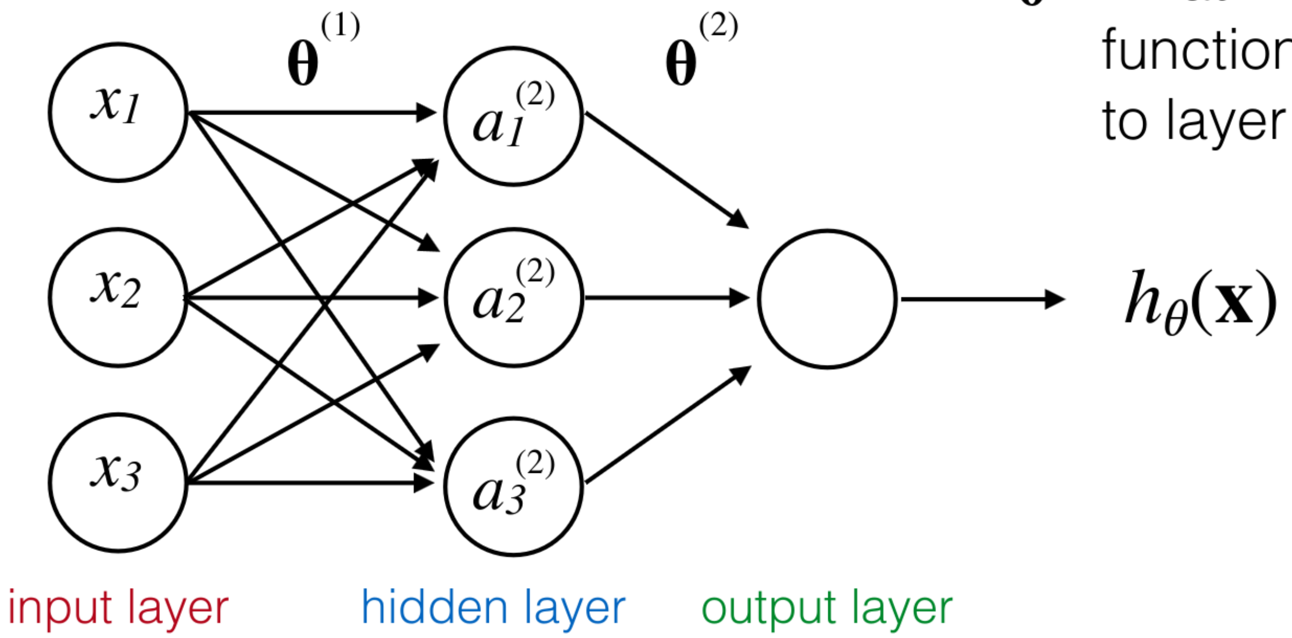
Neural Networks: model representation

- A (artificial) neural network is just a group of this different neurons strong together



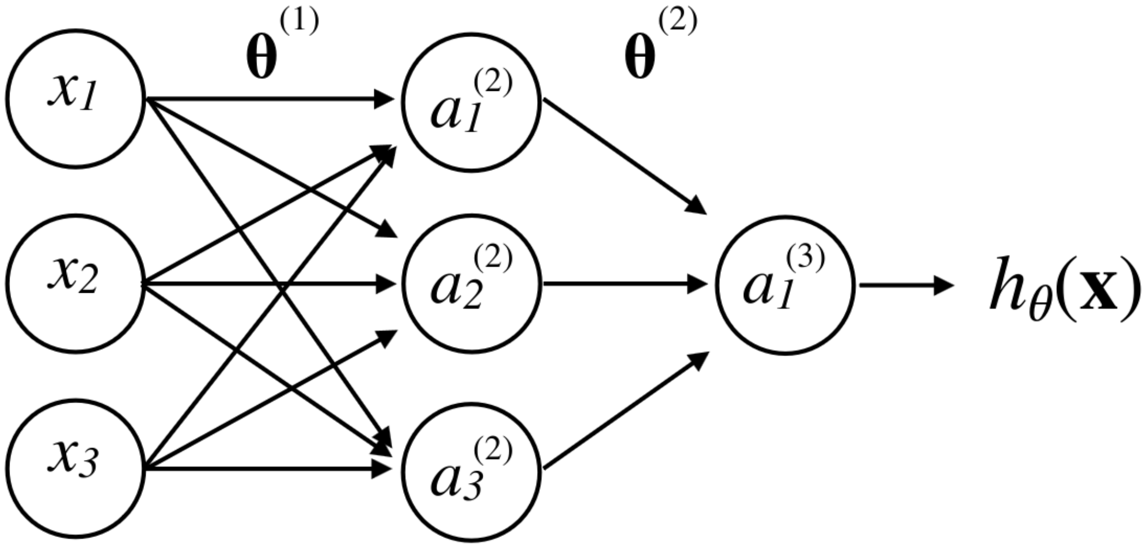
$a_i^{(j)}$: activation of unit i in layer j

$\boldsymbol{\theta}^{(j)}$: matrix of weights controlling function mapping from layer j to layer $j+1$

Neural Networks: feed-forward computation

- Input units are set by some exterior function (it can be generated by sensors) which causes their output links to be activated at the specified level
- Working forward through the network, these outputs are going to be the input for the next layer
  - Each output is just the weighted sum of the activation on the links feeding into a node
  - The activation function transforms this linear combination: typically this is a non linear function (such a sigmoid)
  - This function corresponds to the “threshold” of that node

Neural Networks: feed-forward computation



$\boldsymbol{\theta}^{(1)} \in \mathbb{R}^{3\times 4}$ $\boldsymbol{\theta}^{(2)} \in \mathbb{R}^{1\times 4}$

$a_i^{(j)}$: activation of unit i in layer j

$\boldsymbol{\theta}^{(j)}$: matrix of weights controlling function mapping from layer j to layer $j+1$

If network has u_j units in layer j and $u_\mathrm{j+1}$ units in layer $j+1$, then $\boldsymbol{\theta}^{(\mathrm{j})}$ will be of dimension $u_\mathrm{j+1} \times (u_\mathrm{j} + 1)$

$$a_1^{(2)} = f\,(\theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3)$$

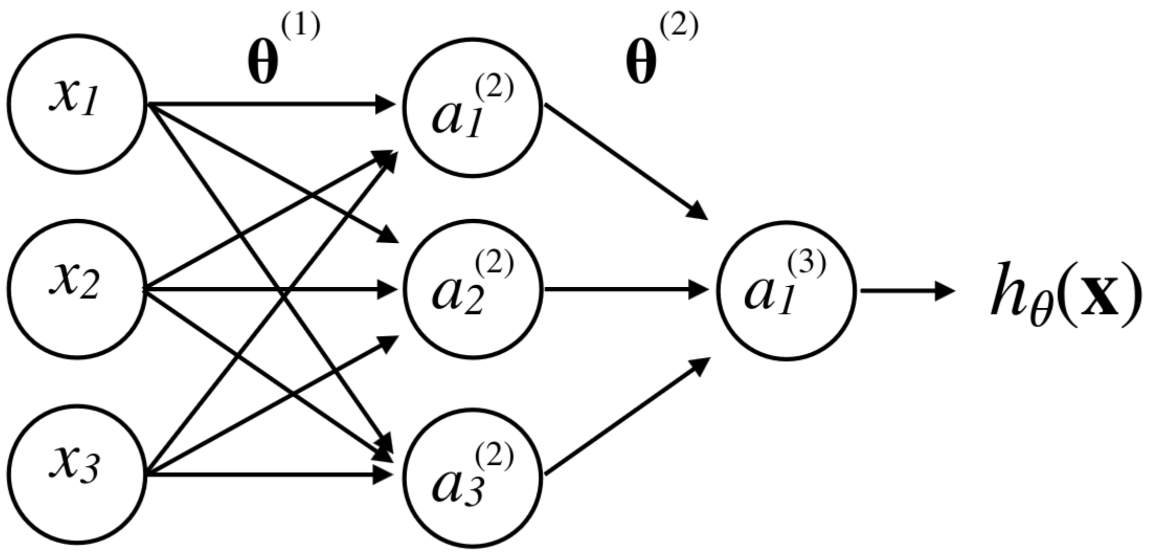
$$a_2^{(2)} = f\,(\theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3)$$

$$a_3^{(2)} = f\,(\theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3)$$

$$h_\theta(\mathbf{x}) = a_1^{(3)} = f\,(\theta_{10}^{(2)} a_0^{(2)} + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)})$$

Neural Networks: feed-forward computation

(vectorization)



$a_i^{(j)}$: activation of unit i in layer j

$\boldsymbol{\theta}^{(j)}$: matrix of weights controlling function mapping from layer j to layer $j+1$

$z_i^{(j)}$: linear combination of the input nodes

$$a_1^{(2)} = f\underbrace{\boxed{(\theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3)}}_{z_1^{(2)}} = f(z_1^{(2)})$$

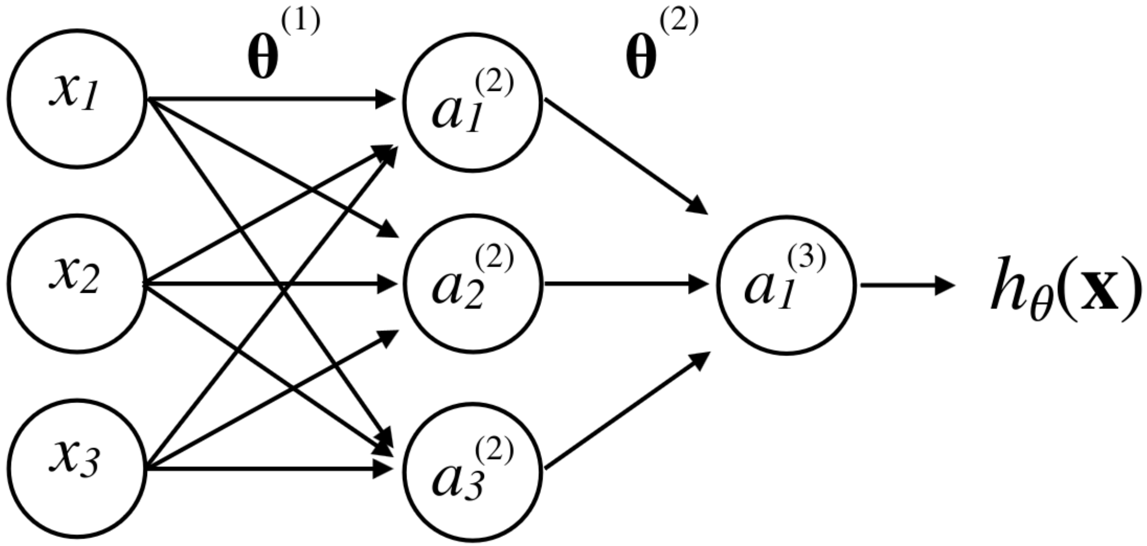
$$a_2^{(2)} = f(\theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3) = f(z_2^{(2)})$$

$$a_3^{(2)} = f(\theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3) = f(z_3^{(2)})$$

$$h_\theta(\mathbf{x}) = a_1^{(3)} = f(\theta_{10}^{(2)} a_0^{(2)} + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)}) = f(z_1^{(3)})$$

Neural Networks: feed-forward computation

(vectorization)



$$\boldsymbol{a}^{(1)} = \mathbf{x}$$

$$\mathbf{x} = \begin{bmatrix} x_0 \\ \vdots \\ x_3 \end{bmatrix} \qquad \mathbf{z}^{(2)} = \begin{bmatrix} z_1^{(2)} \\ \vdots \\ z_3^{(2)} \end{bmatrix}$$

$$a_1^{(2)} = f(\theta_{10}^{(1)} x_0 + \theta_{11}^{(1)} x_1 + \theta_{12}^{(1)} x_2 + \theta_{13}^{(1)} x_3) = f(z_1^{(2)})$$

$$a_2^{(2)} = f(\theta_{20}^{(1)} x_0 + \theta_{21}^{(1)} x_1 + \theta_{22}^{(1)} x_2 + \theta_{23}^{(1)} x_3) = f(z_2^{(2)})$$

$$a_3^{(2)} = f(\theta_{30}^{(1)} x_0 + \theta_{31}^{(1)} x_1 + \theta_{32}^{(1)} x_2 + \theta_{33}^{(1)} x_3) = f(z_3^{(2)})$$

$$\mathbf{z}^{(2)} = \boldsymbol{\theta}^{(1)} \boldsymbol{a}^{(1)}$$

$$\boldsymbol{a}^{(2)} = f(\mathbf{z}^{(2)})$$

Add bias: $a_0^{(2)} = 1$

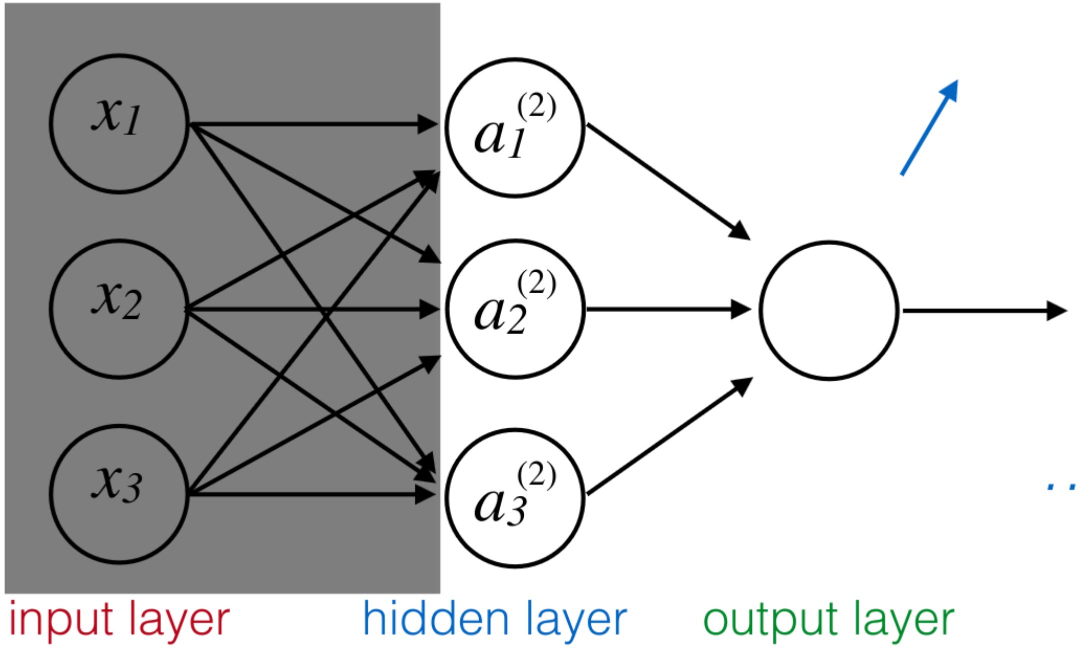
$$\mathbf{z}^{(3)} = \boldsymbol{\theta}^{(2)} \boldsymbol{a}^{(2)}$$

$$h_\theta(\mathbf{x}) = a_1^{(3)} = f(\theta_{10}^{(2)} a_0^{(2)} + \theta_{11}^{(2)} a_1^{(2)} + \theta_{12}^{(2)} a_2^{(2)} + \theta_{13}^{(2)} a_3^{(2)}) = f(z_1^{(3)}) = \boldsymbol{a}^{(3)} = f(\mathbf{z}^{(3)})$$

"forward propagation"

Neural Networks: feed-forward computation

- This forward propagation view also help us to understand what neural networks might be doing



This is basically logistic regression…

$$f(z) = \frac{1}{1 + e^{-z}}$$

$$h_\theta(\mathbf{x}) = f\left(\boldsymbol{\theta}^{(2)}\boldsymbol{a}^{(2)}\right)$$

… but now features ($\boldsymbol{a}^{(2)}$) are learned by the network