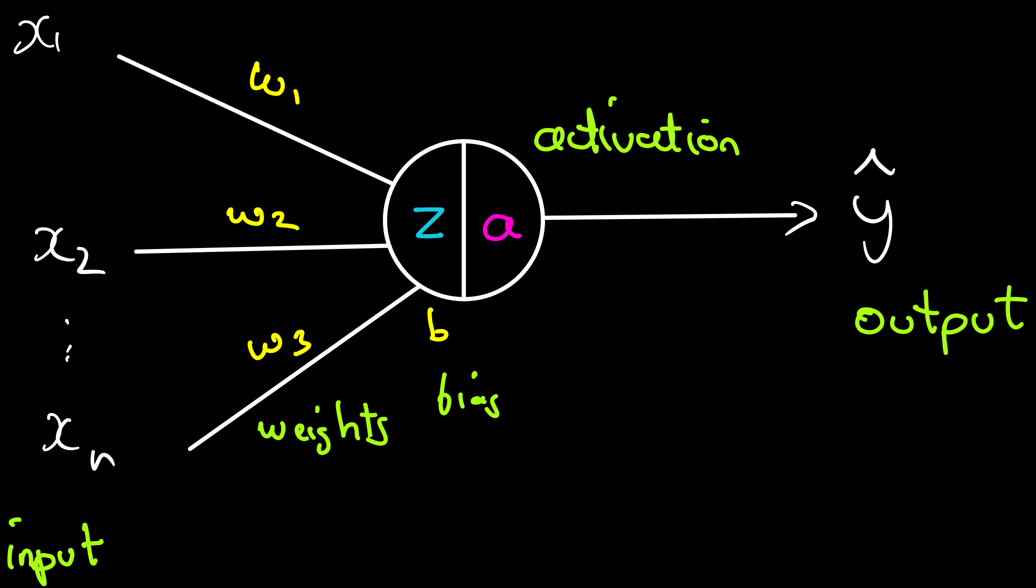
Forward Propogation & Loss

[Neural Networks]

Recap

Neuron



$$\hat{y} = a(w_1 x_1 + w_2 x_2 \ldots + w_n x_n + b)$$

Generalized: if we change this activat^n func^n it behaves like different models

For example:

$a =$ step $\rightarrow$ perceptron

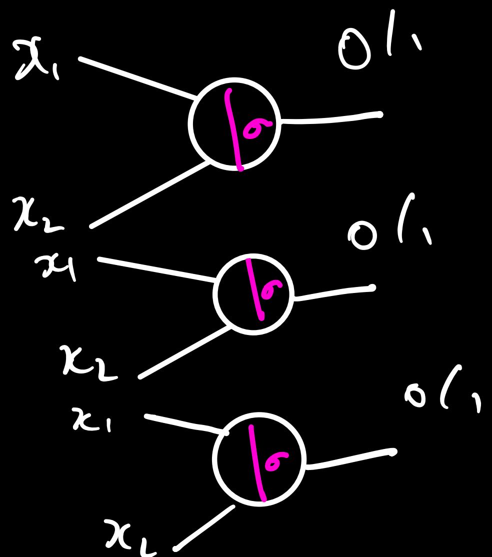
$a =$ linear $\rightarrow$ linear regression

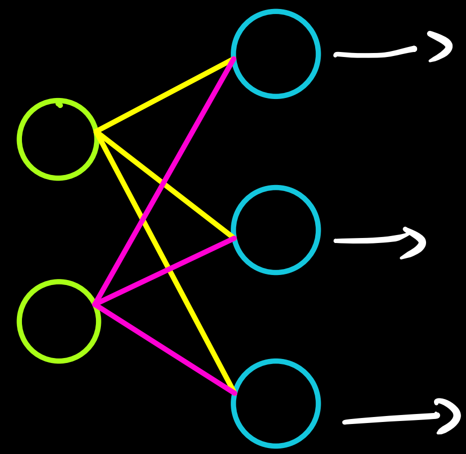
$a =$ sigmoid $\rightarrow$ logistic regression

... and so on.

Let's build a small Network

Using one vs rest approach with binary LRUs (log Reg Units) ; i.e. $\rightarrow a = \underset{\text{sigmoid}}{\sigma}$

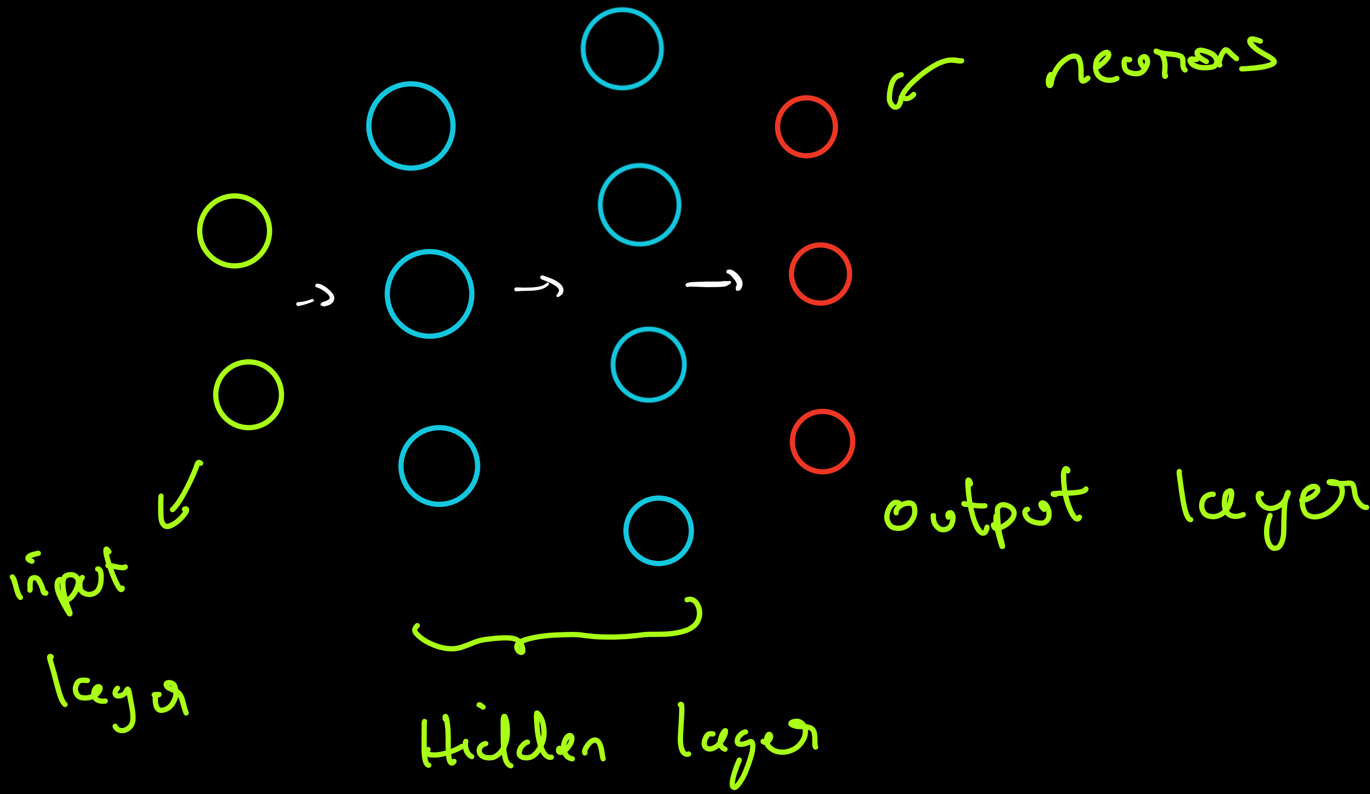




Now this is a single model which will be trained (All LRUs simulteneously)

Obviously we need to fix the probabilities which we will do later...

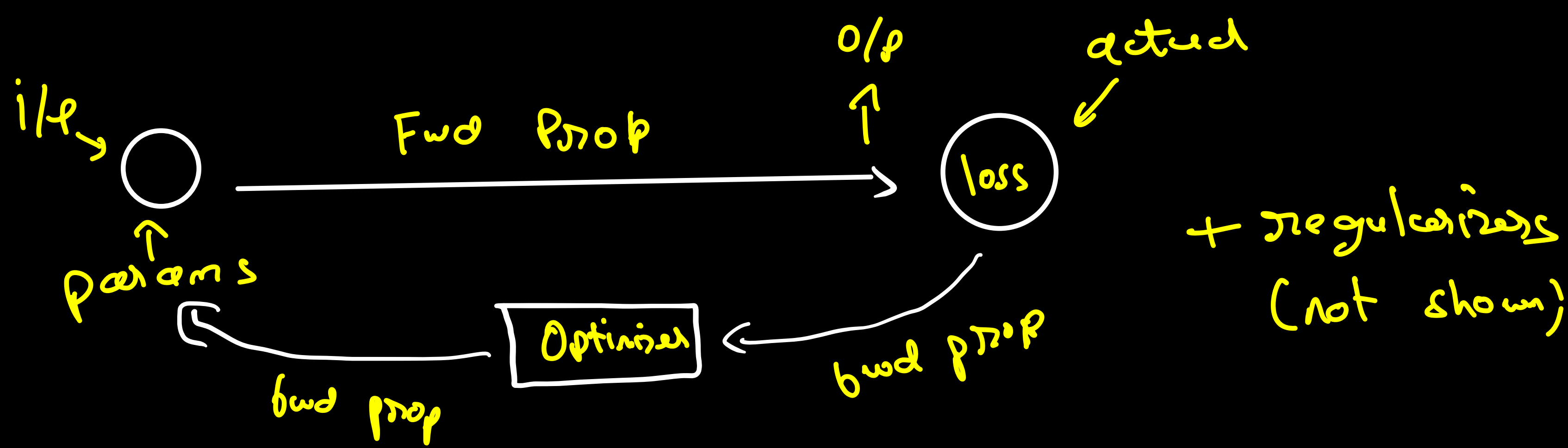
Components of a neural network



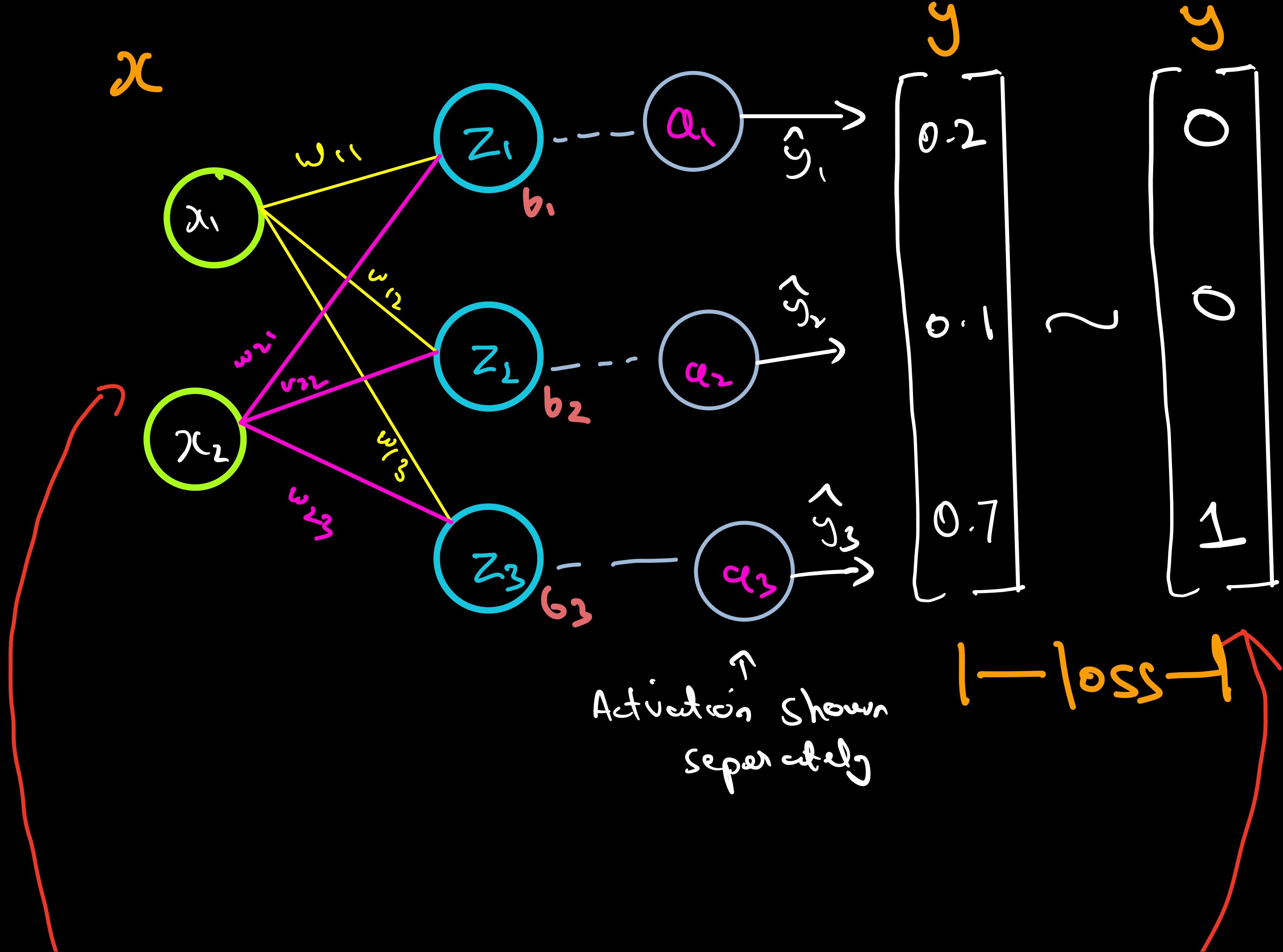
Components

- Neurons + layers

- input / output
- Fwd Prop
- loss
- bwd Prop
- optimizers
- regularisers



Forward Propogation



$$\begin{array}{c|cc|} & f_1 & f_2 \\ \hline 1 & x_{11} & x_{12} \\ 2 & x_{21} & x_{22} \\ \vdots & \vdots & \\ n & x_{n1} & x_{n2} \end{array} \quad \begin{array}{|c|} y \\ \hline y_1 \\ \vdots \\ y_n \end{array} \xrightarrow{\text{OHE}} \begin{array}{|ccc|} y_1 & y_2 & y_3 \\ \hline 0 & 0 & 1 \\ 0 & 1 & 0 \\ & \vdots & \\ 1 & 0 & 0 \end{array}$$

$n \times 2$ $\quad n \times 1$ $\quad n \times 3$

Data Matrix $\quad$ Output Matrix $\quad$ Weights

$n \times d$ $\quad n \times K$ $\quad d \times K$

points ↵ $\quad$ ↳ # features $\quad$ ↳ # classes

$$a(w_{11}x_1 + w_{21}x_2 + b_1) = \hat{y}_1$$

$$a(w_{12}x_1 + w_{22}x_2 + b_2) = \hat{y}_2$$

$$a(w_{13}x_1 + w_{23}x_2 + b_3) = \hat{y}_3$$

$$\begin{bmatrix} x_{11} & \cdots & x_{1d} \\ \vdots & & \\ x_{n1} & \cdots & x_{nd} \end{bmatrix} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \end{bmatrix}$$

$n \times d$ $\quad$ $d \times K$ $\quad$ $n \times K$

Eg: 2×3

$= 6$ ✓

But we also need to add biases

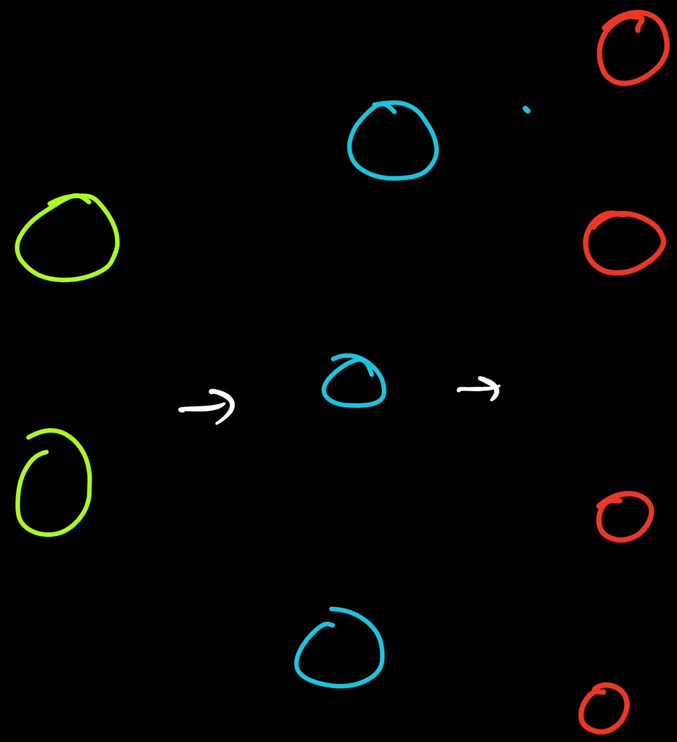
$$\begin{bmatrix} x_{11} & x_{12} \\ x_{21} & x_{22} \\ \vdots & \vdots \end{bmatrix}_{n \times d} \begin{bmatrix} w_{11} & w_{12} & w_{13} \\ w_{21} & w_{22} & w_{23} \end{bmatrix}_{d \times k} + \begin{bmatrix} b_1 & b_2 & b_3 \\ b_1 & b_2 & b_3 \\ \vdots & \vdots & \vdots \\ b_1 & b_2 & b_3 \end{bmatrix}_{n \times k} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} \end{bmatrix}_{n \times k}$$

$$a(X \cdot W + b) = \hat{y} \qquad n \times k$$

Q: What is the total number of learnable params in this model?

→ 6 weights + 3 biases = 9 params

Q: Num of params → ?

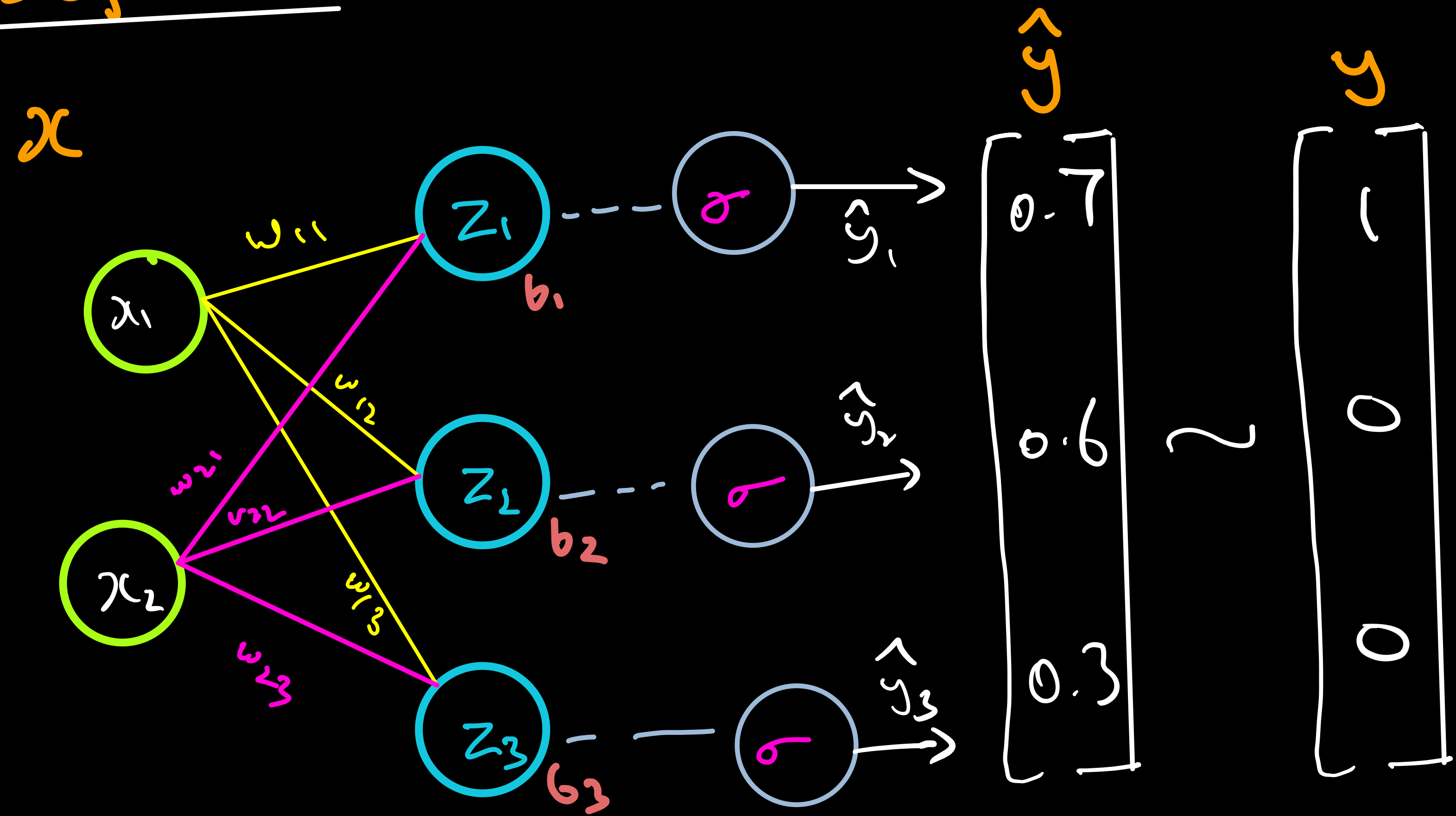


$$\text{Ans} \rightarrow (2\times3)+3 \; + \; (3\times4)+4$$

$$= 6+3 \; + \; 12+4$$

$$= 9 \; + \; 16 = \boxed{25}$$

Softmax



In our model the activatⁿ is sigmoid

So there is no garauntee that $\underbrace{\sum \hat{y}_i}_{\text{probability}} = \underline{1}$

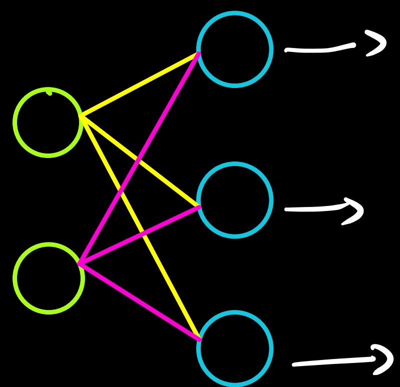
Hence we need to normalise, we have already seen that

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \; ; \quad K = \#\text{ classes}$$

<u>Why e^x ?</u>

→ easy to differentiate

→



$$\begin{bmatrix} 2 \\ 6 \\ 12 \end{bmatrix} \underset{\text{loss}}{\sim} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

direct normalize:

$$\frac{2}{2+6+12}, \quad \frac{6}{2+6+12}, \quad \frac{12}{2+6+12}$$

$$= \quad 0.1 \quad 0.3 \quad 0.6$$

$$= \begin{bmatrix} 0.1 \\ 0.3 \\ 0.6 \end{bmatrix} \sim \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \rightarrow \text{High difference}$$

Softmax

$$\frac{e^2}{e^2+e^6+e^{12}} \quad \ldots = \begin{bmatrix} 0.0004 \\ 0.002 \\ 0.997 \end{bmatrix} \sim \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Low difference

Loss

Q: Can we use rmse to compare $\hat{y}, y$?!

→ No, better choice:

For 2 classes

$$\min_{w,b} J(w,b) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p_i) + (1-y_i)(\log(1-\hat{y}_i))$$

A.K.A → logloss → binary cross entropy loss

Similarly for K classes

$$\min_{w,b} J(w,b) = \sum_{i=1}^{N}\sum_{j=1}^{K} y_{ij} \log(P_{ij})$$

aka → Categorical cross entropy loss

Back Propogation

To update the param w_{ij} & I need,

$$w_{11}' = w_{11} - \alpha \frac{\partial J}{\partial w_{11}}$$

But J is not a direct function. hence