Data Mining & Machine Learning

CS37300
Purdue University

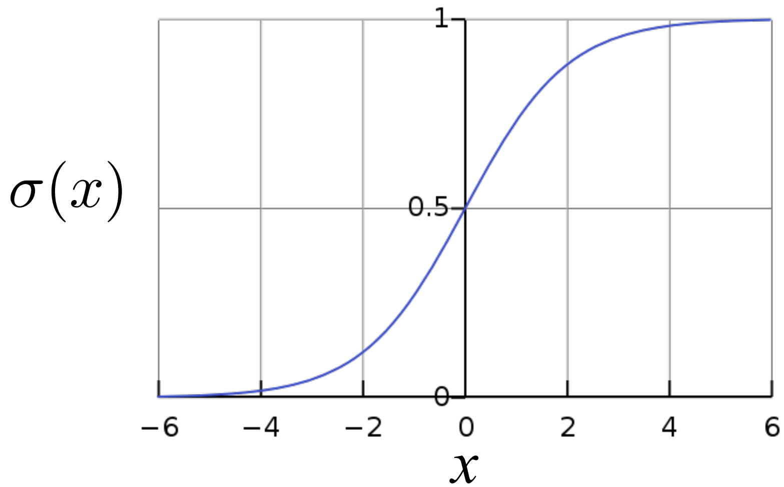
Oct 23, 2023

Training Neural Networks

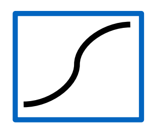
Logistic (neuron) Activation

- If input is $x = \mathbf{w}^T \mathbf{x}$, the output will look like a probability $\sigma(\mathbf{w}^T \mathbf{x}) \in [0, 1]$

- $\mathrm{p}(\mathrm{y} = 1 \mid \mathbf{x}; \mathbf{w}) = \sigma(\mathbf{w}^T \mathbf{x}) \in [0, 1]$



$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- We will represent the logistic function with the symbol: 

- Very simple derivative:

$$\frac{\partial \sigma(x)}{\partial x} = \sigma(x)(1 - \sigma(x))$$

How do we do gradient ascent?

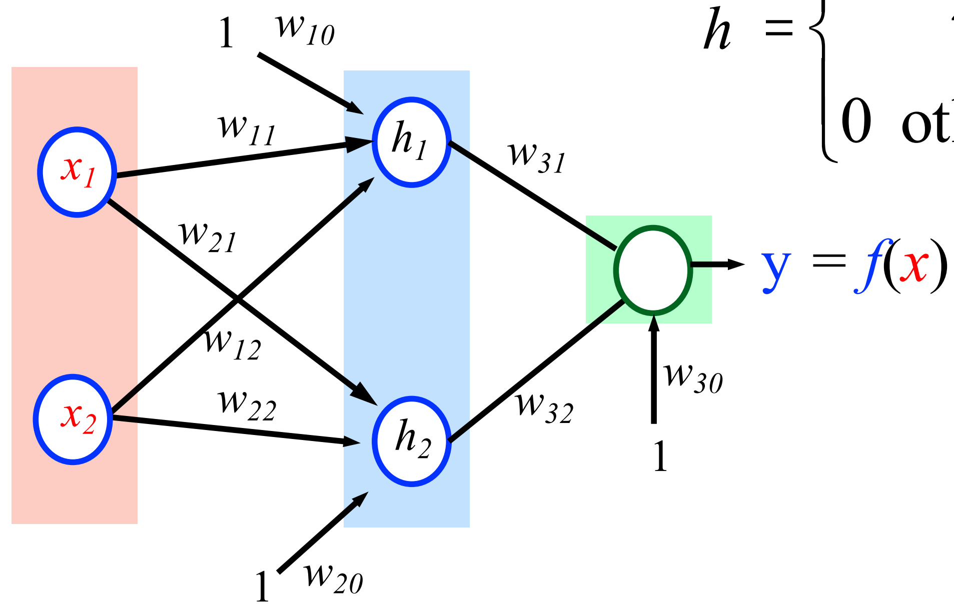
- Seems pretty costly to compute the gradient of this big complicated function
- But fortunately, the “pipeline” structure of the network makes it easier
- We’ll compute the prediction for each data point, and save the intermediate values
- Then we’ll do a “backward” pass to update the weights, by computing the gradients as we go backward
- We update each weight once we compute its gradient (remember, all of this is just about doing gradient ascent: updating each weight w by

$$w \leftarrow w + \epsilon \frac{\partial}{\partial w} \log(L(\text{data}; \text{all weights}))$$

Example: Solving the XOR Problem

$$h = \begin{cases} 1 \text{ if } \sum_{i=0}^{n} w_i x_i \geq 0 \\ 0 \text{ otherwise} \end{cases}$$

Network Topology:
2 hidden nodes
1 output



Desired behavior:

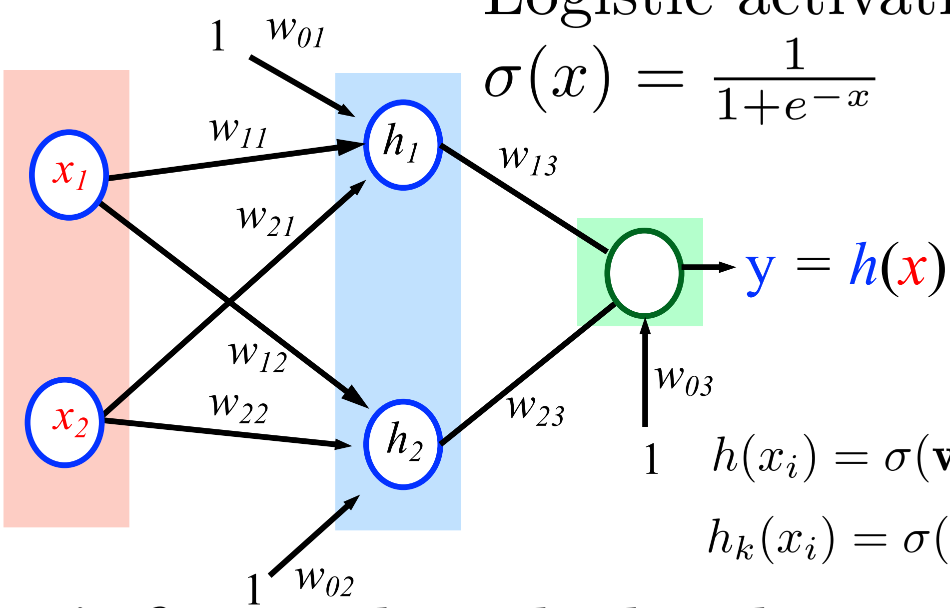
x_1	x_2	h_1	h_2	y
0	0	0	0	0
1	0	0	1	1
0	1	0	1	1
1	1	1	1	0

$h_1 = \text{AND}$, $h_2 = \text{OR}$,
$h = (x_1 \text{ OR } x_2) \text{ AND NOT } (x_1 \text{ AND } x_2)$

Weights:

- h_1=AND: $w_{11} = w_{12} = 1$, $w_{10} = -3/2$
- h_2=OR: $w_{21} = 1$, $w_{22} = 1$, $w_{20} = -1/2$
- h=XOR: $w_{31} = -1$, $w_{32} = 1$, $w_{30} = -1/2$

Example: Learning the XOR Function



Logistic activation functions:

$\sigma(x) = \frac{1}{1+e^{-x}}$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top (1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top (1, x_{i1}, x_{i2})),\ k = 1, 2$$

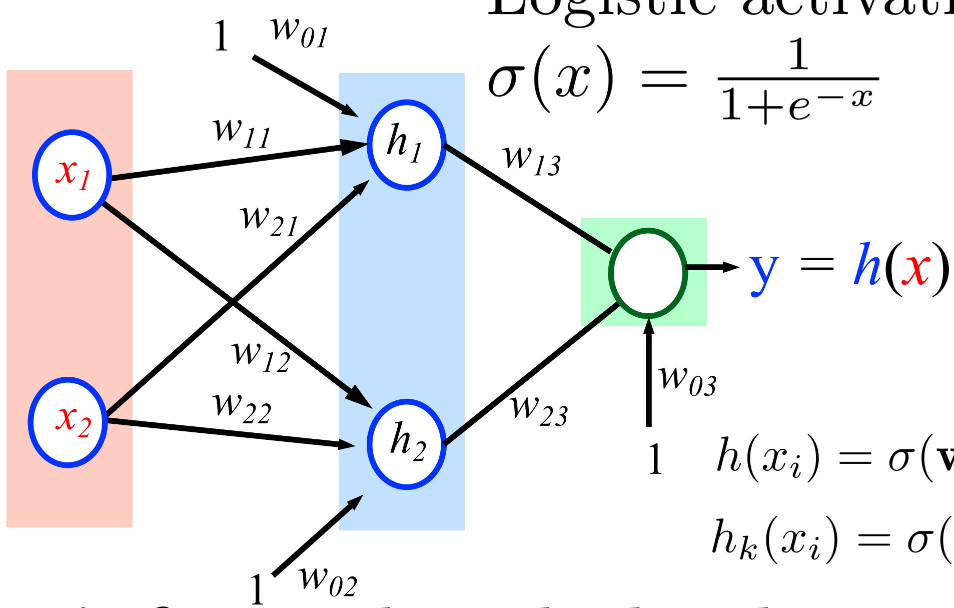
Let's first update the last layer:

$$\frac{\partial}{\partial w_{03}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{03}} h(x_i) \right) \left(\frac{y_i}{h(x_i)} - \frac{1 - y_i}{1 - h(x_i)} \right)$$

$$= \sum_{i=1}^{4} (1)\, h(x_i)(1 - h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1 - y_i}{1 - h(x_i)} \right)$$

Update: $w_{03} \leftarrow w_{03} + \epsilon \frac{1}{4} \sum_{i=1}^{4} h(x_i)(1 - h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$

Example: Learning the XOR Function



Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Let's first update the last layer:

$$\frac{\partial}{\partial w_{13}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{13}} h(x_i) \right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$$

$$= \sum_{i=1}^{4} (h_1(x_i))\, h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$$

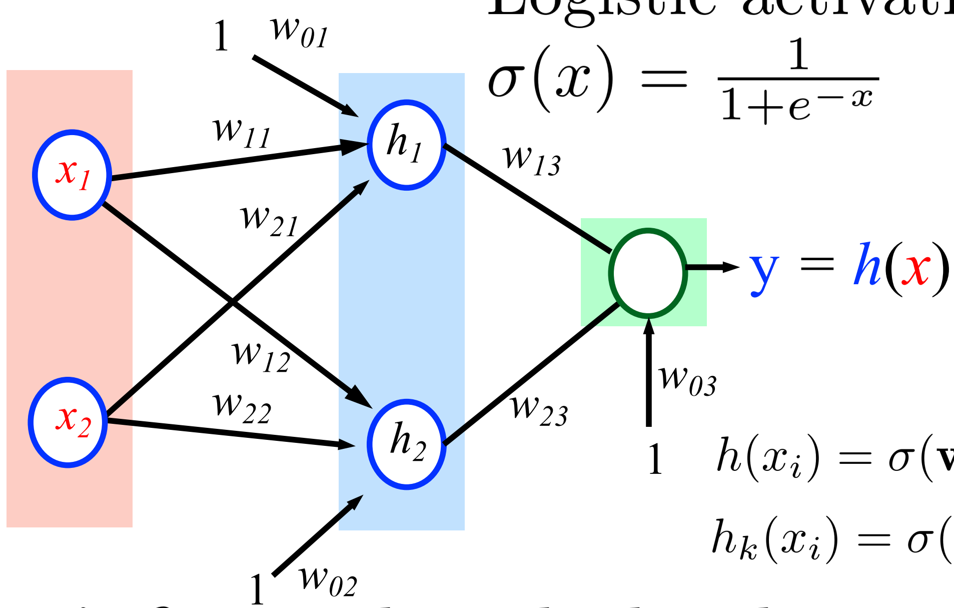
Update: $w_{13} \leftarrow w_{13} + \epsilon \frac{1}{4} \sum_{i=1}^{4} h_1(x_i) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$

Note: We calculated $h_1(x_i)$ while computing the prediction $h(x_i)$ anyway. Just remember it instead of computing it again.

Example: Learning the XOR Function

Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Let's first update the last layer:

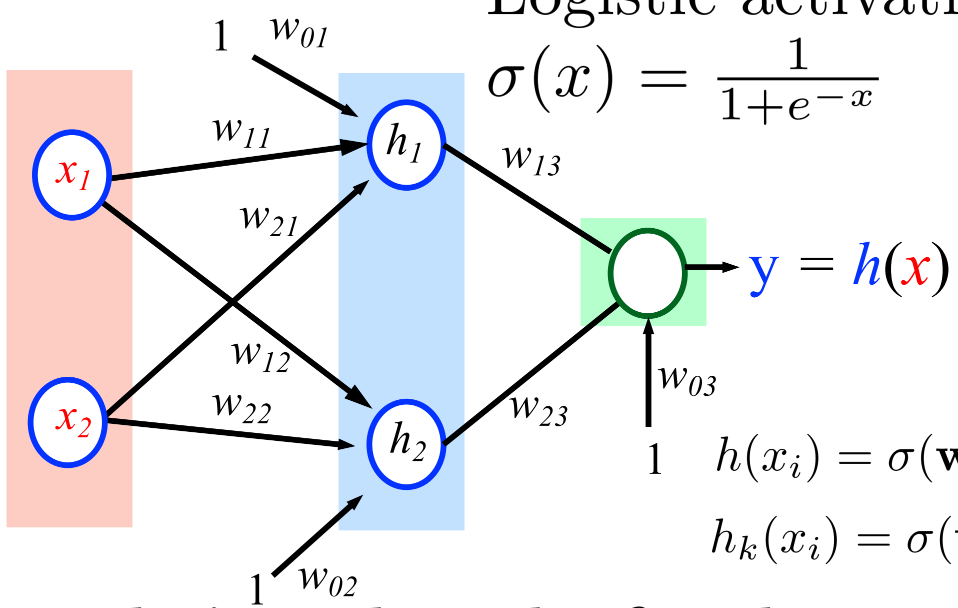
$$\frac{\partial}{\partial w_{23}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{23}} h(x_i) \right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$$

$$= \sum_{i=1}^{4} (h_2(x_i))\, h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$$

Update: $w_{23} \leftarrow w_{23} + \epsilon \frac{1}{4} \sum_{i=1}^{4} h_2(x_i) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$

Note: We calculated $h_2(x_i)$ while computing the prediction $h(x_i)$ anyway. Just remember it instead of computing it again.

Example: Learning the XOR Function



Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Now let's update the first layer:

$$\frac{\partial}{\partial w_{01}} \sum_{i=1}^{4} \log(p(y = y_i|x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{01}} h(x_i)\right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} \left(w_{13} \frac{\partial}{\partial w_{01}} h_1(x_i)\right) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

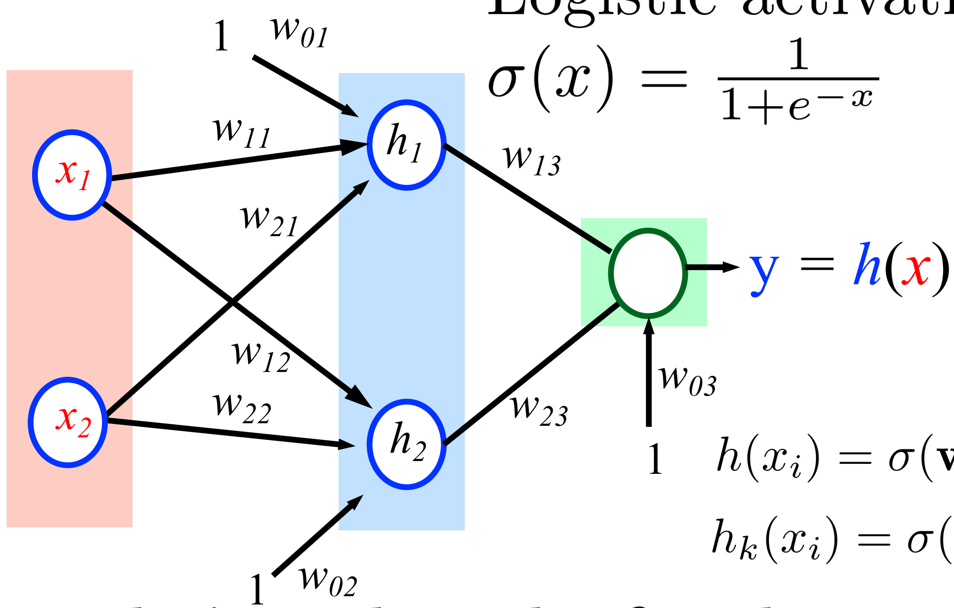
This is the w_{13} from before it was updated

$$= \sum_{i=1}^{4} \quad w_{13} \quad\quad h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Example: Learning the XOR Function

Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Now let's update the first layer:

$$\frac{\partial}{\partial w_{01}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{01}} h(x_i)\right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

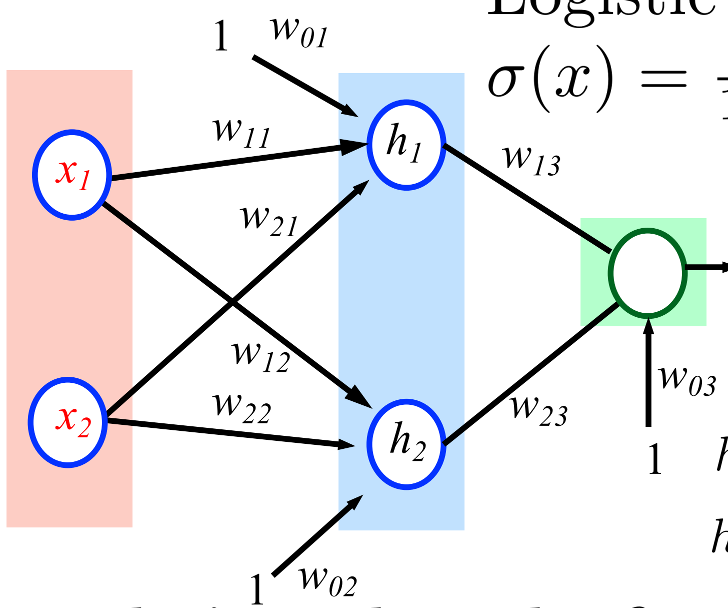
$$= \sum_{i=1}^{4} \left(w_{13} \frac{\partial}{\partial w_{01}} h_1(x_i)\right) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

This is the w_{13} from before it was updated

$$= \sum_{i=1}^{4} (1)\, w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Update: $w_{01} \leftarrow w_{01} + \epsilon \frac{1}{4} \sum_{i=1}^{4} w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$

Example: Learning the XOR Function



Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$$h(x_i) = \sigma(\mathbf{w}^\top_{\{0,1,2\}3}(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}^\top_{\{0,1,2\}k}(1, x_{i1}, x_{i2})),\ k = 1,2$$

Now let's update the first layer:

$$\frac{\partial}{\partial w_{01}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{01}} h(x_i)\right)\left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} \left(w_{13} \frac{\partial}{\partial w_{01}} h_1(x_i)\right) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)}\right.$$

We already calculated this part when updating the last layer

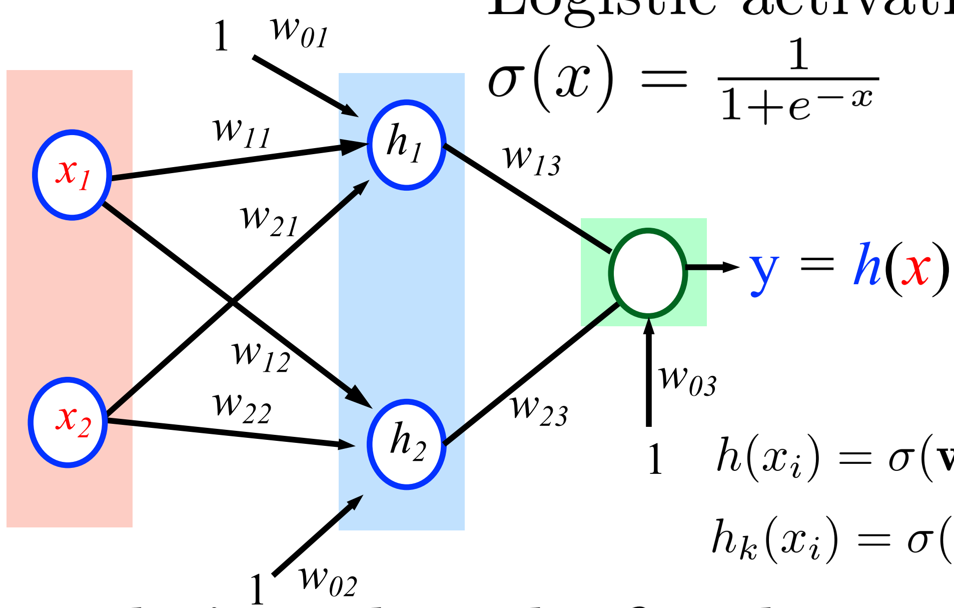
$$= \sum_{i=1}^{4} (1)\, w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Update: $w_{01} \leftarrow w_{01} + \epsilon \frac{1}{4} \sum_{i=1}^{4} w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$

Example: Learning the XOR Function

Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

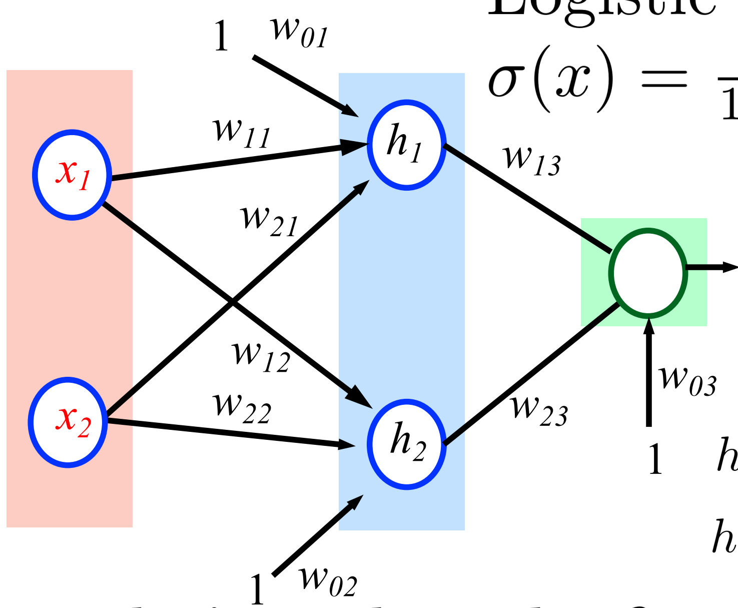
Now let's update the first layer:

$$\frac{\partial}{\partial w_{11}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{11}} h(x_i)\right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} \left(w_{13} \frac{\partial}{\partial w_{11}} h_1(x_i)\right) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} \qquad w_{13} \qquad\qquad h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Example: Learning the XOR Function



$\mathrm{y} = h(x)$

Logistic activation functions:

$\sigma(x) = \frac{1}{1+e^{-x}}$

$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$

$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$

Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

Now let's update the first layer:

$$\frac{\partial}{\partial w_{11}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{11}} h(x_i)\right) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} \left(w_{13} \frac{\partial}{\partial w_{11}} h_1(x_i)\right) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

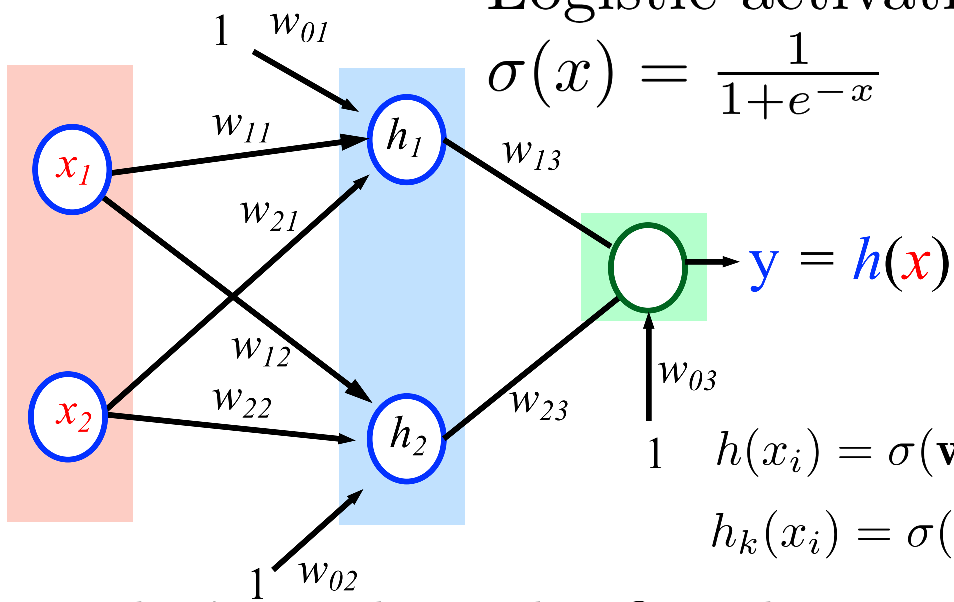
$$= \sum_{i=1}^{4} (x_{i1})\, w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Update: $w_{11} \leftarrow w_{11} + \epsilon \frac{1}{4} \sum_{i=1}^{4} x_{i1} w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$

Example: Learning the XOR Function

Logistic activation functions:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Data set:

x_1	x_2	y
0	0	0
0	1	1
1	0	1
1	1	0

$$h(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}3}^\top(1, h_1(x_i), h_2(x_i)))$$

$$h_k(x_i) = \sigma(\mathbf{w}_{\{0,1,2\}k}^\top(1, x_{i1}, x_{i2})),\ k = 1, 2$$

Now let's update the first layer:

$$\frac{\partial}{\partial w_{21}} \sum_{i=1}^{4} \log(p(y = y_i | x_i; W)) = \sum_{i=1}^{4} \left(\frac{\partial}{\partial w_{21}} h(x_i)\right)\left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

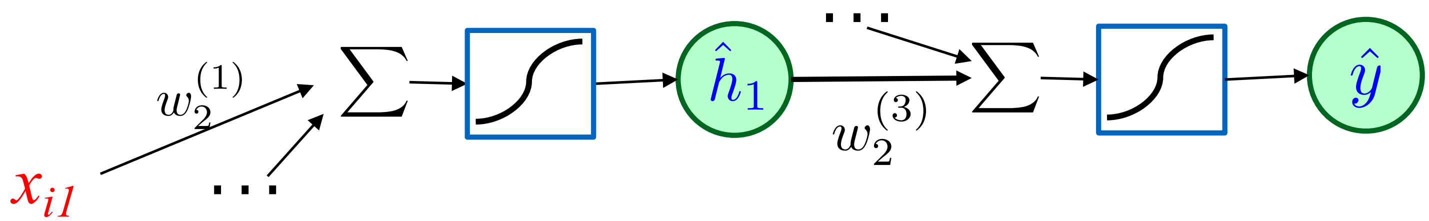
$$= \sum_{i=1}^{4} \left(w_{13}\frac{\partial}{\partial w_{21}} h_1(x_i)\right) h(x_i)(1-h(x_i))\left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

$$= \sum_{i=1}^{4} (x_{i2})\, w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i))\left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$$

Update: $w_{21} \leftarrow w_{21} + \epsilon \frac{1}{4} \sum_{i=1}^{4} x_{i2} w_{13} h_1(x_i)(1-h_1(x_i)) h(x_i)(1-h(x_i))\left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)}\right)$

Computing Gradients of the Lower-Layer Parameters

- In a multi-layer neural network each layer is a composition of previous layers



- The influence of a lower layer parameter in the final error can be recovered by the chain rule, which generally states:

$$\frac{\partial f^l(f^{l-1}(\cdots f^2(f^1(w))))}{\partial w} = \frac{\partial f^l}{\partial f^{l-1}} \cdot \frac{\partial f^{l-1}}{\partial f^{l-2}} \cdots \frac{\partial f^2}{\partial f^1} \cdot \frac{\partial f^1(x)}{\partial w}$$

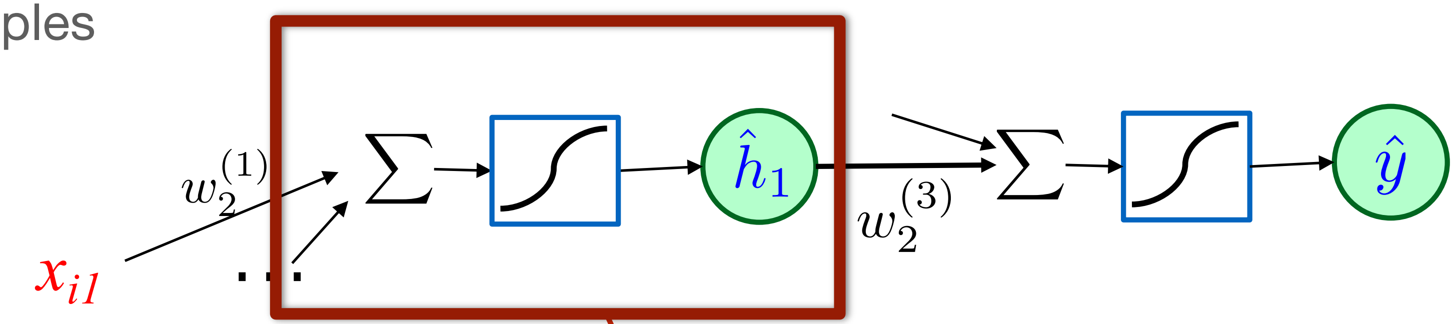
- Specific to the example given above:

$$\frac{\partial \sigma(\ldots + w_2^{(3)} \sigma(\ldots + w_2^{(1)} x_{i1}))}{\partial w_2^{(1)}} = \frac{\partial \sigma(\ldots + w_2^{(3)} \hat{h}_1)}{\partial \hat{h}_1} \cdot \frac{\partial \sigma(\ldots + w_2^{(1)} x_{i1})}{\partial w_2^{(1)}}$$

where $\hat{h}_1 = \sigma(\ldots + w_2^{(1)} x_{i1})$

Neural Network Gradient Ascent

- L is the log-likelihood function with respect to final outputs, on N training examples



Gradient update step (learning rate $\varepsilon \cong 0$):

$$w_3^{(2)} = w_3^{(2)} + \epsilon \frac{\partial L(x)}{\partial w_3^{(2)}}$$

- The local gradient measures how the output changes with the input

$$\boxed{(w_3^{(2)})^T \mathbf{x}}$$

"Local gradient"

$$\frac{\partial \hat{h}(x)}{\partial w_3^{(2)}}$$

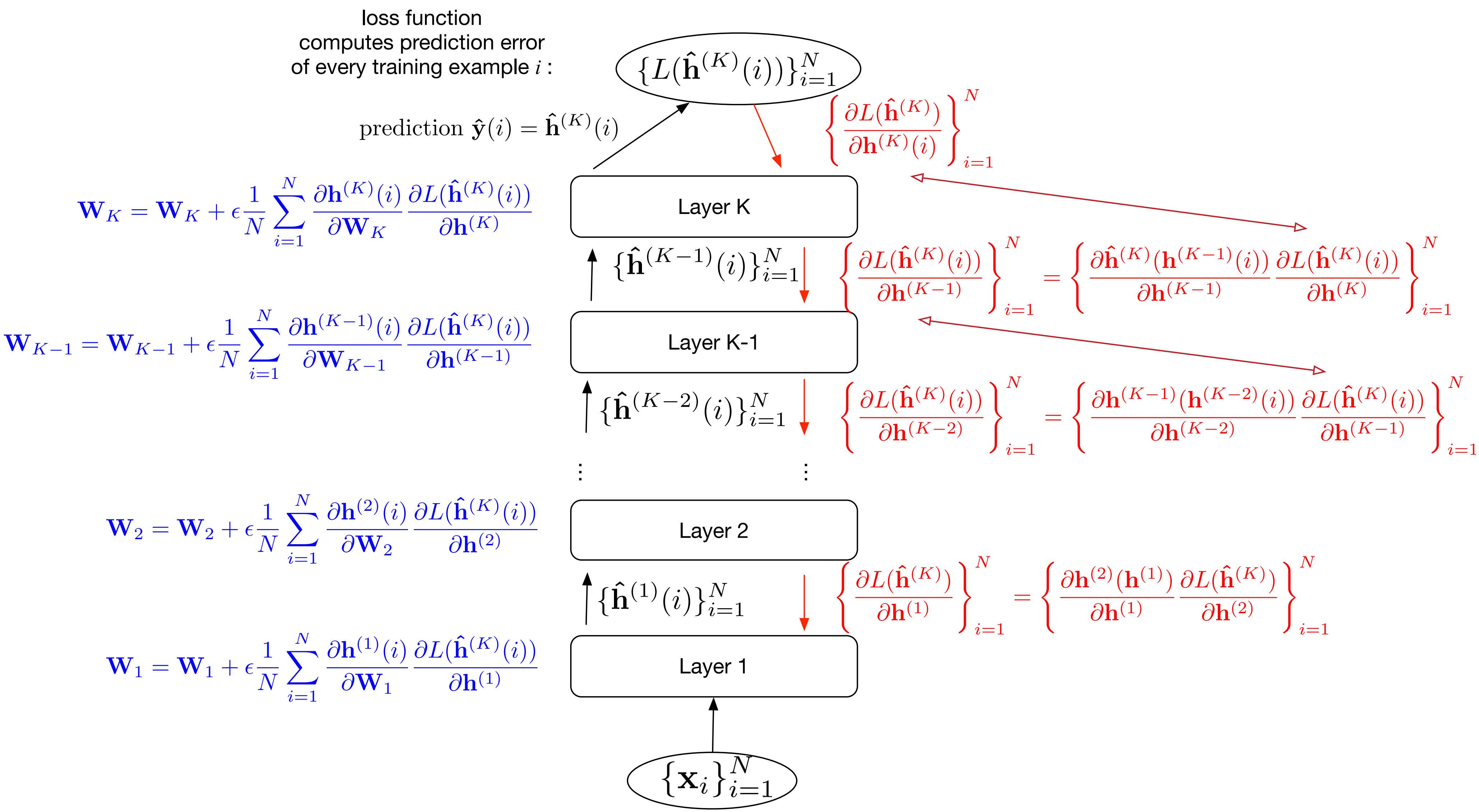
$\hat{h}_1$ forward (prediction)

$$\frac{\partial L(\hat{h}_1)}{\partial h}$$

backward (loss derivative)

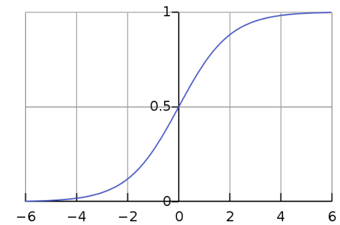
$$\frac{\partial L(x)}{\partial w_3^{(2)}} = \frac{\partial L(\hat{h}_1)}{\partial \hat{h}_1} \frac{\partial \hat{h}_1(x)}{\partial w_3^{(2)}}$$

How it works: Forward and backward updates of last layer parameters



Sigmoid impact

$$\text{sigmoid: } \sigma(x) = \frac{1}{1+e^{-x}}$$



Recall the update for sigmoids:

$$\text{Update: } w_{21} \leftarrow w_{21} + \epsilon \frac{1}{4} \sum_{i=1}^{4} x_{i2} w_{13} h_1(x_i)(1 - h_1(x_i)) h(x_i)(1 - h(x_i)) \left(\frac{y_i}{h(x_i)} - \frac{1-y_i}{1-h(x_i)} \right)$$

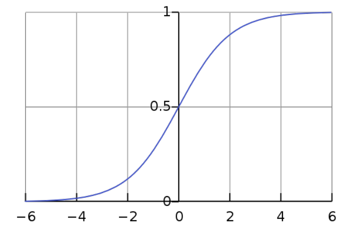
If the network were very deep (many layers), we'd get many factors $\sigma(.)(1- \sigma(.))$

Some of these will probably have $\sigma(.)$ close to 0 or 1, so $\sigma(.)(1 - \sigma(.))$ is very small

Means the updates to early layers are very tiny

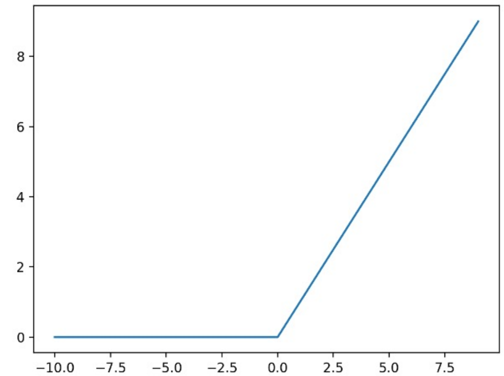
Different activation functions are useful

$$\text{sigmoid: } \sigma(x) = \frac{1}{1+e^{-x}}$$



Rectified Linear Unit (ReLU):

$$\sigma(x) = \max\{0, x\}$$



Has advantages when the network is very deep:
signal doesn't diminish as it propagates backward

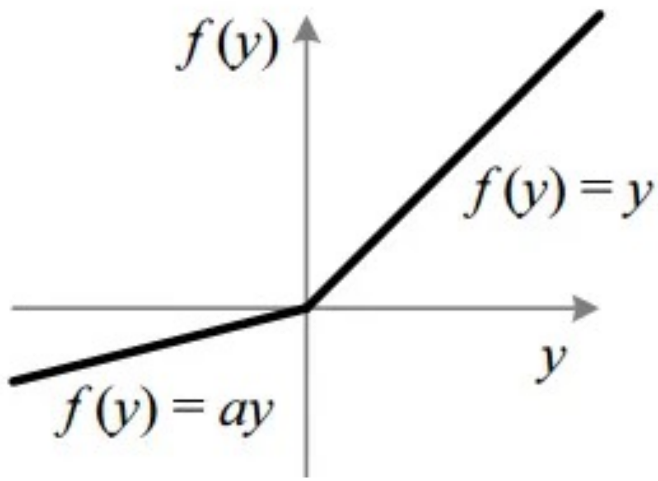
Has disadvantage that it sometimes makes the signal zero

Different activation functions are useful

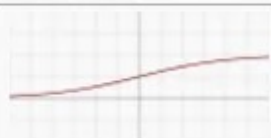
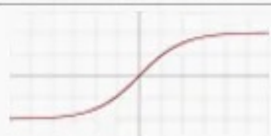
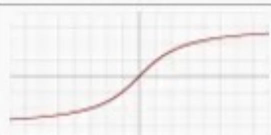
Leaky Rectified Linear Unit (Leaky ReLU):

$$\sigma(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases}$$

$0 < \alpha < 1$ (typically $\alpha = 0.01$)



Doesn't have the "zeros" problem or the diminishing signal problem

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x \neq 0 \\ ? & \text{for} \quad x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for} \quad x < 0 \\ 1 & \text{for} \quad x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Weight Initialization

- How do we initialize the weights before running backpropagation?
- Usually use uniformly random values in a small range, e.g., [-0.1, 0.1].

We can do this with any differentiable loss function

- We can use backpropagation if:
  - If we have a loss function L that is differentiable (e.g., squared loss)
  - And we have activation functions that are differentiable (sigmoid, tanh function, …)