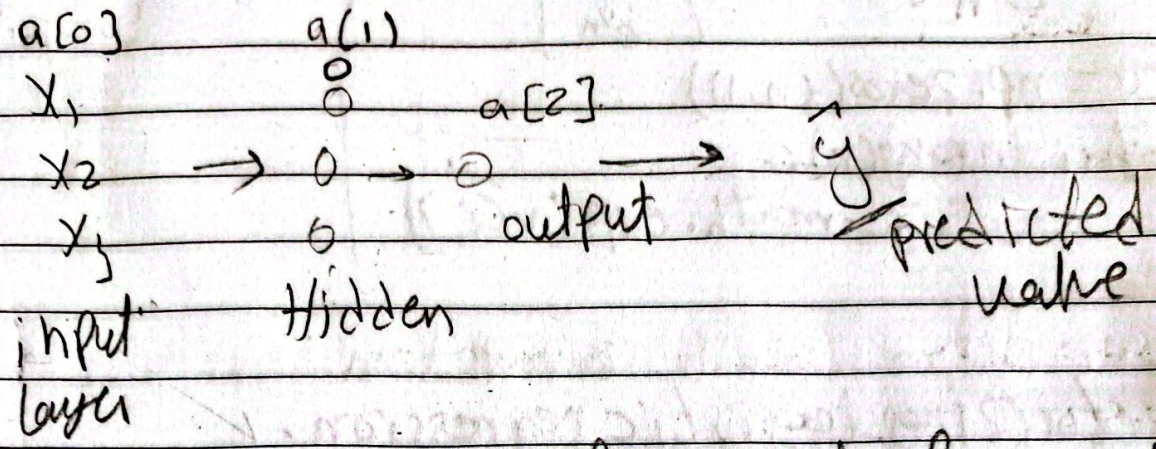
(week 3)

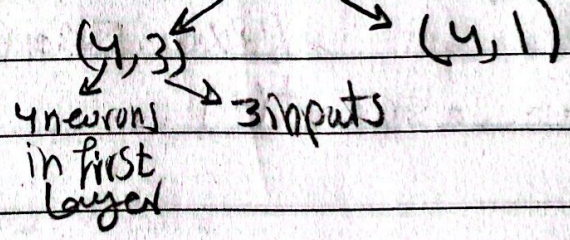
→ Neural Network overview ✓

→ Neural Network representation



- The term hidden refers to the fact that the training set the true values for these node in the middle are not observed, that is you don't see what they should be in the training set

- in hidden layers we have $W^{[1]}, b^{[1]}$

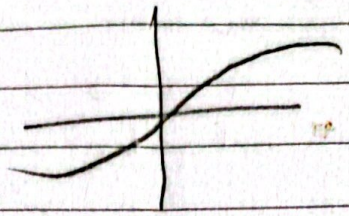


→ Computing a Neural Network's output ✓

→ Activation function

- The sigmoid function goes bet 0 → 1
- tangen fun. works better sigmoid bec it have zero mean bet +1 & -1

formula $= \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$



- sometimes I use different activation funs. to diff layers

- The down side of both Sigmoid & tanh fun. is $z = wx + b$ is very larg (or) very small then the gradient of the derivative of the slope becomes very small

- relu formula $a = \max(0, z)$
- in relu the slope (or) derivative of activation fun is very different from zero

→ why need non linear Act. funs.

→ Derivative of activation fun.

- when we implement backpropagation we need to either compute slope (or) derivative of activation fun.

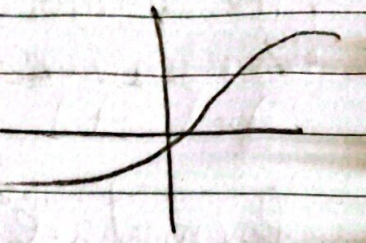
- so if it is Sigmoid $g(z) = \frac{1}{1 + e^{-z}}$

$\frac{d}{dz} g(z) \longrightarrow = g(z)(1 - g(z))$

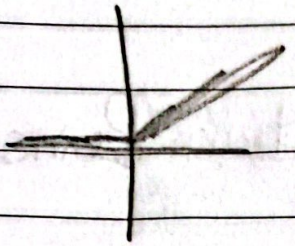
PAGE
DATE

- $g(z) = \frac{d}{d(z)} \longrightarrow 1 - (\tanh(z))^2$

$g(z) = \tanh(z)$



- in relu



$g(z) = \max(0, z)$ $\quad g'(z) = \begin{cases} 0, & z<0 \\ 1, & z \geq 0 \end{cases}$ undefined if $z=0$

→ gradient descent for neural net.

- forward prop.

$z^1 = W^1 X + b^1$

$A^1 = g^1(z^1)$

$z^2 = W^2 X + b^1$

$A^2 = g(z^2) = \sigma(z^2)$

back prop

$dz^2 = A^2 - y, \quad y = y^1 \ldots y^m$

$dw^2 = \frac{1}{m} dz^2 A^{1T}$

$db^1 = \frac{1}{m} np.sum(dz^2, axis=1)$

→ random initialization:

- The problem with if all the inputs are zero is that all the activation functions will be exactly the same

- So the solution is to initialize my parameters random