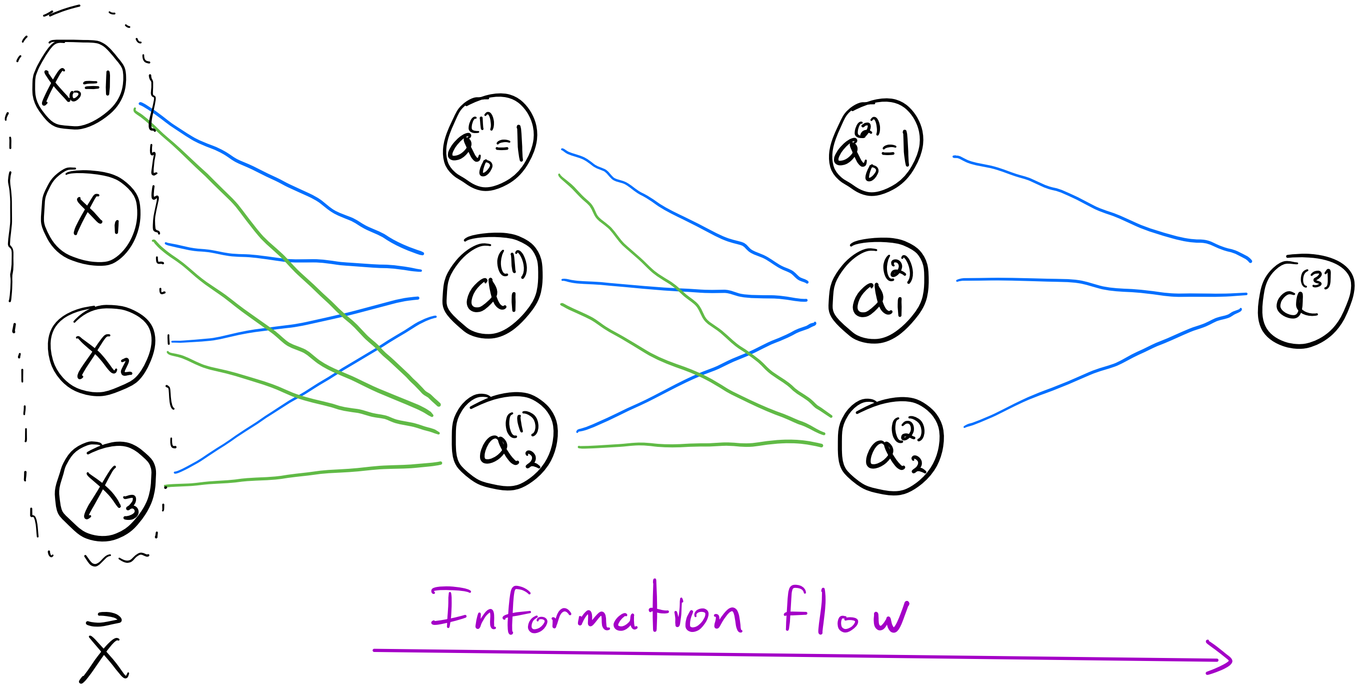
Neural Networks (NN)

A NN is a function $f(\vec{x})$

Ex: $f(\vec{x}) = a^{(3)}$



ERM with NNs (High Level)

$$D = ((\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n))$$

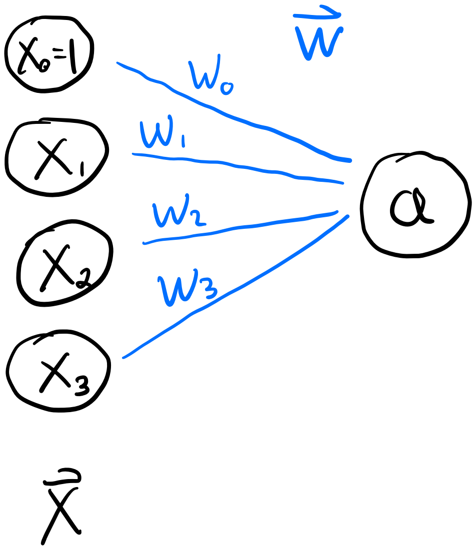
$$\mathcal{A}(D) = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \hat{L}(f)$$

$\mathcal{F} = \{ f \mid f : \mathcal{X} \to \mathcal{Y}$ and f is a NN with a specific architecture$\}$

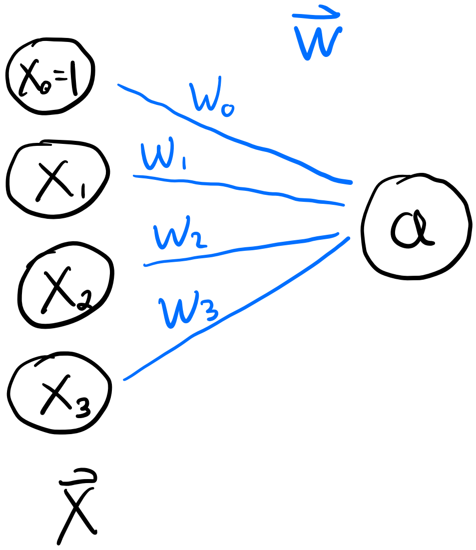
every $f \in \mathcal{F}$ is defined by B weight matrices $W^{(1)}, \ldots, W^{(B)}$

Previously Seen functions as NNs

Linear: $f(\vec{x}) = \vec{x}^T \vec{w}$, $d = 3$

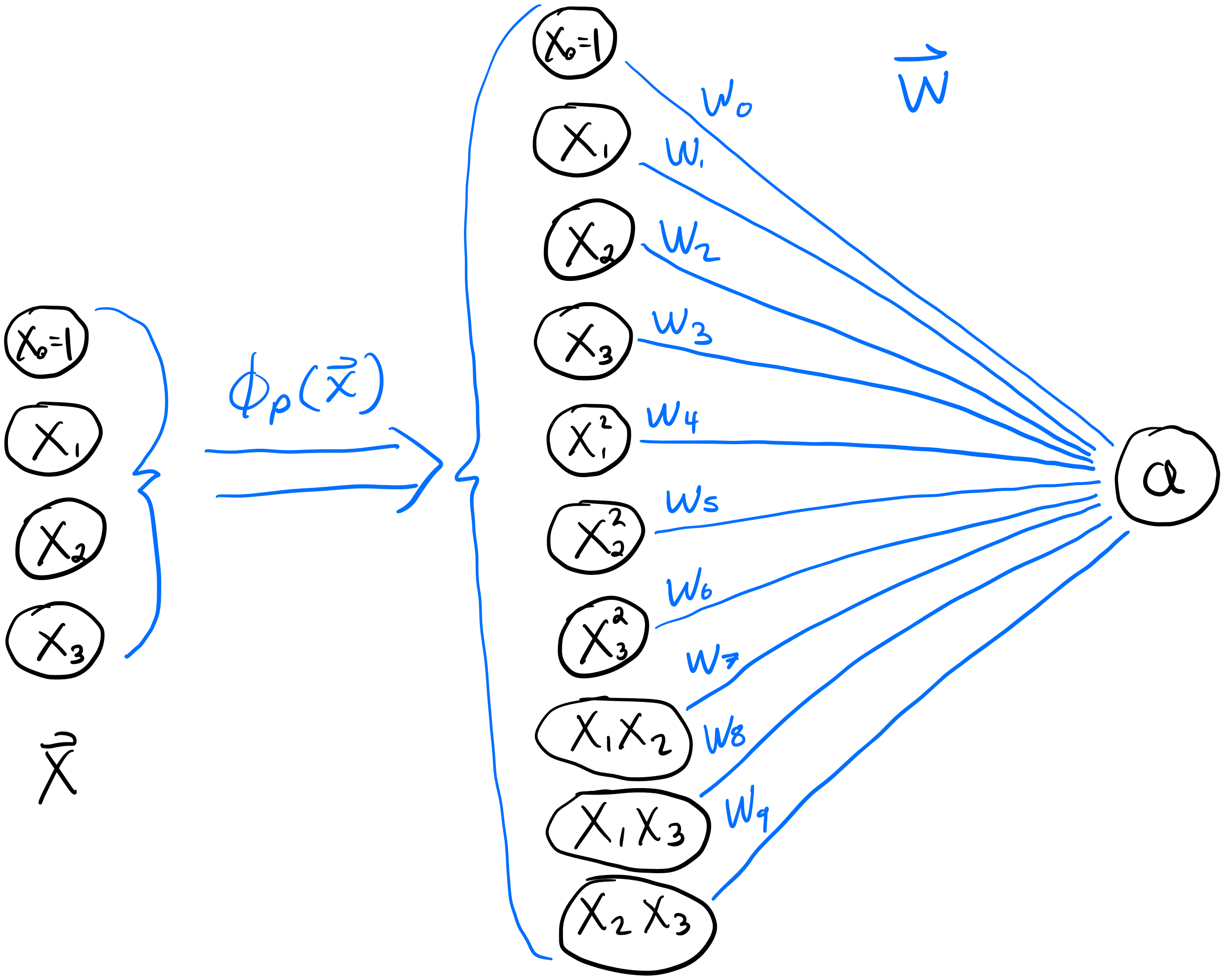


Sigmoid: $f(\vec{x}) = \sigma(\vec{x}^T \vec{w})$



Linear (with polynomial features):

$f(\vec{x}) = \phi_p(\vec{x})^T \vec{w} = a$, $d=3$, $p=2$



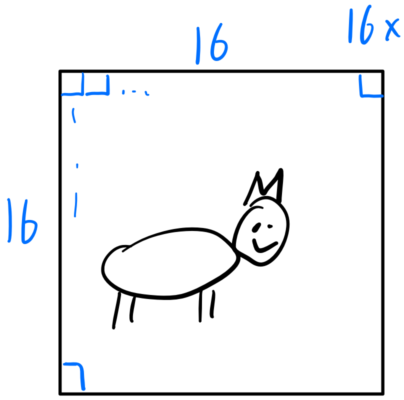
Ex: Binary Classification (Cat or No Cat)

$\mathcal{Y} = \{0, 1\}$ $\qquad \mathcal{X} = \mathbb{R}^{256+1}$

No cat → 0

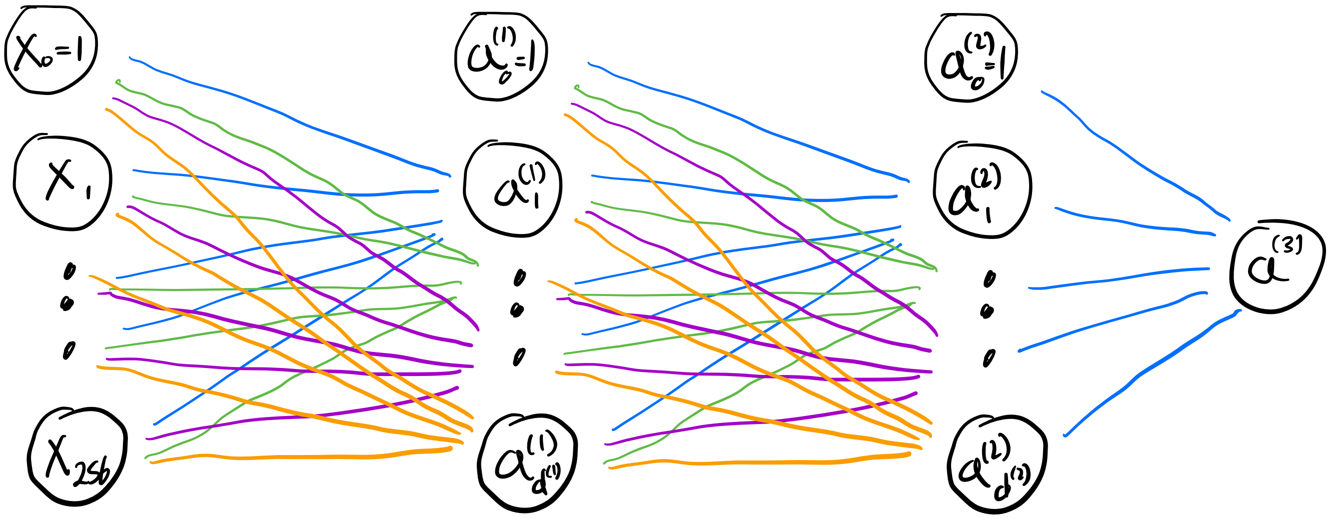
cat → 1

16x16=256=d



$$\longrightarrow \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_{256} \end{pmatrix} = \vec{x}$$

$f(\vec{x}) = a^{(3)}$ probability of a cat

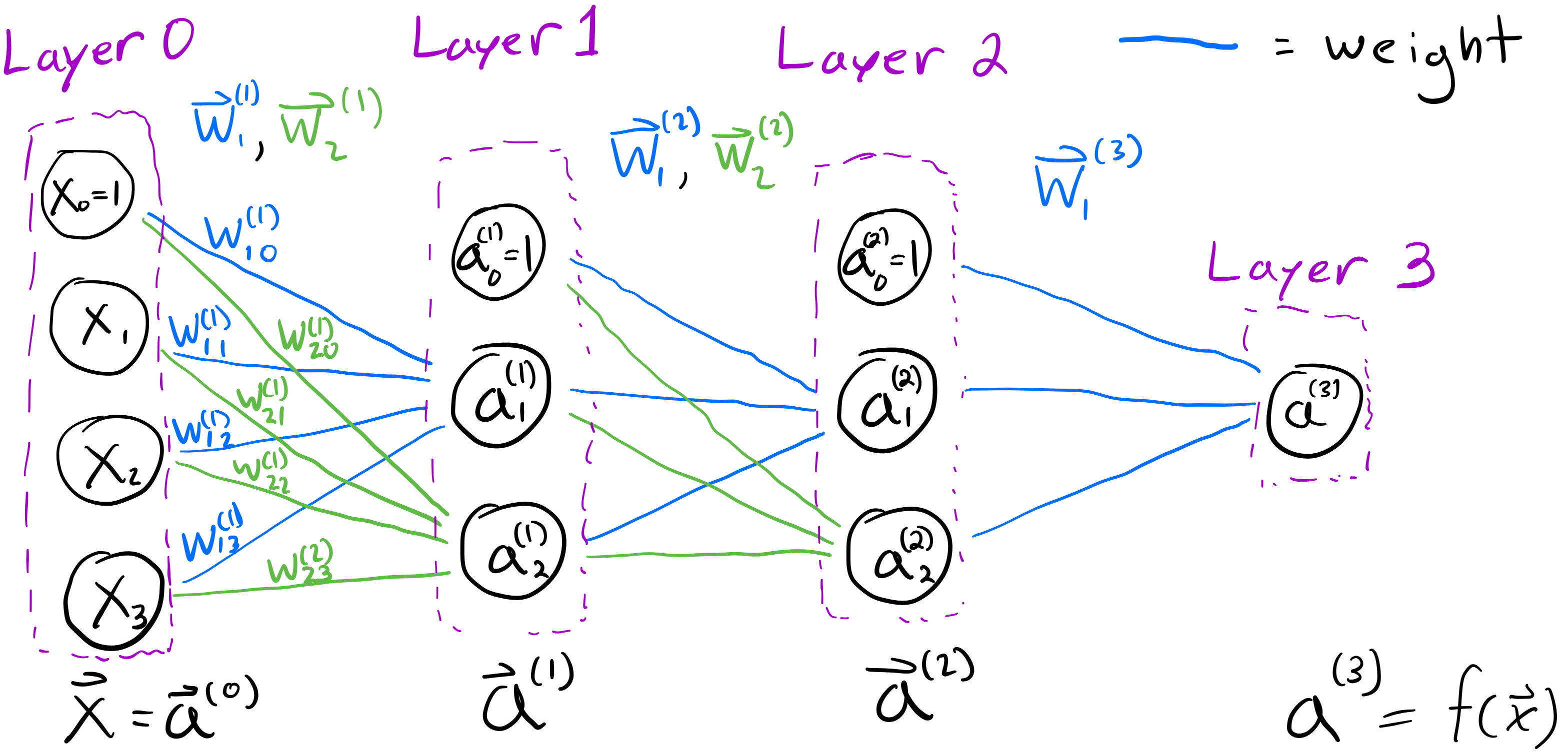


NN Definition: Specific Example

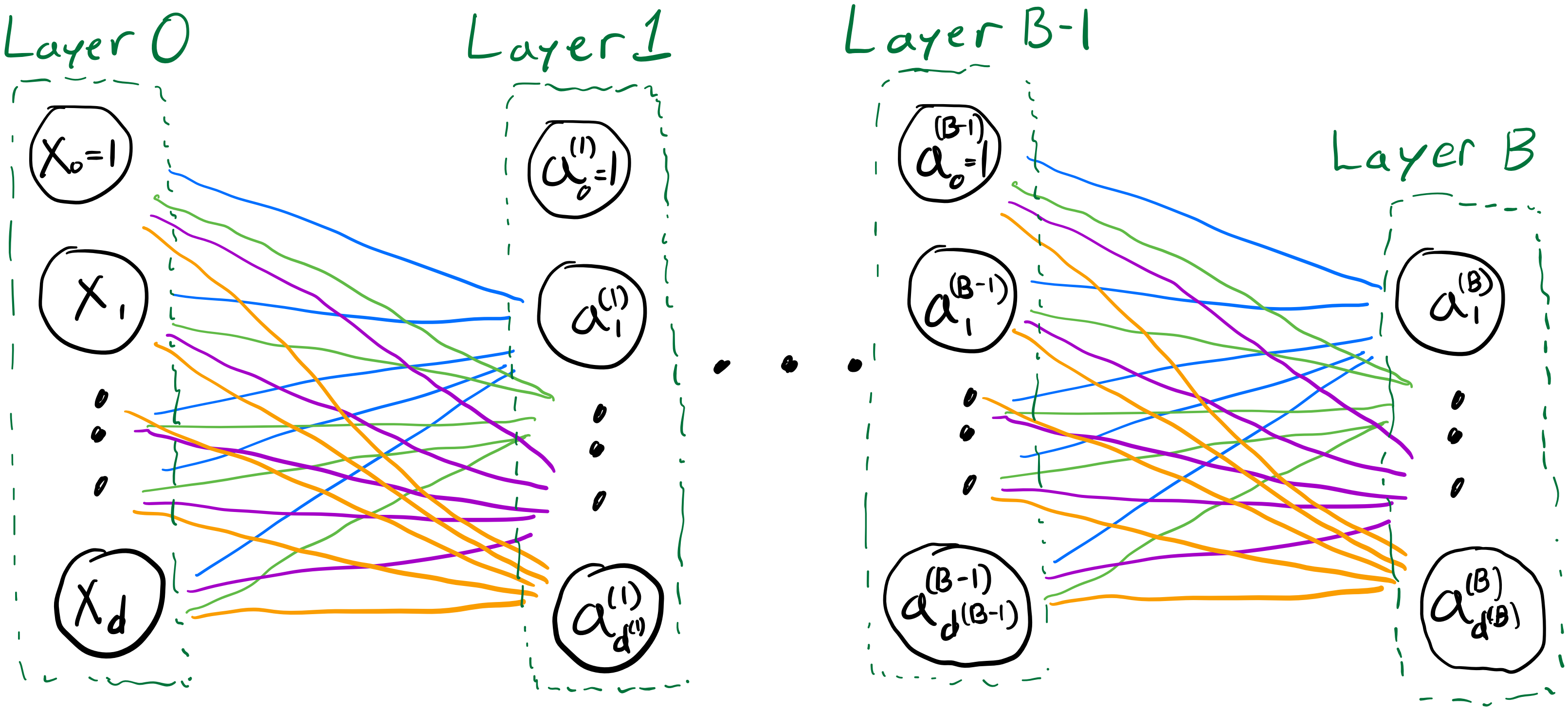
$d = 3$

○ = Neuron

— = weight



In general:



For layer $b \in \{1, \ldots, B\}$

Weights

Pre-activations

Activations

Activation function

Defining the number of layers B and the number of neurons in each layer: $d^{(1)}, \dots, d^{(B)}$, and the activation functions: $h^{(1)}, \dots, h^{(B)}$ defines a "NN architecture"

Matrix Notation

Represent $\vec{w}_1^{(b)}, \ldots, \vec{w}_{d^{(b)}}^{(b)}$ as a matrix:

$$W^{(b)} = \begin{bmatrix} | & | & & | \\ \vec{w}_1^{(b)} & \vec{w}_2^{(b)} & \cdots & \vec{w}_{d^{(b)}}^{(b)} \\ | & | & & | \end{bmatrix}$$

$$\vec{a}^{(b)} = h^{(b)}\left(\left(\vec{a}^{(b-1)}\right)^T W^{(b)}\right)$$

$$h^{(b)}(\vec{v}) = h^{(b)}(v_1, \ldots, v_d)$$
$$= \left(h^{(b)}(v_1), \ldots, h^{(b)}(v_d)\right)$$

ERM with Neural Networks

$D = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$

$\mathcal{A}(D) = \hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \hat{L}(f)$ where $\hat{L}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(\vec{x}_i), y_i)$

$\mathcal{F} = \{ f \mid f : \mathcal{X} \to \mathcal{Y}$ and f is a NN with a specific architecture$\}$

every $f_{W^{(1)}, \dots, W^{(B)}} \in \mathcal{F}$ is defined by B weight matrices $W^{(1)}, \dots, W^{(B)}$

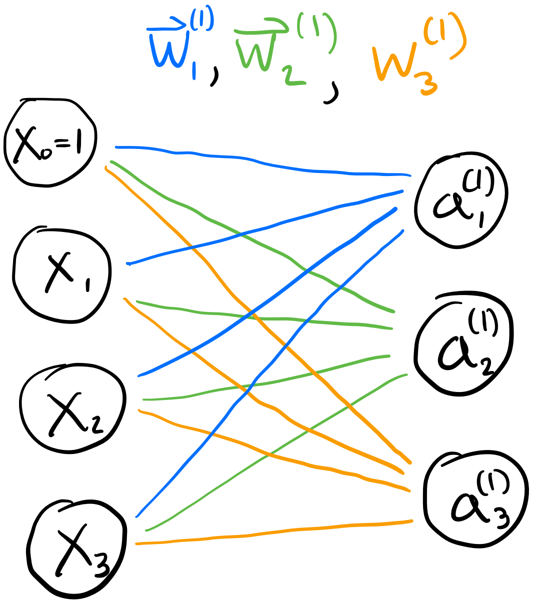
$\mathcal{A}(D) = \hat{f}$ where $\hat{f}(\vec{x}) = \vec{a}^{(B)}$ is defined by

Softmax

$d=3, k=3,$

$$\sigma\left(\vec{x}^T\vec{w}_1^{(1)}, \vec{x}^T\vec{w}_2^{(1)}, \vec{x}\vec{w}_3^{(1)}\right)$$

$$= \left(\sigma_1\left(\vec{x}^T\vec{w}_1^{(1)}, \vec{x}^T\vec{w}_2^{(1)}, \vec{x}\vec{w}_3^{(1)}\right), \sigma_2\left(\vec{x}^T\vec{w}_1^{(1)}, \vec{x}^T\vec{w}_2^{(1)}, \vec{x}\vec{w}_3^{(1)}\right), \sigma_3\left(\vec{x}^T\vec{w}_1^{(1)}, \vec{x}^T\vec{w}_2^{(1)}, \vec{x}\vec{w}_3^{(1)}\right)\right)^T$$



$$a_1^{(1)} = h^{(1)}\left(\vec{x}^T\vec{w}_1^{(1)}\right)$$

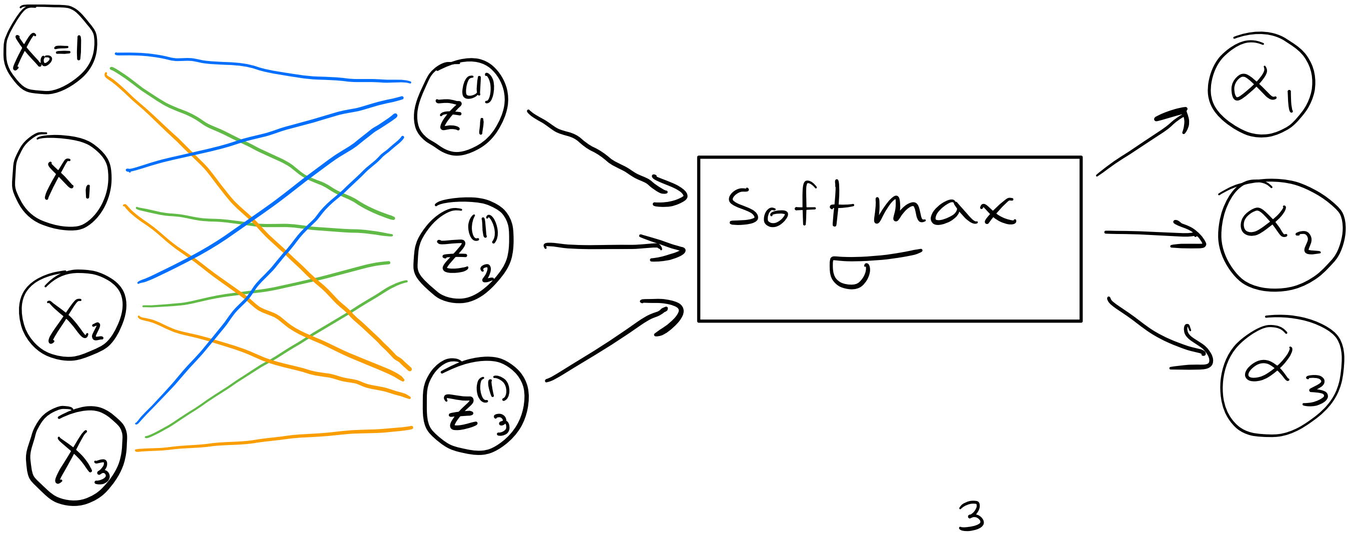
$$a_2^{(1)} = h^{(1)}\left(\vec{x}^T\vec{w}_2^{(1)}\right)$$

$$a_3^{(1)} = h^{(1)}\left(\vec{x}^T\vec{w}_3^{(1)}\right)$$

if $h^{(1)}(z) = z$

$$f(\vec{x}) = \vec{a}^{(1)} = \left(\vec{x}^T\vec{w}_1^{(1)}, \vec{x}^T\vec{w}_2^{(1)}, \vec{x}\vec{w}_3^{(1)}\right)^T$$

$\sigma(f(\vec{x}))$ post-processing step



$$\sum_{q=1}^{3} \alpha_q = 1 \,, \quad \alpha_q \in [0,1]$$

$$\sigma\left(z_1^{(1)}, z_2^{(1)}, z_3^{(1)}\right)$$

$$= \left(\sigma_1\left(z_1^{(1)}, z_2^{(1)}, z_3^{(1)}\right), \sigma_2\left(z_1^{(1)}, z_2^{(1)}, z_3^{(1)}\right) \quad \sigma_3\left(z_1^{(1)}, z_2^{(1)}, z_3^{(1)}\right) \right)^T$$

$$= \left(\frac{\exp\left(z_1^{(1)}\right)}{\sum_{q=1}^{3} \exp\left(z_q^{(1)}\right)} , \frac{\exp\left(z_2^{(1)}\right)}{\sum_{q=1}^{3} \exp\left(z_q^{(1)}\right)} , \frac{\exp\left(z_3^{(1)}\right)}{\sum_{q=1}^{3} \exp\left(z_q^{(1)}\right)} \right)^T$$