Cost Function

Neural Network (Classification)

L: total no. of layers in network

s_l: no. of units in layer l

K: no. of output units/classes

Binary	Multi-class (K classes)
$y = 0$ or 1	$y \in \mathbb{R}^K$
1 output unit ($S_L = 1$) ($K = 1$)	K output units ($S_L = K$) ($K \geq 3$)

Cost f of NN:

$$h_\Theta(x) \in \mathbb{R}^K \quad (h_\Theta(x))_i = i^{th} \text{ output}$$

$$J(\Theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{k=1}^{K} y_k^{(i)} \log\left(h_\Theta(x^{(i)})\right)_k + \left(1-y^{(i)}\right)\log\left(1-h_\Theta(x^{(i)})_k\right)\right]$$

$$+ \frac{\lambda}{2m}\sum_{l=1}^{L-1}\sum_{i=1}^{s_l}\sum_{j=1}^{s_{l+1}}\left(\Theta_{ji}^{(l)}\right)^2$$

→ regularization term

→ loops through the no. of output nodes

Backpropagation Algorithm 역전파 알고리즘

$\min_{\Theta} J(\Theta)$

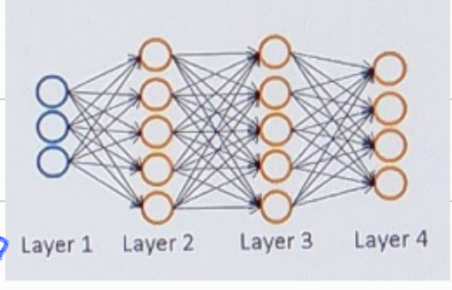
비용 최소화하는 매개변수(가중치, θ, w) 찾기 위해

① $J(\Theta)$

② $\frac{\partial}{\partial \Theta_{ij}^{(l)}} J(\Theta)$ — real number

계산해야 함 (gradient descent 등 사용) 편미분 대상

Gradient Computation 경사 (가중) 계산



Given one training ex. (x, y):

Forward propagation:

$a^{(1)} = x$ (input)

$z^{(2)} = \Theta^{(1)} a^{(1)}$

$a^{(2)} = g(z^{(2)})$ (add $a_0^{(2)}$)

$z^{(3)} = \Theta^{(2)} a^{(2)}$

$a^{(3)} = g(z^{(3)})$ (add $a_0^{(3)}$)

$z^{(4)} = \Theta^{(3)} a^{(3)}$

$a^{(4)} = h_\Theta(x) = g(z^{(4)})$

벡터화되어 구현된 순방향 전파 알고리즘

- 신경망 내 모든 뉴런의 활성화값을 계산하는 것을 가능하게

Backpropagation algorithm

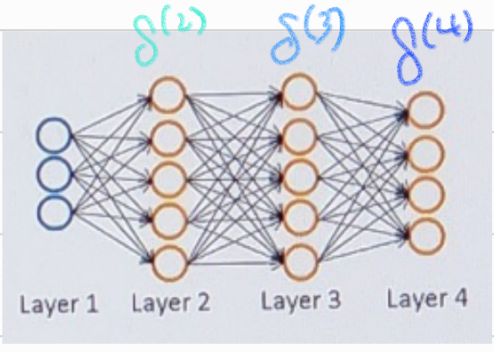
Intuition: $\delta_j^{(l)}$ = "error" of node j in layer l

delta

For each output unit ($L=4$)

$$\delta_j^{(4)} = a_j^{(4)} - y_j$$

activation, label



vectorize

$$\delta^{(4)} = a^{(4)} - y$$

$$\delta^{(3)} = (\Theta^{(3)})^T \delta^{(4)} \cdot g'(z^{(3)})$$

$$\delta^{(2)} = (\Theta^{(2)})^T \delta^{(3)} \cdot g'(z^{(2)})$$

element-wise

모든 parameters에 대한 partial derivatives 편미분 계산 방법

i.g. Training set $\{(x^{(1)}, y^{(1)}), \cdots, (x^{(m)}, y^{(m)})\}$

Set $\Delta_{ij}^{(l)} = 0$ (for all l, i, j)

For $i=1$ to m

Set $a^{(1)} = x^{(i)}$

Perform forward propagation to compute $a^{(l)}$ for $l = 1, 2, \cdots, L$

Using $y^{(i)}$, compute $\delta^{(L)} = a^{(L)} - y^{(i)}$

Compute $\delta^{(L-1)}, \delta^{(L-2)}, \cdots \delta^{(2)}$

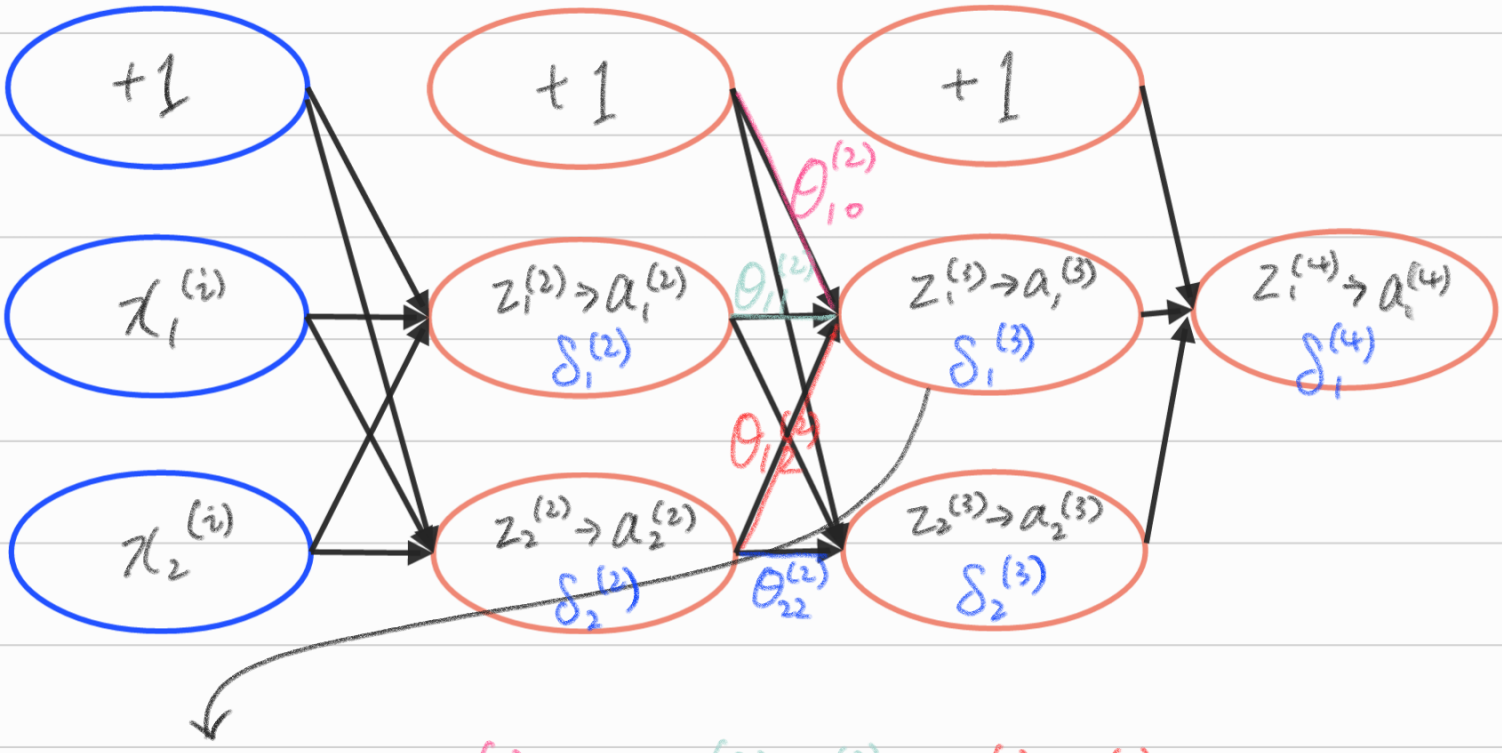
$\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$

$$D_{ij}^{(l)} := \frac{1}{m}\left(\Delta_{ij}^{(l)} + \lambda\theta_{ij}^{(l)}\right), \text{ if } j \neq 0$$
$$:= \frac{1}{m}\Delta_{ij}^{(l)}, \text{ if } j = 0$$

"accumulator" to add up our values as we go along and eventually compute our 편미분

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

Forward Propagation



$$z_1^{(3)} = \theta_{10}^{(2)} \times 1 + \theta_{11}^{(2)} \times a_1^{(2)} + \theta_{12}^{(2)} \times a_2^{(2)}$$

$$\delta_2^{(2)} = \theta_{12}^{(2)}\,\delta_1^{(3)} + \theta_{22}^{(2)}\,\delta_2^{(3)}$$