Feed forward networks

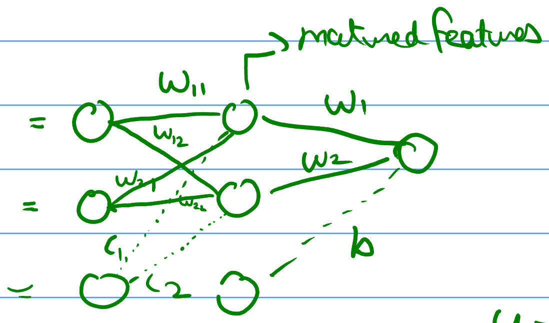
x_1	x_2	t
0	0	0
0	1	1
1	0	1
1	1	0

4 examples

$$\sum_{i=1}^{4}\left(\left(w_1 x_1^{(i)} + w_2 x_2^{(i)} + b\right) - t_i\right)^2$$

Linear regression != neural network

$w_1 = 0$ $w_2 = 0$ & $b = 0.5$



$$h = f^{(1)}(x; W, c)$$

$$y = f^{(2)}(h, w, b) = w^T h + b$$

$$y = f^{(2)} f^{(1)}(x; W, c, w, b)$$

What will be $f^{(1)}$?

L.T. $[2 \quad 1]$

$\begin{bmatrix}2\\1\end{bmatrix} \in \mathbb{R}^2$ to $\mathbb{R}^3$ can be $\begin{bmatrix} f_1\left(\begin{bmatrix}2\\1\end{bmatrix}\right) \\ f_2\left(\begin{bmatrix}2\\1\end{bmatrix}\right) \\ f_3\left(\begin{bmatrix}2\\1\end{bmatrix}\right) \end{bmatrix}$

$$\begin{bmatrix}3 & 1 & 5\\ 4 & 2 & 6\end{bmatrix}\begin{bmatrix}2\\1\end{bmatrix} \rightarrow \begin{bmatrix}10\\4\\16\end{bmatrix}$$

$$\left.\begin{aligned} f_1 &= 3x_1 + 4x_2 \\ f_2 &= x_1 + 2x_2 \\ f_3 &= 5x_1 + 6x_2 \end{aligned}\right\}$$ linear combination

⇒ linear transformation

Affine transformation

Affine transform: Linear transformation + constant vector added to result

$$f^{(1)}(x : W, C) = \underbrace{W^T x + b}_{\text{Affine transform}} \Big\} \text{matured space}$$

For brevity (and to make life easier), let us just consider a linear transformation

$$y = \omega^T (W^T x)$$

$$= (\omega')^T x \text{ where } (\omega')^T = \omega^T W^T$$

$$\sum_{i=1}^{4} (\omega^T x - t_i)^2$$

1' SO, The resultant matured features of a linear transformation is a linear model. (Which can't fit a non linear function like XOR)

So as a result of this problem, all of deep learning and
Some machine learning algorithms use something else on top of it

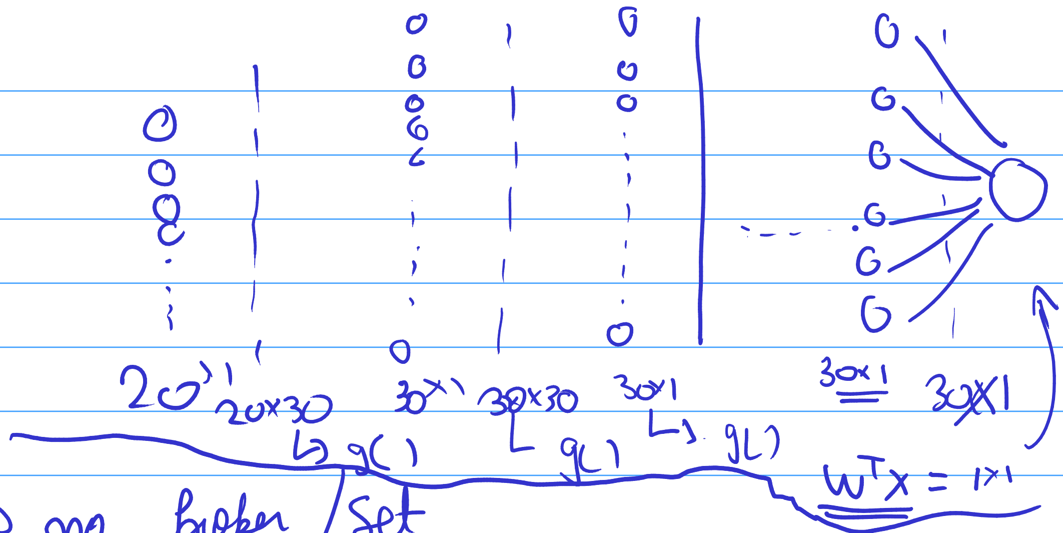
$$\begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} = \begin{bmatrix} w_{11} + w_{21} + c_1 \\ w_{12} + w_{22} + c_2 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$

Add a non linear function $= \begin{bmatrix} g(w_{11} + w_{21} + c_1) \\ g(w_{12} + w_{22} + c_2) \end{bmatrix}$

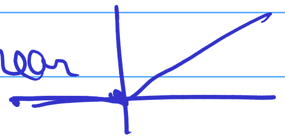
one of them could be

$g(z) = \max(z, 0) \Rightarrow$ This function is an activation function

applied to every hidden layer



Theres no proper / set "algorithm" to choose an activation function, You have to experiment/make a judgement to choose

RELU: piecewise linear  but not actually linear ⇒ nonlinear

→ It gives nonlinear properties, and can be used as activation

⇒ but you need to do the math involved.

→ It even has some biological motivation.

Similarly, there is not set algo for Number of layers we will needs insights for the domain/ experimentation if ~~to~~ the ~~above~~ insights ~~and~~ are not there.