★ ML Theory

What is a model?

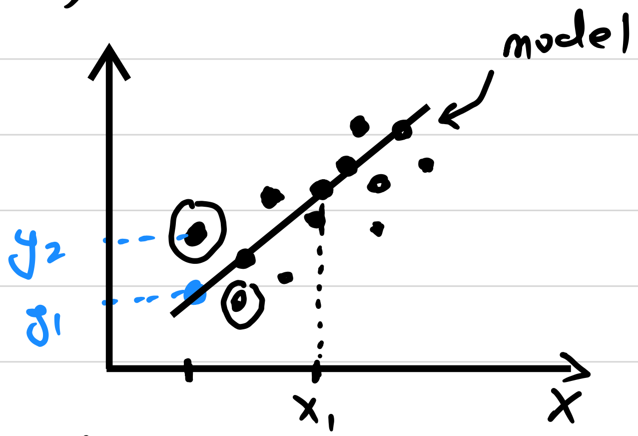
$$f(x, \theta)$$

Predict ice cream sales using temp.

$x \rightarrow$ inputs
$\theta \rightarrow$ parameters

$$y = mx + c \longleftrightarrow f = wx + b$$

$$f_\theta(x) = wx + b \qquad \underline{\underline{(\theta \sim (w, b))}}$$



$$\underline{|y_2 - y_1|}$$

Cost f^n $\rightarrow$ tells us how well our model is doing.

Based on this cost f^n we tune w & b to make model better

$$\boxed{J(w, b) = \boxed{\frac{1}{2m}} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2}$$

How do we make model better?
(Grad. Descent).

We minimize loss using iterations.
→ start with initial w, b
→ keep changing w, b to reduce $J(w, b)$ on each iteration.

<u>Algo</u>

→ w

$$w = w - \boxed{\alpha} \frac{\partial}{\partial w} J(w, b)$$

$f(x, y) = x^2 + 3y^2 + xy$

$\frac{\partial f}{\partial x} =$

$J(w, b)$

$$b = b - \alpha \frac{\partial}{\partial b} J(w, b)$$

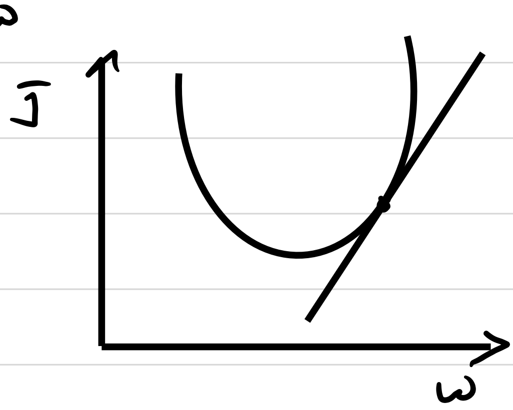
$w \rightarrow (w, b)$

$b \rightarrow J(w, b)$
model

$\alpha \rightarrow$ learning rate

<u>Case 1</u> $\frac{\partial}{\partial w} \rightarrow +ve$

$\frac{\partial(J)}{\partial w} \rightarrow +ve$



+ve slope
i.e.

w should decrease

<u>Case 2</u> $\frac{\partial}{\partial w} \rightarrow -ve$

w should increase.

Now, for multiple features $x_1, x_2, x_3, \ldots$

$\vec{w} = [w_1, w_2, w_3, \ldots]$

$\vec{x} = [x_1, x_2, x_3, \ldots]$

b = number.

$x_1 \to$ ear shape
$x_2 \to$ tail
$x_3 \to$ stripes.

$\boxed{\vec{x} = [x_1, x_2, x_3]}$

$\boxed{\vec{w} = [w_1, w_2, w_3]}$

$\boxed{\vec{w} \cdot \vec{x} = w_1x_1 + w_2x_2 + w_3x_3}$

$$\therefore \boxed{w_j = w_j \underbrace{- \alpha \frac{1}{m} \sum \left(f(\vec{x}^{(i)}) - y^{(i)} \right) x_j^{(i)}}_{-\alpha \frac{1}{m} \sum \frac{\partial}{\partial w_j}(J,w)}}$$

$x_j \to j^{th}$ feature.

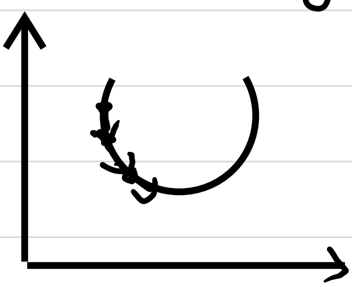
n: no. of features

$x^{(i)} \to$ features of i^{th} training example.

$\Rightarrow x^{(2)} = [x_1^{(2)}, x_2^{(2)}, x_3^{(2)}]$

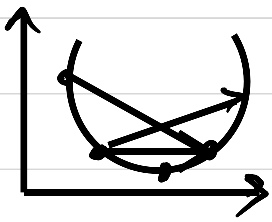
Now, Learning Rate $\alpha \to$ b/w 0 to 1

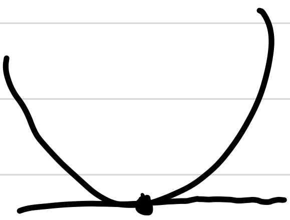
C1 $\alpha \ll$



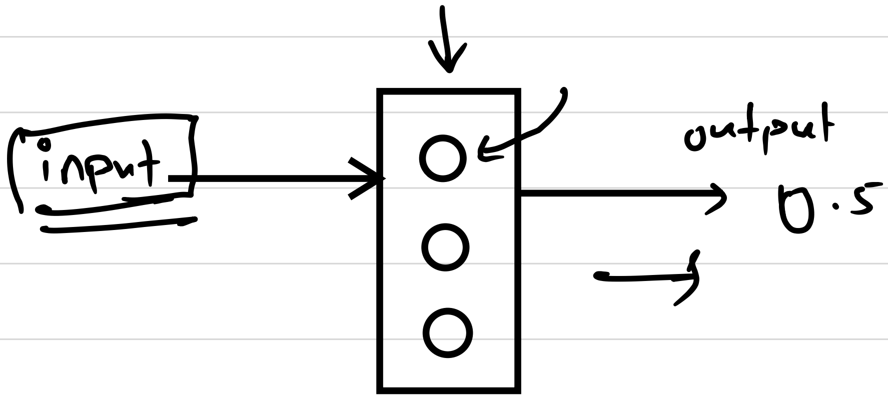
$\hat{y} - y_i = 0$

C2 $\alpha \gg$





★ Neural Networks



computation

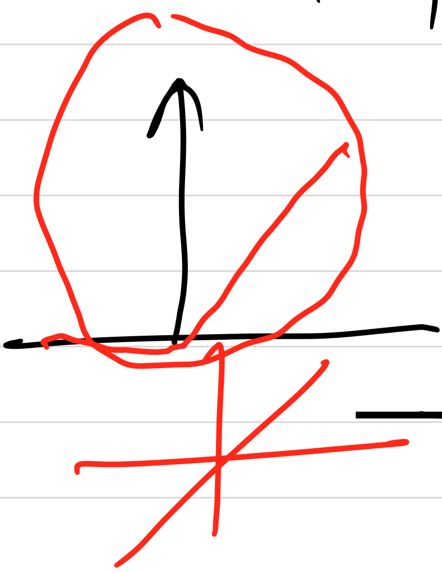
$$z = w \cdot x + b$$

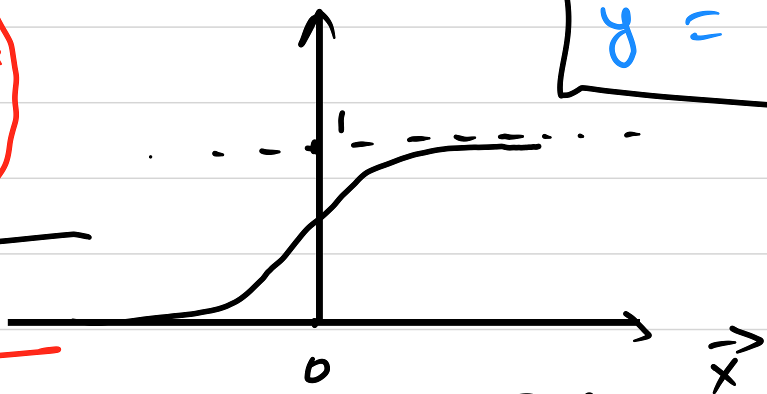
$$f(z) = \frac{1}{1+e^{-(z)}}$$

$\underline{\underline{a}} \rightarrow$ activation (for output)

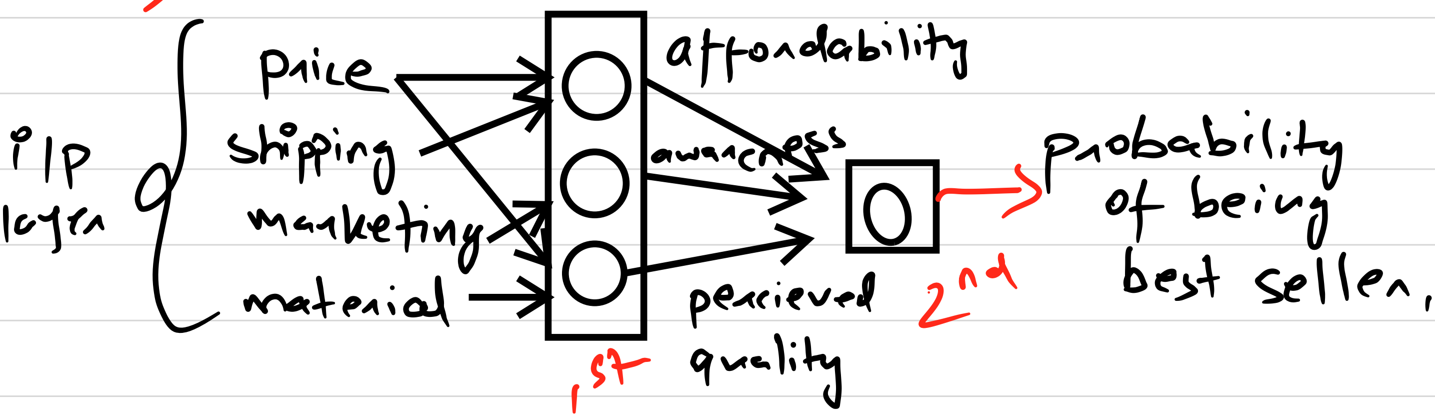
ex: $a = f(z) = \frac{1}{1+e^{-(wx+b)}}$ → sigmoid

$$\boxed{y = \max(0, x)}$$





a = price, shipp



each neuron will have access to each feature.

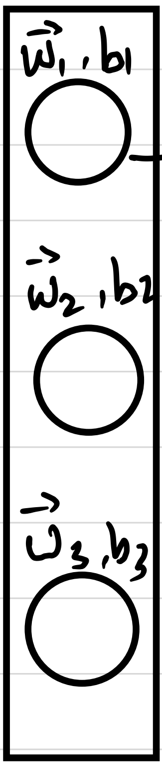
Model learns to ignore marketing & material for affordability using weights.

ex: $\boxed{z} = w_a\,\text{price} + w_b\,\text{shipping} + w_c\,\text{marketing} + w_d\,\text{material}$

then for affordability,

$\boxed{w_c \approx 0}$

$w_d \ll w_a \,\&\, w_b$



$a_1 = g(\vec{w}_1 \cdot \vec{x} + b_1)$ max(0, z)

$a_2 = g(\vec{w}_2 \cdot \vec{x} + b_2)$

$a_3 = g(\vec{w}_3 \cdot \vec{x} + b_3)$

$\vec{w}_j = [w_a, w_b, w_c, w_d]$

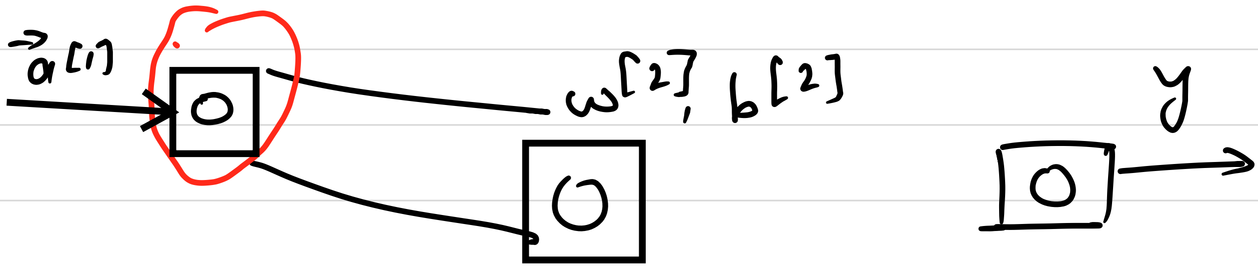
$a^{[1]} = [a_1^{[1]}, a_2^{[1]}, a_3^{[1]}]$

$a_1^{[1]}$

$a_{neuron}^{[layer]}$

output of first layer.
i/p for second layer. $= [a_{neuron}^{[layer]}]$

$a^{[layer]}$



$$a^{[2]} = g(\vec{w}^{[2]} \cdot \vec{a}^{[1]} + b^{[2]})$$

then we can classify

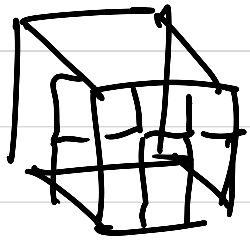
$a^{[2]} \geq 0.5$?

No: $\hat{y} = 0$

Yes: $\hat{y} = 1$

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

$$\begin{bmatrix} [2 & 3] \\ [1 & 4] \end{bmatrix}$$



$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$