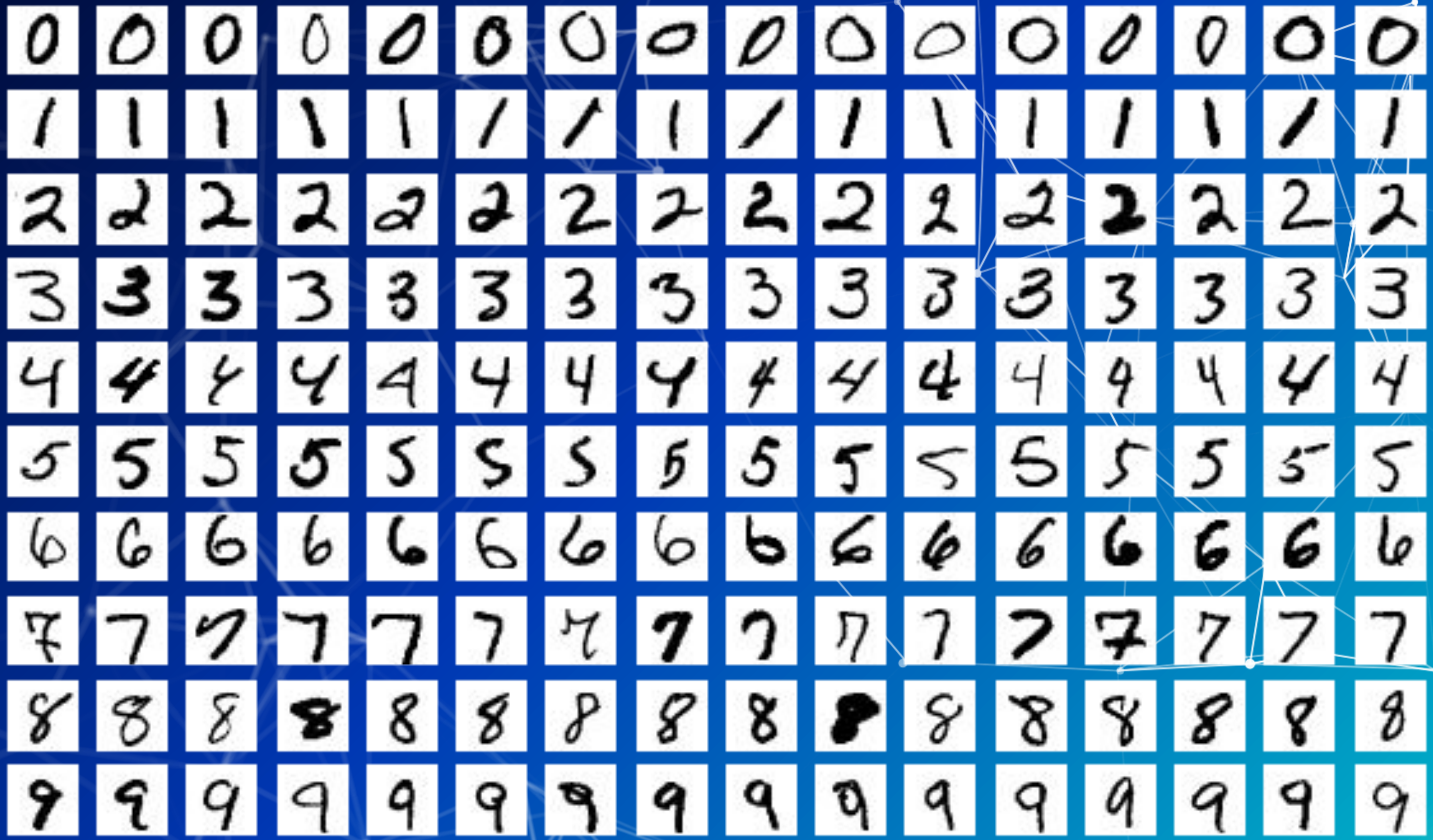
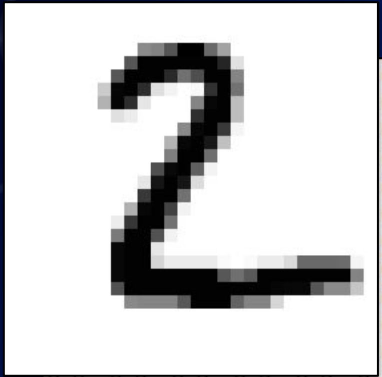
Feedforward Neural Networks

Calvin Osborne

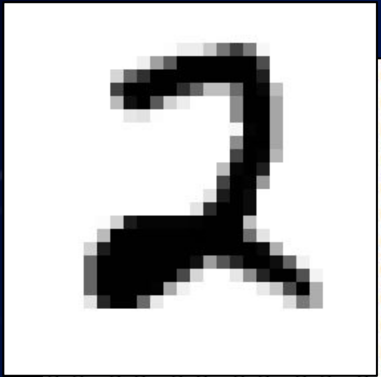
Feedforward Neural Networks

—Background/Motivation—





[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.5,	0.5,	0.7,	1.0,	1.0,	0.6,	0.4,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.7,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	0.9,	0.1,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.7,	1.0,	1.0,	1.0,	0.8,	0.6,	0.7,	1.0,	1.0,	0.5,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.2,	1.0,	1.0,	0.8,	0.1,	0.0,	0.0,	0.0,	0.8,	1.0,	0.5,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.3,	1.0,	0.8,	0.1,	0.0,	0.0,	0.0,	0.5,	1.0,	1.0,	0.3,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.1,	0.1,	0.0,	0.0,	0.0,	0.0,	0.8,	1.0,	1.0,	0.3,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.5,	1.0,	1.0,	0.8,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.3,	1.0,	1.0,	0.9,	0.2,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.5,	1.0,	1.0,	0.6,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.7,	1.0,	1.0,	0.6,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.1,	0.9,	1.0,	0.9,	0.1,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.8,	1.0,	1.0,	0.6,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.3,	1.0,	1.0,	0.7,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.1,	0.8,	1.0,	1.0,	0.6,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.5,	1.0,	1.0,	0.7,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	1.0,	1.0,	1.0,	0.1,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	1.0,	1.0,	1.0,	0.2,	0.1,	0.1,	0.1,	0.1,	0.0,	0.0,	0.0,	0.1,	0.1,	0.1,	0.6,	0.6,	0.6,	0.6,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	0.7,	0.6,	0.7,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	0.5,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.7,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	1.0,	0.7,	0.5,	0.5,	0.2,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.5,	0.5,	0.5,	0.5,	0.7,	1.0,	1.0,	1.0,	0.6,	0.5,	0.5,	0.2,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,
0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,	0.0,

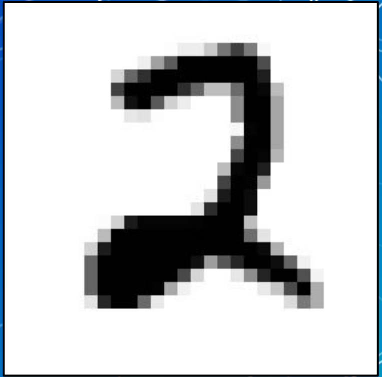


```
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.1, 0.2, 0.5, 0.8, 0.5, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.4, 0.6, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.7, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.2, 0.6, 0.8, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.9, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.7, 1.0, 1.0, 1.0, 1.0, 0.8, 0.6, 0.3, 0.3, 0.6, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.3, 0.9, 1.0, 0.7, 0.2, 0.1, 0.0, 0.0, 0.0, 0.4, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.1, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.4, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.7, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 1.0, 1.0, 0.9, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.7, 1.0, 1.0, 0.4, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.9, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.8, 1.0, 1.0, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.9, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.3, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.6, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.2, 0.9, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.6, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.9, 0.4, 0.7, 0.9, 1.0, 1.0, 0.9, 0.6, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.6, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.7, 0.2, 0.0, 0.0, 0.1, 0.4, 0.9, 1.0, 1.0, 0.7, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.6, 1.0, 1.0, 1.0, 1.0, 1.0, 0.3, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.2, 0.7, 1.0, 0.3, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.8, 1.0, 1.0, 0.6, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.1, 0.5, 0.3, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
```

28 x 28 =
784 inputs

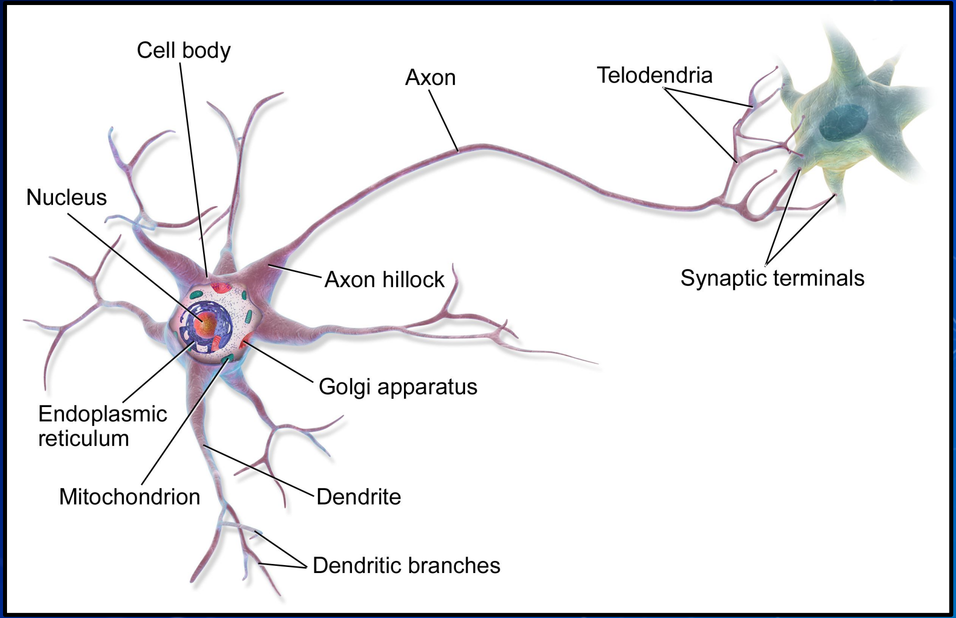
```
.
.
.
0.0,
0.0,
0.0,
0.0,
0.0,
0.2,
0.7,
0.3,
0.0,
0.0,
0.0,
0.0,
0.0,
0.0,
0.0,
0.0,
0.2,
1.0,
1.0,
1.0,
1.0,
1.0,
1.0,
0.0,
0.0,
0.0,
0.0,
0.0,
.
.
.
```

$$f(x)$$



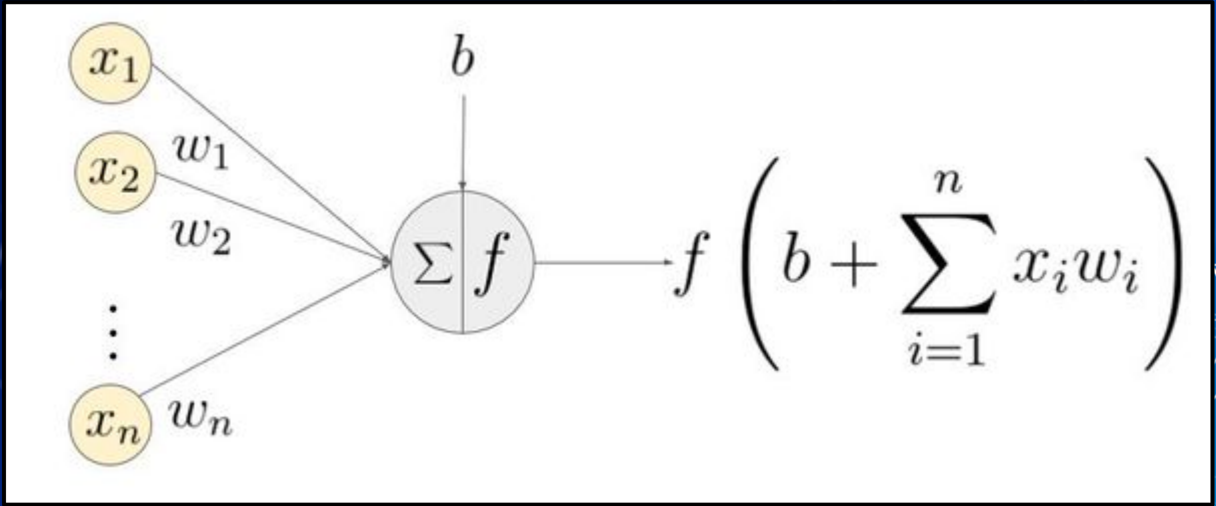
Why a Neural Networks?

Complexity Versatility Extrapolation



Feedforward Neural Networks

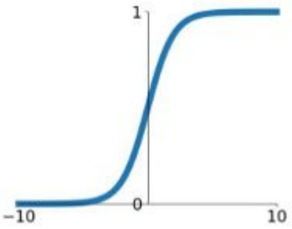
—Single Perceptron Model—



$$y = \sigma\left(\sum_i x_i w_i + b\right)$$

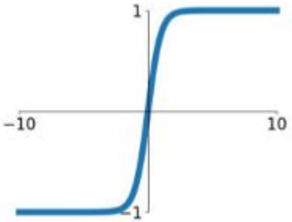
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



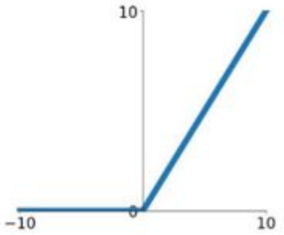
tanh

$$\tanh(x)$$



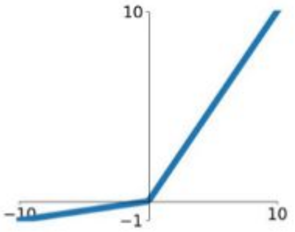
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$

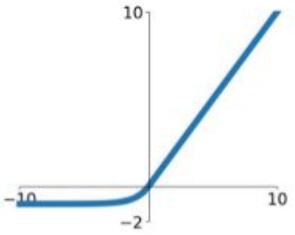


Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

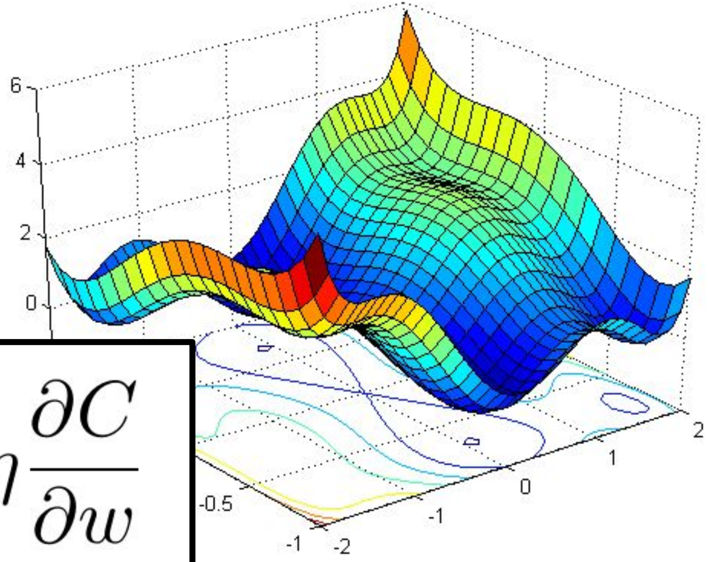


https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044

$$C = (y - \hat{y})^2$$

$$z = \sum_i x_i w_i + b$$

$$y = \sigma(z)$$



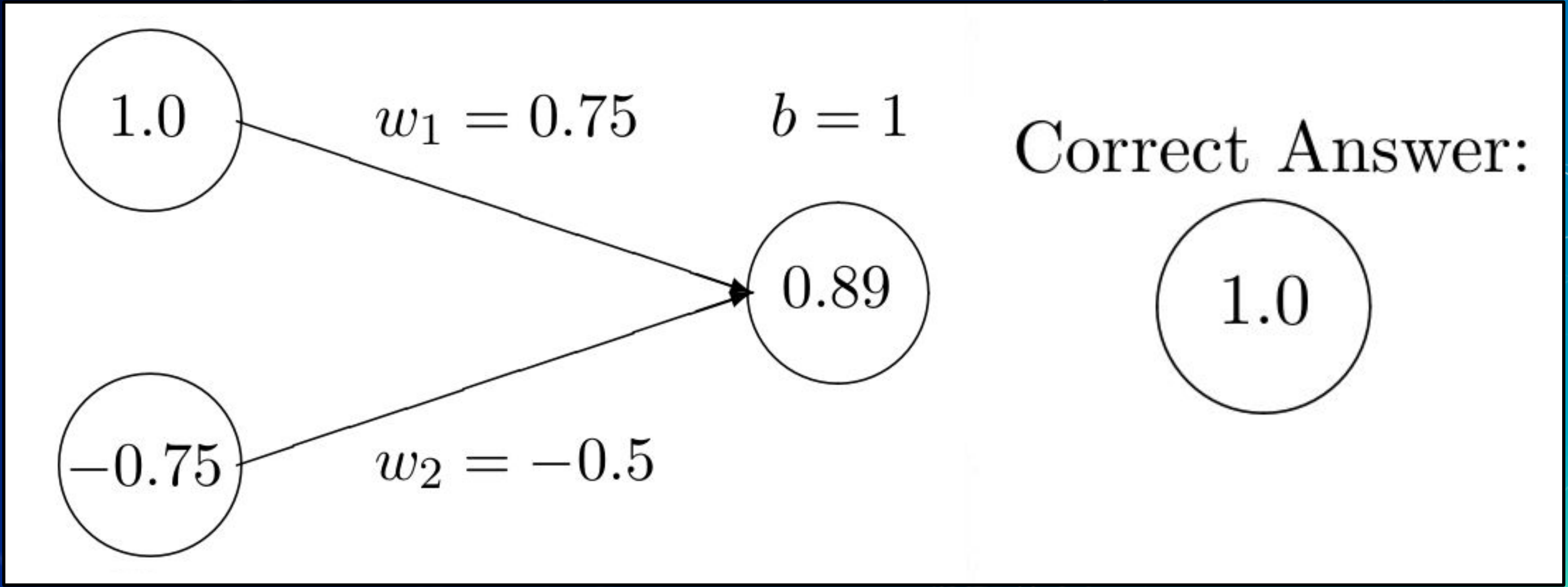
$$w \to w - \eta \frac{\partial C}{\partial w}$$

$$b \to w - \eta \frac{\partial C}{\partial b}$$

https://medium.com/ml-cheat-sheet/winning-at-loss-functions-2-important-loss-functions-in-computer-vision-b2b9d293e15a

$$
\begin{aligned}
\frac{\partial C}{\partial w_i} &= \frac{\mathrm{d}C}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}z}\frac{\partial z}{\partial x_i} \\
&= 2(y-\hat{y})\sigma'(z)x_i \\
\frac{\partial C}{\partial b} &= \frac{\mathrm{d}C}{\mathrm{d}y}\frac{\mathrm{d}y}{\mathrm{d}z}\frac{\partial z}{\partial b} \\
&= 2(y-\hat{y})\sigma'(z)
\end{aligned}
$$

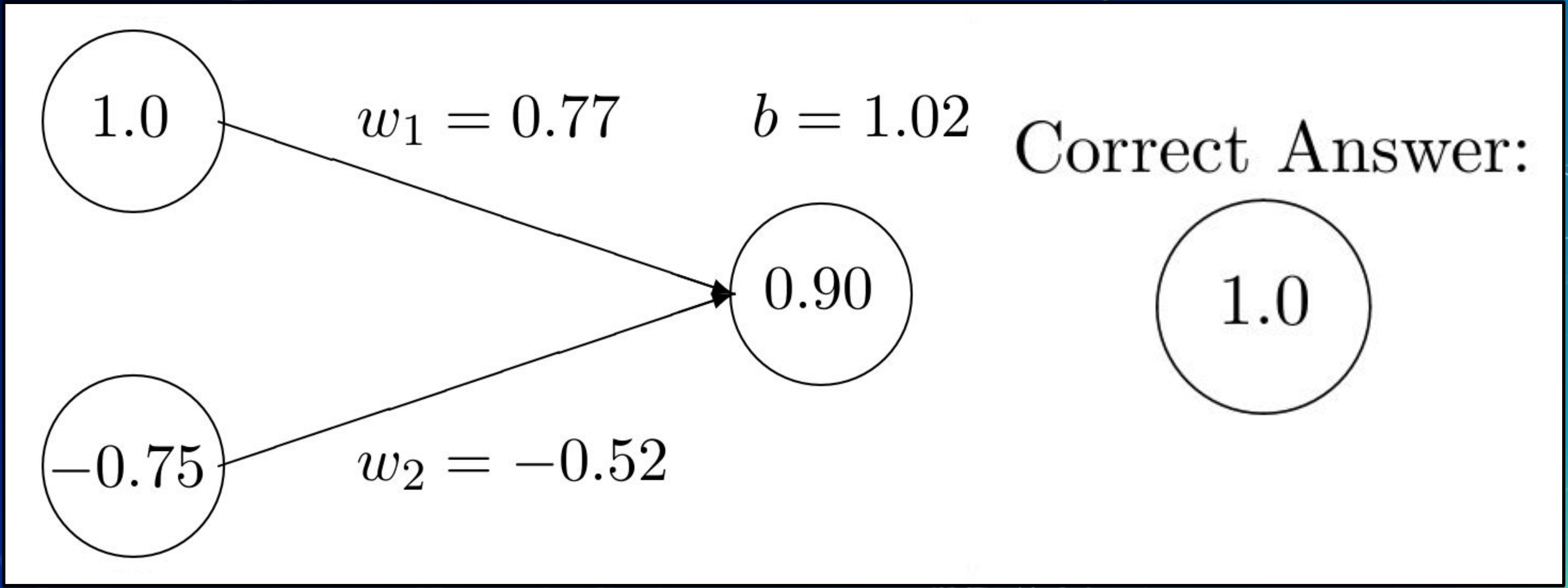
Problem: Is x_1 larger than x_2?



Computation:

$$
\begin{aligned}
C &= (0.89 - 1)^2 = 0.0121 \\
\partial C/\partial w_1 &= 2(0.89 - 1) \cdot \sigma'(1.0 \cdot 0.75 + -0.75 \cdot -0.5 + 1) \cdot 1.0 \\
&= -0.021 \\
\partial C/\partial w_2 &= 2(0.89 - 1) \cdot \sigma'(1.0 \cdot 0.75 + -0.75 \cdot -0.5 + 1) \cdot -0.75 \\
&= 0.016 \\
\partial C/\partial b &= 2(0.89 - 1) \cdot \sigma'(1.0 \cdot 0.75 + -0.75 \cdot -0.5 + 1) \\
&= -0.021
\end{aligned}
$$

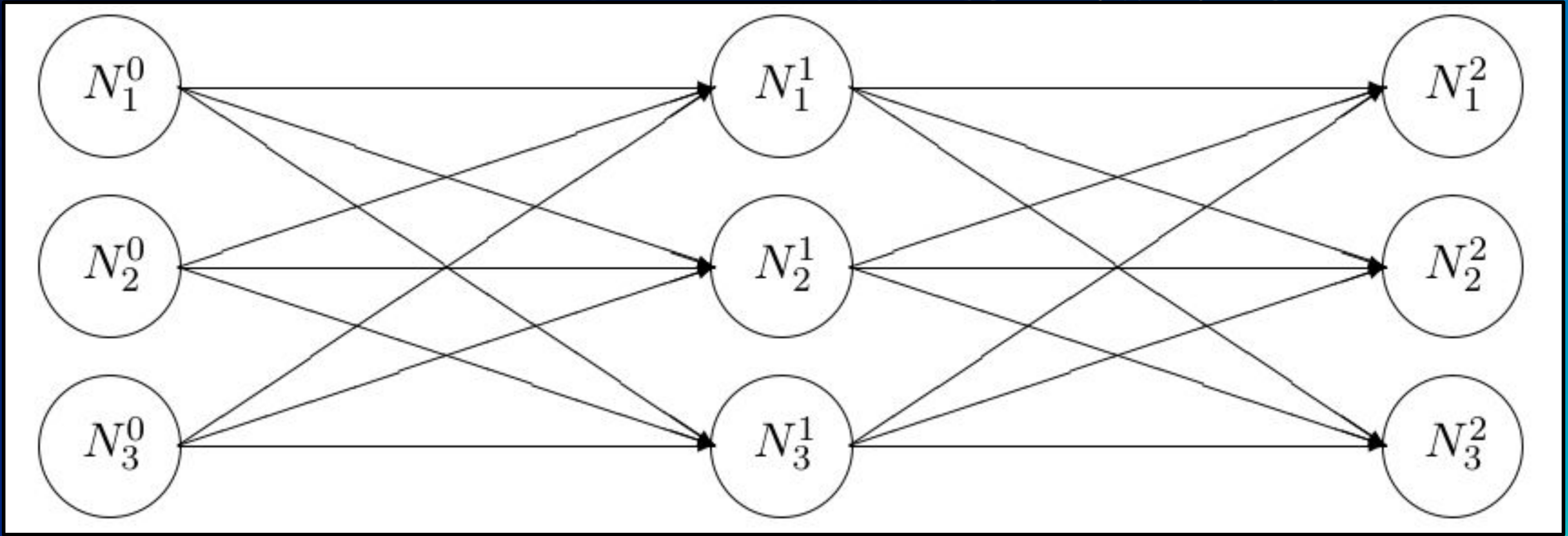
Problem: Is x_1 larger than x_2?



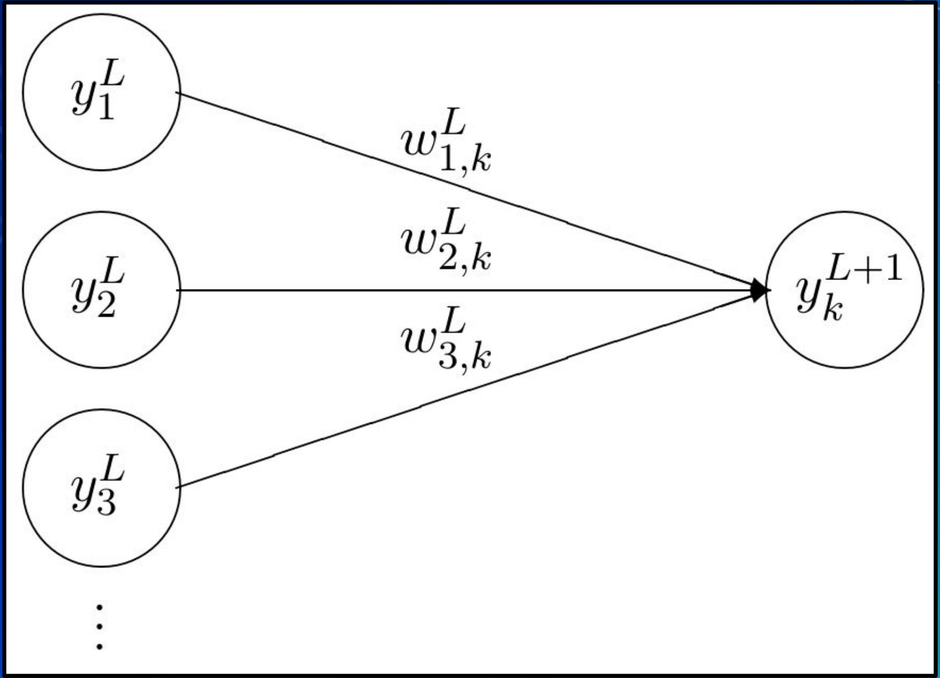
Feedforward Neural Networks

—Neural Network Model—

General Feedforward Network



$$y_k^{L+1} = \sigma(z_k^{L+1}) = \sigma\left(b_k^L + \sum_j w_{j,k}^L y_j^L\right)$$



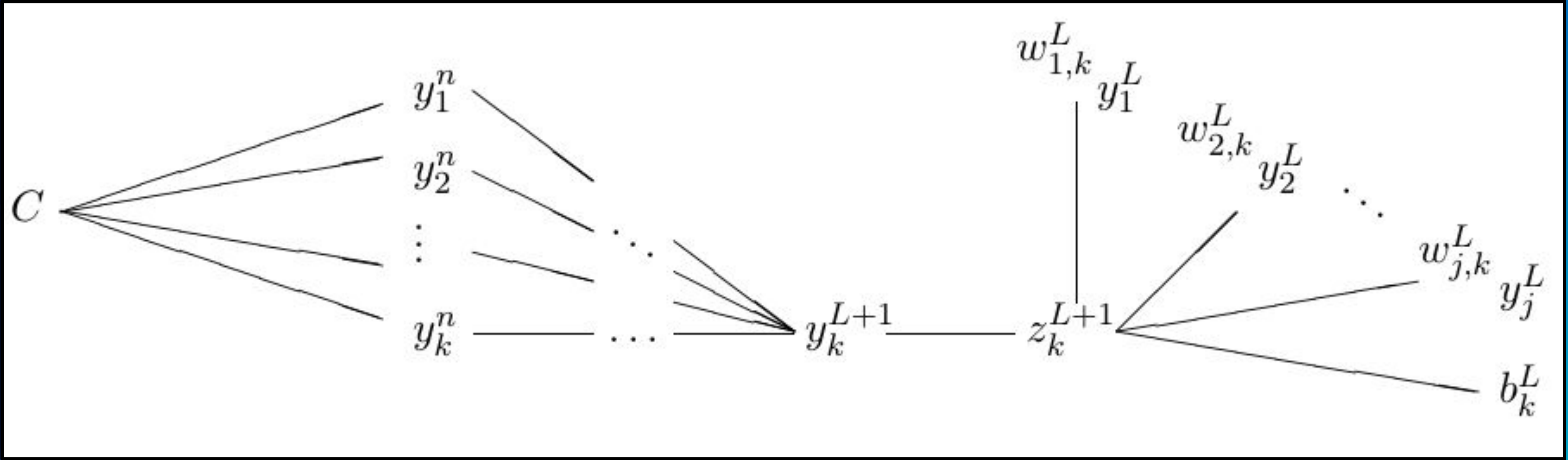
$$
\begin{bmatrix} y_1^{L+1} \\ y_2^{L+1} \\ \vdots \\ y_k^{L+1} \end{bmatrix} = \sigma\left(\begin{bmatrix} z_1^{L+1} \\ z_2^{L+1} \\ \vdots \\ z_k^{L+1} \end{bmatrix}\right) = \sigma\left(\begin{bmatrix} b_1^L \\ b_2^L \\ \vdots \\ b_k^L \end{bmatrix} + \begin{bmatrix} \sum_j w_{j,1}^L y_j^L \\ \sum_j w_{j,2}^L y_j^L \\ \vdots \\ \sum_j w_{j,k}^L y_j^L \end{bmatrix}\right)
$$

$$
= \sigma\left(\begin{bmatrix} b_1^L \\ b_2^L \\ \vdots \\ b_k^L \end{bmatrix} + \begin{bmatrix} w_{1,1}^L & w_{2,1}^L & \cdots & w_{j,1}^L \\ w_{1,2}^L & w_{2,2}^L & \cdots & w_{j,2}^L \\ \vdots & \vdots & \ddots & \vdots \\ w_{1,k}^L & w_{2,k}^L & \cdots & w_{j,k}^L \end{bmatrix} \begin{bmatrix} y_1^L \\ y_2^L \\ \vdots \\ y_j^L \end{bmatrix}\right)
$$

$$
\boldsymbol{y}^{L+1} = \boldsymbol{\sigma}(\boldsymbol{z}^{L+1}) = \boldsymbol{\sigma}(\boldsymbol{b}^L + \boldsymbol{w}^L \boldsymbol{y}^L)
$$

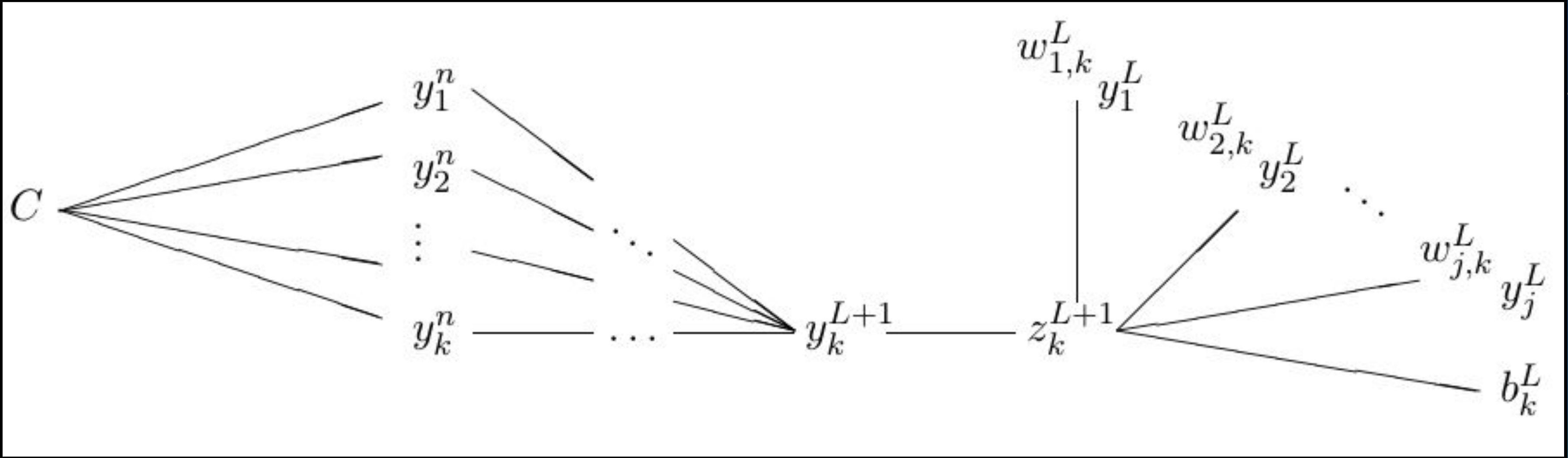
$$
C = \sum_i (y_i^n - \hat{y}_i)^2
$$

Chain Rule Diagram



$$\begin{aligned}\frac{\partial C}{\partial w_{j,k}^L} &= \frac{\partial C}{\partial y_k^{L+1}} \cdot \frac{\partial y_k^{L+1}}{\partial z_k^{L+1}} \cdot \frac{\partial z_k^{L+1}}{\partial w_{j,k}^L} \\ &= \frac{\partial C}{\partial y_k^{L+1}} \cdot \sigma'(z_k^{L+1}) \cdot y_j^L\end{aligned}$$

$$\begin{aligned}\frac{\partial C}{\partial b_k^L} &= \frac{\partial C}{\partial y_k^{L+1}} \cdot \frac{\partial y_k^{L+1}}{\partial z_k^{L+1}} \cdot \frac{\partial z_k^{L+1}}{\partial b_k^L} \\ &= \frac{\partial C}{\partial y_k^{L+1}} \cdot \sigma'(z_k^{L+1})\end{aligned}$$



$$
\begin{aligned}
\frac{\partial C}{\partial y_j^L} &= \frac{\partial C}{\partial y_1^{L+1}} \cdot \frac{\partial y_1^{L+1}}{\partial z_1^{L+1}} \cdot \frac{\partial z_1^{L+1}}{\partial y_j^L} + \frac{\partial C}{\partial y_2^{L+1}} \cdot \frac{\partial y_2^{L+1}}{\partial z_2^{L+1}} \cdot \frac{\partial z_2^{L+1}}{\partial y_j^L} + \cdots + \frac{\partial C}{\partial y_k^{L+1}} \cdot \frac{\partial y_k^{L+1}}{\partial z_k^{L+1}} \cdot \frac{\partial z_k^{L+1}}{\partial y_j^L} \\
&= \sum_k \frac{\partial C}{\partial y_k^{L+1}} \cdot \frac{\partial y_k^{L+1}}{\partial z_k^{L+1}} \cdot \frac{\partial z_k^{L+1}}{\partial y_j^L} \\
&= \sum_k \frac{\partial C}{\partial y_k^{L+1}} \cdot \sigma'(z_k^{L+1}) \cdot w_{j,k}^L.
\end{aligned}
$$

$$\begin{bmatrix} \partial C/\partial w_{1,1}^L & \partial C/\partial w_{2,1}^L & \dots & \partial C/\partial w_{j,1}^L \\ \partial C/\partial w_{1,2}^L & \partial C/\partial w_{2,2}^L & \dots & \partial C/\partial w_{j,2}^L \\ \vdots & \vdots & \ddots & \vdots \\ \partial C/\partial w_{1,k}^L & \partial C/\partial w_{2,k}^L & \dots & \partial C/\partial w_{j,k}^L \end{bmatrix} = \left(\begin{bmatrix} \partial C/\partial y_L^{l+1} \\ \partial C/\partial y_2^{L+1} \\ \vdots \\ \partial C/\partial y_k^{L+1} \end{bmatrix} \circ \begin{bmatrix} \sigma'(z_1^{L+1}) \\ \sigma'(z_2^{L+1}) \\ \vdots \\ \sigma'(z_k^{L+1}) \end{bmatrix} \right) \begin{bmatrix} y_1^L \\ y_2^L \\ \vdots \\ y_j^L \end{bmatrix}^\top$$

$$\begin{bmatrix} \partial C/\partial b_1^L \\ \partial C/\partial b_2^L \\ \vdots \\ \partial C/\partial b_k^L \end{bmatrix} = \begin{bmatrix} \partial C/\partial y_1^{L+1} \\ \partial C/\partial y_2^{L+1} \\ \vdots \\ \partial C/\partial y_k^{L+1} \end{bmatrix} \circ \begin{bmatrix} \sigma'(z_1^{L+1}) \\ \sigma'(z_2^{L+1}) \\ \vdots \\ \sigma'(z_k^{L+1}) \end{bmatrix}$$

$$\begin{bmatrix} \partial C/\partial y_1^L \\ \partial C/\partial y_2^L \\ \vdots \\ \partial C/\partial y_j^L \end{bmatrix} = \begin{bmatrix} w_{1,1}^L & w_{2,1}^L & \dots & w_{j,1}^L \\ w_{1,2}^L & w_{2,2}^L & \dots & w_{j,2}^L \\ \vdots & \vdots & \ddots & \vdots \\ w_{1,k}^L & w_{2,k}^L & \dots & w_{j,k}^L \end{bmatrix}^\top \cdot \left(\begin{bmatrix} \sigma'(z_1^{L+1}) \\ \sigma'(z_2^{L+1}) \\ \vdots \\ \sigma'(z_k^{L+1}) \end{bmatrix} \circ \begin{bmatrix} \partial C/\partial y_1^{L+1} \\ \partial C/\partial y_2^{L+1} \\ \vdots \\ \partial C/\partial y_k^{L+1} \end{bmatrix} \right)$$

The Gradient Decent Equations (Unsimplified)

$$1)\ \nabla_{w^L}\boldsymbol{C} = (\nabla_{y^{L+1}}\boldsymbol{C} \circ \sigma'(\boldsymbol{z}^{L+1})) \cdot (\boldsymbol{y}^L)^\top$$

$$2)\ \nabla_{b^L}\boldsymbol{C} = \nabla_{y^{L+1}}\boldsymbol{C} \circ \sigma'(\boldsymbol{z}^{L+1})$$

$$3)\ \nabla_{y^L}\boldsymbol{C} = (\boldsymbol{w}^L)^\top \cdot (\nabla_{y^{L+1}}\boldsymbol{C} \circ \sigma'(\boldsymbol{z}^{L+1}))$$

Getting rid of the y-Gradient: Delta Notation

$$\boldsymbol{\delta}^n = \nabla_{y^n} \boldsymbol{C} \circ \sigma'(\boldsymbol{z}^n)$$

$$\boldsymbol{\delta}^L = ((\boldsymbol{w}^L)^\top \boldsymbol{\delta}^{L+1}) \circ \sigma'(\boldsymbol{z}^L)$$

$$\Rightarrow \boldsymbol{\delta}^L = \nabla_{y^L} \boldsymbol{C} \circ \sigma'(\boldsymbol{z}^L)$$

The Gradient Decent Equations

1) $\boldsymbol{\delta}^n = \nabla_{y^n} \boldsymbol{C} \circ \sigma'(\boldsymbol{z}^n)$

2) $\boldsymbol{\delta}^L = ((\boldsymbol{w}^L)^\top \boldsymbol{\delta}^{L+1}) \circ \sigma'(\boldsymbol{z}^L)$

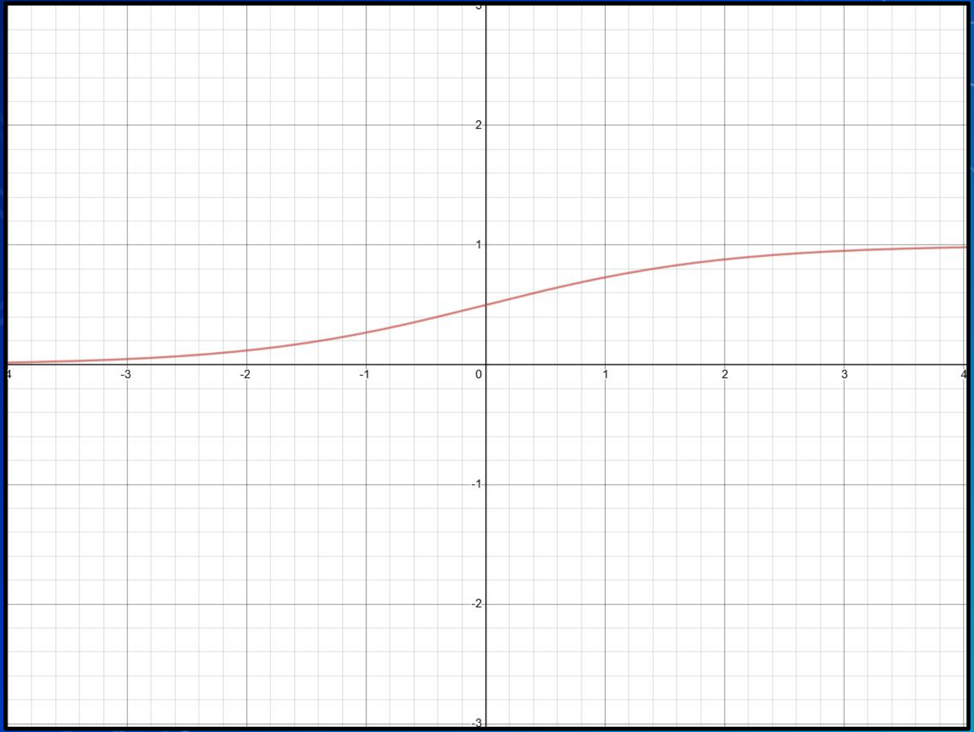
3) $\nabla_{w^L} \boldsymbol{C} = \boldsymbol{\delta}^L \cdot (\boldsymbol{y}^L)^\top$

4) $\nabla_{b^L} \boldsymbol{C} = \boldsymbol{\delta}^L$

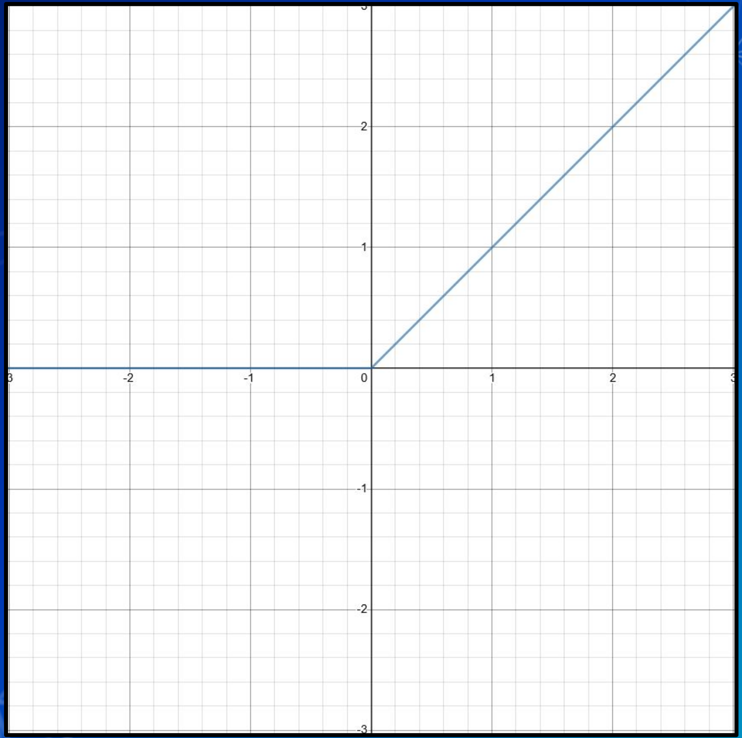
Feedforward Neural Networks

—Issues/Potential Solutions—

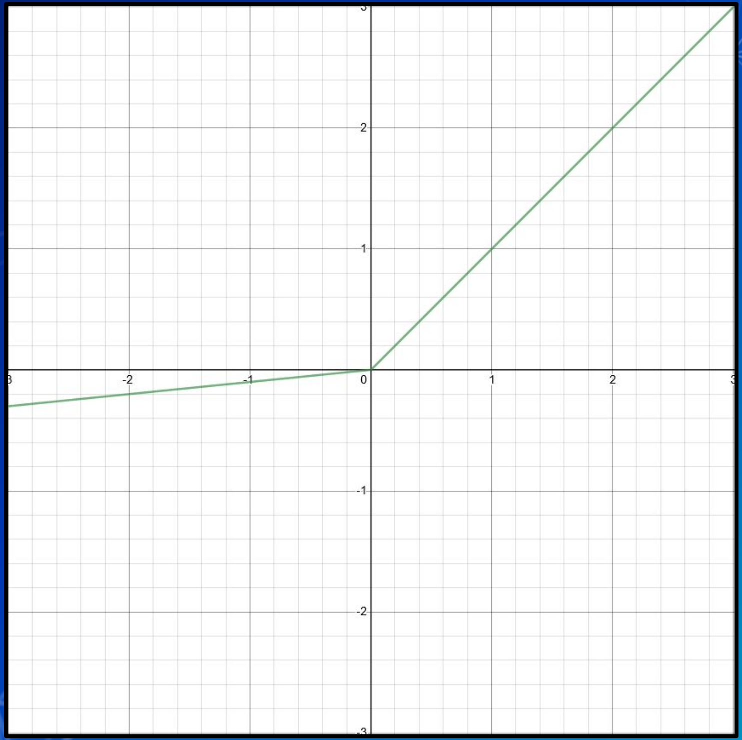
Vanishing Gradient Problem



Vanishing Gradient Problem



Vanishing Gradient Problem



Vanishing Gradient Problem (Cross-Entropy)

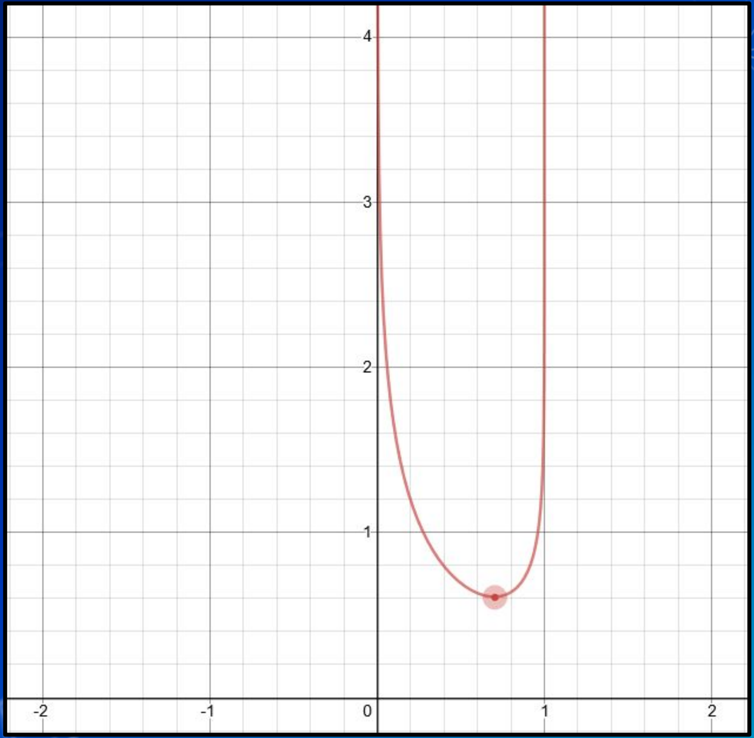
$$\frac{\partial C}{\partial w^{n-1}} = \frac{\partial C}{\partial y^n} \cdot \sigma'(z^n) \cdot y^{n-1}$$

Quadratic Cost Function: $C = \frac{1}{2}(y^n - \hat{y})^2$

$$\Rightarrow \frac{\partial C}{\partial w^{n-1}} = (y^n - \hat{y}) \cdot \sigma'(z^n) \cdot y^{n-1}$$

Cross-Entropy Cost Function: $C = -\hat{y} \ln y^n - (1 - \hat{y}) \ln (1 - y^n)$

Vanishing Gradient Problem (Cross-Entropy)



$$\frac{\partial C}{\partial y^n} = \frac{1-\hat{y}}{1-y^n} - \frac{\hat{y}}{y^n} = \frac{y^n - \hat{y}}{y^n(1-y^n)}$$

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$\sigma'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}}\left(\frac{1+e^{-x}-1}{1+e^{-x}}\right)$$

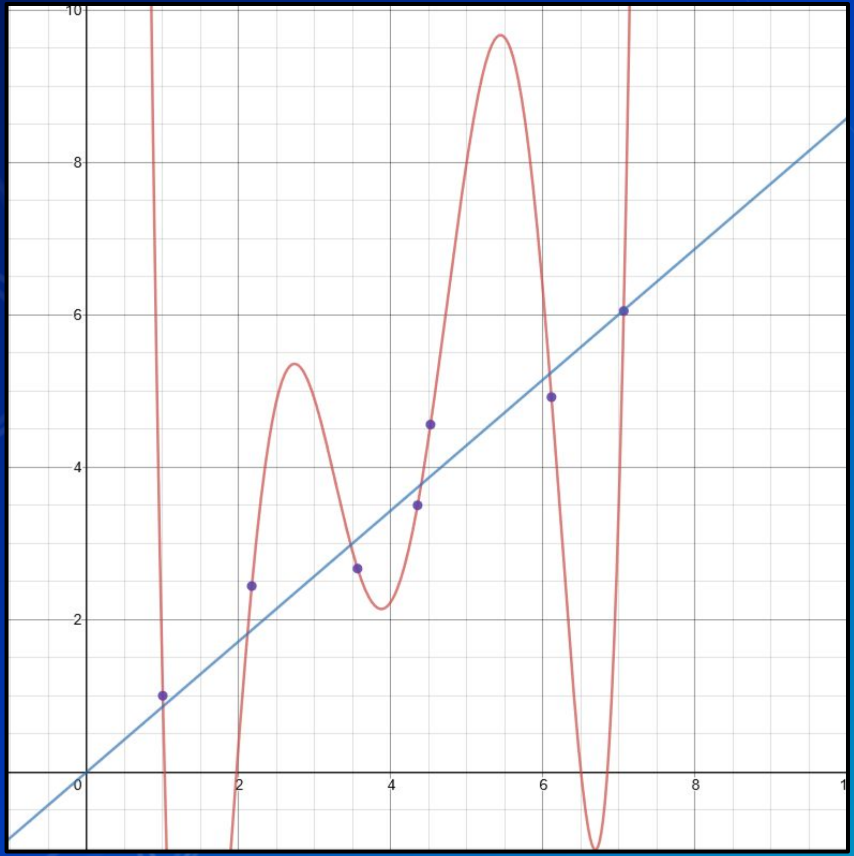
$$\sigma'(x) = \sigma(x)(1-\sigma(x))$$

Vanishing Gradient Problem (Cross-Entropy)

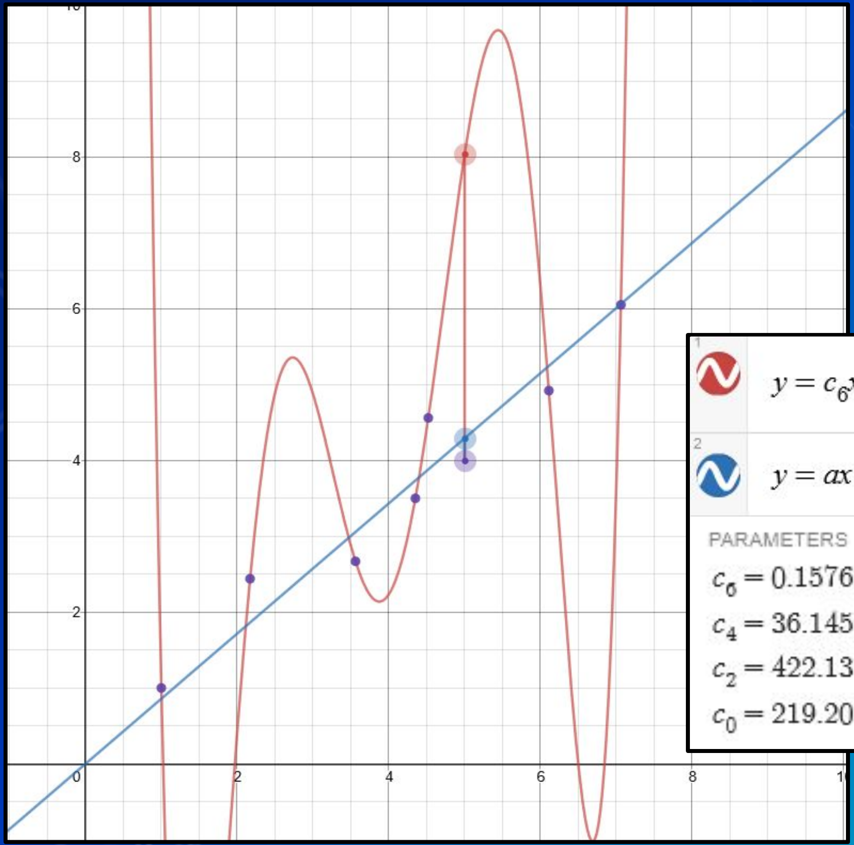
Cross-Entropy Cost Function: $C = -\hat{y}\ln y^n - (1-\hat{y})\ln(1-y^n)$

$$
\begin{aligned}
\Rightarrow \frac{\partial C}{\partial w^{n-1}} &= \frac{y^n - \hat{y}}{y^n(1-y^n)} \cdot \sigma'(z^n) \cdot y^{n-1} \\
&= \frac{y^n - \hat{y}}{y^n(1-y^n)} \cdot y^n(1-y^n) \cdot y^{n-1} \\
&= (y^n - \hat{y}) \cdot y^{n-1}
\end{aligned}
$$

Overfitting and Regularization



Overfitting and Regularization



$$y = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$

$$y = ax$$

PARAMETERS

$c_6 = 0.157635$ $c_5 = -3.80403$
$c_4 = 36.1454$ $c_3 = -171.388$
$c_2 = 422.13$ $c_1 = -501.443$
$c_0 = 219.202$

PARAMETERS

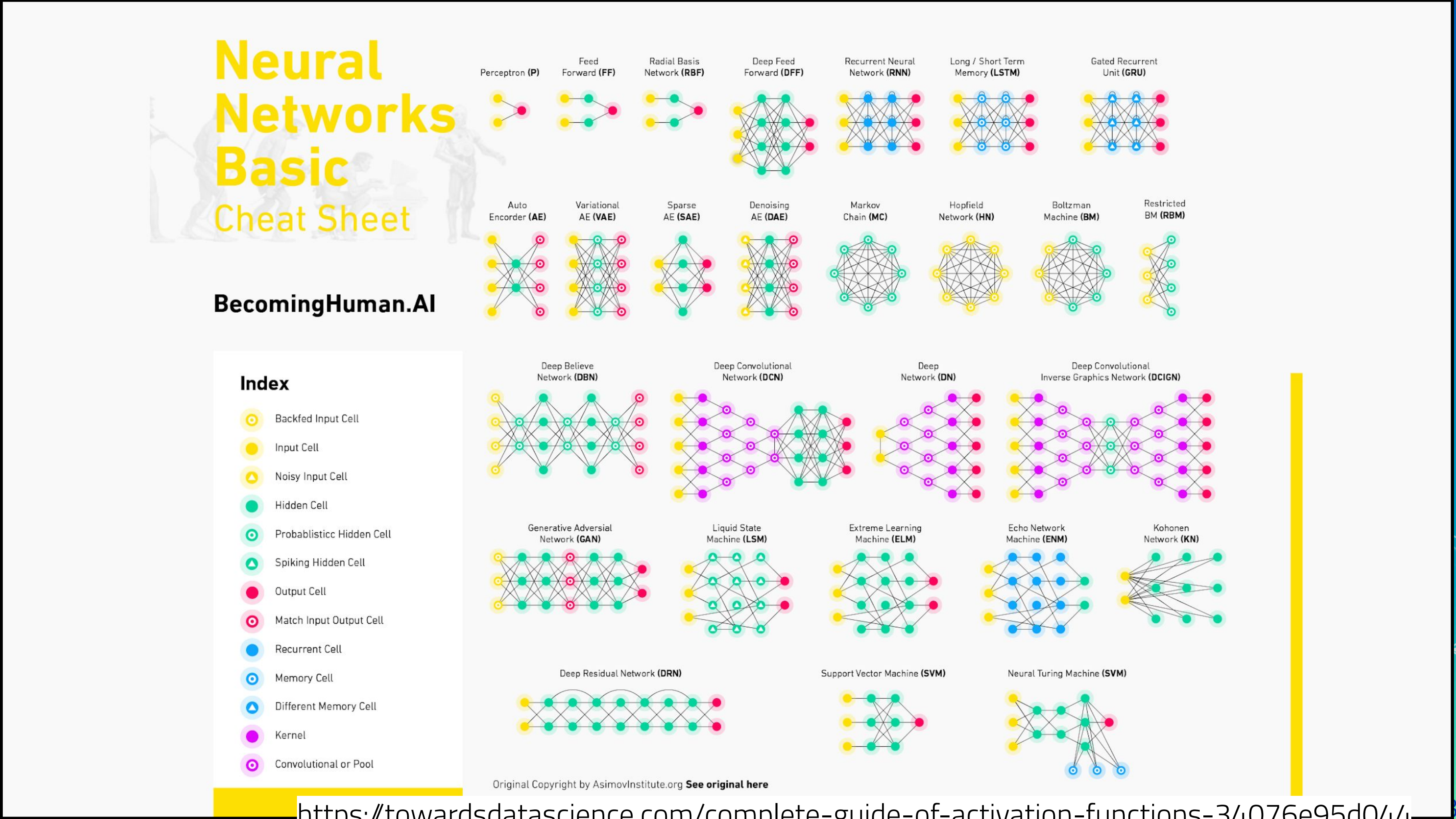
$a = 0.858529$

Overfitting and Regularization

$$C = C_0 + \frac{\lambda}{2|w|} \sum_w w^2$$

$$w \rightarrow w - \eta \frac{\partial C}{\partial w} - \eta \frac{\lambda}{|w|} w$$

$$b \rightarrow w - \eta \frac{\partial C}{\partial b}$$



https://towardsdatascience.com/complete-guide-of-activation-functions-34076e95d044

Feedforward Neural Networks

—Questions?—

Feedforward Neural Networks

—Thank you!—

http://neuralnetworksanddeeplearning.com/chap3http://neuralnetworksanddeeplearning.com/index.html.html

https://rohanvarma.me/Loss-Functions/

https://www.desmos.com/

https://towardsdatascience.com/how-to-build-your-own-neural-network-from-scratch-in-python-68998a08e4f6

https://iamtrask.github.io/2015/07/12/basic-python-network/

—Sources—