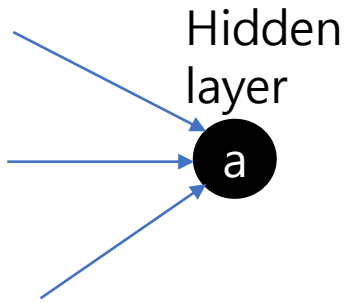
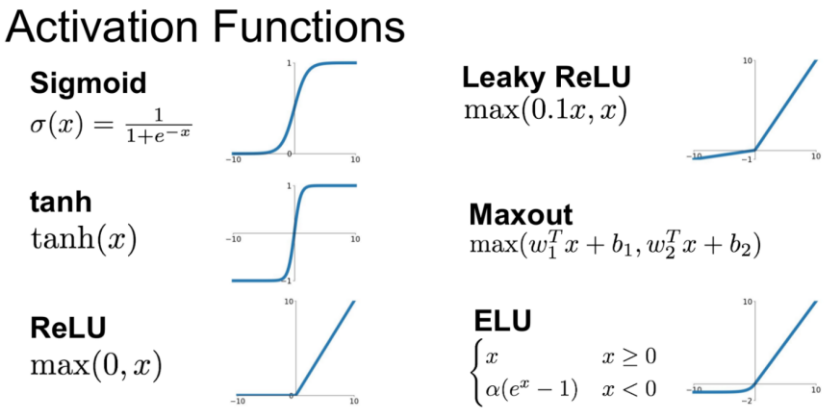
Activation Function



- Activation function
- $a = g(w^T x + b)$, $\mathrm{z} = \mathrm{w}^{\mathrm{T}}\mathrm{x} + b$

- Sigmoid : $\mathrm{a} = \frac{1}{1+e^{-z}}$
- Tanh : $\mathrm{a} = \frac{sinhz}{coshz}$ -> mean value ~ 0
- ReLU : $\mathrm{a} = \max(0, \mathrm{z})$



Why Nonlinear Activation function

- Linear activation : matrix multiplication
- Output depends on only input. -> No meaning of hidden layers

- Assume $g(z) = z$
- $a^{[1]} = w_1^T x + b_1$, $a^{[2]} = w_2^T a^{[1]} + b_2$
- $a^{[2]} = w_2^T w_1^T x + w_2^T b_1 + b_2$

- Linear activation is used in regression problem (in output layer)

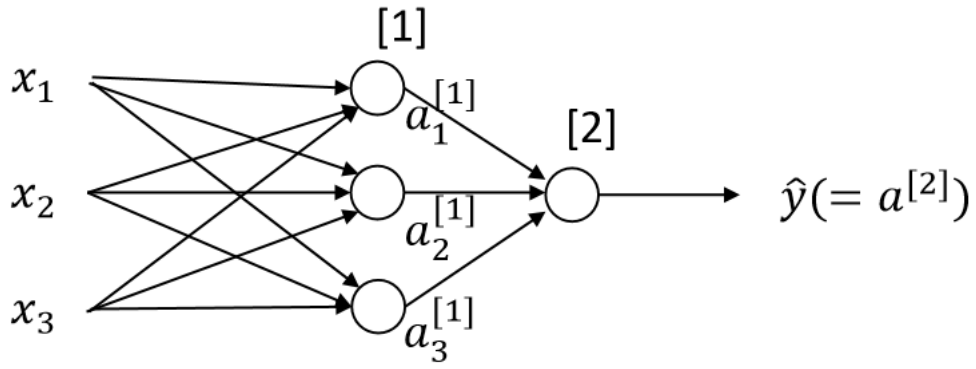
Derivatives of activation functions

- Pass

- Derivatives is used in gradient descent for neural network.

Gradient descent

- Neural network



- Back propagation to give feedback to the neural network

- Parameters : w^1, w^2, b^1, b^2
- Cost function(loss function) : $\mathrm{J}(w^1, w^2, b^1, b^2) = 1/m \sum L(\hat{y}, y)$

- Parameter random initialization is important.

Gradient descent

$z^1 = W^1 x + b^1$	$a^1 = \sigma(z^1)$	$z^2 = W^2 a^1 + b^2$	$a^2 = \sigma(z^2)$	$L(a^2, y)$

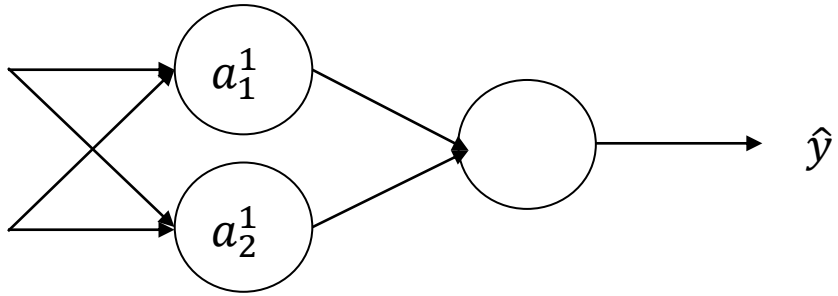
- Primary input: x, W1, b1
- $w^i = w^i - \alpha \frac{\partial J}{\partial w^i}$
- $b^i = b^i - \alpha \frac{\partial J}{\partial b^i}$
- α is learning rate.

- Assume Loss function : $-\mathrm{ylog}\hat{y} - (1-y)\mathrm{log}(1-\hat{y})$

Gradient descent

- $dz^2 = a^2 - y$
- $dW^2 = dz^2 a^{1T}$
- $db^2 = dz^2$, $db^1 = dz^1$
- $dz^1 = W^{2T} dz^2 * g^{1\prime}(z^1)$
- $dW^1 = dz^1 x^T$

Weight initialization



- If we initialize the weight 0,…

- $W^1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, b^1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$
- Make $a_1^1 = a_2^1$ and $dz_1^1 = dz_2^1$ -> Full symmetric, a hidden layer's nodes always have same value.

- Random generation is good. But give small value.