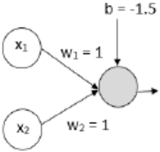
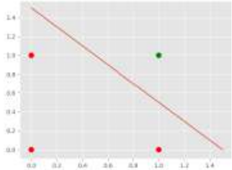
Neural Networks

Perceptron: Linearly separable data

Boolean AND

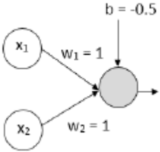
Input x_1	Input x_2	Output
0	0	0
0	1	0
1	0	0
1	1	1

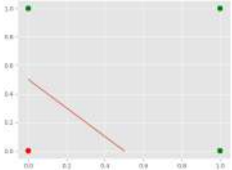




Boolean OR

Input x_1	Input x_2	Output
0	0	0
0	1	1
1	0	1
1	1	1

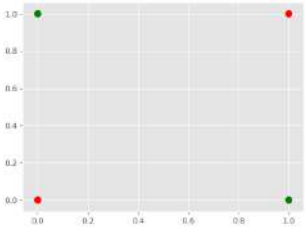




Perceptron: Linearly inseparable data

Boolean XOR

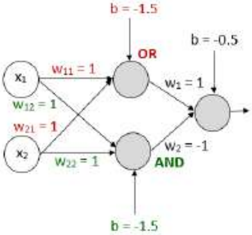
Input x_1	Input x_2	Output
0	0	0
0	1	1
1	0	1
1	1	0

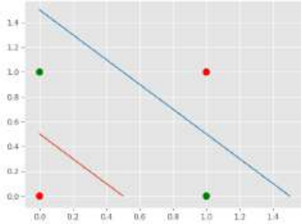


Perceptron: Linearly inseparable data

Boolean XOR

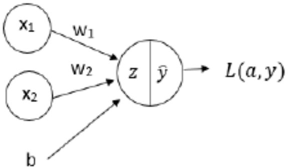
Input x_1	Input x_2	Output
0	0	0
0	1	1
1	0	1
1	1	0





Logistic Regression to Neural Network

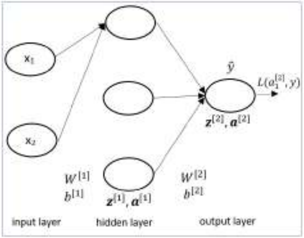
Logistic regression (sigmoid neuron)

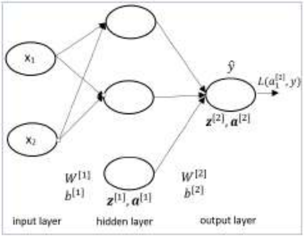


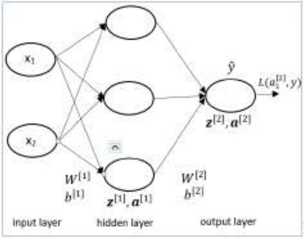
$$z = w_1 x_1 + w_2 x_2 + b$$

$$\hat{y} = a = \sigma(z)$$

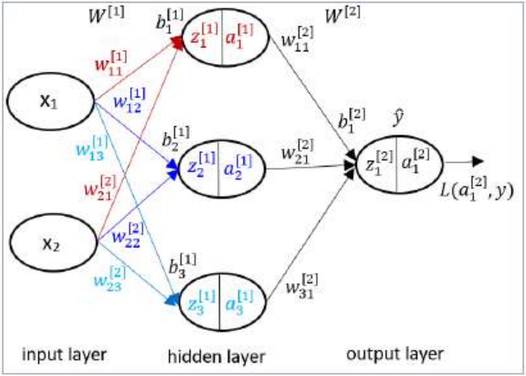
Stack multiple sigmoid neurons to create Neural network







Neural Network: Representations

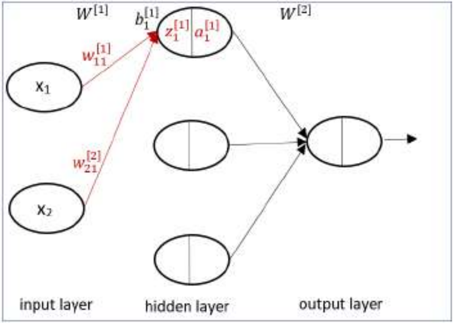


Neural Network: Forward Pass

Compute output at first hidden unit:

$$z_1^{[1]} = w_{11}^{[1]} x_1 + w_{21}^{[1]} x_2 + b_1^{[1]}$$

$$a_1^{[1]} = \sigma(z_1^{[1]})$$

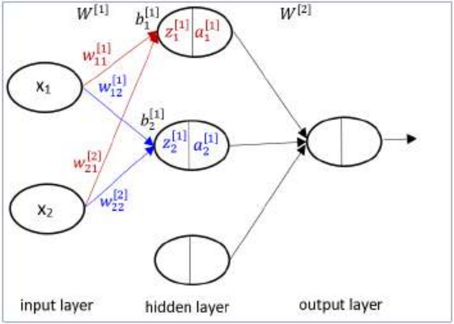


Neural Network: Forward Pass

Compute output at second hidden unit:

$$z_2^{[1]} = w_{12}^{[1]} x_1 + w_{22}^{[1]} x_2 + b_2^{[1]}$$

$$a_2^{[1]} = \sigma(z_2^{[1]})$$

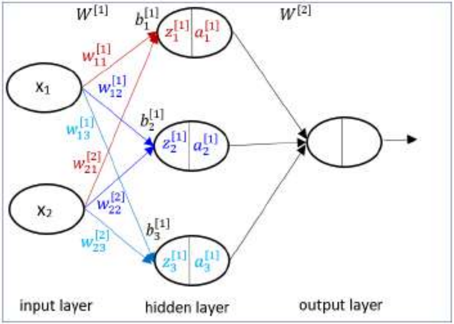


Neural Network: Forward Pass

Compute output at third hidden unit:

$$z_3^{[1]} = w_{13}^{[1]} x_1 + w_{23}^{[1]} x_2 + b_3^{[1]}$$

$$a_3^{[1]} = \sigma(z_3^{[1]})$$

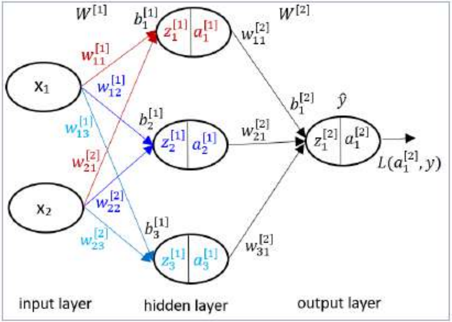


Neural Network: Forward Pass

Compute output at the output layer:

$$z_1^{[2]} = w_{11}^{[2]} a_1^{[1]} + w_{21}^{[2]} a_2^{[1]} + w_{31}^{[2]} a_3^{[1]} + b_1^{[2]}$$

$$a_1^{[2]} = \sigma(z_1^{[2]}) = \hat{y}$$



Neural Network: Forward Pass

Putting everything together:

$$z_1^{[1]} = w_{11}^{[1]} x_1 + w_{21}^{[1]} x_2 + b_1^{[1]}$$

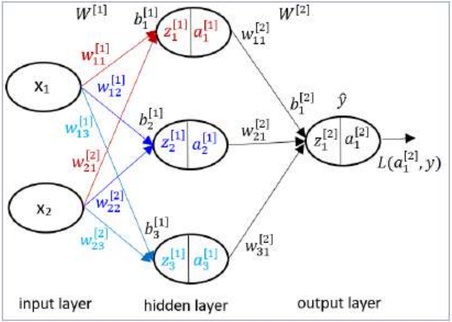
$$a_1^{[1]} = \sigma(z_1^{[1]})$$

$$z_2^{[1]} = w_{12}^{[1]} x_1 + w_{22}^{[1]} x_2 + b_2^{[1]}$$

$$a_2^{[1]} = \sigma(z_2^{[1]})$$

$$z_3^{[1]} = w_{13}^{[1]} x_1 + w_{23}^{[1]} x_2 + b_3^{[1]}$$

$$a_3^{[1]} = \sigma(z_3^{[1]})$$



$$z_1^{[2]} = w_{11}^{[2]} a_1^{[1]} + w_{21}^{[2]} a_2^{[1]} + w_{31}^{[2]} a_3^{[1]} + b_1^{[2]};\ a_1^{[2]} = \sigma(z_1^{[2]}) = \hat{y}$$

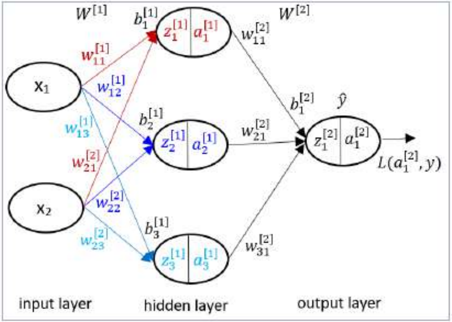
NN: outputs at different layers

$$\mathbf{z}^{[1]} = W^{[1]T}\mathbf{x} + b^{[1]}$$

$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]})$$

$$\mathbf{z}^{[2} = W^{[2]T}\mathbf{a}^{[1]} + b^{[2]}$$

$$\hat{y} = \mathbf{a}^{[2]} = \sigma(\mathbf{z}^{[2]})$$



Neural Network: Vector & Matrix Form

$$\mathbf{z}^{[1]} = W^{[1]T}\mathbf{x} + b^{[1]}$$

$$\mathbf{z}^{[1]} = \begin{bmatrix} z_1^{[1]} \\ z_2^{[1]} \\ z_3^{[1]} \end{bmatrix}_{3\times 1} = \begin{bmatrix} w_{11}^{[1]} & w_{21}^{[1]} \\ w_{12}^{[1]} & w_{22}^{[1]} \\ w_{13}^{[1]} & w_{23}^{[1]} \end{bmatrix}_{3\times 2} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}_{2\times 1} + \begin{bmatrix} b_1^{[1]} \\ b_2^{[1]} \\ b_3^{[1]} \end{bmatrix}_{3\times 1}$$

$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]}) = \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \end{bmatrix}_{3\times 1}$$

Neural Network: Vector & Matrix Form

$$\mathbf{z}^{[2} = W^{[2]T}\mathbf{a}^{[1]} + b^{[2]}$$

$$\mathbf{z}^{[2]} = \begin{bmatrix} z_1^{[2]} \end{bmatrix}_{1\times 1} = \begin{bmatrix} w_{11}^{[2]} & w_{21}^{[2]} & w_{31}^{[2]} \end{bmatrix}_{1\times 3} \begin{bmatrix} a_1^{[1]} \\ a_2^{[1]} \\ a_3^{[1]} \end{bmatrix}_{3\times 1} + b^{[2]}$$

$$\mathbf{a}^{[2]} = \sigma(\mathbf{z}^{[2]}) = \hat{y} \text{ (scalar)}$$

NN: single training example

$$\mathbf{z}^{[\mathbf{1}]} = W^{[1]T}\mathbf{x} + b^{[1]}$$

$$\mathbf{a}^{[\mathbf{1}]} = \sigma(\mathbf{z}^{[\mathbf{1}]})$$

$$\mathbf{z}^{[\mathbf{2}} = W^{[2]T}\mathbf{a}^{[1]} + b^{[2]}$$

$$\hat{y} = \mathbf{a}^{[\mathbf{2}]} = \sigma(\mathbf{z}^{[\mathbf{2}]})$$

NN: *m* training examples (2-dim)

$$Z^{[1]}{}_{3\times m} = W^{[1]T}{}_{3\times 2} X_{2\times m} + b^{[1]}$$

$$A^{[1]}{}_{3\times m} = \sigma(Z^{[1]}{}_{3\times m})$$

$$Z^{[2]}{}_{1\times m} = W^{[2]T}{}_{1\times 3} A^{[1]}{}_{3\times m} + b^{[2]}$$

$$A^{[2]}{}_{1\times m} = \sigma(Z^{[2]})$$

$$X = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \dots & x_1^{(m)} \\ x_2^{(1)} & x_2^{(2)} & \dots & x_2^{(m)} \end{bmatrix}_{2\times m}$$

$$Z^{[1]} = \begin{bmatrix} z_1^{[1](1)} & \dots & z_1^{[1](m)} \\ z_2^{[1](1)} & \dots & z_2^{[1](m)} \\ z_3^{[1](1)} & \dots & z_3^{[1](m)} \end{bmatrix}_{3\times m}$$

NN: for m training examples (d-dim)

- **to generalise:**
  - m training examples
  - d-dimensional features
  - h hidden units

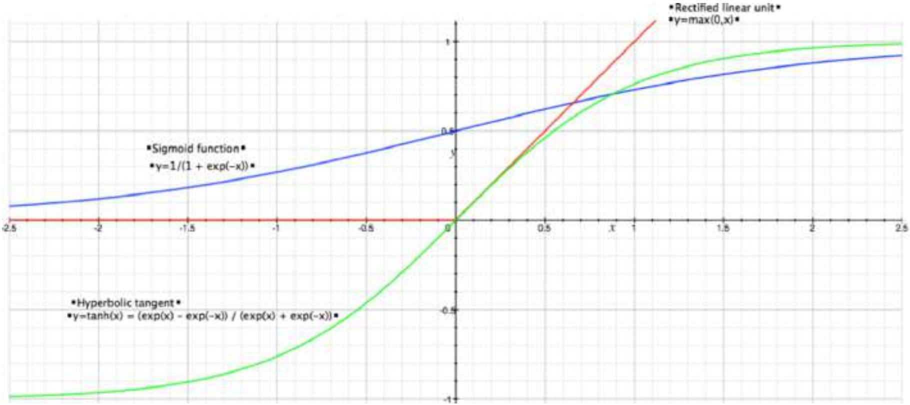
$$Z^{[1]}{}_{h\times m} = W^{[1]T}{}_{h\times d} X_{d\times m} + b^{[1]}$$

$$A^{[1]}{}_{h\times m} = \sigma(Z^{[1]}{}_{h\times m})$$

$$Z^{[2]}{}_{1\times m} = W^{[2]T}{}_{1\times h} A^{[1]}{}_{h\times m} + b^{[2]}$$

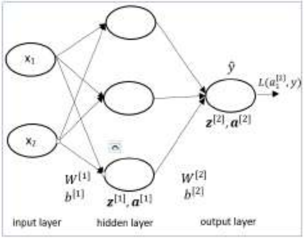
$$A^{[2]}{}_{1\times m} = \sigma(Z^{[2]})$$

Activation Functions



NN: *k*-classes

- so far, we had NN one output unit, computed $P(y = 1 \mid \mathbf{x})$
- used sigmoid to classify, useful for binary classification
- what if we have k outputs (e.g., sentiment analysis)
  - 2 classes (positive, negative) vs.
  - 5 classes (strongly negative, weakly negative, neutral, weakly positive, strongly positive)



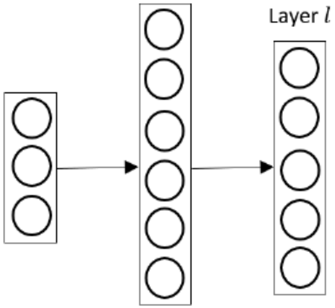
NN: softmax activation

- k-classes $\rightarrow$ k-output nodes $\rightarrow$ we output K-length vector

$\mathbf{z^{[l]}}_{5\times 1} = W^{[l]}\mathbf{a}^{[l-1]} + b^{[l]}$

$\mathbf{a^{[l]}}_{5\times 1}$,

where $a_i = \dfrac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$, for $i = 1, \ldots, K$



NN: softmax - Example

$$\mathbf{z}^{[l]} = \begin{bmatrix} 7 \\ 5 \\ 2 \\ -1 \\ 3 \end{bmatrix}; \mathbf{e}^{\mathbf{z}^{[l]}} = \begin{bmatrix} e^7 \\ e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} = \begin{bmatrix} 1096.63 \\ 148.4 \\ 7.4 \\ 0.4 \\ 20.1 \end{bmatrix}$$

$let \sum_{k=1}^{K} e^{z_k} = 1272.93$

where $a_1 = \frac{e^{z_1}}{\sum_{k=1}^{K} e^{z_k}} = \frac{1096.63}{1272.93}$

where $a_1 = 0.86$

Following this:

$$\mathbf{a}^{[l]} = \begin{bmatrix} 0.8615 \\ 0.1165 \\ 0.0058 \\ 0.0003 \\ 0.0157 \end{bmatrix}$$

NN: Backpropagation

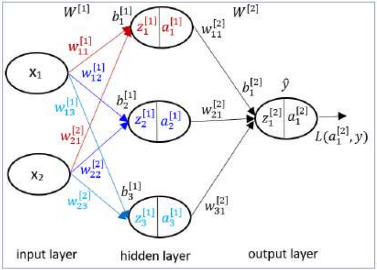
- we have computed *loss* or *error* function at the final output layer.

$$\mathbf{z}^{[1]} = W^{[1]}\mathbf{x} + b^{[1]}$$

$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]})$$

$$\mathbf{z}^{[2} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\hat{y} = \mathbf{a}^{[2]} = \sigma(\mathbf{z}^{[2]})$$



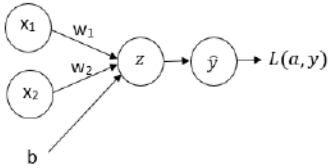
- update parameters $W^{[1]}, W^{[2]}, b^{[1]}, b^{[2]}$

Logistic Regression: Compute Gradients

Forward Pass

$$z = w_1 x_1 + w_2 x_2 + b$$

$$\hat{y} = a = \sigma(z)$$



$$L = L(a, y) = -(y \log(a) + (1 - y) \log(1 - a))$$

Backpropagate Errors

$$\frac{\partial L}{\partial w_1} = \frac{\partial L}{\partial a} \frac{\partial a}{\partial z} \frac{\partial z}{\partial w_1}$$

$$\frac{\partial L}{\partial w_1} = -\frac{y}{a} + \frac{1-y}{1-a} = \frac{a-y}{a(1-a)}; \frac{\partial a}{\partial z} = a(1 - a); \frac{\partial z}{\partial w_1} = x_1$$

$$\frac{\partial L}{\partial w_1} = (a - y)x_1; \ \frac{\partial L}{\partial w_2} = (a - y)x_2; \ \frac{\partial L}{\partial b} = a - y$$

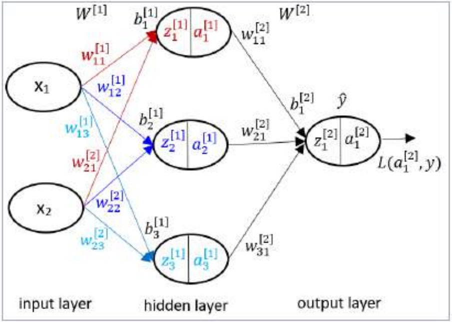
NN: Backpropagation

$$\mathbf{z}^{[1]} = W^{[1]}\mathbf{x} + b^{[1]}$$

$$\mathbf{a}^{[1]} = \sigma(\mathbf{z}^{[1]})$$

$$\mathbf{z}^{[2} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\hat{y} = \mathbf{a}^{[2]} = \sigma(\mathbf{z}^{[2]})$$



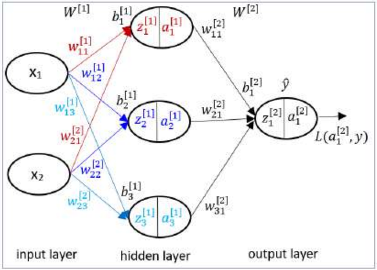
- update parameters $W^{[1]}, W^{[2]}, b^{[1]}, b^{[2]}$

NN: Backpropagation

update parameters $W^{[2]}$; compute $\frac{\partial L}{\partial W^{[2]}}$

$$\frac{\partial L}{\partial W^{[2]}} = \frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial W^{[2]}}$$

(chain rule: $\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z}\frac{\partial z}{\partial x}$)



NN: Backpropagation

1. compute: $\frac{\partial L}{\partial \mathbf{a}^{[2]}}$

$$L = L(\mathbf{a}^{[2]}, y) = -\left(y \log\left(\mathbf{a}^{[2]}\right) + (1-y)\log\left(1-\mathbf{a}^{[2]}\right)\right)$$

(using log-likelihood of cross entropy function)

$$\frac{\partial L}{\partial \mathbf{a}^{[2]}} = -(y \times \frac{1}{\mathbf{a}^{[2]}} + (1-y) \times \frac{1}{1-\mathbf{a}^{[2]}} \times (-1))$$

$$\frac{\partial L}{\partial \mathbf{a}^{[2]}} = -\left(\frac{y}{\mathbf{a}^{[2]}} - \frac{1-y}{1-\mathbf{a}^{[2]}}\right)$$

$$\frac{\partial L}{\partial \mathbf{a}^{[2]}} = \frac{\mathbf{a}^{[2]} - y}{\mathbf{a}^{[2]}(1-\mathbf{a}^{[2]})}$$

NN: Backpropagation

2. compute: $\frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}}$

$$\mathbf{a}^{[2]} = \frac{1}{1 + e^{-(\mathbf{z}^{[2]})}}$$

$$\frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} = \mathbf{a}^{[2]}(1 - \mathbf{a}^{[2]})$$

NN: Backpropagation

3. compute: $\frac{\partial \mathbf{z}^{[2]}}{\partial W^{[2]}}$

$$\mathbf{z}^{[2]} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\frac{\partial \mathbf{z}^{[2]}}{\partial W^{[2]}} = \mathbf{a}^{[1]}$$

We have now computed: $\frac{\partial L}{\partial \mathbf{a}^{[2]}}$; $\frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}}$; $\frac{\partial \mathbf{z}^{[2]}}{\partial W^{[2]}}$

$$\frac{\partial L}{\partial W^{[2]}} = \frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial W^{[2]}}$$

$$\frac{\partial L}{\partial W^{[2]}} = \frac{\mathbf{a}^{[2]} - y}{\mathbf{a}^{[2]}(1 - \mathbf{a}^{[2]})}\mathbf{a}^{[2]}(1 - \mathbf{a}^{[2]})\mathbf{a}^{[1]}$$

$$\frac{\partial L}{\partial W^{[2]}} = (\mathbf{a}^{[2]} - y)\mathbf{a}^{[1]}$$

NN: Backpropagation

update parameters $b^{[2]}$; compute $\frac{\partial L}{\partial b^{[2]}}$

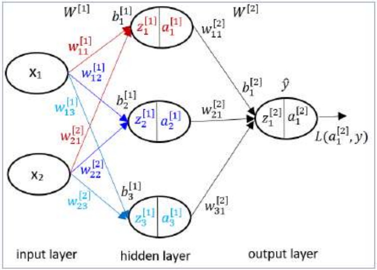
$$\frac{\partial L}{\partial b^{[2]}} = \frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial b^{[2]}}$$

we know: $\frac{\partial L}{\partial \mathbf{a}^{[2]}}$; $\frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}}$

$$\mathbf{z}^{[2]} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\frac{\partial \mathbf{z}^{[2]}}{\partial b^{[2]}} = 1$$

Thus $\frac{\partial L}{\partial b^{[2]}} = (\mathbf{a}^{[2]} - y)$



NN: Backpropagation

update parameters $W^{[1]}$; compute $\frac{\partial L}{\partial W^{[1]}}$

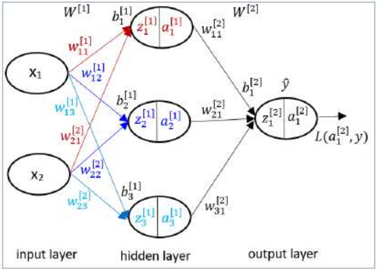
$$\frac{\partial L}{\partial W^{[1]}} = \frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} \frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{a}^{[1]}} \frac{\partial \mathbf{a}^{[1]}}{\partial \mathbf{z}^{[1]}} \frac{\partial \mathbf{z}^{[1]}}{\partial W^{[1]}}$$

we know: $\frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}} = (\mathbf{a}^{[2]} - y)$

compute $\frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{a}^{[1]}}$

$$\mathbf{z}^{[2} = W^{[2]}\mathbf{a}^{[1]} + b^{[2]}$$

$$\frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{a}^{[1]}} = W^{[2]}$$



NN: Backpropagation

compute: $\frac{\partial \mathbf{a}^{[1]}}{\partial \mathbf{z}^{[1]}}$

$$\mathbf{a}^{[1]} = \frac{1}{1 + e^{-(\mathbf{z}^{[1]})}}$$

$$\frac{\partial \mathbf{a}^{[1]}}{\partial \mathbf{z}^{[1]}} = g^{[1]\prime}(\mathbf{z}^{[1]}) = \mathbf{a}^{[1]}(1 - \mathbf{a}^{[1]})$$

Finally: $\mathbf{z}^{[1]} = W^{[1]}\mathbf{x} + b^{[1]}$

$$\frac{\partial \mathbf{z}^{[1]}}{\partial W^{[1]}} = \mathbf{x}$$

NN: Backpropagation

Putting everything together:

$$\frac{\partial L}{\partial W^{[1]}} = \underbrace{\frac{\partial L}{\partial \mathbf{a}^{[2]}} \frac{\partial \mathbf{a}^{[2]}}{\partial \mathbf{z}^{[2]}}}_{\mathbf{a}^{[2]}-y} \underbrace{\frac{\partial \mathbf{z}^{[2]}}{\partial \mathbf{a}^{[1]}}}_{W^{[2]}} \underbrace{\frac{\partial \mathbf{a}^{[1]}}{\partial \mathbf{z}^{[1]}}}_{g^{[1]\prime}(\mathbf{z}^{[1]})} \underbrace{\frac{\partial \mathbf{z}^{[1]}}{\partial W^{[1]}}}_{\mathbf{x}}$$

$$\underbrace{\frac{\partial L}{\partial W^{[1]}}}_{3\times 2} = \underbrace{\mathbf{a}^{[2]} - y}_{1\times 1} \underbrace{W^{[2]}}_{3\times 1} \underbrace{g^{[1]\prime}(\mathbf{z}^{[1]})}_{3\times 1} \underbrace{\mathbf{x}}_{2\times 1}$$

$$\underbrace{\frac{\partial L}{\partial W^{[1]}}}_{3\times 2} = \underbrace{W^{[2]T}}_{3\times 1} \circ \underbrace{g^{[1]\prime}(\mathbf{z}^{[1]})}_{3\times 1} \underbrace{\mathbf{a}^{[2]} - y}_{1\times 1} \underbrace{\mathbf{x}^{T}}_{1\times 2}$$

NN: Gradient Descent

for any single layer ℓ, the update rule is defined by:

$$W^{[\ell]} = W^{[\ell]} - \alpha \frac{\partial J}{\partial W^{[\ell]}}$$

where $J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}^{(i)}$, where $\mathcal{L}^{(i)}$ is the loss for the single example.