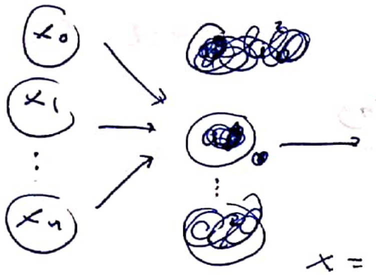
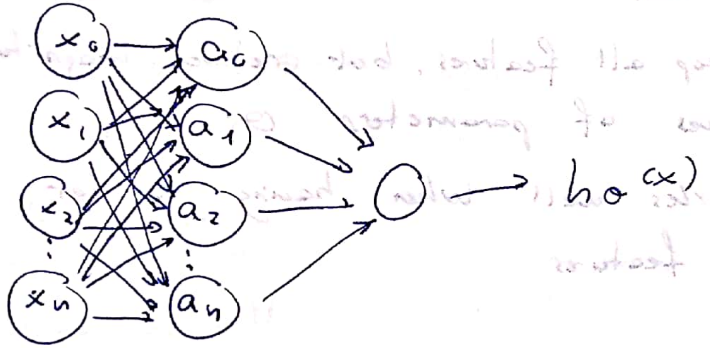
NEURAL NETWORKS

• Logistic unit



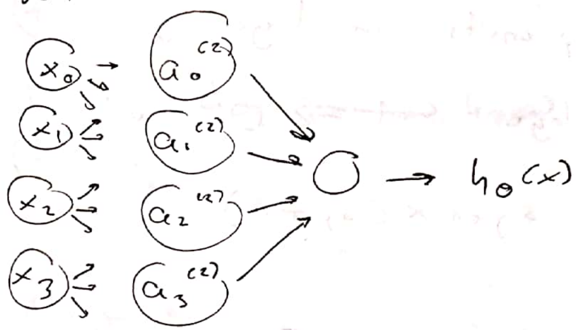
$$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}$$



pág 20

NN



$a_i^{(j)}$ = activation of unit i in layer j

$\theta^{(j)}$ = matrix of weights controlling function mapping from layer j to layer $j+1$

$$a_1^2 = g(\theta_{10}^1 x_0 + \theta_{11}^1 x_1 + \theta_{12}^1 x_2 + \theta_{13}^1 x_3)$$

$$a_2^2 = g(\theta_{20}^1 x_0 + \theta_{21}^1 x_1 + \theta_{22}^1 x_2 + \theta_{23}^1 x_3)$$

$$a_3^2 = g(\theta_{30}^1 x_0 + \theta_{31}^1 x_1 + \theta_{32}^1 x_2 + \theta_{33}^1 x_3)$$

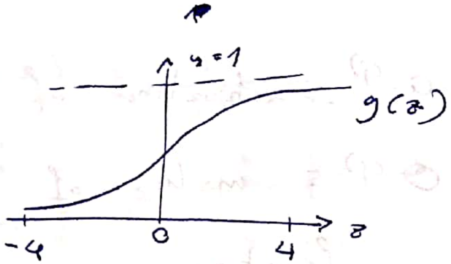
$$h_\theta(x) = a_1^3 = g(\theta_{10}^2 a_0^2 + a_{11}^2 a_1^2 + \theta_{12}^2 a_2^2 + \theta_{13}^2 a_3^2)$$

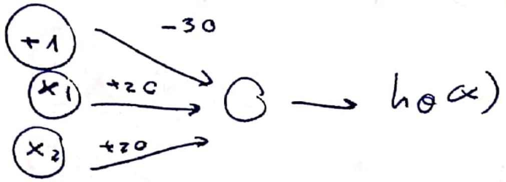
If network has s_j units in layer j, s_{j+1} units in layer $j+1$ $\Rightarrow$ Θ^{j} will be of dimension $s_{j+1} \times (s_j + 1)$

Simple example: AND

$x_1, x_2 \in \{0, 1\}$

$y = x_1$ AND x_2



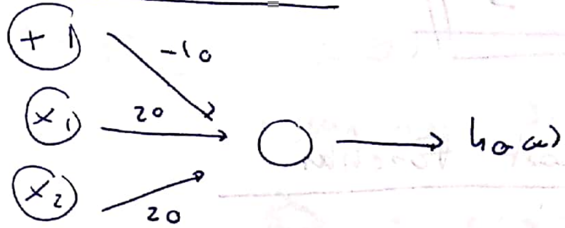


$$h_\Theta(x) = g(\underbrace{-30}_{\Theta^1_{10}} + \underbrace{20}_{\Theta^1_{11}} x_1 + \underbrace{20}_{\Theta^1_{12}} x_2)$$

x_1	x_2	$h_\Theta(x)$
0	0	$g(-30) \approx 0$
0	1	$g(-10) \approx 0$
1	0	$g(-10) \approx 0$
1	1	$g(10) \approx 1$

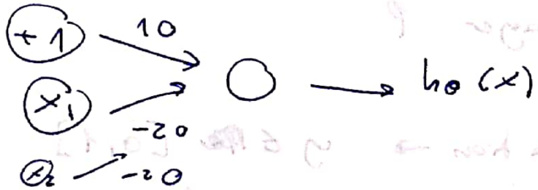
$h_\Theta(x) \approx x_1$ and x_2

OR function



x_1	x_2	$h_\Theta(x)$
0	0	$g(-10) \approx 0$
0	1	$g(10) \approx 1$
1	0	≈ 1
1	1	≈ 1

Negation



x_1	$h_\Theta(x)$
0	$g(10) \approx 1$
1	$g(-10) \approx 0$

$h_\Theta(x) = g(10 - 20x_1)$

NOT x_1 AND NOT x_2

$= 1 \iff x_1, x_2 = 0$

" I need more info about this "

WEEK 5

Neural Network Cost Function

L = nº of layers in neural network

s_l = nº of units (without counting bias unit) in layer l

- Binary classification → $y \in [0,1]$
- Multiclass class → $y \in \mathbb{R}^K$

Logistic regression $\}$ $J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log(1-h_\theta(x^{(i)}))\right] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2$

$h_\theta(x) \in \mathbb{R}^K$; $(h_\theta(x))_i$ = ith output

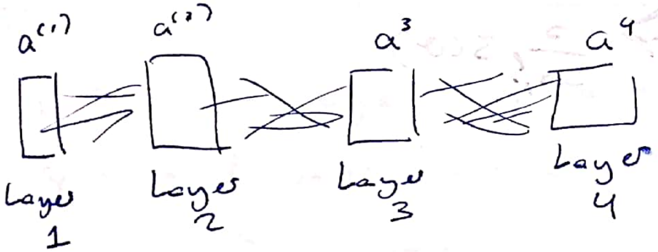
Neural Network $\}$ $J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log(h_\theta(x^{(i)}))_k + (1-y_k^{(i)}) \log(1-h_\theta(x^{(i)}))_k\right] + \frac{\lambda}{2m}\sum_{l}^{L-1}\sum_{i}^{s_l}\sum_{j=1}^{s_l+1}(\theta_{ji}^{(l)})^2$

- steps
  - $J(\theta)$
  - Minimize $J(\theta)$

$\Rightarrow \frac{\partial}{\partial \theta_{ij}^{(l)}} J(\theta)$

• Gradient computation

$(x, y) \equiv$ training example



$$\begin{cases} a^1 = x \\ z^2 = \Theta^1 a^1 \\ a^2 = g(z^2) \quad \text{add } a_0^2 \end{cases}$$

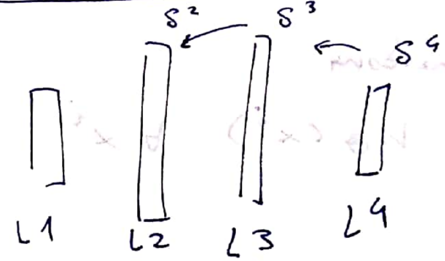
$$\boxed{\begin{aligned} z^l &= \Theta^{l-1} a^{l-1} \\ a^l &= g(z^l) \end{aligned}}$$

$$\begin{cases} z^3 = \Theta^2 a^2 \\ a^3 = g(z^3) \quad \text{add } a_0^3 \end{cases}$$

$$\begin{cases} z^4 = \Theta^3 a^3 \\ a^4 = h_\Theta(x) = g(z^4) \end{cases}$$

• Backpropagation

$$\boxed{\delta_j^l = a_j^l - y_j}$$



$\delta_j^4 = a_j^4 - y_j$

$\delta^3 = (\Theta^3)^T \delta^4 .\, g'(z^3)$; $g'(z^l) = a^l(1 - a^l)$

$\delta^2 = (\Theta^2)^T \delta^3 .\, g'(z^2)$

• Pseudo algorithm

- Training set $\rightarrow \{(x^1, y^1) \dots (x^m, y^m)\}$
- $\Delta_{ij}^l = 0 \quad \forall\, i, j, l$
- For $i = 1, m$

  $a^1 = x^i$

  Foward prop for a^l, $l = 2, 3 \dots L$

  Compute $\delta^L = a^L - y^i$

  Compute $\delta^{L-1}, \delta^{L-2} \dots \delta^2$

  $\Delta_{ij}^l := \Delta_{ij}^l + \underbrace{a_j^l \delta_i^{l+1}} \rightarrow \delta_i^{l+1} a_j^{l\,T}$

  $D_{ij}^l := \frac{1}{m} \Delta_{ij}^l + \lambda \Theta_{ij}^l$ if $j \neq 0$

  $D_{ij} := \frac{1}{m} \Delta_{ij}^l$ if $j = 0$

$\left\|\ \frac{\partial}{\partial \Theta_{ij}^l} J(\Theta) = D_{ij}^l \right.$

Training a Neural Network

1 - Initialize weights random

2 - Foward prop to get $h_\Theta(x^i)$ $\forall x^i$

3 - Cost function $J(\Theta)$

4 - Backprop to get $\frac{\partial}{\partial_{jk}^{l}} J(\Theta)$

5 - Gradient checking to compare this with a numerical estimation of $\vec{\nabla} J(\Theta)$

6 - Disable 5

7 - Gradient descent or other algorithm ~~t~~with backprop to minimize $J(\Theta)$

Scanned with CamScanner