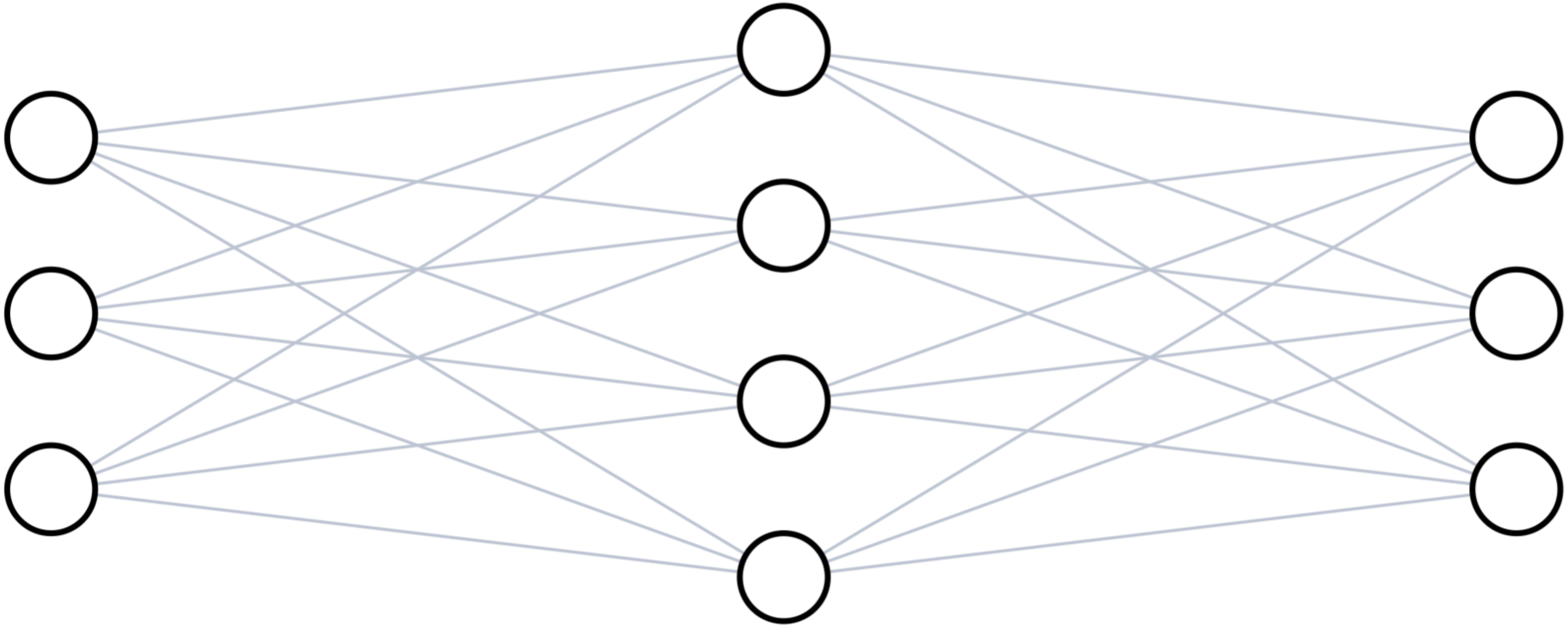
Neural Networks

Back Propagation Equations for Binary Classification

Rahul Singh
rsingh@arrsingh.com

Neural Networks



n_0 Neurons

Input Layer (L_0)

n_1 Neurons

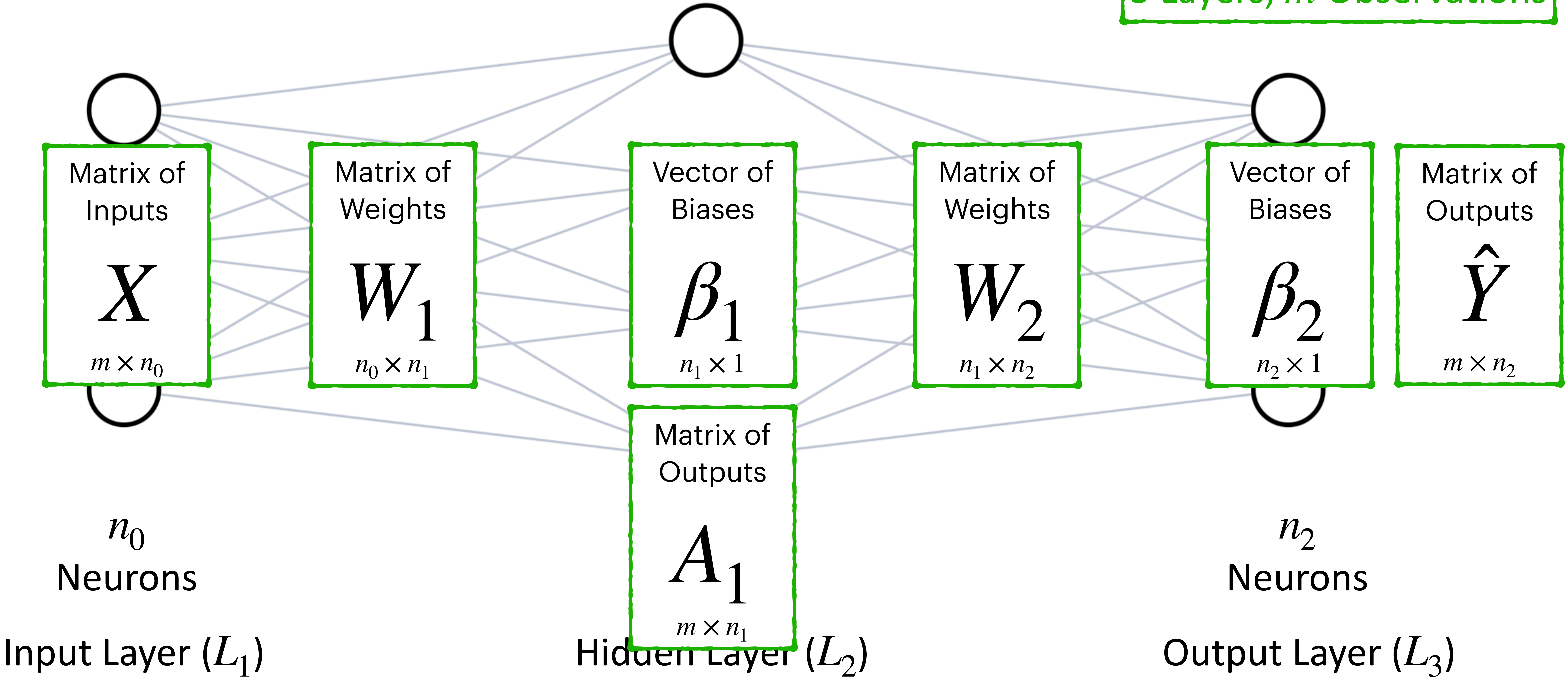
Hidden Layer (L_1)

n_2 Neurons

Output Layer (L_2)

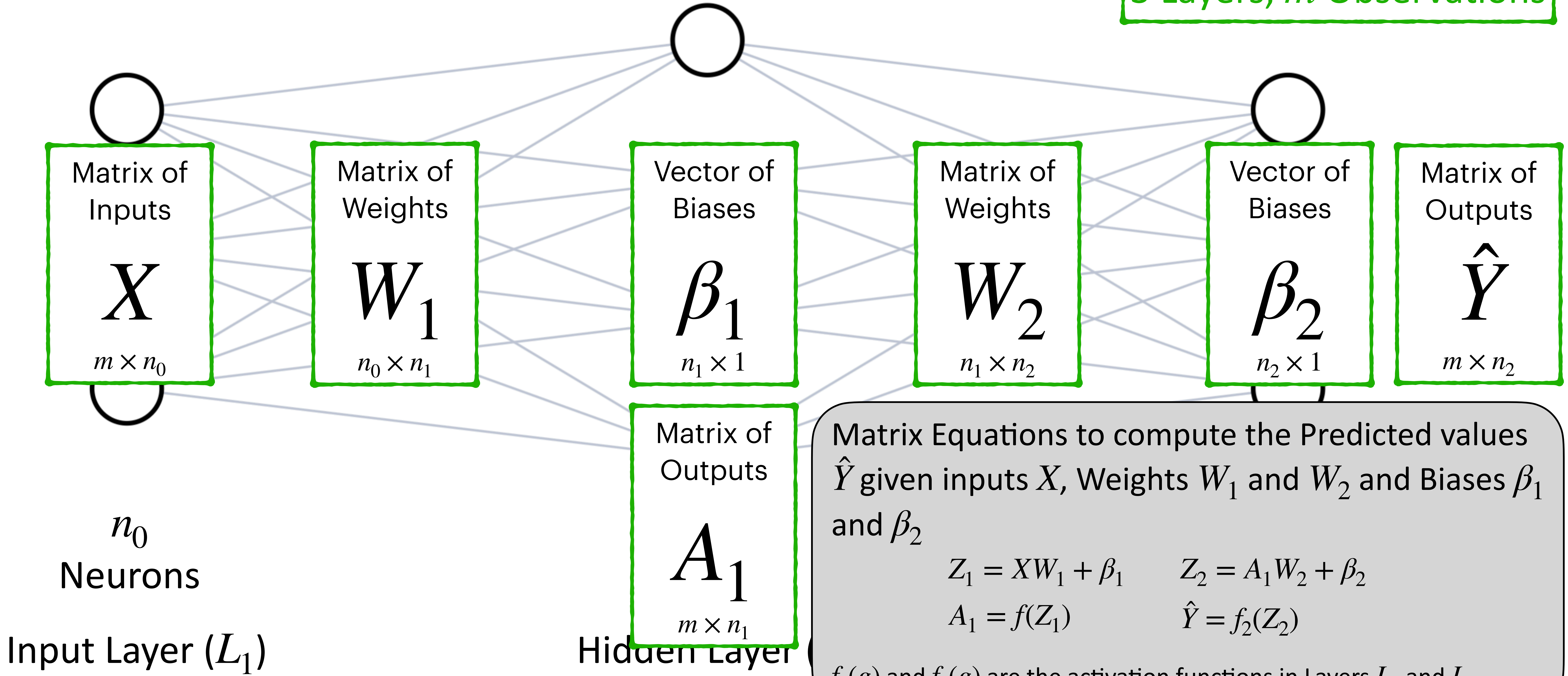
Neural Networks

3 Layers, m Observations



Neural Networks

3 Layers, m Observations



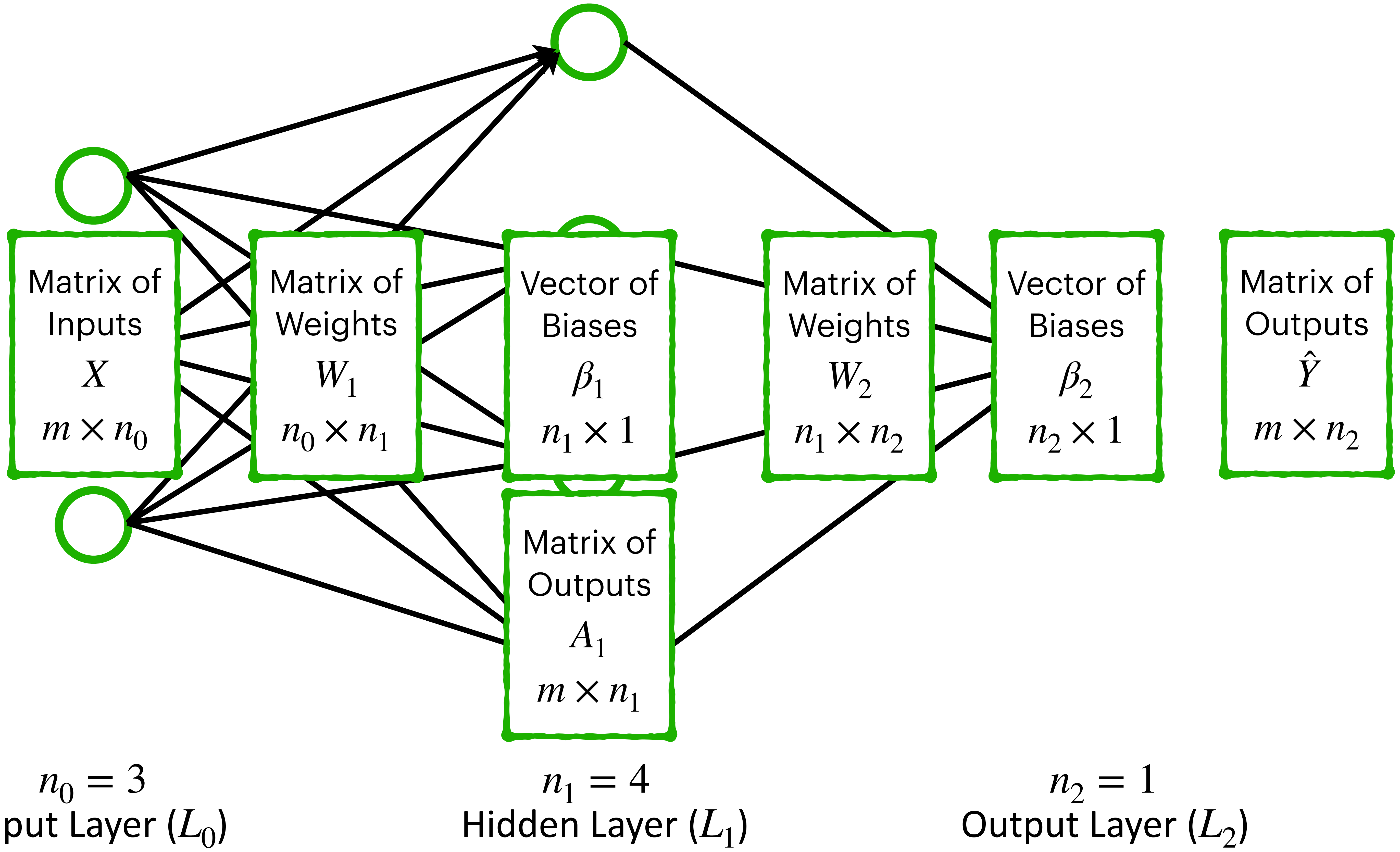
Matrix Equations to compute the Predicted values $\hat{Y}$ given inputs X, Weights W_1 and W_2 and Biases β_1 and β_2

$$Z_1 = XW_1 + \beta_1 \qquad Z_2 = A_1 W_2 + \beta_2$$

$$A_1 = f(Z_1) \qquad \hat{Y} = f_2(Z_2)$$

$f_1(g)$ and $f_2(g)$ are the activation functions in Layers L_1 and L_2

Let's derive the equations for Back Propagation for a Feedforward Neural Network for Binary Classification



$Z_1 = XW_1 + \beta_1$ $\quad Z_2 = A_1W_2 + \beta_2$

$A_1 = f_1(Z_1)$ $\quad \hat{Y} = f_2(Z_2)$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y\,log_e\hat{y} + (1-y)\,log_e(1-\hat{y})]$$

Loss function:

$$L = -\,[y\,log_e\hat{y} + (1-y)\,log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Let's derive the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

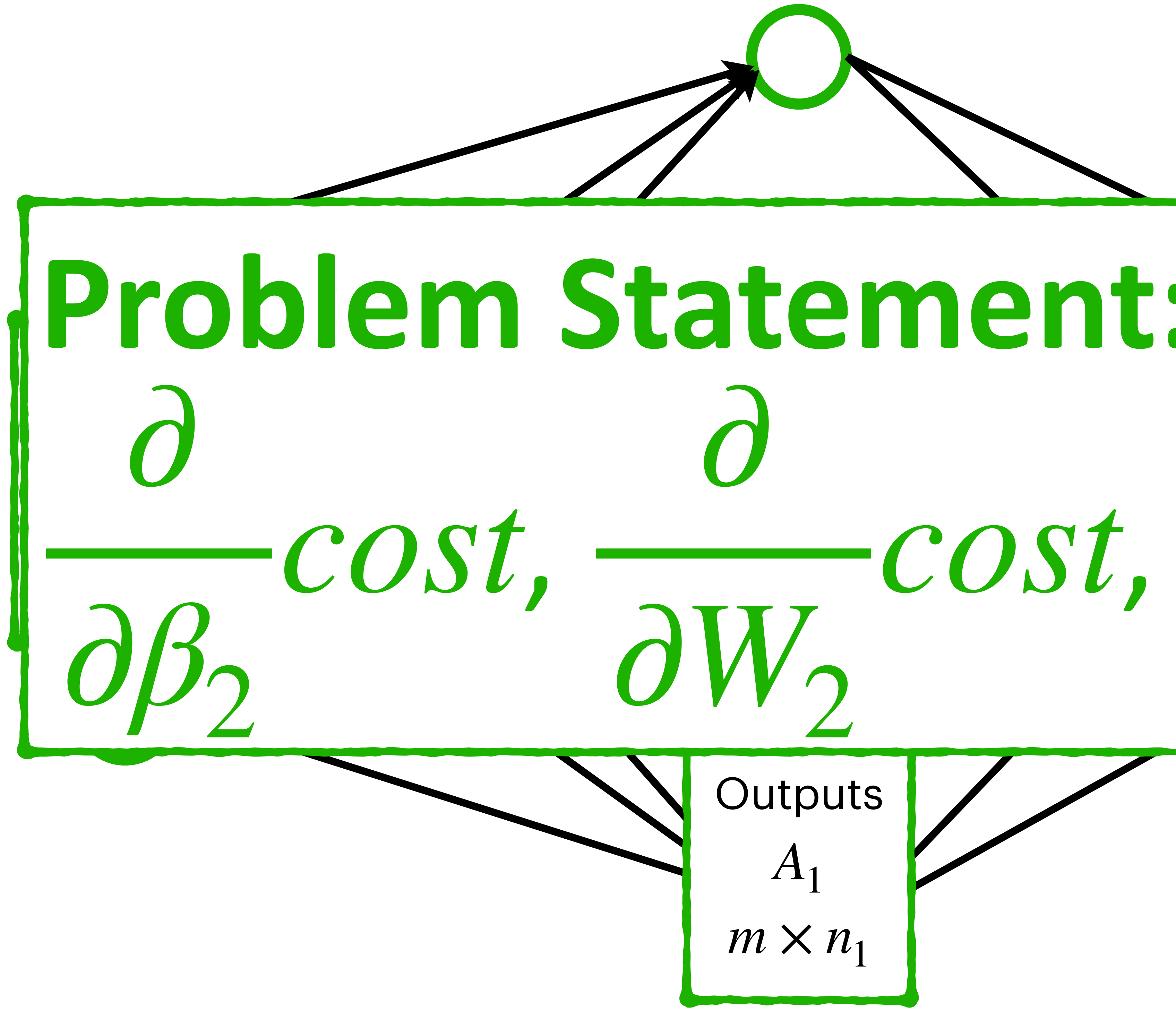
Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y\,log_e\hat{y} + (1-y)\,log_e(1-\hat{y})]$$

Loss function:

Problem Statement: Calculate the four partial derivatives:

$$\frac{\partial}{\partial\beta_2}cost,\ \frac{\partial}{\partial W_2}cost,\ \frac{\partial}{\partial\beta_1}cost,\ \frac{\partial}{\partial W_1}cost$$



$n_0 = 3$ Input Layer (L_0)

$n_1 = 4$ Hidden Layer (L_1)

$n_2 = 1$ Output Layer (L_2)

$$Z_1 = XW_1 + \beta_1 \quad Z_2 = A_1W_2 + \beta_2$$
$$A_1 = f_1(Z_1) \quad \hat{Y} = f_2(Z_2)$$

Neural Networks

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

$$\frac{\partial}{\partial W_2}L = \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial W_2}z_2$$

Chain Rule:
L depends on $\hat{y}, z_2$ in that order

First let's calculate $\frac{\partial}{\partial \hat{y}}L$

$$\Rightarrow \frac{\partial}{\partial \hat{y}}L = \frac{\partial}{\partial \hat{y}}[-y\,log_e\hat{y} - (1-y)\,log_e(1-\hat{y})]$$

$$\frac{d}{dx}log_e\,x = \frac{1}{x}$$

$$\Rightarrow \frac{\partial}{\partial \hat{y}}L = -\frac{y}{\hat{y}} - \frac{(1-y)}{(1-\hat{y})}(-1)$$

$$\Rightarrow \frac{\partial}{\partial \hat{y}}L = \frac{-y(1-\hat{y}) + \hat{y}(1-y)}{\hat{y}(1-\hat{y})}$$

$+y\hat{y} - y\hat{y}$ cancels out

$$\Rightarrow \frac{\partial}{\partial \hat{y}}L = \frac{-y + y\hat{y} + \hat{y} - y\hat{y}}{\hat{y}(1-\hat{y})}$$

$$\Rightarrow \frac{\partial}{\partial \hat{y}}L = \frac{-y + \hat{y}}{\hat{y}(1-\hat{y})}$$

$$\boxed{\Rightarrow \frac{\partial}{\partial \hat{y}}L = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})}}$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i\,log_e\hat{y}_i + (1-y_i)\,log_e(1-\hat{y}_i)]$$

Loss function:

$$L = -\,[y\,log_e\hat{y} + (1-y)\,log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Next lets calculate $\frac{\partial}{\partial z_2}\hat{y}$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = \frac{\partial}{\partial z_2}\left(\frac{1}{1+e^{-z_2}}\right)$$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = \frac{\partial}{\partial z_2}(1+e^{-z_2})^{-1}$$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = (-1)(1+e^{-z_2})^{-2}\frac{\partial}{\partial z_2}(1+e^{-z_2})$$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = (-1)(1+e^{-z_2})^{-2}(-1)e^{-z_2}$$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = \frac{e^{-z_2}}{(1+e^{-z_2})^2}$$

$$\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = \frac{\hat{y}^2(1-\hat{y})}{\hat{y}}$$

$$\boxed{\Rightarrow \frac{\partial}{\partial z_2}\hat{y} = \hat{y}(1-\hat{y})}$$

Power Rule:

$$\frac{d}{dx}x^n = n \cdot x^{(n-1)}$$

$$\frac{d}{dx}e^{-x} = -\,e^{-x}$$

$$\hat{y} = \frac{1}{1+e^{-z2}}$$

$$\Rightarrow \hat{y}^2 = \frac{1}{(1+e^{-z_2})^2}$$

$$\hat{y} = \frac{1}{1+e^{-z2}}$$

$$\Rightarrow e^{-z_2} = \frac{(1-\hat{y})}{\hat{y}}$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i\, log_e \hat{y}_i + (1-y_i)\, log_e(1-\hat{y}_i)]$$

Loss function:

$$L = -\,[y\, log_e \hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Neural Networks

Next lets calculate $\frac{\partial}{\partial W_2} z_2$

$$\Rightarrow \frac{\partial}{\partial W_2} z_2 = \frac{\partial}{\partial W_2}(a_1 W_2 + \beta_2)$$

$$\boxed{\Rightarrow \frac{\partial}{\partial W_2} z_2 = a_1}$$

Putting all three terms together: $\frac{\partial}{\partial W_2} L = \frac{\partial}{\partial \hat{y}} L \frac{\partial}{\partial z_2} \hat{y} \frac{\partial}{\partial W_2} z_2$

$$\Rightarrow \frac{\partial}{\partial W_2} L = \frac{\hat{y} - y}{\hat{y}(1-\hat{y})} \hat{y}(1-\hat{y})\, a_1$$

$$\Rightarrow \frac{\partial}{\partial W_2} L = (\hat{y} - y)\, a_1$$

Derivative of the cost over m observations:

$$\boxed{\Rightarrow \frac{\partial}{\partial W_2} cost = \frac{1}{m} A_1^T (\hat{Y} - Y)}$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n} \sum_{i=1}^{n} [y_i \, log_e \hat{y}_i + (1 - y_i)\, log_e (1 - \hat{y}_i)]$$

Loss function:

$$L = -\,[y \, log_e \hat{y} + (1 - y)\, log_e (1 - \hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1 + e^{-x}}$$

A_1 is an $m \times n_1$ matrix
$\hat{Y}$ is a $m \times n_2$ matrix
Y is a $m \times n_2$ matrix

$\Rightarrow A_1^T$ is a $n_1 \times m$ matrix

$\Rightarrow \frac{1}{m} A_1^T (\hat{Y} - Y)$ is a $n_1 \times n_2$ matrix

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

$$\frac{\partial}{\partial \beta_2}L = \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial \beta_2}z_2$$

Chain Rule:
L depends on $\hat{y}, z_2$ in that order

$$\frac{\partial}{\partial \hat{y}}L = \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})} \qquad \frac{\partial}{\partial z_2}\hat{y} = \hat{y}(1 - \hat{y})$$

Let's calculate $\frac{\partial}{\partial \beta_2}z_2$

$$\Rightarrow \frac{\partial}{\partial \beta_2}z_2 = \frac{\partial}{\partial \beta_2}(a_1 W_2 + \beta_2) = 1$$

Putting all three terms together: $\frac{\partial}{\partial \beta_2}L = \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial \beta_2}z_2$

$$\Rightarrow \frac{\partial}{\partial \beta_2}L = \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})}\hat{y}(1 - \hat{y})$$

$$\Rightarrow \frac{\partial}{\partial \beta_2}L = (\hat{y} - y)$$

Derivative of the cost over m observations:

$$\boxed{\Rightarrow \frac{\partial}{\partial \beta_2}cost = \frac{1}{m}\sum_{i=1}^{m}(\hat{Y} - Y)}$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i\, log_e \hat{y}_i + (1 - y_i)\, log_e(1 - \hat{y}_i)]$$

Loss function:

$$L = -\,[y\, log_e \hat{y} + (1 - y)\, log_e(1 - \hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1 + e^{-x}}$$

$\hat{Y}$ is a $m \times n_2$ matrix
Y is a $m \times n_2$ matrix

$\Rightarrow \frac{1}{m}\sum_{i=1}^{m}(\hat{Y} - Y)$ is a $1 \times n_2$ vector

Summation from $1..m$ collapses the m dimension

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Neural Networks

$$\frac{\partial}{\partial W_1}L = \frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y}\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1\frac{\partial}{\partial W_1}z_1$$

Chain Rule:
L depends on $\hat{y}, z_2, a_1, z_1$ in that order

$$\frac{\partial}{\partial \hat{y}}L = \frac{\hat{y}-y}{\hat{y}(1-\hat{y})} \qquad \frac{\partial}{\partial z_2}\hat{y} = \hat{y}(1-\hat{y})$$

First let's calculate $\frac{\partial}{\partial a_1}z_2$

$$\Rightarrow \frac{\partial}{\partial a_1}z_2 = \frac{\partial}{\partial a_1}(a_1W_2+\beta_2) = W_2$$

Next let's calculate $\frac{\partial}{\partial z_1}a_1$

$$\Rightarrow \frac{\partial}{\partial z_1}a_1 = \frac{\partial}{\partial z_1}f_1(z_1)$$

Next let's calculate $\frac{\partial}{\partial W_1}z_1$

$$\Rightarrow \frac{\partial}{\partial W_1}z_1 = \frac{\partial}{\partial W_1}(xW_1+\beta_1) = x$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i\, log_e\hat{y}_i + (1-y_i)\, log_e(1-\hat{y}_i)]$$

Loss function:

$$L = -\,[y\, log_e\hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

$$\frac{\partial}{\partial W_1}L = \frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y}\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1\frac{\partial}{\partial W_1}z_1$$

Chain Rule:
L depends on $\hat{y}, z_2, a_1, z_1$ in that order

Putting all five terms together:

$$\frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y} = \frac{\partial}{\partial z_2}L$$

$$\Rightarrow \frac{\partial}{\partial W_1}L = \frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y}\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1\frac{\partial}{\partial W_1}z_1$$

$$\Rightarrow \frac{\partial}{\partial W_1}L = \frac{\partial}{\partial z_2}L\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1\frac{\partial}{\partial W_1}z_1$$

$$\Rightarrow \frac{\partial}{\partial W_1}L = \left[\frac{\partial}{\partial z_1}L\right]x$$

$$\Rightarrow \frac{\partial}{\partial z_1}L = \frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y}\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1$$
$$= \frac{\partial}{\partial z_2}L\frac{\partial}{\partial a_1}z_2\frac{\partial}{\partial z_1}a_1$$
$$= \left(\left[\frac{\partial}{\partial z_2}L\right]W_2^T\right) \odot \left[\frac{\partial}{\partial z_1}f_1(z_1)\right]$$

Derivative of the cost over m observations:

$$\Rightarrow \frac{\partial}{\partial W_1}cost = \frac{1}{m}X^T\left[\frac{\partial}{\partial Z_1}cost\right]$$

$$\frac{\partial}{\partial Z_1}cost = \left(\left[\frac{\partial}{\partial Z_2}cost\right]W_2^T\right) \odot \left[\frac{\partial}{\partial Z_1}f_1(Z_1)\right]$$

$$\Rightarrow \frac{\partial}{\partial Z_1}cost = \left((\hat{Y} - Y)W_2^T\right) \odot \left[\frac{\partial}{\partial Z_1}f_1(Z_1)\right]$$

X^T is a $n_0 \times m$ matrix

$\Rightarrow \frac{1}{m}X^T\left[\frac{\partial}{\partial Z_1}cost\right]$ is a $n_0 \times n_1$ matrix

$\frac{\partial}{\partial Z_1}cost$ is an $m \times n_1$ matrix

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i\,log_e\hat{y}_i + (1 - y_i)\,log_e(1 - \hat{y}_i)]$$

Loss function:

$$L = -\,[y\,log_e\hat{y} + (1 - y)\,log_e(1 - \hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1 + e^{-x}}$$

$$\Rightarrow \frac{\partial}{\partial z_2}L = \frac{\partial}{\partial \hat{y}}L\frac{\partial}{\partial z_2}\hat{y}$$
$$= \frac{\hat{y} - y}{\hat{y}(1 - \hat{y})}\hat{y}(1 - \hat{y})$$
$$= \hat{y} - y$$

Vectorized over m observations

$$\Rightarrow \frac{\partial}{\partial Z_2}cost = \hat{Y} - Y$$

$\hat{Y} - Y$ is an $m \times n_2$ matrix

Deriving the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Neural Networks

$$\frac{\partial}{\partial \beta_1}L = \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial a_1}z_2 \frac{\partial}{\partial z_1}a_1 \frac{\partial}{\partial \beta_1}z_1$$

Chain Rule:
L depends on $\hat{y}, z_2, a_1, z_1$ in that order

Putting all five terms together:

$$\Rightarrow \frac{\partial}{\partial \beta_1}L = \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial a_1}z_2 \frac{\partial}{\partial z_1}a_1 \frac{\partial}{\partial \beta_1}z_1$$

$$\frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} = \frac{\partial}{\partial z_2}L$$

$$\Rightarrow \frac{\partial}{\partial \beta_1}L = \frac{\partial}{\partial z_2}L \frac{\partial}{\partial a_1}z_2 \frac{\partial}{\partial z_1}a_1 \frac{\partial}{\partial \beta_1}z_1$$

$$\Rightarrow \frac{\partial}{\partial \beta_1}L = \left[\frac{\partial}{\partial z_1}L\right] \frac{\partial}{\partial \beta_1}z_1$$

$$\begin{aligned}\Rightarrow \frac{\partial}{\partial z_1}L &= \frac{\partial}{\partial \hat{y}}L \frac{\partial}{\partial z_2}\hat{y} \frac{\partial}{\partial a_1}z_2 \frac{\partial}{\partial z_1}a_1 \\ &= \frac{\partial}{\partial z_2}L \frac{\partial}{\partial a_1}z_2 \frac{\partial}{\partial z_1}a_1 \\ &= \left(\left[\frac{\partial}{\partial z_2}L\right] W_2^T\right) \odot \left[\frac{\partial}{\partial z_1}f_1(z_1)\right]\end{aligned}$$

$$\Rightarrow \frac{\partial}{\partial \beta_1}L = \left[\frac{\partial}{\partial z_1}L\right] \frac{\partial}{\partial \beta_1}(xW_1 + \beta_1)$$

$$\Rightarrow \frac{\partial}{\partial \beta_1}L = \frac{\partial}{\partial z_1}L$$

Derivative of the cost over m observations:

$$\boxed{\Rightarrow \frac{\partial}{\partial \beta_1}cost = \frac{1}{m}\sum_{1=1}^{m}\frac{\partial}{\partial z_1}L}$$

$\frac{\partial}{\partial z_1}L$ is a $n_1 \times 1$ vector

$\frac{1}{m}\sum_{1=1}^{m}\frac{\partial}{\partial z_1}L$ is an $n_1 \times 1$ vector

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i \, log_e\hat{y}_i + (1-y_i)\, log_e(1-\hat{y}_i)]$$

Loss function:

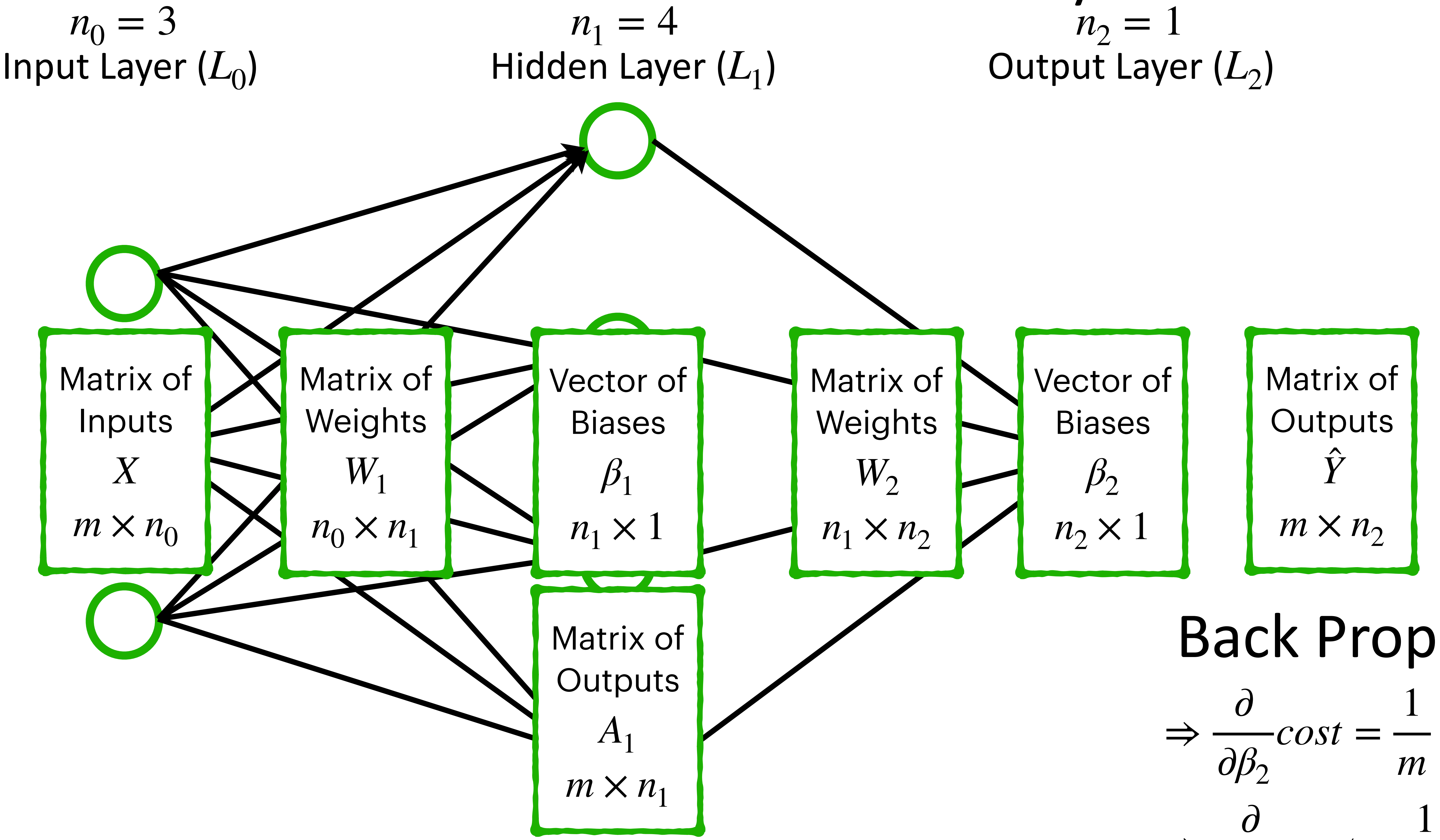
$$L = -\,[y \, log_e\hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Neural Networks

Let's derive the equations for Back Propagation for a Feedforward Neural Network for Binary Classification



$$Z_1 = XW_1 + \beta_1 \qquad Z_2 = A_1W_2 + \beta_2$$
$$A_1 = f_1(Z_1) \qquad \hat{Y} = f_2(Z_2)$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n}\sum_{i=1}^{n}[y\, log_e\hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Loss function:

$$L = -\,[y\, log_e\hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Back Propagation (Summary):

$$\Rightarrow \frac{\partial}{\partial \beta_2}cost = \frac{1}{m}\sum_{i=1}^{m}(\hat{Y} - Y) \qquad \beta_2 = \beta_2 - learning\_rate \times \frac{\partial}{\partial \beta_2}cost$$

$$\Rightarrow \frac{\partial}{\partial W_2}cost = \frac{1}{m}A_1^T(\hat{Y} - Y) \qquad W_2 = W_2 - learning\_rate \times \frac{\partial}{\partial W_2}cost$$

$$\Rightarrow \frac{\partial}{\partial \beta_1}cost = \frac{1}{m}\sum_{1=1}^{m}\frac{\partial}{\partial z_1}L \qquad \beta_1 = \beta_1 - learning\_rate \times \frac{\partial}{\partial \beta_1}cost$$

$$\Rightarrow \frac{\partial}{\partial W_1}cost = \frac{1}{m}X^T\left[\frac{\partial}{\partial Z_1}cost\right] \qquad W_1 = W_1 - learning\_rate \times \frac{\partial}{\partial W_1}cost$$

Let's derive the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Back Propagation (Summary for a 3 layer network):

$$\Rightarrow \frac{\partial}{\partial Z_2} cost = dZ_2 = \hat{Y} - Y$$

$$\Rightarrow \frac{\partial}{\partial \beta_2} cost = dB_2 = \frac{1}{m} \sum_{i=1}^{m} dZ_2$$

$$\Rightarrow \frac{\partial}{\partial W_2} cost = dW_2 = \frac{1}{m} A_1^T dZ_2$$

$$\Rightarrow \frac{\partial}{\partial Z_1} cost = dZ_1 = \left(dZ_2 W_2^T\right) \odot \left[\frac{\partial}{\partial Z_1} f_1(Z_1)\right]$$

$$\Rightarrow \frac{\partial}{\partial W_1} cost = dW_1 = \frac{1}{m} X^T dZ_1$$

$$\Rightarrow \frac{\partial}{\partial \beta_1} cost = dB_1 = \frac{1}{m} \sum_{1=1}^{m} dZ_1$$

$$\beta_2 = \beta_2 - learning\_rate \times \frac{\partial}{\partial \beta_2} cost$$

$$W_2 = W_2 - learning\_rate \times \frac{\partial}{\partial W_2} cost$$

$$\beta_1 = \beta_1 - learning\_rate \times \frac{\partial}{\partial \beta_1} cost$$

$$W_1 = W_1 - learning\_rate \times \frac{\partial}{\partial W_1} cost$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n} \sum_{i=1}^{n} [y\, log_e \hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Loss function:

$$L = -\,[y\, log_e \hat{y} + (1-y)\, log_e(1-\hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1+e^{-x}}$$

Let's derive the equations for Back Propagation for a Feedforward Neural Network for Binary Classification

Back Propagation (Summary for a 4 layer network):

$$\Rightarrow \frac{\partial}{\partial Z_3} cost = dZ_3 = \hat{Y} - Y$$

$$\Rightarrow \frac{\partial}{\partial \beta_3} cost = dB_3 = \frac{1}{m} \sum_{i=1}^{m} dZ_3$$

$$\Rightarrow \frac{\partial}{\partial W_3} cost = dW_3 = \frac{1}{m} A_2^T dZ_3$$

$$\Rightarrow \frac{\partial}{\partial Z_2} cost = dZ_2 = \left(dZ_3 W_3^T\right) \odot \left[\frac{\partial}{\partial Z_2} f_2(Z_2)\right]$$

$$\Rightarrow \frac{\partial}{\partial W_2} cost = dW_2 = \frac{1}{m} A_1^T dZ_2$$

$$\Rightarrow \frac{\partial}{\partial \beta_2} cost = dB_2 = \frac{1}{m} \sum_{1=1}^{m} dZ_2$$

$$\Rightarrow \frac{\partial}{\partial Z_1} cost = dZ_1 = \left(dZ_2 W_2^T\right) \odot \left[\frac{\partial}{\partial Z_1} f_1(Z_1)\right]$$

$$\Rightarrow \frac{\partial}{\partial W_1} cost = dW_1 = \frac{1}{m} X^T dZ_1$$

$$\Rightarrow \frac{\partial}{\partial \beta_1} cost = dB_1 = \frac{1}{m} \sum_{1=1}^{m} dZ_1$$

$$\beta_3 = \beta_3 - learning\_rate \times \frac{\partial}{\partial \beta_3} cost$$

$$W_3 = W_3 - learning\_rate \times \frac{\partial}{\partial W_3} cost$$

$$\beta_2 = \beta_2 - learning\_rate \times \frac{\partial}{\partial \beta_2} cost$$

$$W_2 = W_2 - learning\_rate \times \frac{\partial}{\partial W_2} cost$$

$$\beta_1 = \beta_1 - learning\_rate \times \frac{\partial}{\partial \beta_1} cost$$

$$W_1 = W_1 - learning\_rate \times \frac{\partial}{\partial W_1} cost$$

Cost function is the **Binary Cross Entropy:**

$$-\frac{1}{n} \sum_{i=1}^{n} [y \, log_e \hat{y} + (1 - y) \, log_e(1 - \hat{y})]$$

Loss function:

$$L = -[y \, log_e \hat{y} + (1 - y) \, log_e(1 - \hat{y})]$$

Sigmoid Activation function

$$\hat{y} = f_2(x) = \frac{1}{1 + e^{-x}}$$

Related Tutorials & Textbooks

[Neural Networks]

An introduction to Neural Networks starting from a foundation of linear regression, logistic classification and multi class classification models along with the matrix representation of a neural network generalized to *l* layers with *n* neurons

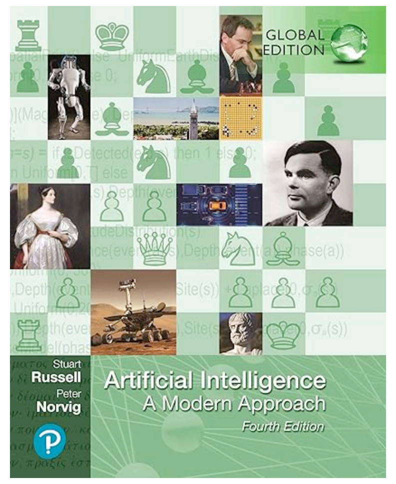
[Forward and Back Propagation in Neural Networks]

A deep dive into how Neural Networks are trained using Gradient Descent. Output predictions, are compared to observations to calculate loss and Backward propagation then computes gradients by working backward through the network

[Gradient Descent for Multiple Regression]

Gradient Descent algorithm for multiple regression and how it can be used to optimize k + 1 parameters for a Linear model in multiple dimensions.

Recommended Textbooks



Artificial Intelligence: A Modern Approach

by Peter Norvig, Stuart Russell

For a complete list of tutorials see:
https://arrsingh.com/ai-tutorials