1)} \dots t^{(N)}$

make the network learn their relationship.

Find the values of the weights so that $y^{(n)} \approx t^{(n)}$

"learning" ~ "finding parameters"

classification problem

$A \rightarrow t=1$

$O \rightarrow t=0$

w?

$y(A|w) \simeq 1$

$y(O|w) \simeq 0$

$$\text{error} = \{ y_w^{(1)} - t^{(1)}, \dots, y_w^{(N)} - t^{(N)} \}$$

learning = adjust w's so that error is small.

The Error Function

inputs: $\{\bar{x}^{(n)}, t^{(n)}\}_{n=1}^{N}$ $\quad \{x_i^{(n)}\}_{i=0}^{I}$

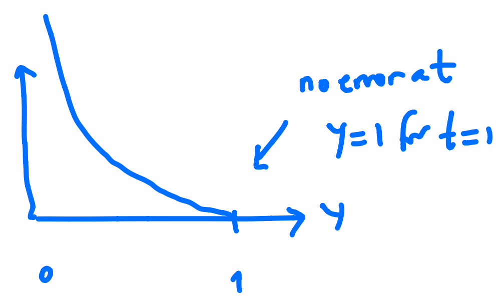
outputs $\quad y^{(n)}(\bar{x}^{(n)}, \bar{w}) = \dfrac{1}{1 + e^{-\bar{w}\bar{x}^{(n)}}}$

Introduce the error function

$$G(w) = -\sum_{n=1}^{N}\left[t^{(n)} \cdot \log y^{(n)} + (1-t^{(n)}) \log (1-y^{(n)})\right]$$

why?

$t^{(n)} = 1 : \; -\log y^{(n)}$



$t^{(n)} = 0 \; : \; -\log(1-y^{(n)})$

$$G(w) = -\sum_{n=1}^{N} \Big[t^{(n)} \log y^{(n)}(\bar{w},\bar{x})$$
$$+ (1-t^{(n)}) \log\big(1 - y^{(n)}(\bar{w},\bar{x})\big) \Big]$$

Back propagation

want to minimize error $G(w)$

$$G(\bar{w}) \geq 0 \quad , \quad G(w) = 0 \iff y^{(n)} = t^{(n)}$$

let's take the derivative of $G(\bar{w})$

$$\frac{\delta G(\bar{w})}{\delta w_j} = -\sum_n \left[\frac{t^{(n)}}{y^{(n)}} - \frac{1-t^{(n)}}{1-y^{(n)}} \right] \cdot \frac{\delta y^{(n)}}{\delta w_i}$$

$$= -\sum_n \frac{t^{(n)}(1-y^{(n)}) - y^{(n)}(1-t^{(n)})}{y^{(n)}(1-y^{(n)})} \frac{\delta y^{(n)}}{\delta w_i}$$

$$= -\sum_n \frac{t^{(n)} - y^{(n)}}{y^{(n)}(1-y^{(n)})} \cdot \frac{\delta y^{(n)}}{\delta w_i}$$

$$y^{(n)}(\bar{w},\bar{x}) = \frac{1}{1+e^{-\bar{w}\bar{x}^{(n)}}}$$

$$\frac{\delta y^{(n)}}{\delta w_i} = (-1)\left(1+e^{-\bar{w}\bar{x}^{(n)}}\right)^{-2} \cdot \frac{\delta}{\delta w_i}\left[e^{-\bar{w}\bar{x}^{(n)}}\right]$$

$$= \frac{-1}{\left[1+e^{-\bar{w}\bar{x}^{(n)}}\right]^{2}} \cdot (-w_i)\, e^{-\bar{w}\bar{x}^{(n)}}$$

$$= x_i^{(n)} \cdot \frac{e^{-\bar{w}\bar{x}^{(n)}}}{\left(1+e^{-\bar{w}\bar{x}^{n}}\right)^{2}} = x_i^{(n)}\, y^{(n)}\left(1-y^{(n)}\right)$$

$$\frac{\delta G(w)}{\delta w_j} = -\sum_n \left(t^{(n)}-y^{(n)}\right) x_j^{(n)} = -\sum_n e^{(n)} x_j^{(n)}$$

$$\left(\frac{\delta G}{\delta w_0}, \dots, \frac{\delta G}{\delta w_I}\right) = \underbrace{\boxed{\bar{g} = -\sum_n e^{(n)}\, \bar{x}^{(n)}}}_{\text{gradient vector}}$$

Backpropagation

Update weights by a quantity η in the opposite direction to the gradient

$$\bar{w}^{(old)}$$

$$\bar{w}^{(new)} = \bar{w}^{(old)} + \eta \sum_n e^{(n)} \bar{x}^{(n)}$$

$$= \bar{w}^{(old)} + \eta \sum_n \left(t^{(n)} - y^{(n)}(\bar{w}^{old}, \bar{x})\right) \bar{x}^{(n)}$$

η = learning rate, a free parameter

Different ways to update the weight

- batch gradient descent learning
- on-line gradient descent learning

batch learning

all weights are updated by looking at all data points at the time

- $\bar{w}^{(0)}$
- $\bar{w}^{(1)} = \bar{w}^{(0)} + \eta \sum_n \left(t^{(n)} - y_0^{(n)}\right) \bar{x}^{(n)}$

$$y_0^{(n)} = y^{(n)}\left(\bar{w}^{(0)}, \bar{x}^{(n)}\right)$$

On-line learning

Change all the weights by looking at one data point at the time

- $\bar{w}^{(0)}$
- take $m \in M$

$$\bar{w}^{(1)} = \bar{w}^{(0)} + \eta \left[t^{(m)} - y^{(m)}\left(\bar{w}^{(0)}, \bar{x}^{(m)}\right)\right] \bar{x}^{(m)}$$

How well does the batch learning algorithm do?

Apples / oranges

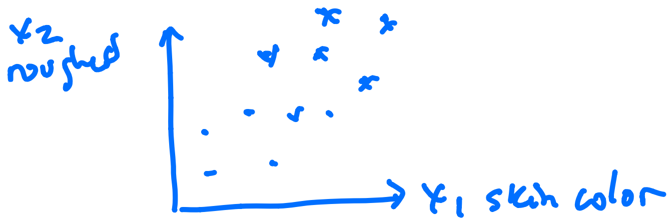
2 inputs x^1 - skin color $\quad w_0$

x^2 - surface roughness $\quad w_1$

w_2

$$Y(\bar{w}, \bar{x}) = \frac{1}{1+e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

$N = 10$ data points

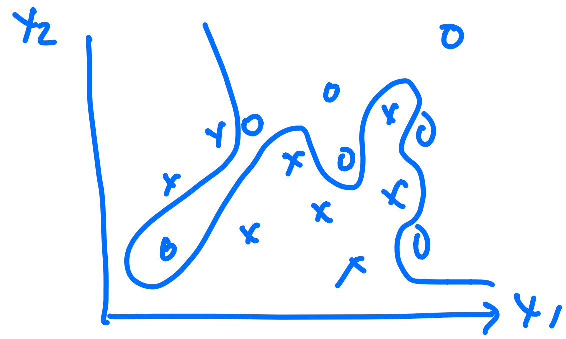


$\eta = 0.01 \quad w_0^{(0)} = w_1^{(0)} = w_2^{(0)} = 0$

→ class code

Regularization

To avoid overfitting



$G(w)$

$$M(\bar{w}) = G(\bar{w}, \bar{x}) + \alpha R(\bar{w})$$

$$R(\bar{w}) = \frac{1}{2} \sum_i w_i^2.$$

$$\frac{\delta M}{\delta \bar{w}_i} = \frac{\delta G}{\delta w_i} + \frac{\delta R}{\delta w_i} = \frac{\delta G}{\delta w_i} + \alpha w_i$$

$$\bar{w}^{(new)} = \bar{w}^{old}(1 - \eta\alpha) + \eta \sum_n (t^n - y_i^{(n)}) \bar{x}^{(n)}$$