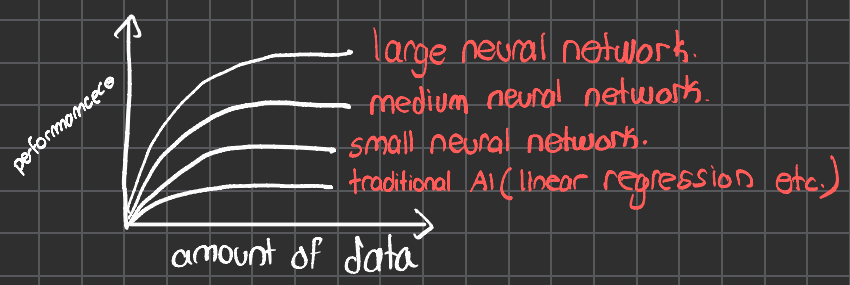
Neural Networks: → Origins: Algorithms that try to mimic the brain.



Demand Prediction:

$x \to$ price = input

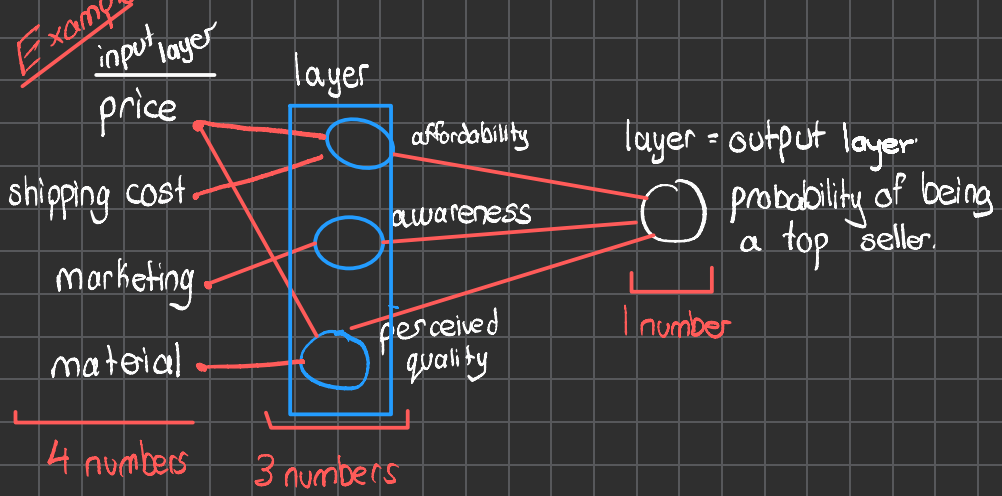
$$a = f(x) = \frac{1}{1+e^{-(wx+b)}} = \text{output}$$

a ↓ activation

$x \to \boxed{\bigcirc} \to a$

price → probability.

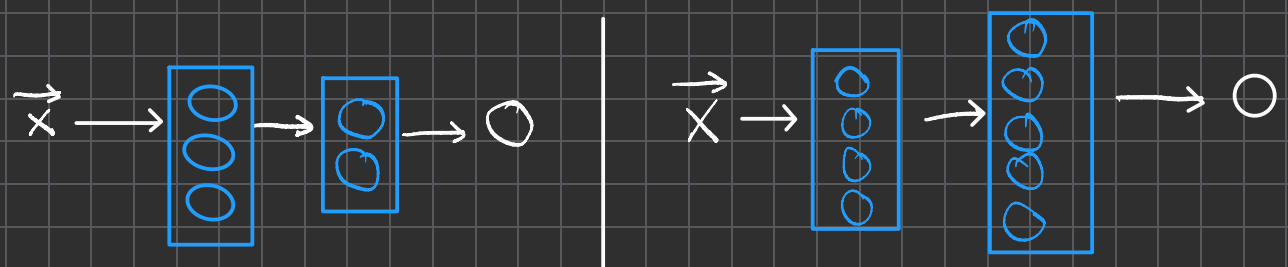
Example



$(\vec{x})$ input layer → $(\vec{a})$ hidden layer → (a) output layer

→ Each neuron in a layer is called an activation.

Multiple Hidden Layer: Neural network architecture.



Note: Multi-Layer NN are also called multilayer perceptrons

Made with Goodnotes

Neural Network Layer:

$$\begin{bmatrix} 197 \\ 184 \\ 136 \\ 214 \end{bmatrix} \xrightarrow{\vec{x}}$$

layer 0.

$w_1^{[1]}, b_1^{[1]} \rightarrow a_1^{[1]} = g(\vec{w}_1^{[1]} \cdot x + b_1^{[1]})$ 0.3

$W_2^{[1]}, b_2^{[1]} \rightarrow a_2^{[1]} = g(\vec{w}_2^{[1]} \cdot x + b_2^{[1]})$ 0.7

$W_3^{[1]}, b_3^{[1]} \rightarrow a_3^{[1]} = g(\vec{w}_3^{[1]} \cdot x + b_3^{[1]})$ 0.2

Layer 1

$$\xrightarrow{\vec{a}^{[1]}} \begin{bmatrix} 0.3 \\ 0.7 \\ 0.2 \end{bmatrix}$$

Layer 2

output from layer 1. becomes input of Layer 2.
And so on until we reach the output layer.

$$\begin{bmatrix} 0.3 \\ 0.7 \\ 0.2 \end{bmatrix} \xrightarrow{\vec{a}^{[1]}} \bigcirc \xrightarrow{a^{[2]}} 0.84 \text{ (scalar val)}$$

layer 1

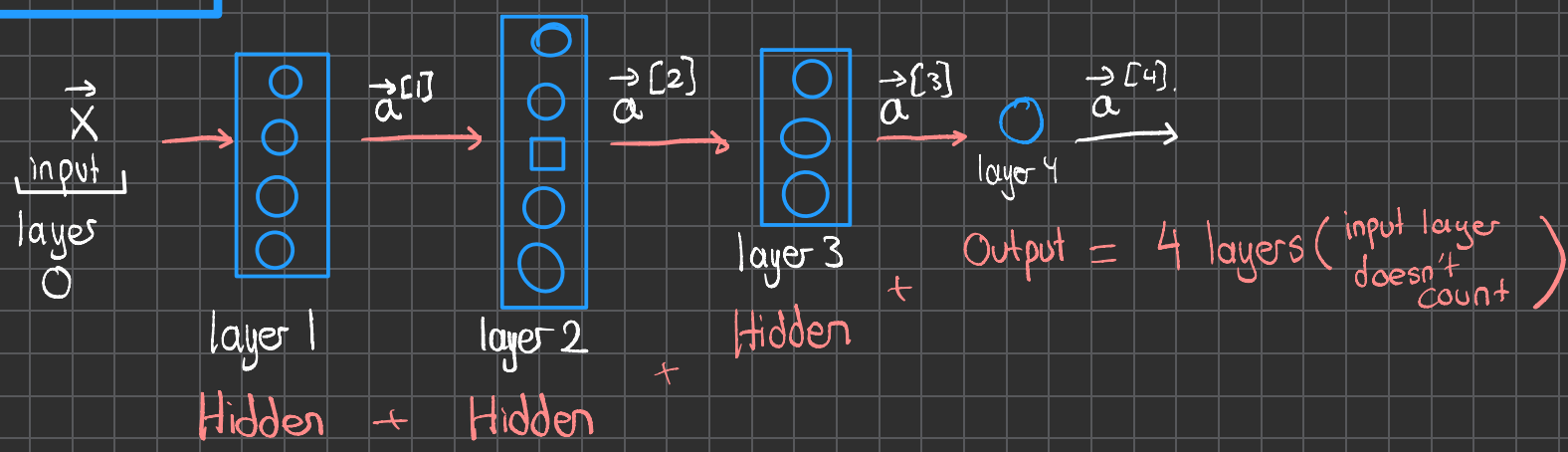
Layer 2.

$a^{[2]} \geq 0.5$?

yes: $\hat{y} = 1$

no: $\hat{y} = 0$

Cont....



Output = 4 layers (input layer doesn't count)

$$\boxed{a_j^{[l]} = g(\vec{w}_j^{[l]} \cdot \vec{a}^{[l-1]} + b_j^{[l]})}$$

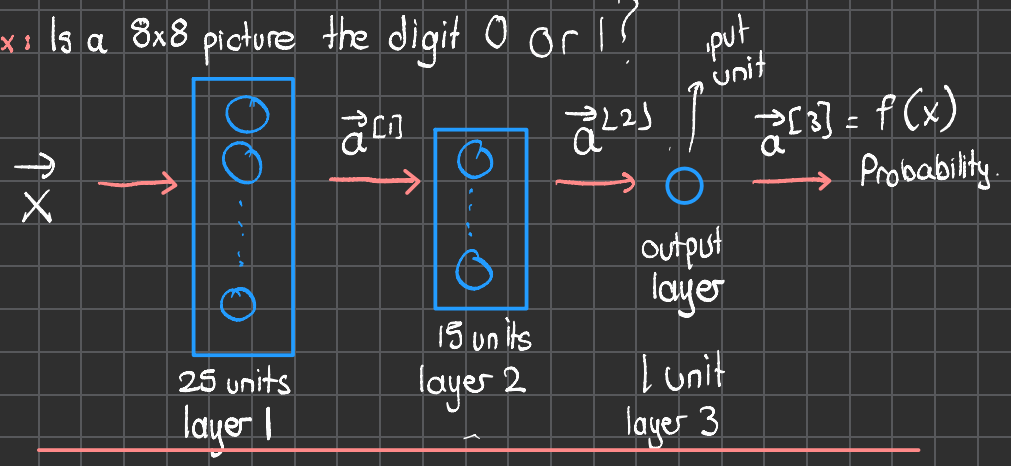
Notation of parameters w, b in layer l, unit j

g → activation function.

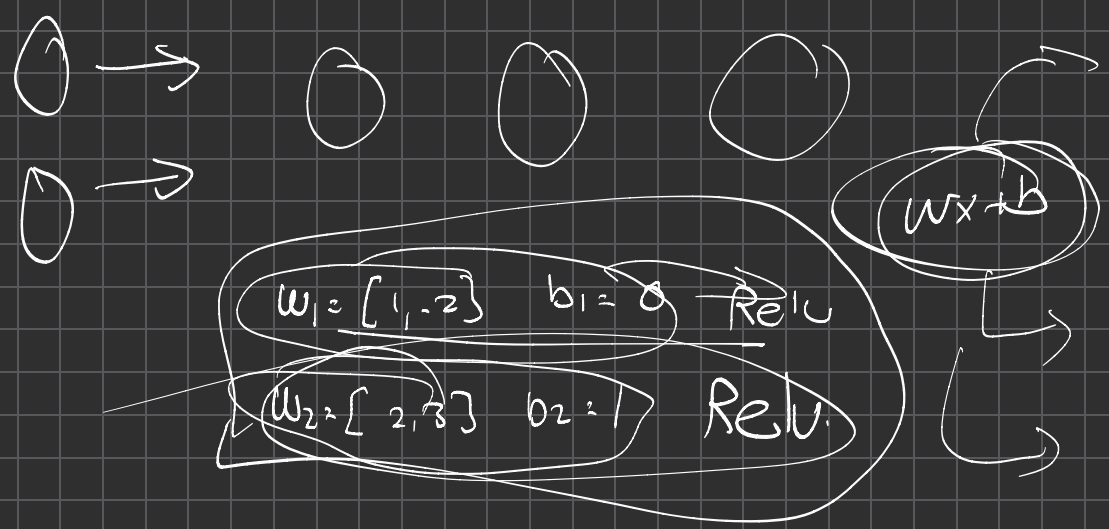
Made with Goodnotes

Inference: Make prediction.

Ex: Is a 8x8 picture the digit 0 or 1?



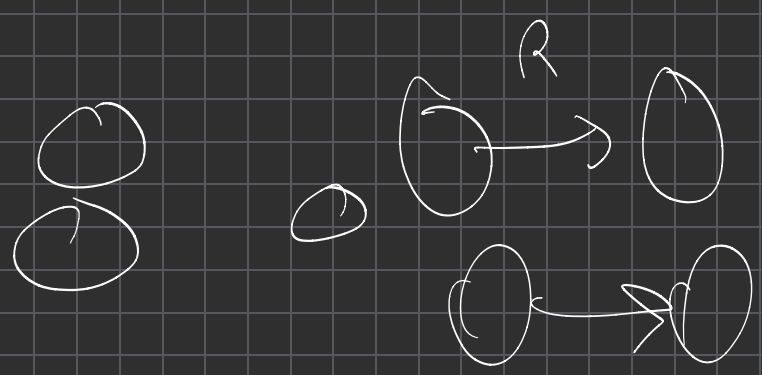
FORWARD PROROGATION



$N_1 = 1 \times 1 + (-2 \times 2) + 0 = -3$ Relu $\rightarrow 0$

$N_2 = 2 \times 1 + (3 \times 2) + 1 = 7$ Relu $\rightarrow 7$

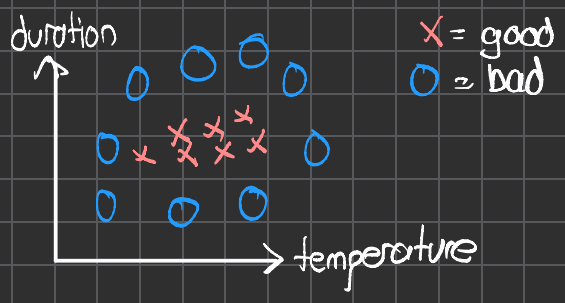
relu = (0, x)



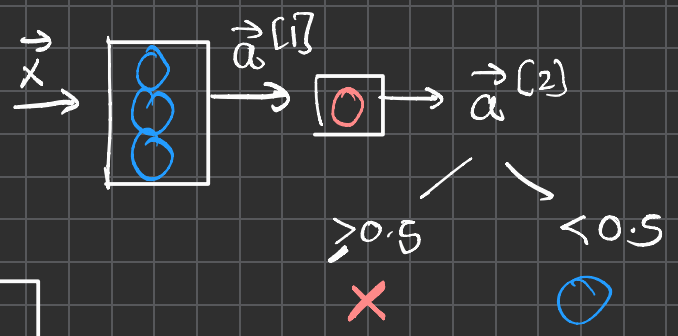
Made with Goodnotes

Inference:

Ex. Coffee roasting.



2 features = $\begin{bmatrix} \text{duration} \\ \text{temp} \end{bmatrix} = \begin{bmatrix} 200 \\ 17 \end{bmatrix} = x$




```
x = np.array([[200, 17.0]])
layer_1 = Dense(units = 3, activation = 'sigmoid')
a_1 = layer_1(x)

layer2 = Dense(units = 1, activation = 'sigmoid')
a_2 = layer2(a_1)
```

(tensorflow)

Building a Neural Network:

```
layer_1 = Dense(units = 3, activation = 'sigmoid')
layer_2 = Dense(units = 1, activation = 'sigmoid')
model = Sequential([ layer_1 , layer_2 ])
x = np.array([[200,4],[100,2],[300,5],[150,4]])
y = np.array([1,0,0,1])
model.compile(....)
model.fit(x,y)
model.predict(x_new)
```

Made with Goodnotes

		1	10	2	8	3, 4
time	left	sent	total	dropped	delivered

Made with Goodnotes