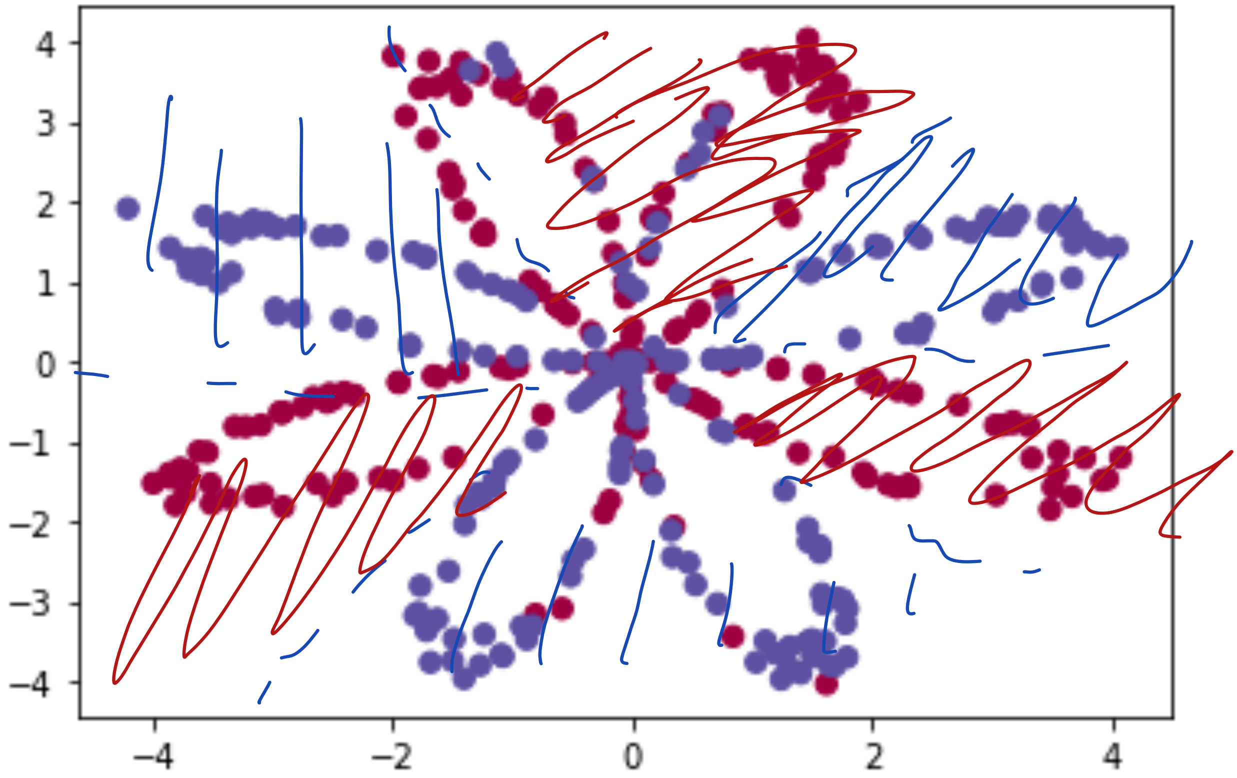


Dataset ↑ Want to classify regions as RED (0) or BLUE (1).

∴

$$X = n_x \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix}$$

$\longleftarrow m \longrightarrow$

$X \langle n_x, m \rangle$. where $n_x = 2$, $x_1^{(i)}$, $x_2^{(i)}$

↓ x-axis ↓ y-axis.

$Y = [\quad]$ $\langle 1, m \rangle$

$\in m \rightarrow$

0 or 1 for input red or blue

One example $m = 400$ $X: \langle 2, 400 \rangle$ $Y: \langle 1, 400 \rangle$

Lin model not good b/c data not lin.

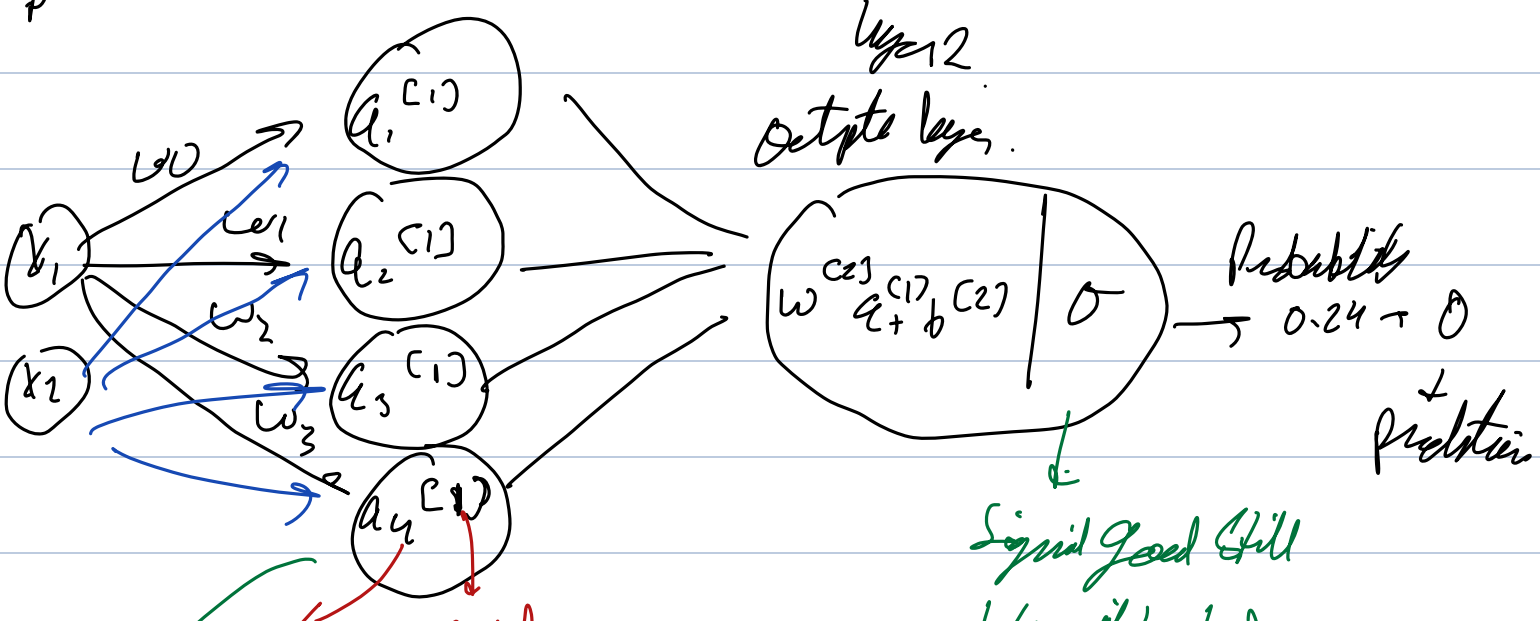
Need NON-LINEAR ACTIVATION FXN

Neural Network model (1-hidden layer).

input layer = 0

input → hidden ∴ 2 layer total

layer 1. layer



Sigmoid good still b/c it's being classification problem.

non input layer

neuron. [layer] (example)

a neuron #.

$g^{[1]}$ activation fxn non-lin

So for one example. $X^{(i)} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ $A^{(i)} = \begin{bmatrix} a_1^{(i)} \\ a_2^{(i)} \\ a_3^{(i)} \\ a_4^{(i)} \end{bmatrix}$

$Z^{(i)} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \end{bmatrix}$.

$\Rightarrow X = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(m)} \\ x_2^{(1)} & x_2^{(2)} & & x_2^{(m)} \end{bmatrix} \langle 2, m \rangle$

$A^{[1]} = \begin{bmatrix} a_1^{[1](1)} & & a_1^{[1](m)} \\ a_2^{[1](1)} & \cdots & a_2^{[1](m)} \\ a_3^{[1](1)} & & a_3^{[1](m)} \\ a_4^{[1](1)} & & a_4^{[1](m)} \end{bmatrix}$ $\langle 4, m \rangle$

A bcc could be more layers

Z is same shape.

$Z^{[1]} = W^{[1]} X + b^{[1]}$

So if: $Z^{[1]}$: $\langle 4, m \rangle$, $W^{[1]}$: $\langle 4, 2 \rangle$, X: $\langle 2, m \rangle$, $b^{[1]}$: $\langle 4, m \rangle$

$\langle 4, 2 \rangle$: n_h → output layer neurons; n_x → input layer neurons

$W = \langle$ new neurons, old rows $\rangle$

Just 1 example (i)

$Z^{[1](i)} = W^{[1](i)} X^{(i)} + b^{[1]}$ ← Same for all m → So right now $\langle 4, 1 \rangle$

$\langle 4, 1 \rangle$

$A^{[1](i)} = \tanh\left(Z^{[1](i)}\right)$

$Z^{[2](i)} = W^{[2](i)} A^{[1](i)} + b^{[2]}$

$$\hat{y} = A^{[2](i)} = \sigma(z^{[2](i)})$$

if $\hat{y}^{n(i)} = \begin{cases} 1 & \text{if } a^{[2](i)} > 0.5 \\ 0 & \text{else} \end{cases}$

We know friendly Cost is

$$J = -\frac{1}{m} \sum_{i=0}^{m} \left(y^{(i)} \log(a^{[2](i)}) + (1-y^{(i)}) \log(1-a^{[2](i)}) \right)$$

↑ Cost.

And we know derivatives.

General Methodology:

1) Define NN structure
2) Initialize params (→ NOT ZERO).
3) Loop
   a) Forward prop.
   b) loss calculation
   c) backward prop.
   d) update parameters (grad. descent).

TODO:

⇒ Review the MATH behind the derivative Hens solves.