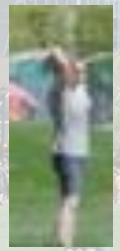
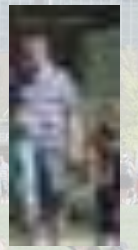
Neural Network

Hyun Soo Park

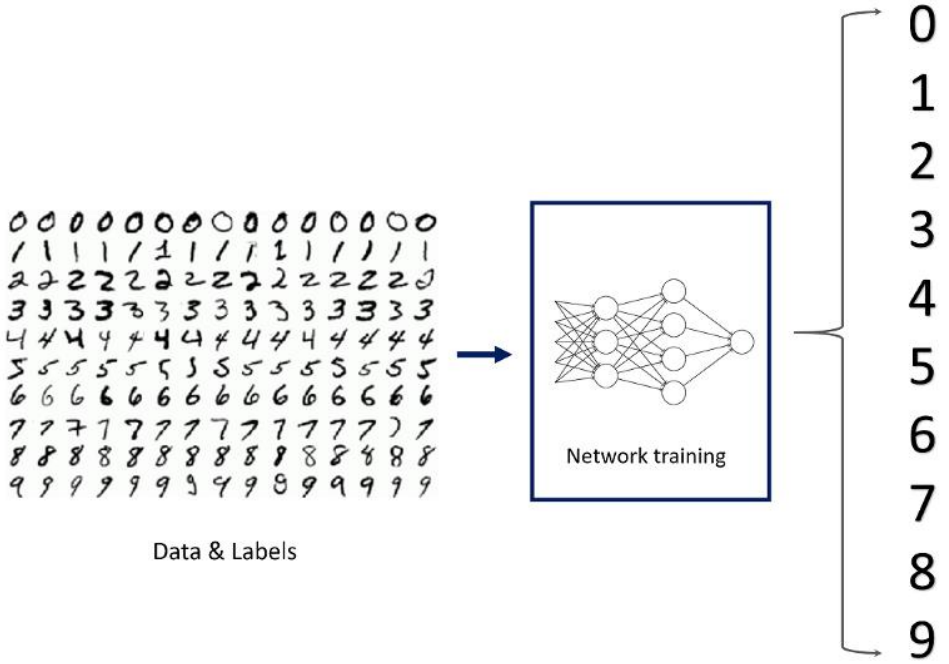
Challenges of Visual Recognition

- Appearance
  - DOF: texture, illumniation, material, shading, …
- **Shape**
  - DOF: object category, geometric pose, viewpoint, …



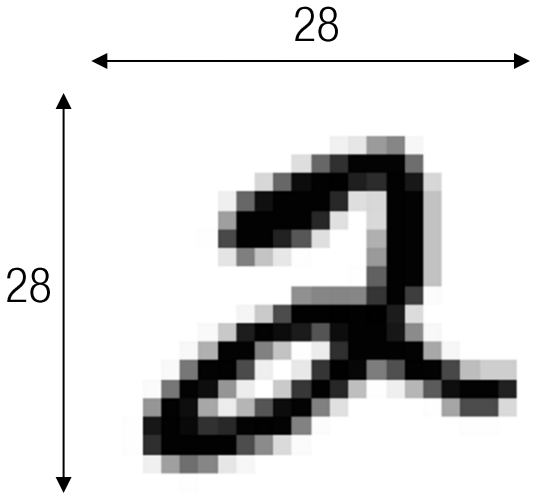
MNIST Digit Recognition



Data & Labels

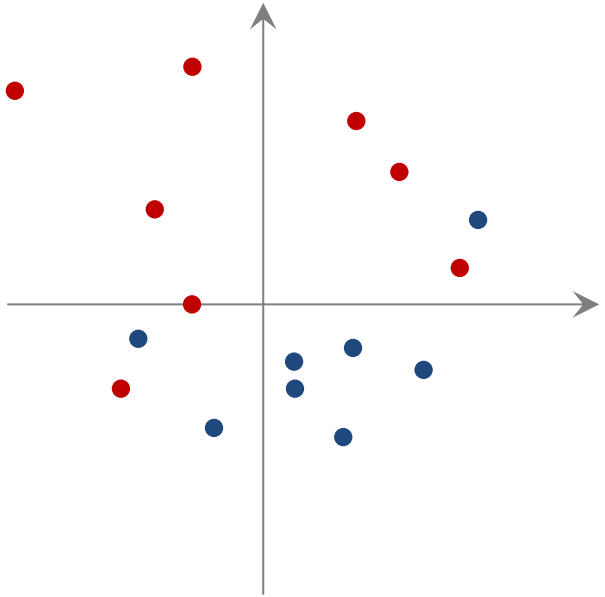
Data

- 60,000 training
- 10,000 testing



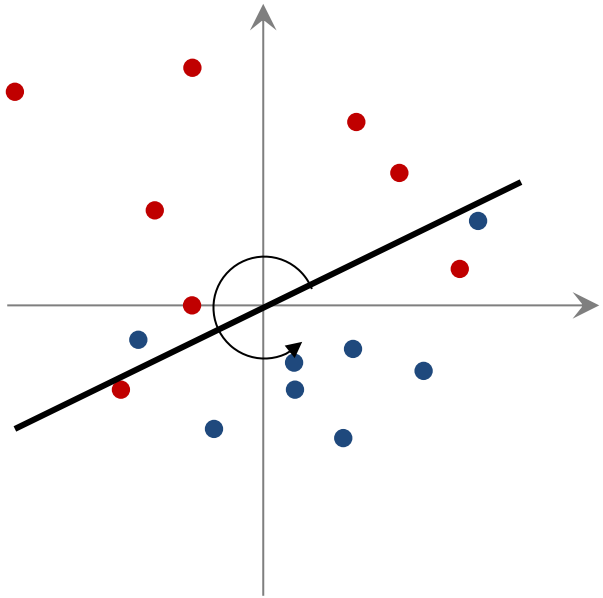
PERCEPTRON

Binary classification

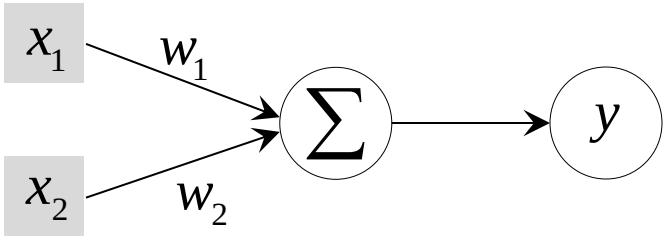


PERCEPTRON

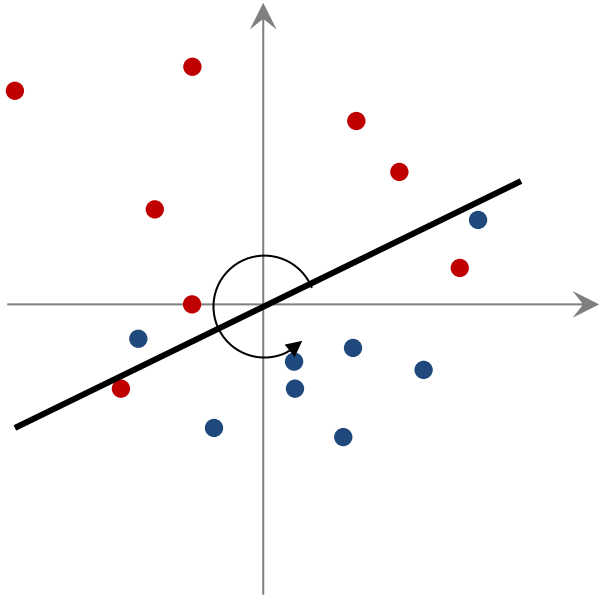
Binary classification



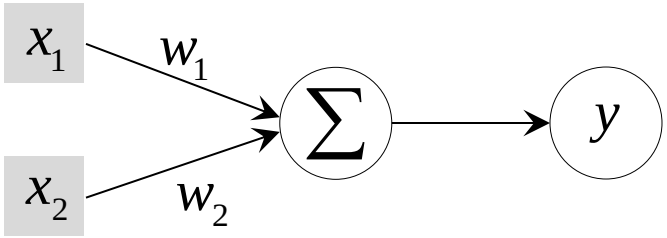
PERCEPTRON



Binary classification

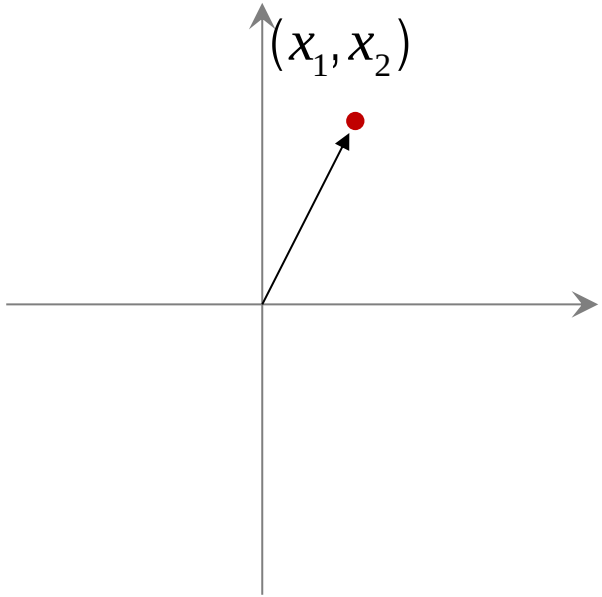


PERCEPTRON

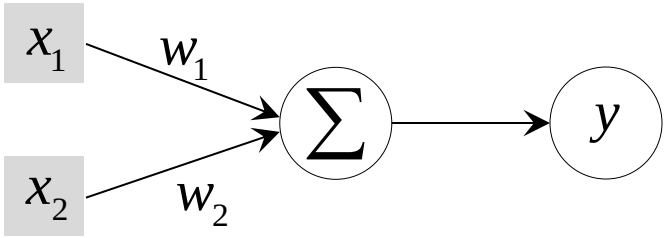


$$y = x_1 w_1 + x_2 w_2$$

Binary classification

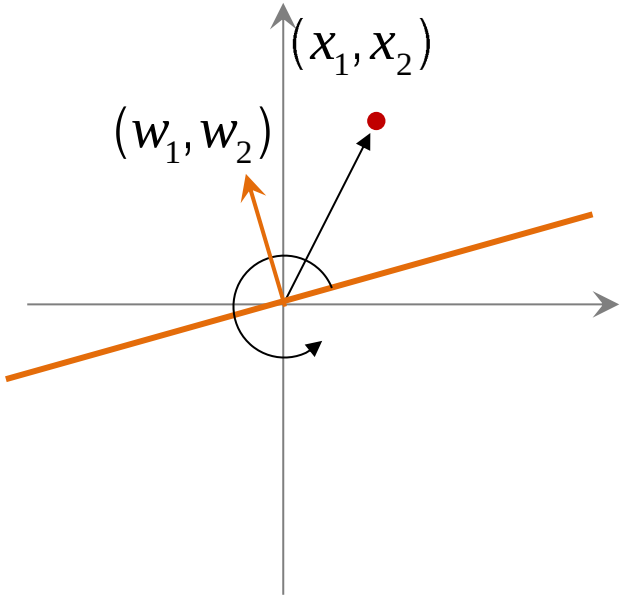


PERCEPTRON

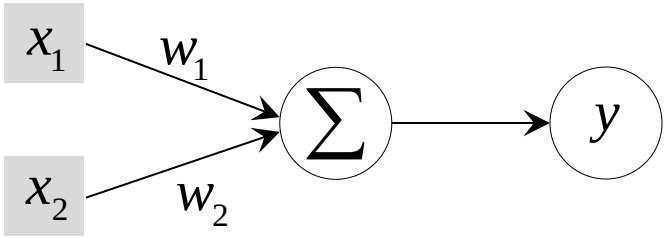


$$y = x_1 w_1 + x_2 w_2$$

Binary classification

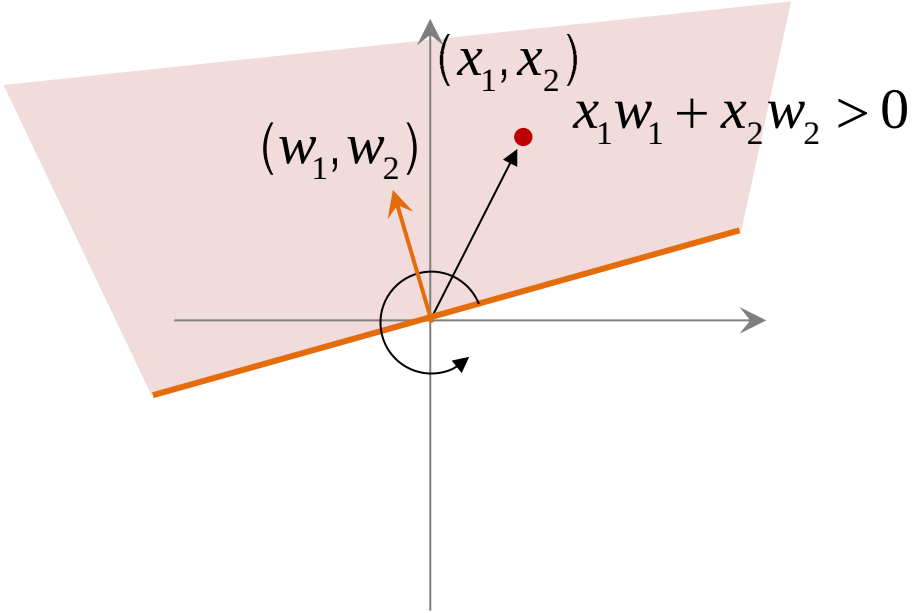


PERCEPTRON

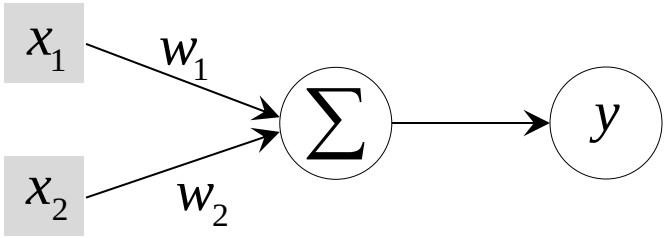


$$y = x_1 w_1 + x_2 w_2$$

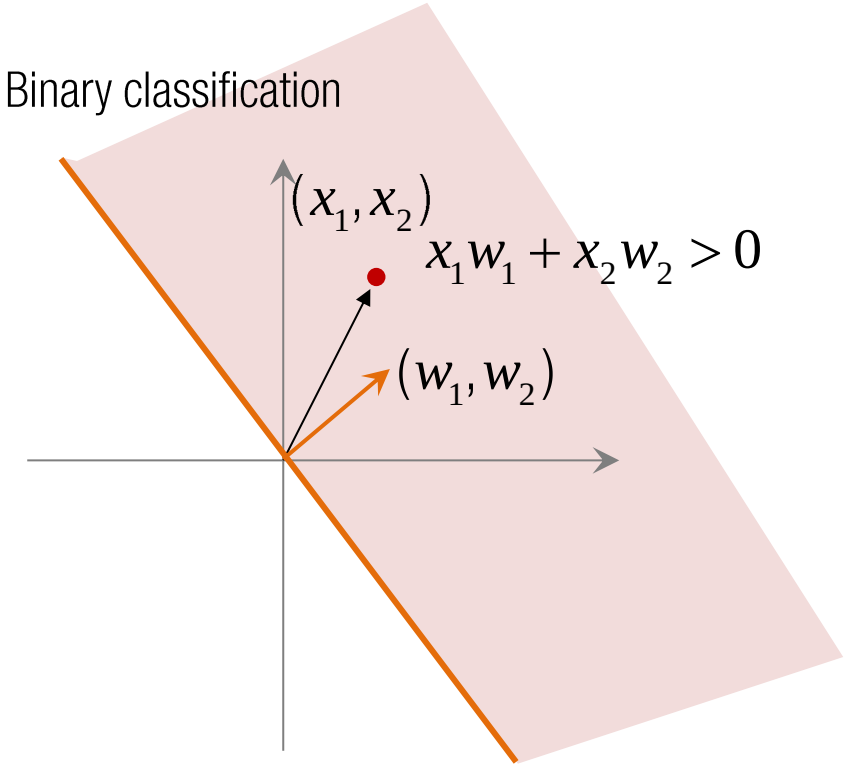
Binary classification



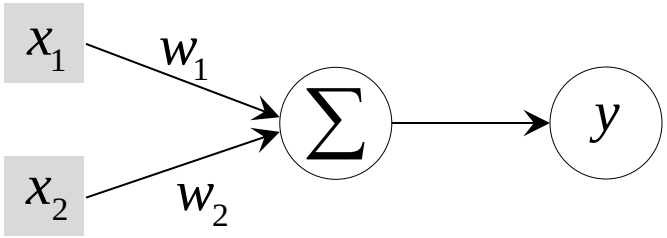
PERCEPTRON



$$y = x_1 w_1 + x_2 w_2$$

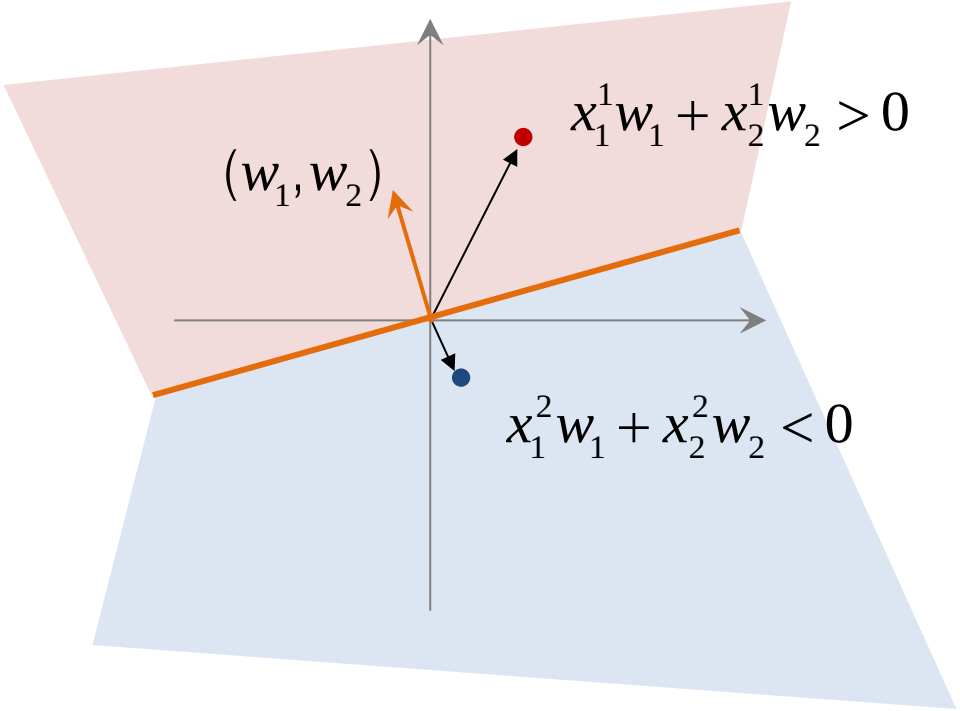


PERCEPTRON

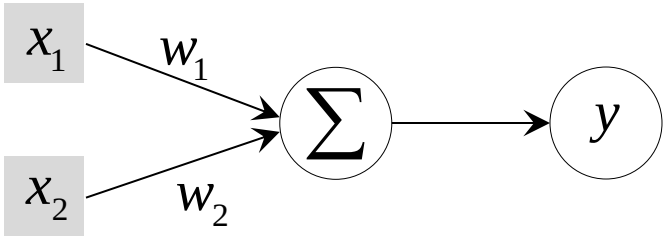


$$y = x_1 w_1 + x_2 w_2$$

Binary classification

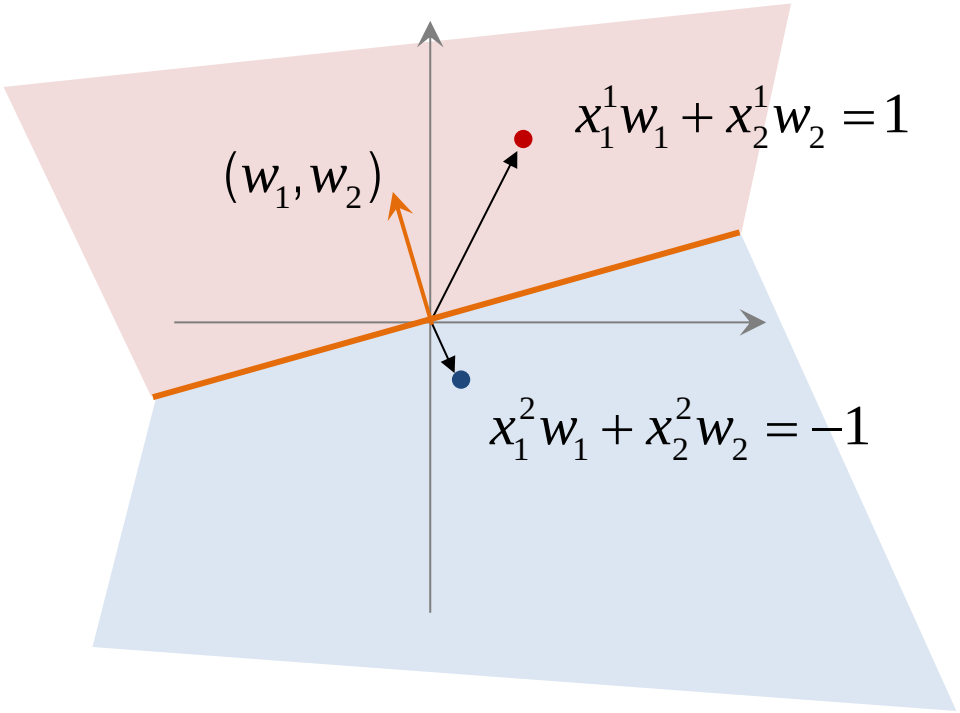


PERCEPTRON

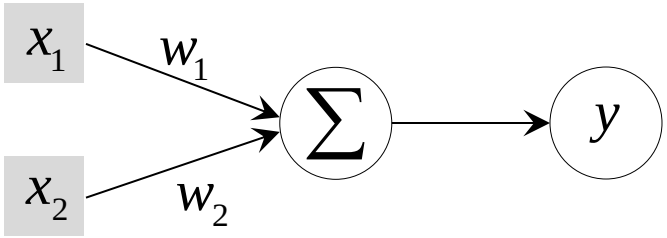


$$y = \{-1, 1\}$$

Binary classification

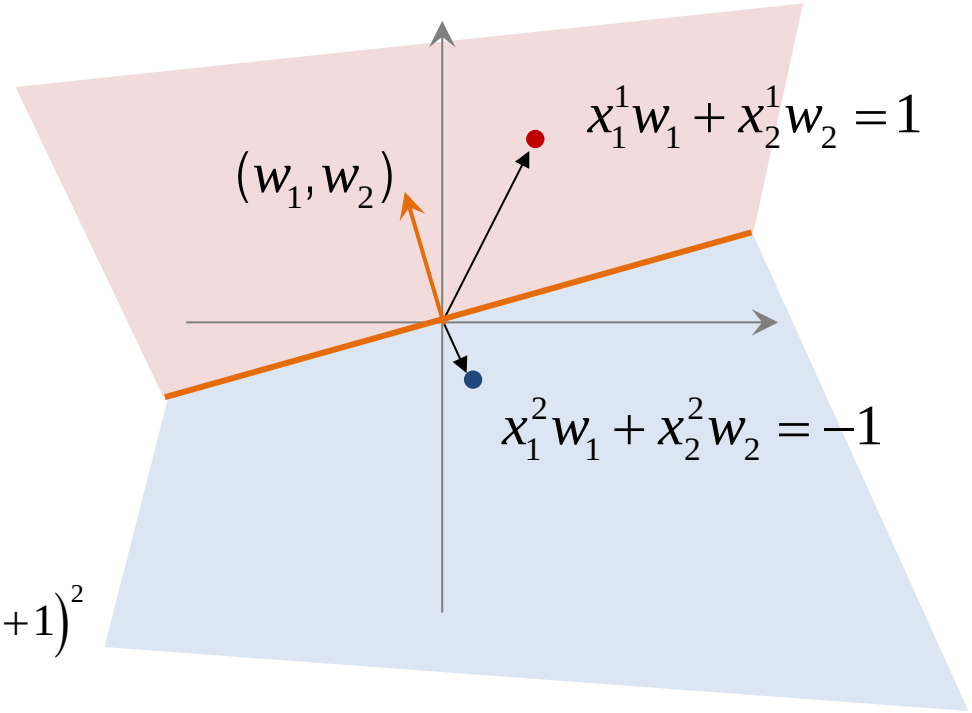


PERCEPTRON



$$y = \{-1, 1\}$$

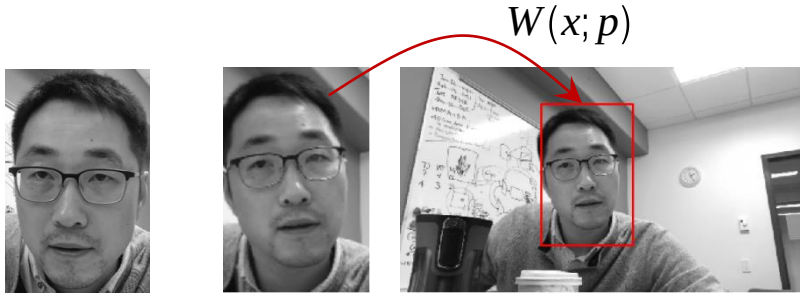
Binary classification



$$\underset{w_1, w_2}{\text{minimize}} \quad L(w_1, w_2)$$

$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - 1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 + 1\right)^2$$

RECALL: IMAGE ALIGNMENT OBJECTIVE

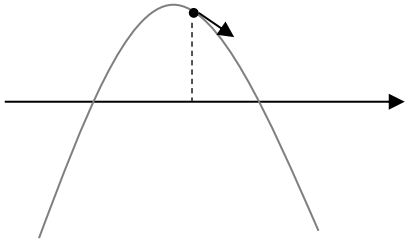


$$p* = \underset{p}{\text{minimize}} \sum_{x} \left(I(W(x;p) - T(x) \right)^2$$

Guass-Newton's method

1.Linearize the obj. function at p

$$I(W(x;p+\Delta p)) \approx I(W(x;p)) + \frac{\partial I}{\partial p}\Delta p$$



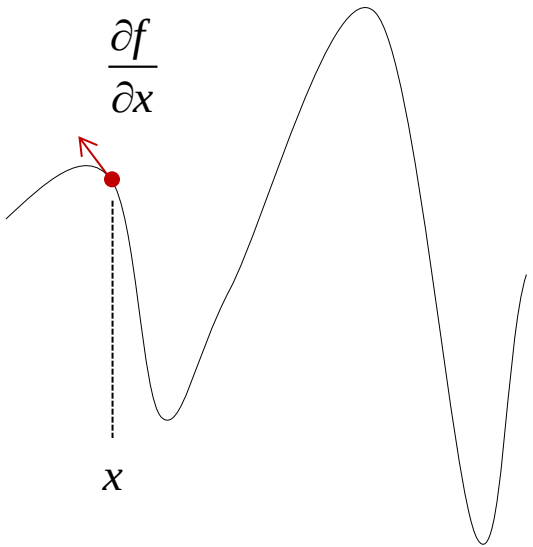
Ex) affine transform

$$W(x;p) = \begin{bmatrix} u \\ v \end{bmatrix} + \begin{bmatrix} p_1 u + p_2 v + p_3 \\ p_4 u + p_5 v + p_6 \end{bmatrix}$$

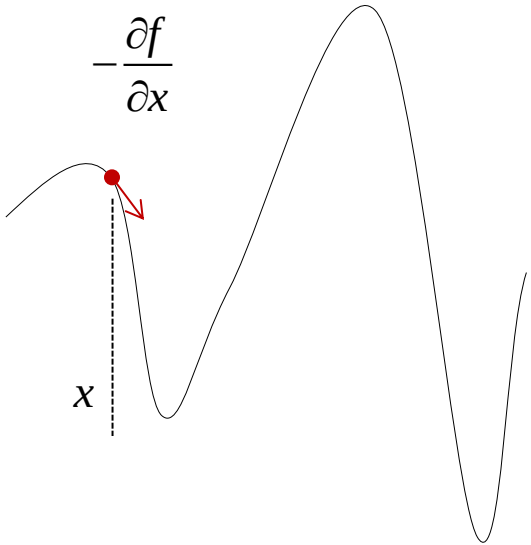
$$\rightarrow \quad \frac{\partial I}{\partial p}\Delta p = \frac{\partial I}{\partial x}\frac{\partial W}{\partial p}\Delta p = \nabla I \frac{\partial W}{\partial p}\Delta p$$

Jacobian: $\frac{\partial W}{\partial p} = \begin{bmatrix} \frac{\partial u}{\partial p_1} & \cdots & \frac{\partial u}{\partial p_6} \\ \frac{\partial v}{\partial p_1} & \cdots & \frac{\partial v}{\partial p_6} \end{bmatrix}$

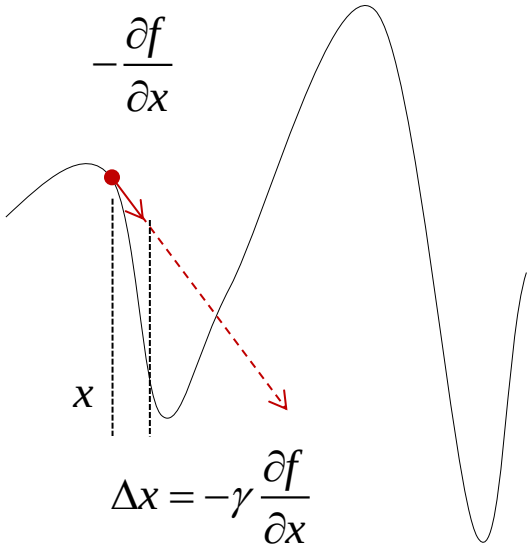
Gradient Descent



Gradient Descent



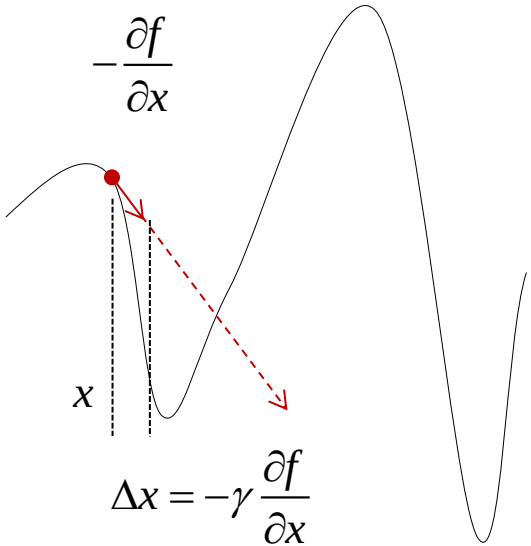
LINE SEARCH



$$\Delta x = -\gamma \frac{\partial f}{\partial x}$$

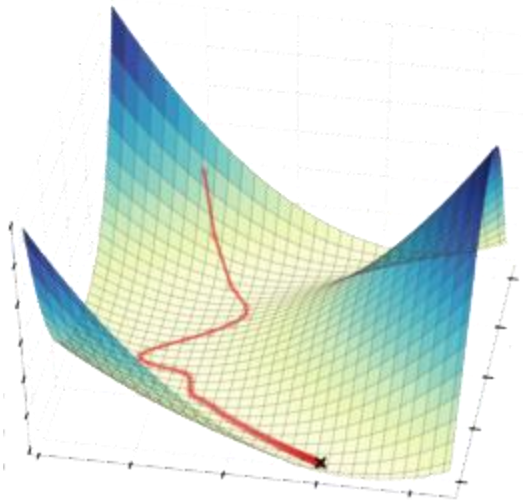
$$f(x) > f(x + \Delta x)$$

GRADIENT DESCENT



$$\Delta x = -\gamma \frac{\partial f}{\partial x}$$

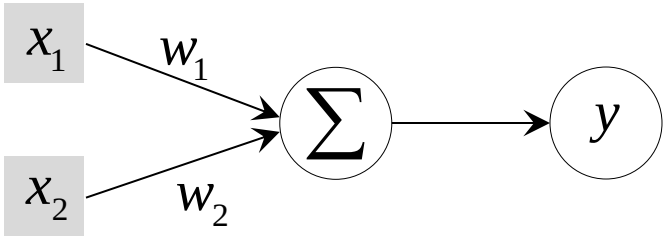
$$f(x) > f(x + \Delta x)$$



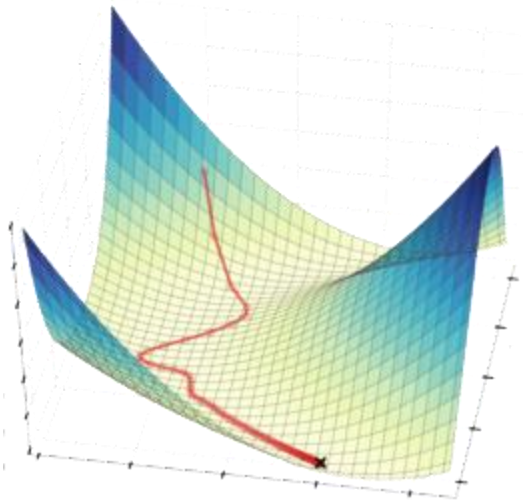
$$\underset{x}{\text{minimize}} \; f(x)$$

$$x \leftarrow x - \gamma \frac{\partial f}{\partial x}$$

GRADIENT DESCENT



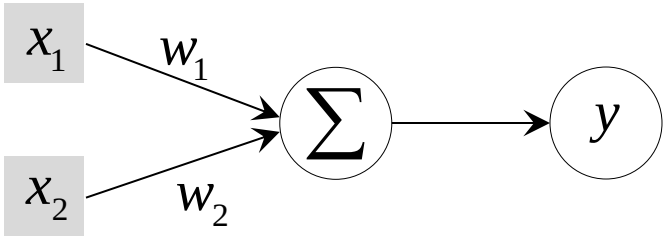
$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - 1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 + 1\right)^2$$

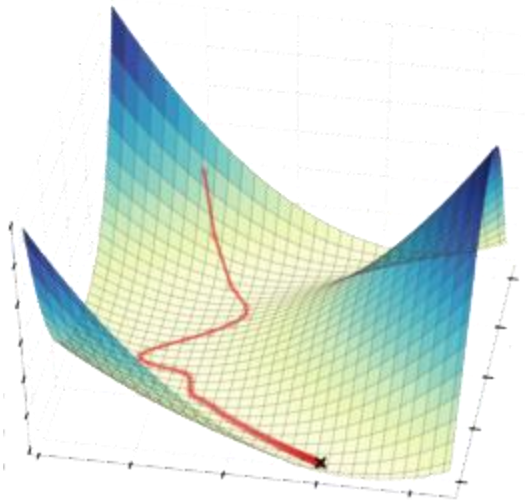


$$\underset{x}{\text{minimize}} \quad f(x)$$

$$x \leftarrow x - \gamma \frac{\partial f}{\partial x}$$

GRADIENT DESCENT





$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - 1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 + 1\right)^2$$

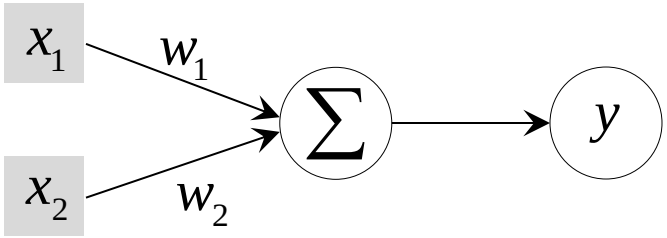
$$\underset{x}{\text{minimize}} \quad f(x)$$

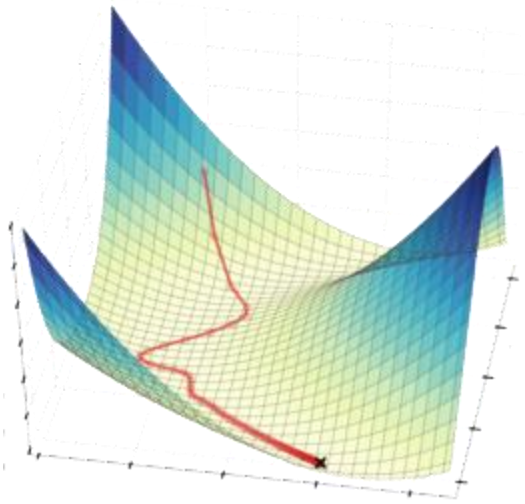
Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$x \leftarrow x - \gamma \frac{\partial f}{\partial x}$$

GRADIENT DESCENT





$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - y^1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 - y^2\right)^2$$

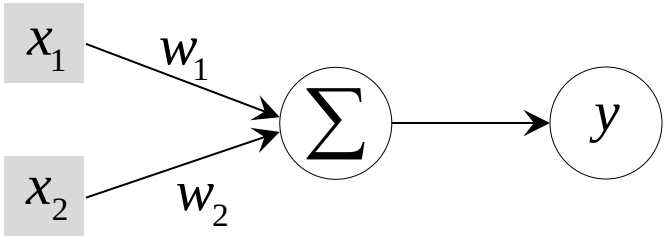
$$\underset{x}{\text{minimize}} \; f(x)$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$x \leftarrow x - \gamma \frac{\partial f}{\partial x}$$

GRADIENT DESCENT



Prediction:

$$\tilde{y} = x_1 w_1 + x_2 w_2$$

Error for each data:

$$L(w_1, w_2) = \left(\tilde{y} - y\right)^2$$

Gradient:

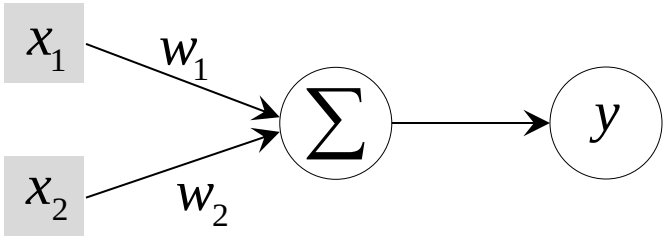
$$\frac{\partial L}{\partial w_1} =$$

$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - y^1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 - y^2\right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

GRADIENT DESCENT



$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - y^1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 - y^2\right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

Prediction:

$$\tilde{y} = x_1 w_1 + x_2 w_2$$

Error for each data:

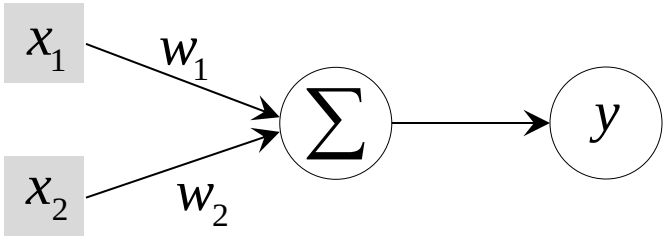
$$L(w_1, w_2) = \left(\tilde{y} - y\right)^2$$

Gradient:

$$\frac{\partial L}{\partial w_1} = \left(\tilde{y} - y\right) \frac{\partial \hat{y}}{\partial w_1}$$

$$= \left(\tilde{y} - y\right) x_1$$

GRADIENT DESCENT



Prediction:

$$\tilde{y} = x_1 w_1 + x_2 w_2$$

Error for each data:

$$L(w_1, w_2) = \left(\tilde{y} - y\right)^2$$

Gradient:

$$\frac{\partial L}{\partial w_1} = \left(\tilde{y} - y\right)\frac{\partial \hat{y}}{\partial w_1}$$

$$= \left(\tilde{y} - y\right)x_1$$

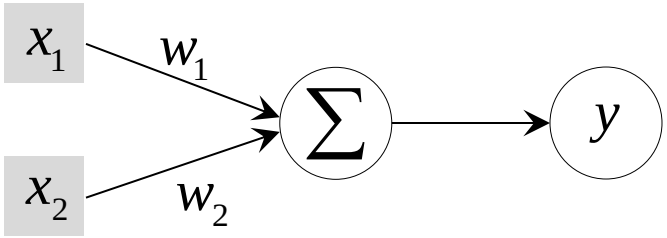
$$L(w_1, w_2) = \left(x_1^1 w_1 + x_2^1 w_2 - y^1\right)^2 + \left(x_1^2 w_1 + x_2^2 w_2 - y^2\right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma(\tilde{y}^1 - y^1)x_1^1 - \gamma(\tilde{y}^2 - y^2)x_1^2$$

$$w_2 = w_2 - \gamma(\tilde{y}^1 - y^1)x_2^1 - \gamma(\tilde{y}^2 - y^2)x_2^2$$

GRADIENT DESCENT

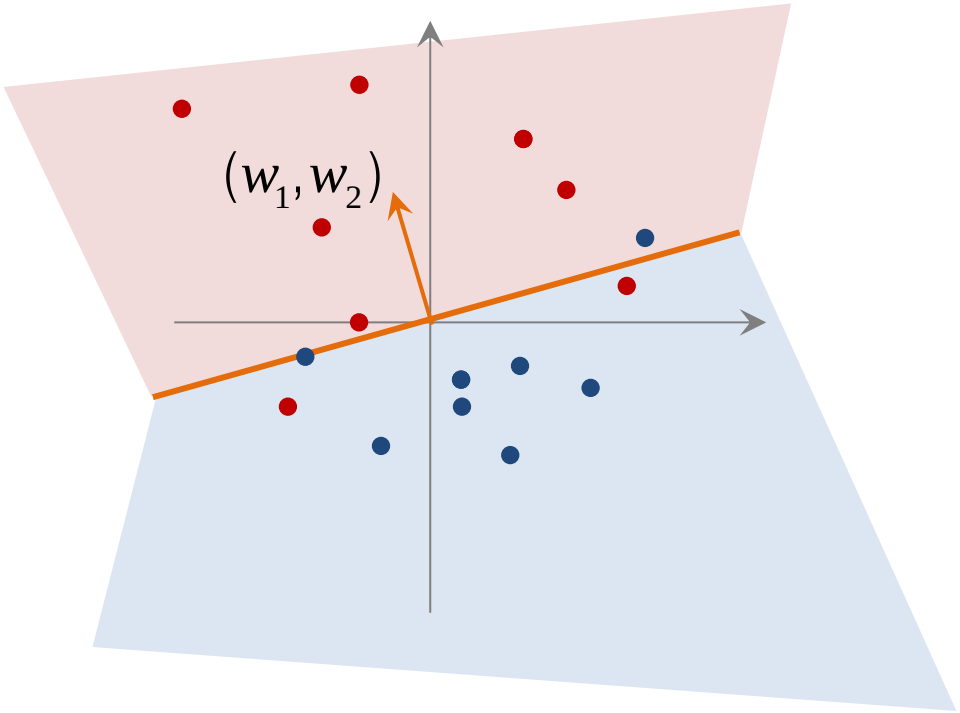


$$L(w_1, w_2) = \sum_i \left(x_1^i w_1 + x_2^i w_2 - y^i \right)^2$$

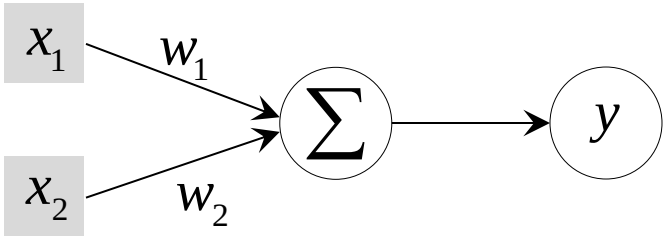
Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

Binary classification



GRADIENT DESCENT



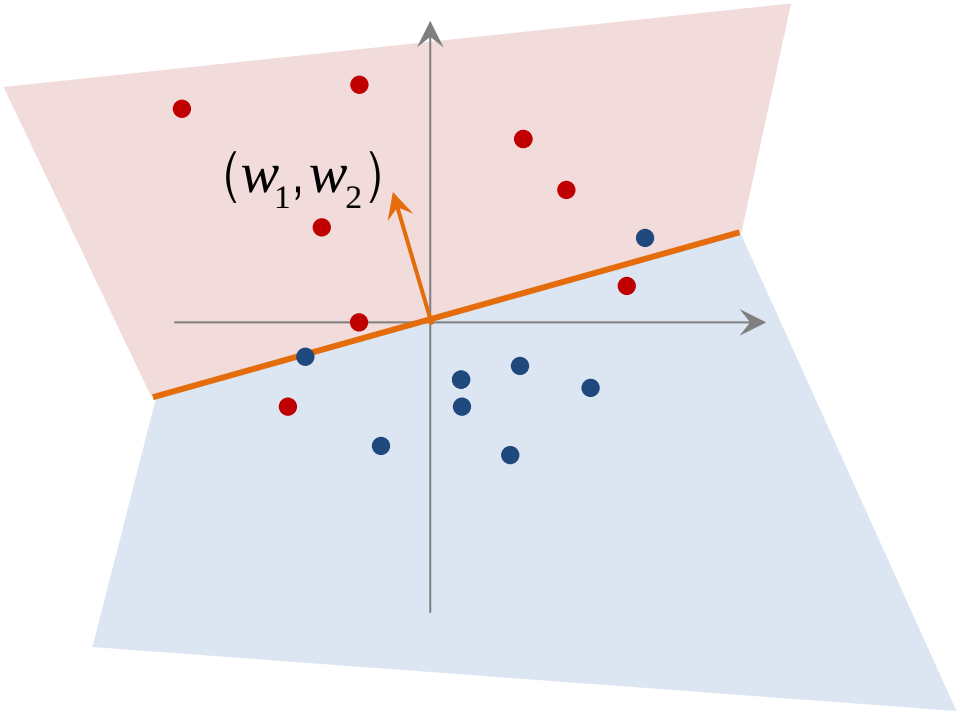
$$L(w_1, w_2) = \sum_i \left(x_1^i w_1 + x_2^i w_2 - y^i \right)^2$$

Weight update rule:

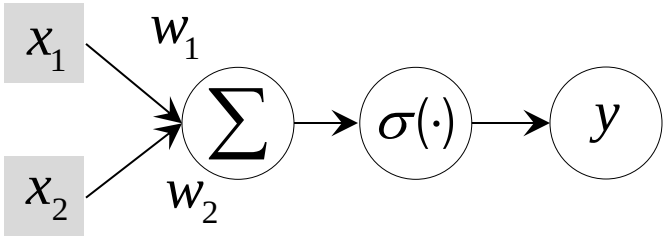
$$w_1 = w_1 - \frac{\gamma}{n} \sum_i (\tilde{y}^i - y^i) x_1^i$$

$$w_2 = w_2 - \frac{\gamma}{n} \sum_i (\tilde{y}^i - y^i) x_2^i$$

Binary classification



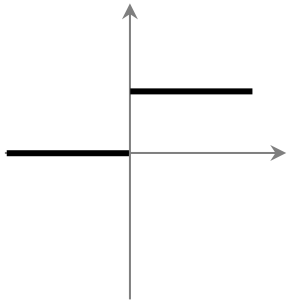
ACTIVATION



$$y = f\left(a\right)$$

$$a = x_1 w_1 + x_2 w_2$$

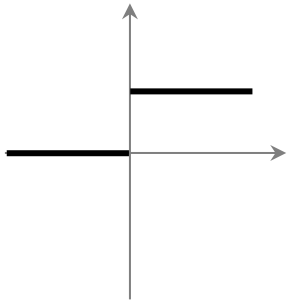
ACTIVATION



Step

$$\sigma(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

ACTIVATION

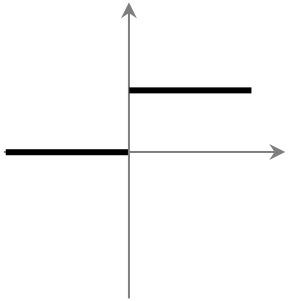


Step

$$\sigma(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$

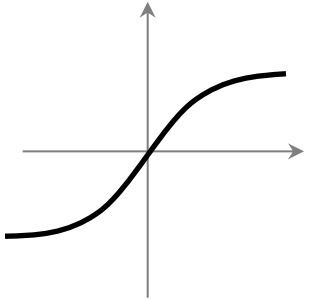
$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

ACTIVATION



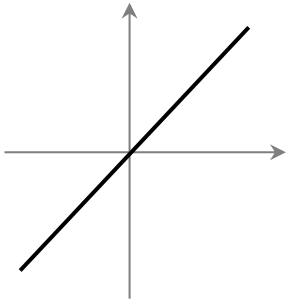
Step

$$\sigma(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$



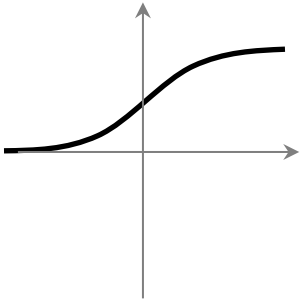
Tanh

$$\sigma(x) = \tanh(x)$$



Linear

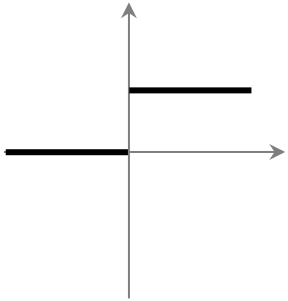
$$\sigma(x) = ax$$



Sigmoid

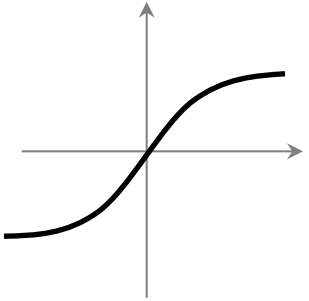
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

ACTIVATION



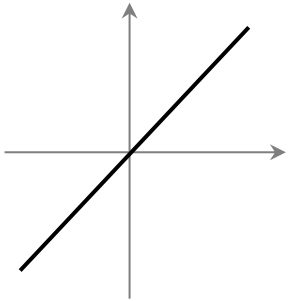
Step

$$\sigma(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$$



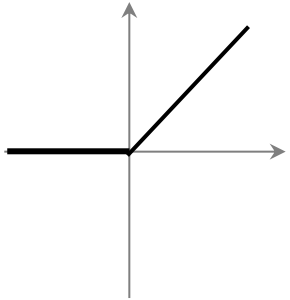
Tanh

$$\sigma(x) = \tanh(x)$$



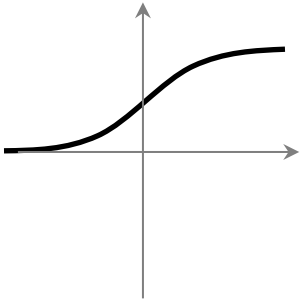
Linear

$$\sigma(x) = ax$$



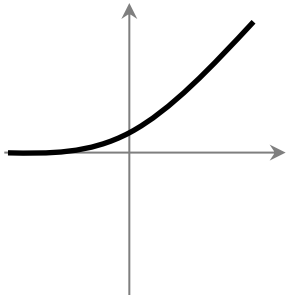
Rect. Linear (ReLu)

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



Sigmoid

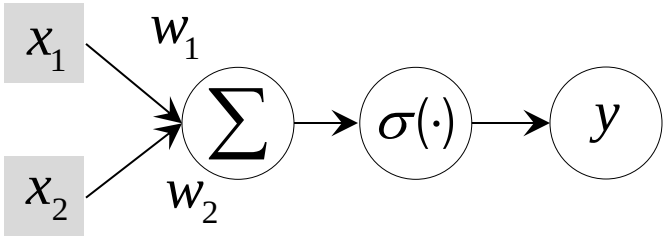
$$\sigma(x) = \frac{1}{1+e^{-x}}$$



Soft Rect. Linear

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

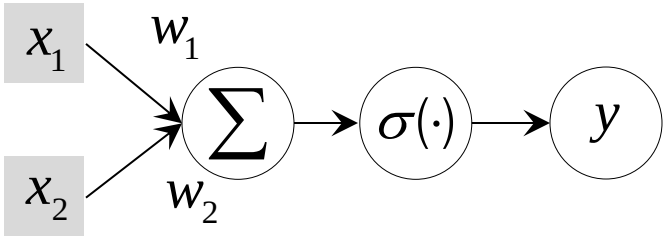
ACTIVATION



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2$$

ACTIVATION



$$f = \sigma(a)$$

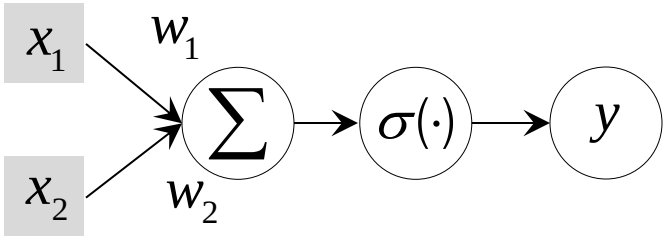
$$a = x_1 w_1 + x_2 w_2$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

ACTIVATION



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

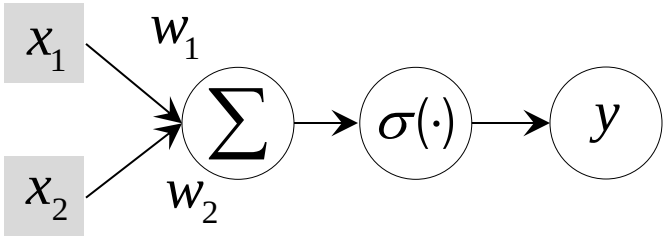
$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f} \frac{\partial f}{\partial a} \frac{\partial a}{\partial w}$$: chain rule

Sigmoid activation

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

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

ACTIVATION



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial w} \quad : \text{chain rule}$$

$$= (\tilde{y} - y)\frac{\partial f}{\partial a}\frac{\partial a}{\partial w}$$

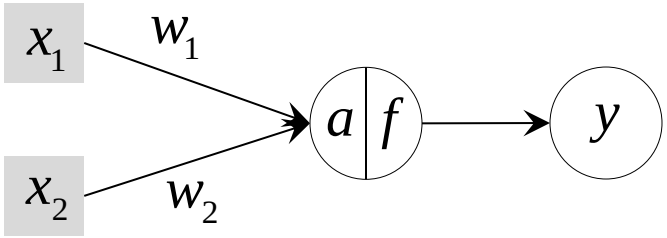
$$= (\tilde{y} - y) f(1 - f) x$$

Sigmoid activation

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

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

ACTIVATION



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial w} \quad : \text{chain rule}$$

$$= (\tilde{y} - y)\frac{\partial f}{\partial a}\frac{\partial a}{\partial w}$$

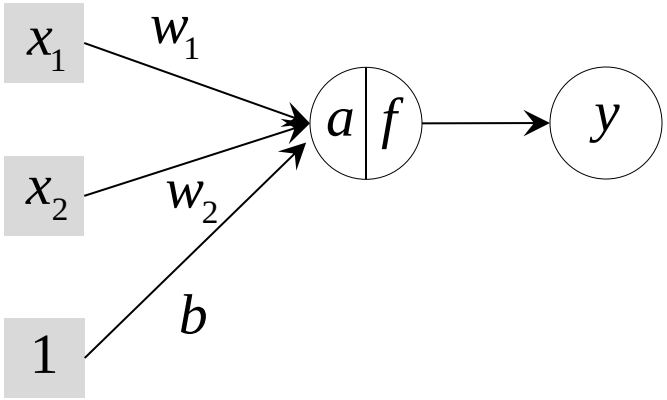
$$= (\tilde{y} - y) f(1 - f) x$$

Sigmoid activation

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

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

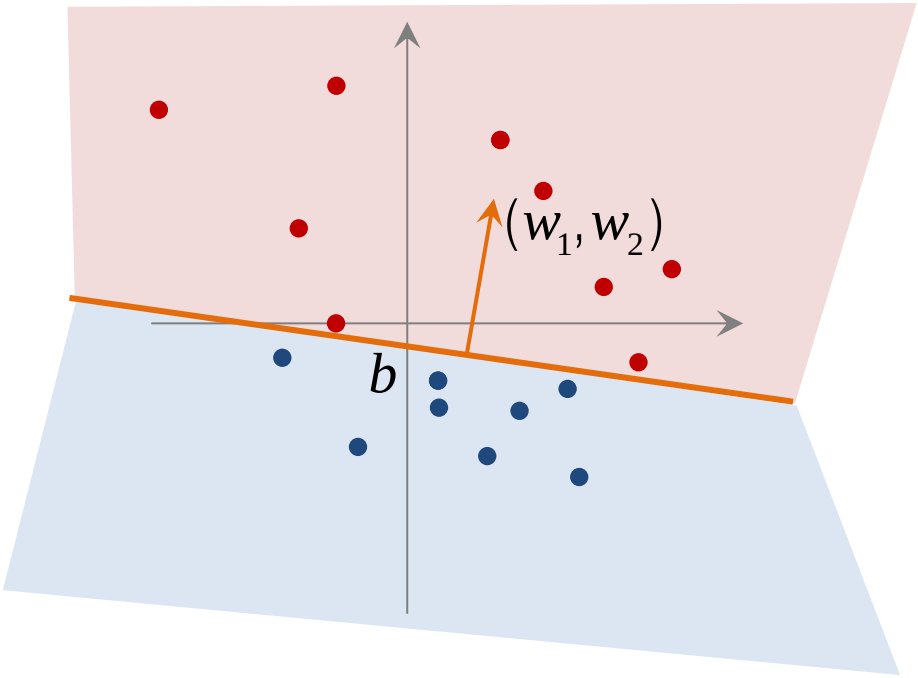
BIAS



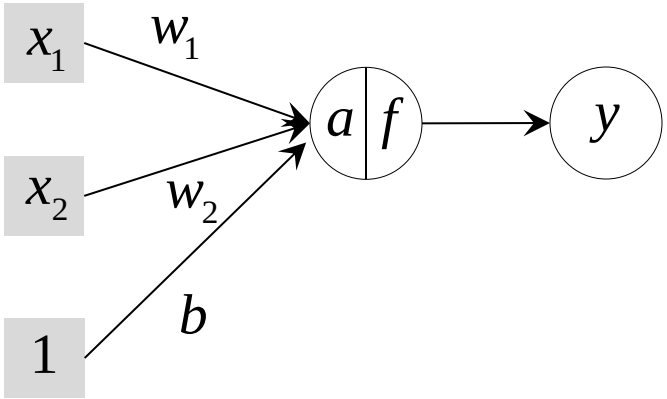
$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2 + 1 \cdot b$$

Binary classification



BIAS



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2 + 1 \cdot b$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

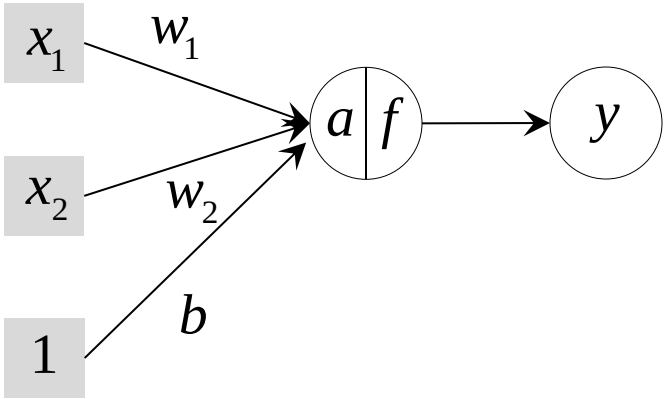
$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f} \frac{\partial f}{\partial a} \frac{\partial a}{\partial b}$$: chain rule

Sigmoid activation

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

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

BIAS



$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2 + 1 \cdot b$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial b} \quad \text{: chain rule}$$

$$= (\tilde{y} - y)\frac{\partial f}{\partial a}\frac{\partial a}{\partial b}$$

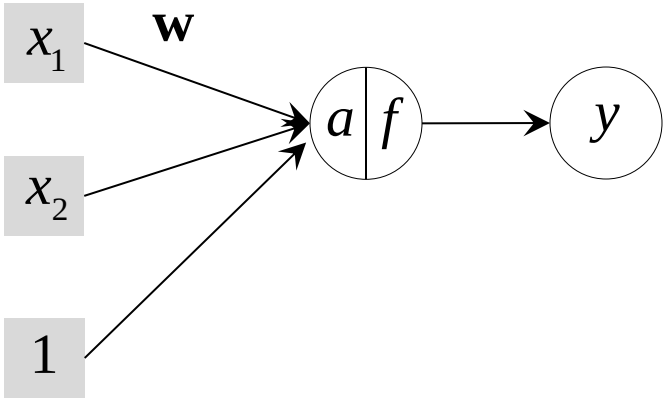
$$= (\tilde{y} - y) f(1 - f)$$

Sigmoid activation

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

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

EXPRESSION



$$f = \sigma(\mathbf{w} \cdot \mathbf{x})$$

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$

Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial b} \quad \text{: chain rule}$$

$$= (\tilde{y} - y)\frac{\partial f}{\partial a}\frac{\partial a}{\partial b}$$

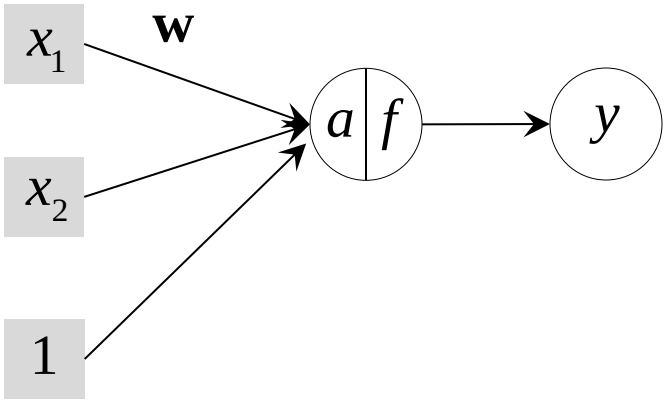
$$= (\tilde{y} - y) f(1 - f)$$

Sigmoid activation

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

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

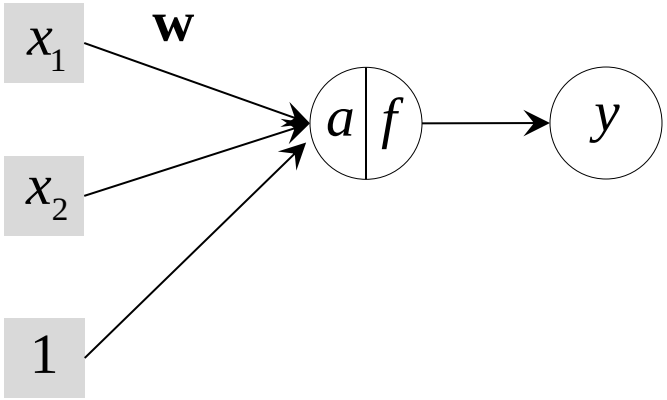
EXPRESSION



How many variables?

$$f = \sigma(\mathbf{w} \cdot \mathbf{x})$$

EXPRESSION



$$f = \sigma(\mathbf{w} \cdot \mathbf{x})$$

How many variables?

n+1

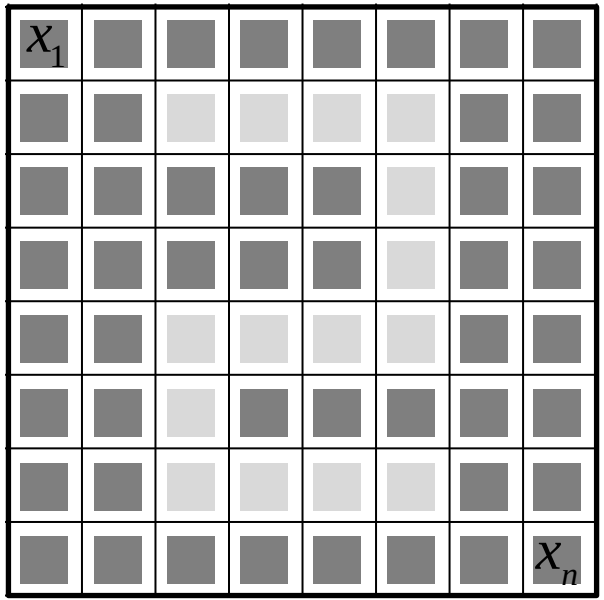
$$\frac{\partial L}{\partial \mathbf{w}} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial \mathbf{w}}$$

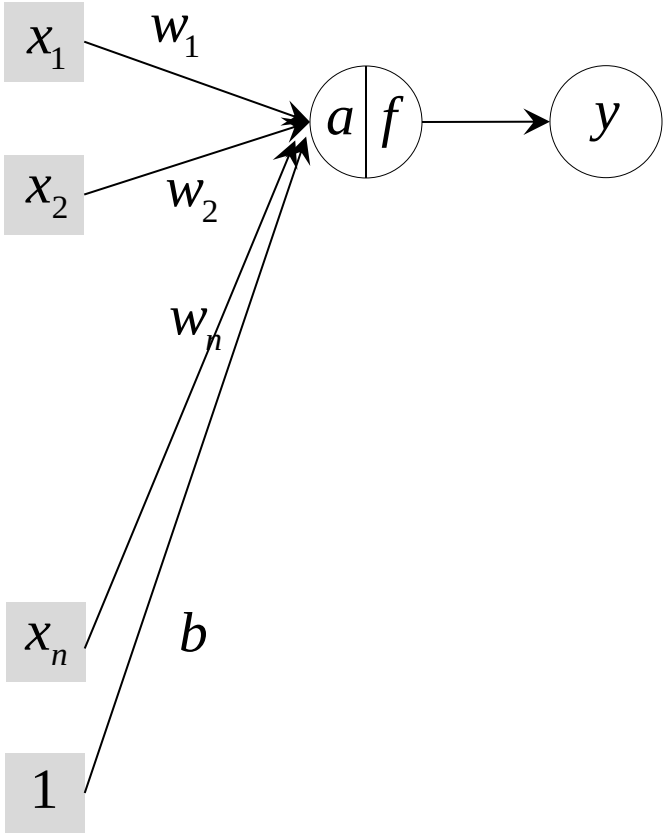
1x3 1x1 1x1 1x3

$$f = \sigma(a)$$

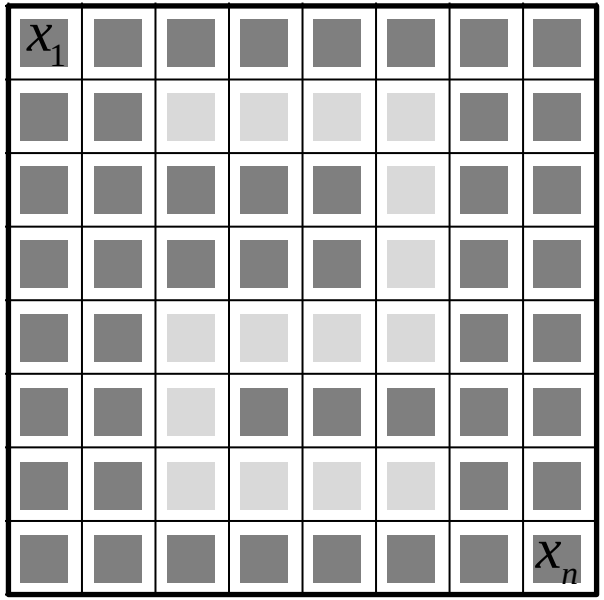
$$a = x_1 w_1 + x_2 w_2 + 1 \cdot b$$

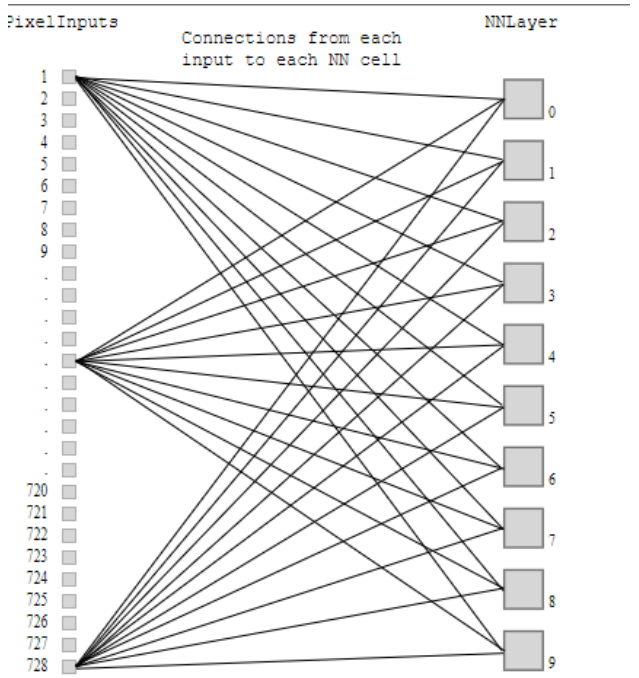
DIGIT RECOGNITION

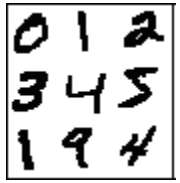




DIGIT RECOGNITION





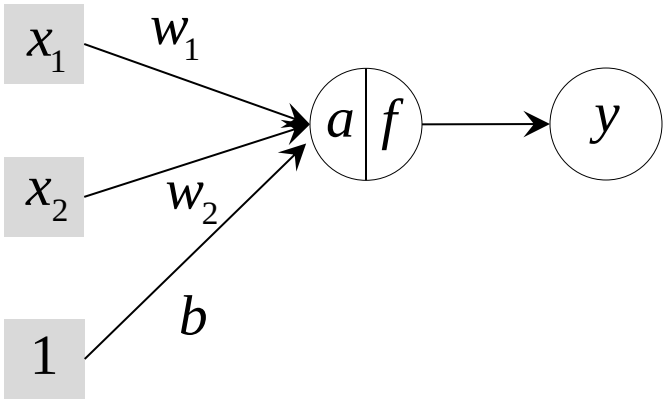


Training error: 99.97%

Testing error: 85% (88% by LeCunn)

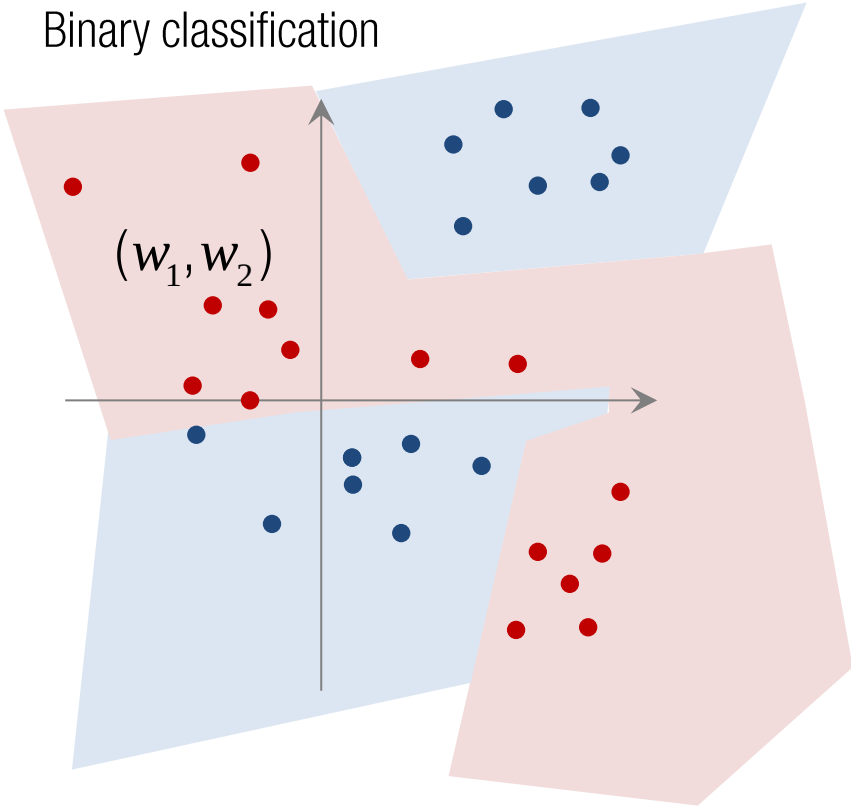
Can we do better?

LINEARLY NON-SEPARABLE DATA

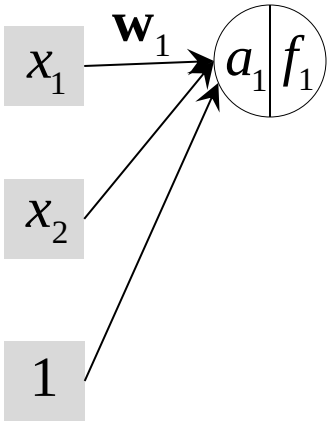


$$f = \sigma(\mathbf{w} \cdot \mathbf{x})$$

Binary classification



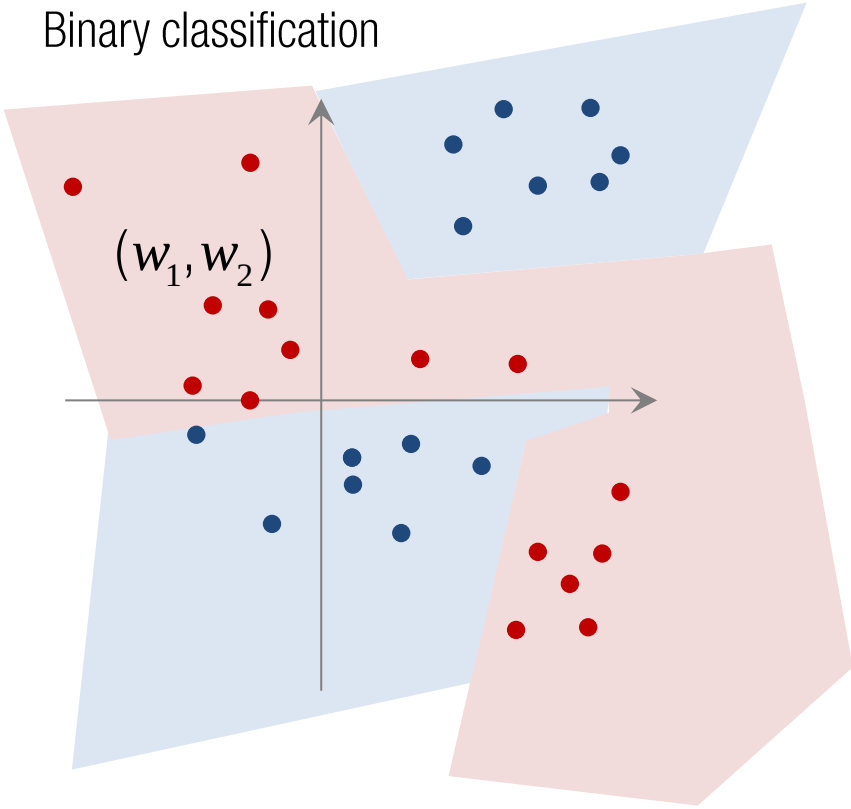
MULTI-LAYER PERCEPTRON



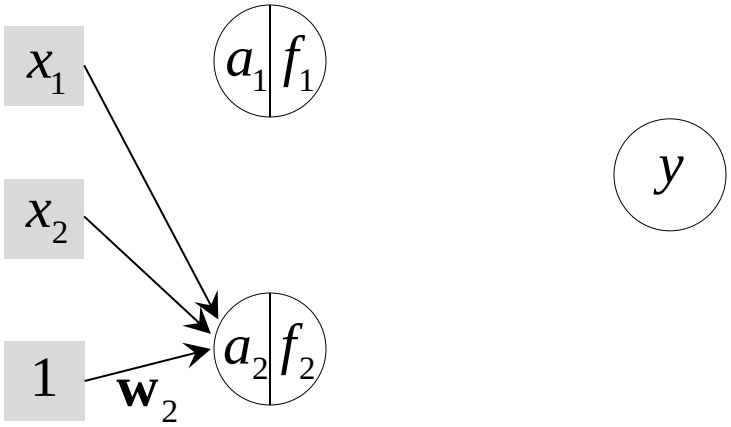
y

$$f_1 = \sigma(\mathbf{w}_1 \cdot \mathbf{x})$$

Binary classification



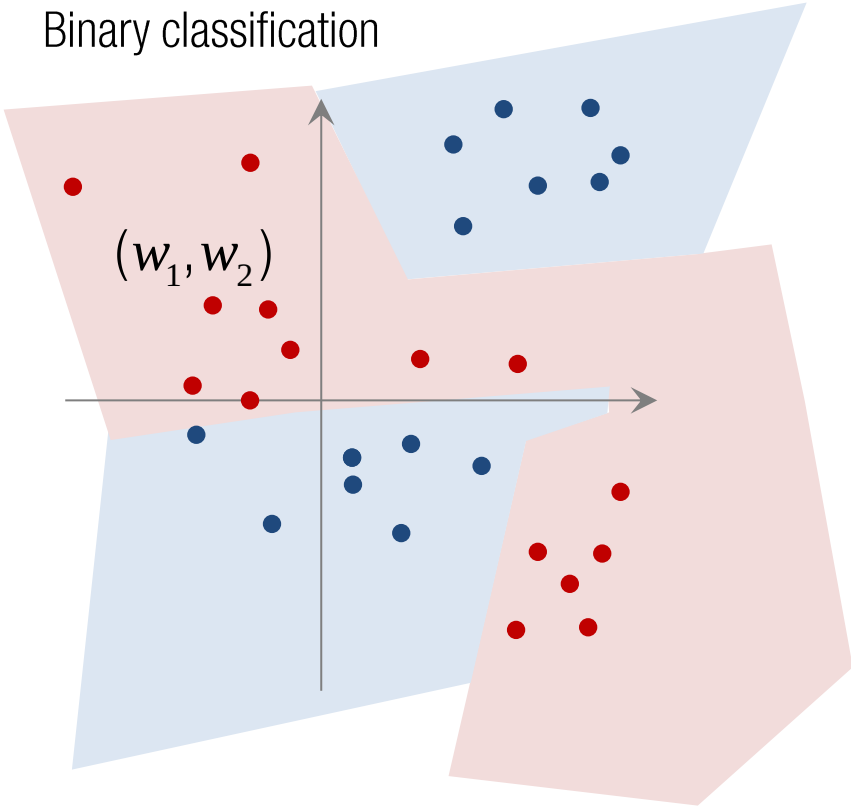
MULTI-LAYER PERCEPTRON



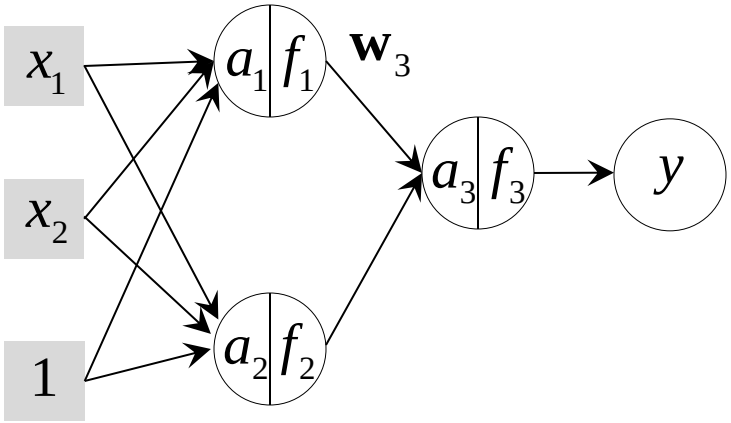
y

$$f_1 = \sigma(\mathbf{w}_1 \cdot \mathbf{x})$$
$$f_2 = \sigma(\mathbf{w}_2 \cdot \mathbf{x})$$

Binary classification

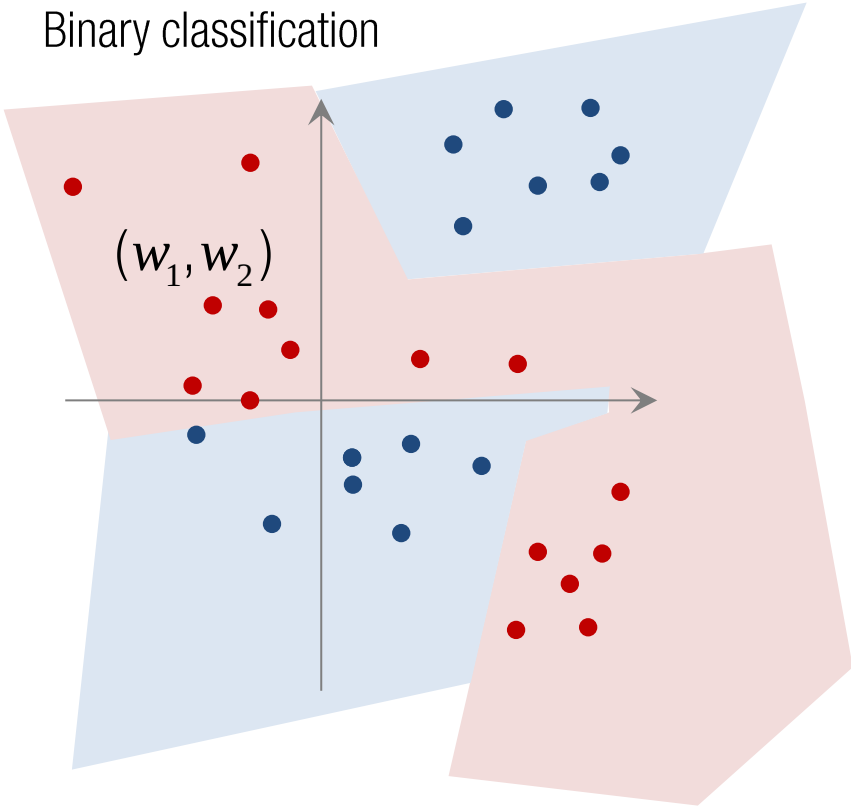


MULTI-LAYER PERCEPTRON

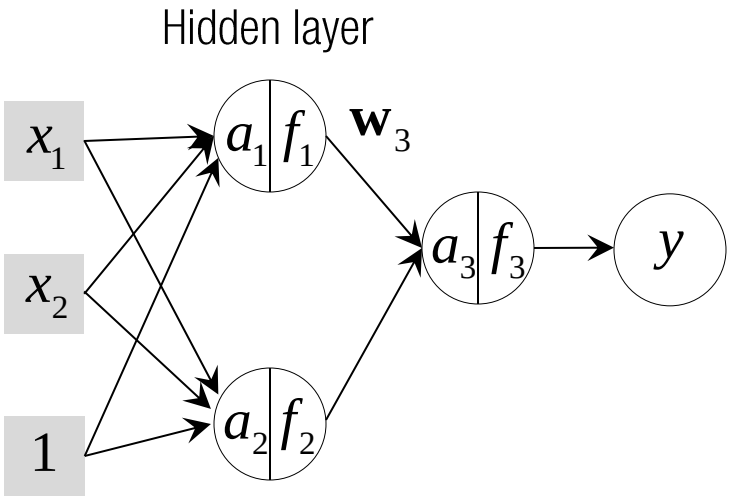


$$f_1 = \sigma(\mathbf{w}_1 \cdot \mathbf{x})$$
$$f_2 = \sigma(\mathbf{w}_2 \cdot \mathbf{x})$$
$$f_3 = \sigma\left(\mathbf{w}_3 \cdot \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}\right)$$

Binary classification



MULTI-LAYER PERCEPTRON

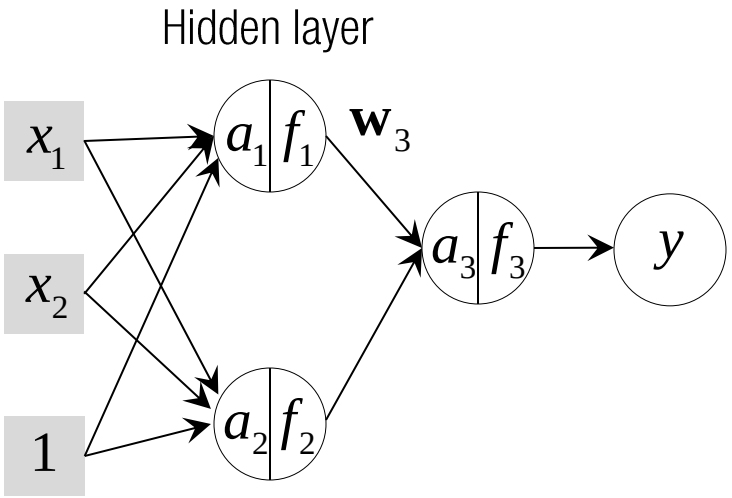


How many variables?

$$f_1 = \sigma(\mathbf{w}_1 \cdot \mathbf{x})$$
$$f_2 = \sigma(\mathbf{w}_2 \cdot \mathbf{x})$$

$$f_3 = \sigma\left(\mathbf{w}_3 \cdot \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}\right)$$

MULTI-LAYER PERCEPTRON



How many variables?

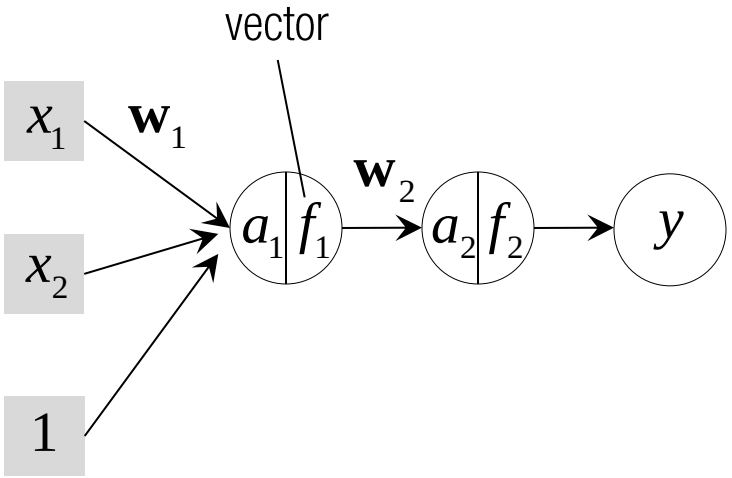
m(n+1) + m

n: input dimension
m: # of hidden variables

$$f_1 = \sigma(\mathbf{w}_1 \cdot \mathbf{x})$$
$$f_2 = \sigma(\mathbf{w}_2 \cdot \mathbf{x})$$

$$f_3 = \sigma\left(\mathbf{w}_3 \cdot \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}\right)$$

MULTI-LAYER PERCEPTRON



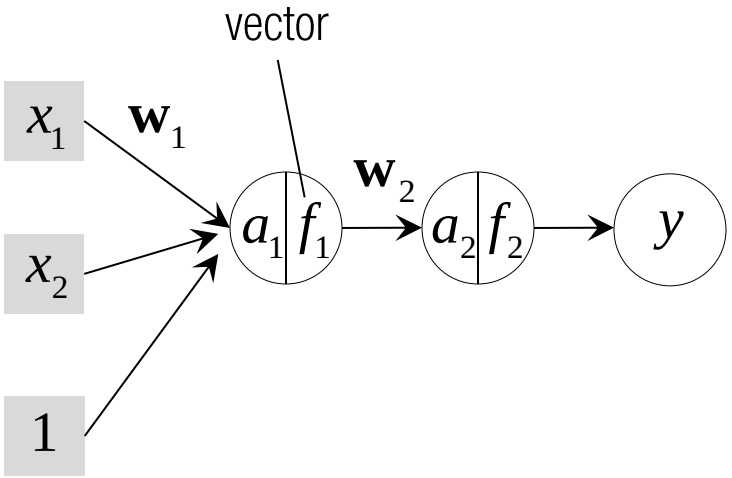
How many variables?

m(n+1) + m

n: input dimension
m: # of hidden variables

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

MULTI-LAYER PERCEPTRON



How many variables?

m(n+1) + m

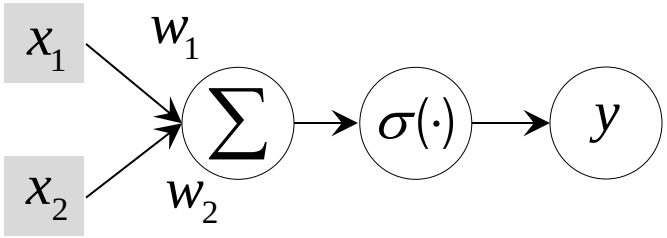
n: input dimension
m: # of hidden variables

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

How to train?

RECALL: SINGLE PERCEPTRON

$$L(w_1, w_2) = \sum_i \left(f(x_1^i w_1 + x_2^i w_2) - y^i \right)^2$$



Weight update rule:

$$w_1 = w_1 - \gamma \frac{\partial L}{\partial w_1} \qquad w_2 = w_2 - \gamma \frac{\partial L}{\partial w_2}$$

$$f = \sigma(a)$$

$$a = x_1 w_1 + x_2 w_2$$

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial a}\frac{\partial a}{\partial w} \quad \text{: chain rule}$$

$$= (\tilde{y} - y)\frac{\partial f}{\partial a}\frac{\partial a}{\partial w}$$

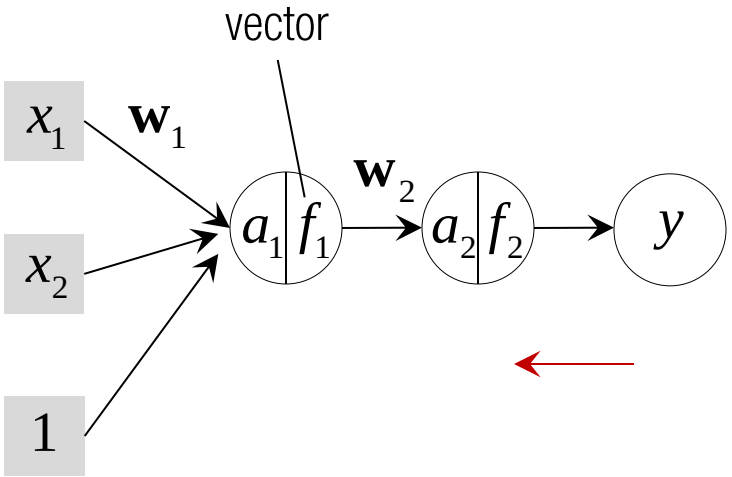
$$= (\tilde{y} - y) f(1 - f) x$$

Sigmoid activation

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

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

BACK-PROPAGATION



$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i \right)^2$$

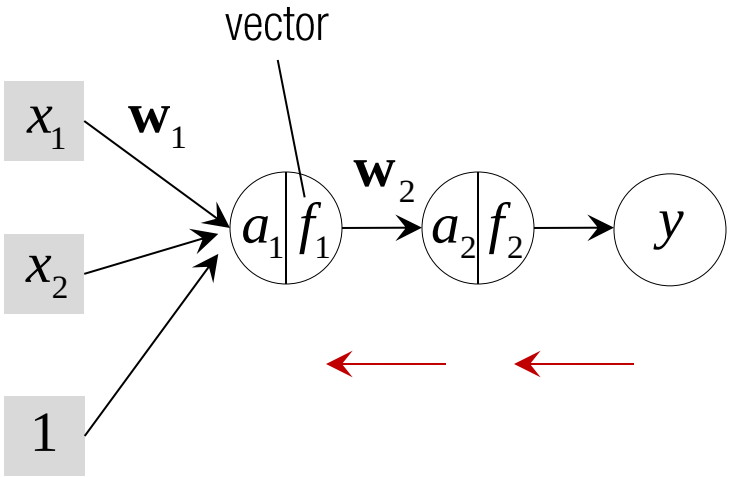
Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial \mathbf{w}_2}$$

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x} \right) \right)$$

BACK-PROPAGATION



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

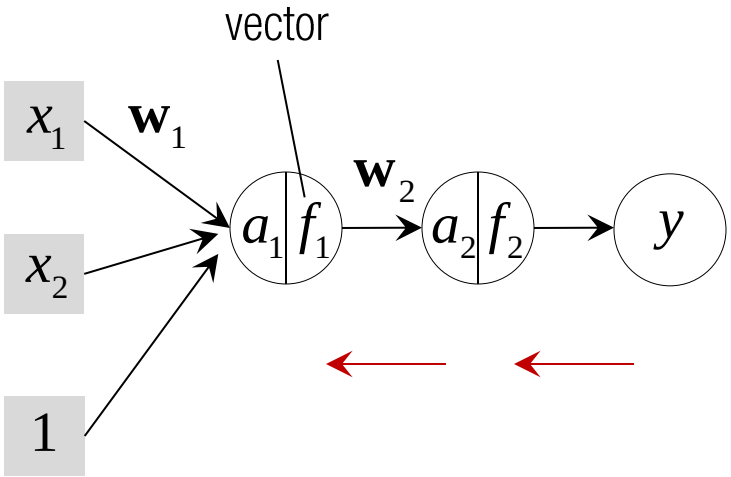
Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial \mathbf{w}_2}$$

$$\frac{\partial L}{\partial \mathbf{w}_1} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial f_1} \frac{\partial f_1}{\partial a_1} \frac{\partial a_1}{\partial \mathbf{w}_1}$$

BACK-PROPAGATION



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

Weight update rule:

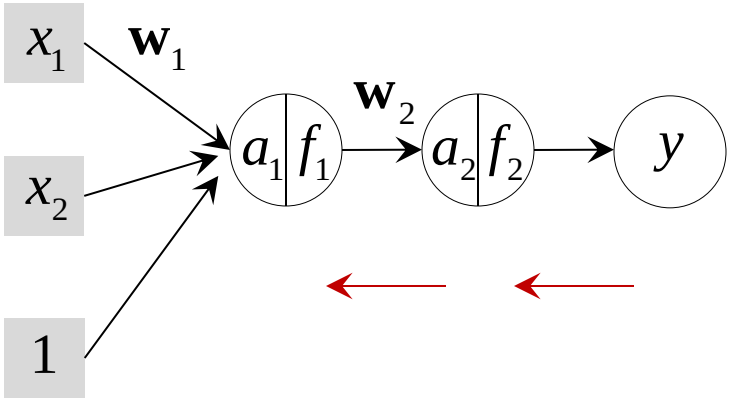
$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \boxed{\frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}}\frac{\partial a_2}{\partial \mathbf{w}_2}$$

$$\frac{\partial L}{\partial \mathbf{w}_1} = \boxed{\frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

BACK-PROPAGATION

5 hidden var.



$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i \right)^2$$

Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial \mathbf{w}_2}$$

$$f_2 = \sigma(a_2)$$

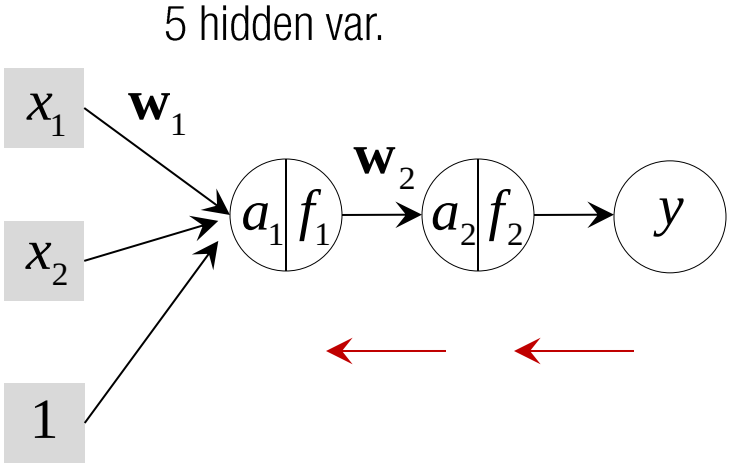
$$a_2 = \mathbf{w}_2 \cdot f_1$$

$$f_1 = \sigma(a_1)$$

$$a_1 = \mathbf{w}_1 \cdot x$$

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x} \right) \right)$$

BACK-PROPAGATION



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial \mathbf{w}_2}$$

1x5 1x1 1x1 1x5

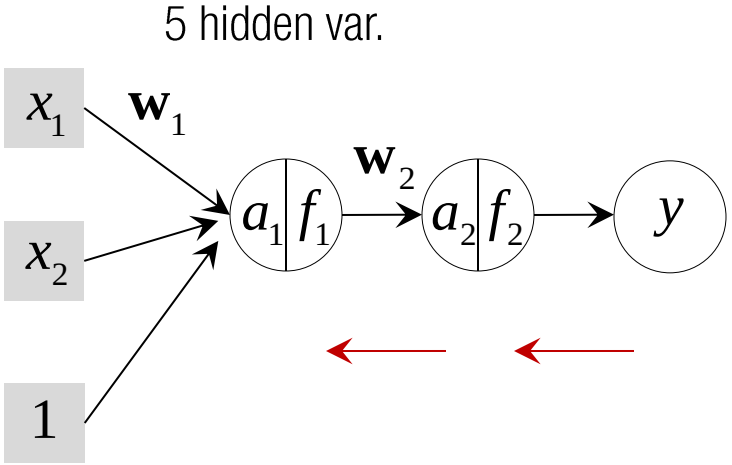
$$f_2 = \sigma(a_2)$$

$$a_2 = \mathbf{w}_2 \cdot f_1$$

$$f_1 = \sigma(a_1)$$

$$a_1 = \mathbf{w}_1 \cdot x$$

BACK-PROPAGATION



$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i \right)^2$$

Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

$$\frac{\partial L}{\partial \mathbf{w}_1} = \frac{\partial L}{\partial f_2} \frac{\partial f_2}{\partial a_2} \frac{\partial a_2}{\partial f_1} \frac{\partial f_1}{\partial a_1} \frac{\partial a_1}{\partial \mathbf{w}_1}$$

$$f_2 = \sigma(a_2)$$

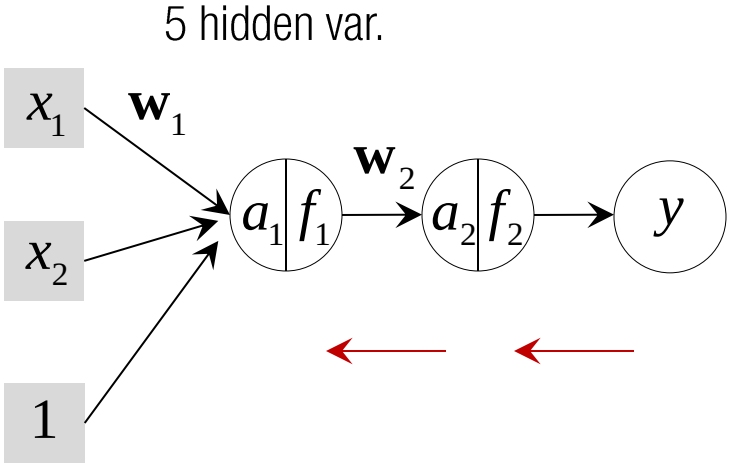
$$a_2 = \mathbf{w}_2 \cdot f_1$$

$$f_1 = \sigma(a_1)$$

$$a_1 = \mathbf{w}_1 \cdot x$$

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x} \right) \right)$$

BACK-PROPAGATION



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

Weight update rule:

$$\mathbf{w} = \mathbf{w} - \gamma \frac{\partial L}{\partial \mathbf{w}}$$

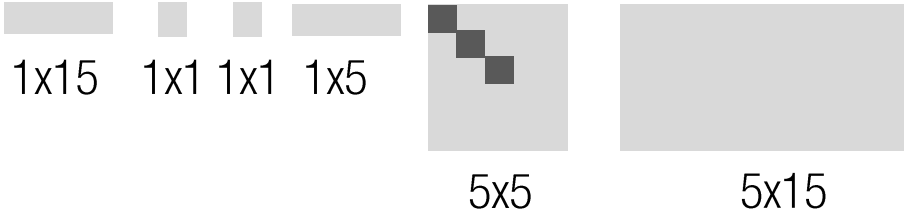
$$\frac{\partial L}{\partial \mathbf{w}_1} = \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

$$f_2 = \sigma(a_2)$$

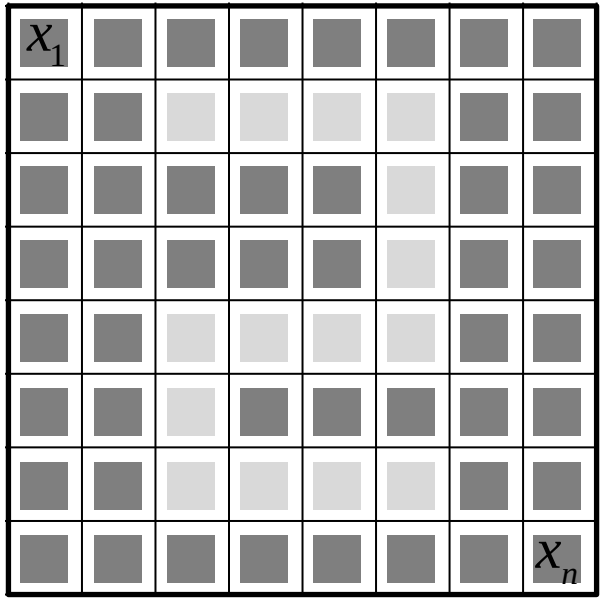
$$a_2 = \mathbf{w}_2 \cdot f_1$$

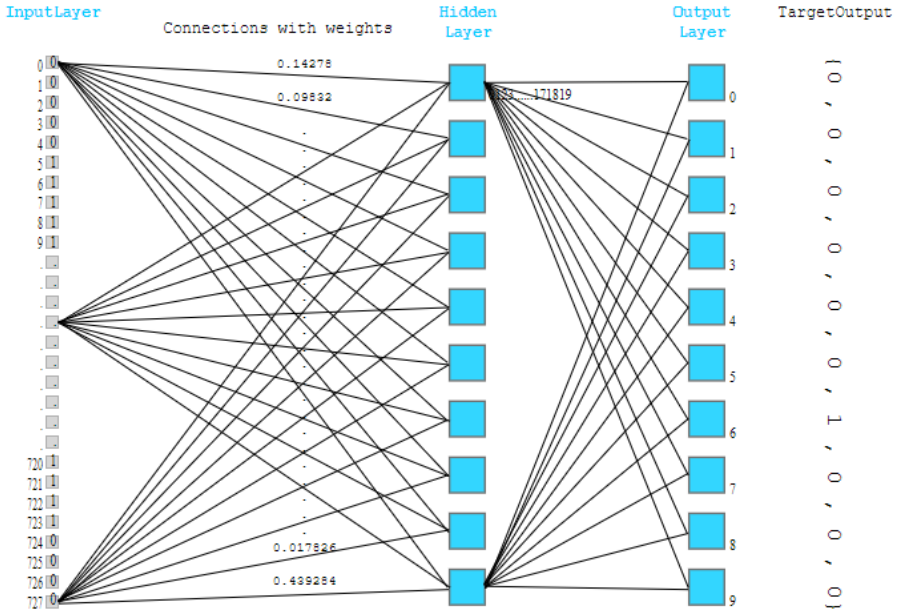
$$f_1 = \sigma(a_1)$$

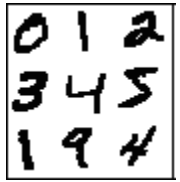
$$a_1 = \mathbf{w}_1 \cdot x$$



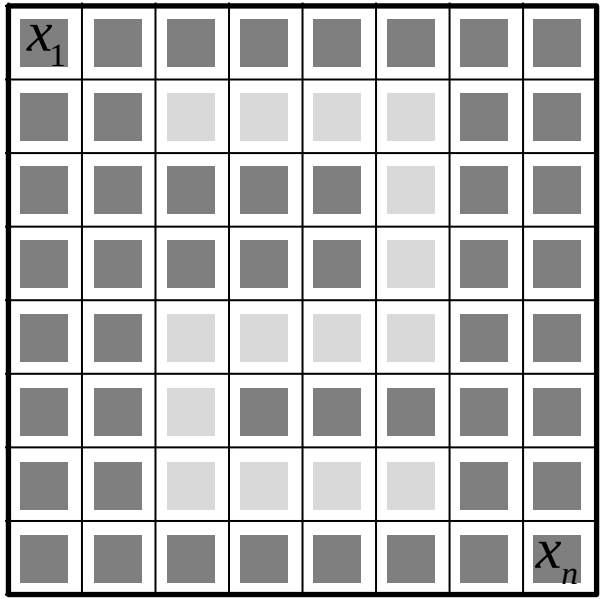
DIGIT RECOGNITION

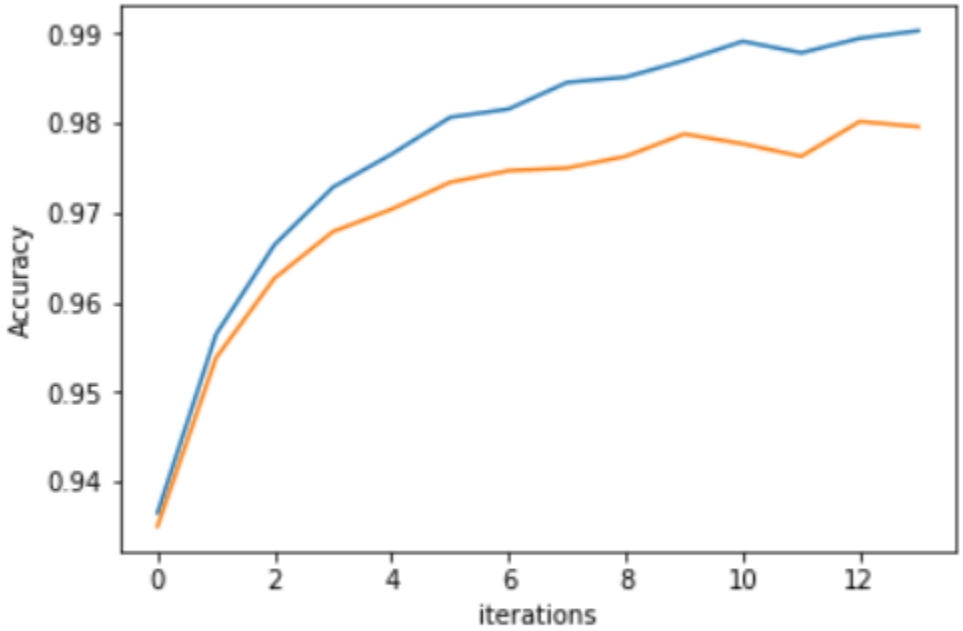






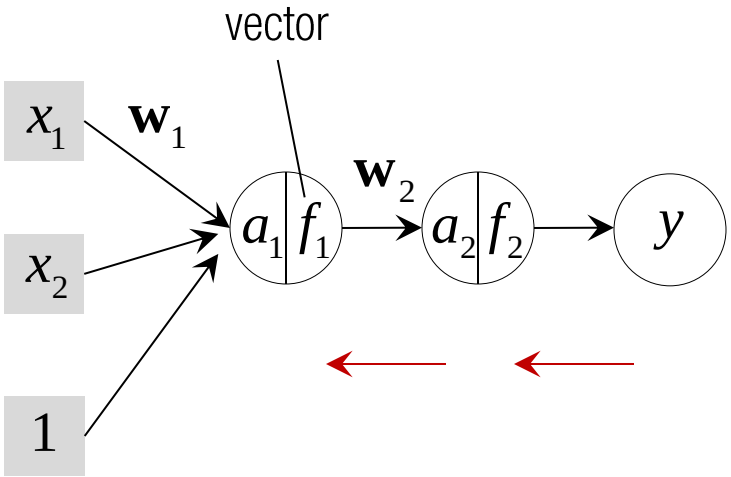
DIGIT RECOGNITION






```
Train Accuracy: 0.99
Test Accuracy:0.98
```

BACK-PROPAGATION ALGORITHM W/ GRADIENT DESCENT



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

While until converges:

For each data sample,

1. Prediction

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

2. Measure error

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

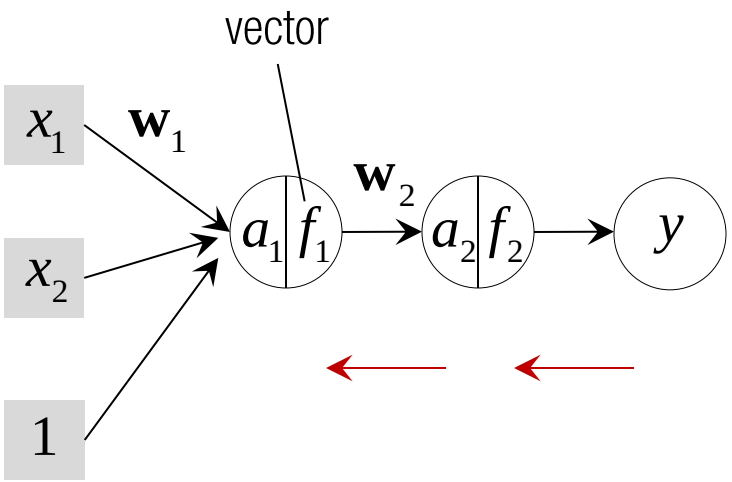
3. Back-propagation

$$\frac{\partial L}{\partial \mathbf{w}_2} \mathrel{+}= \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{w}_2} \qquad \frac{\partial L}{\partial \mathbf{w}_1} \mathrel{+}= \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

Update gradient

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1} \qquad \mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

GRADIENT DESCENT



Weight update rule:

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1}$$

$$\mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

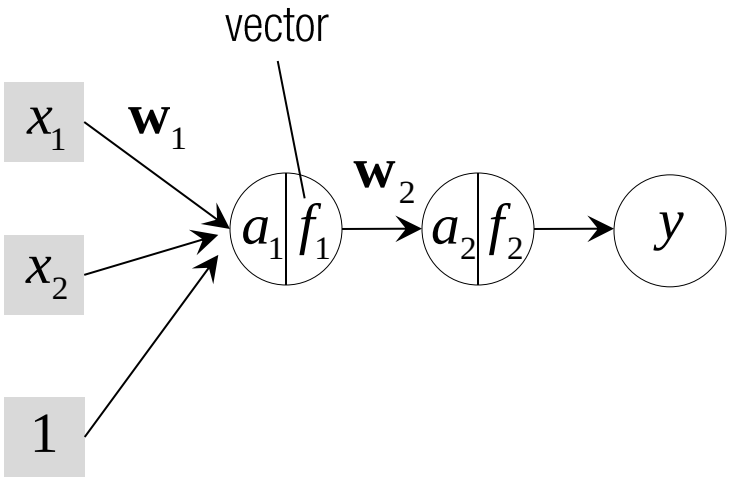
where

$$\frac{\partial L}{\partial \mathbf{w}_1} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial L_i}{\partial \mathbf{w}_1}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial L_i}{\partial \mathbf{w}_2}$$

n : the number of training data (> millions)

STOCHASTIC GRADIENT DESCENT (SGD)



Weight update rule:

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1}$$

$$\mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

where

$$\frac{\partial L}{\partial \mathbf{w}_1} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial L_i}{\partial \mathbf{w}_1}$$

$$\frac{\partial L}{\partial \mathbf{w}_2} = \frac{1}{n}\sum_{i=1}^{n} \frac{\partial L_i}{\partial \mathbf{w}_2}$$

n : the number of training data (> millions)

$$\frac{\partial L}{\partial \mathbf{w}_1} \approx \frac{\partial L_i}{\partial \mathbf{w}_1} \qquad \frac{\partial L}{\partial \mathbf{w}_2} \approx \frac{\partial L_i}{\partial \mathbf{w}_2}$$

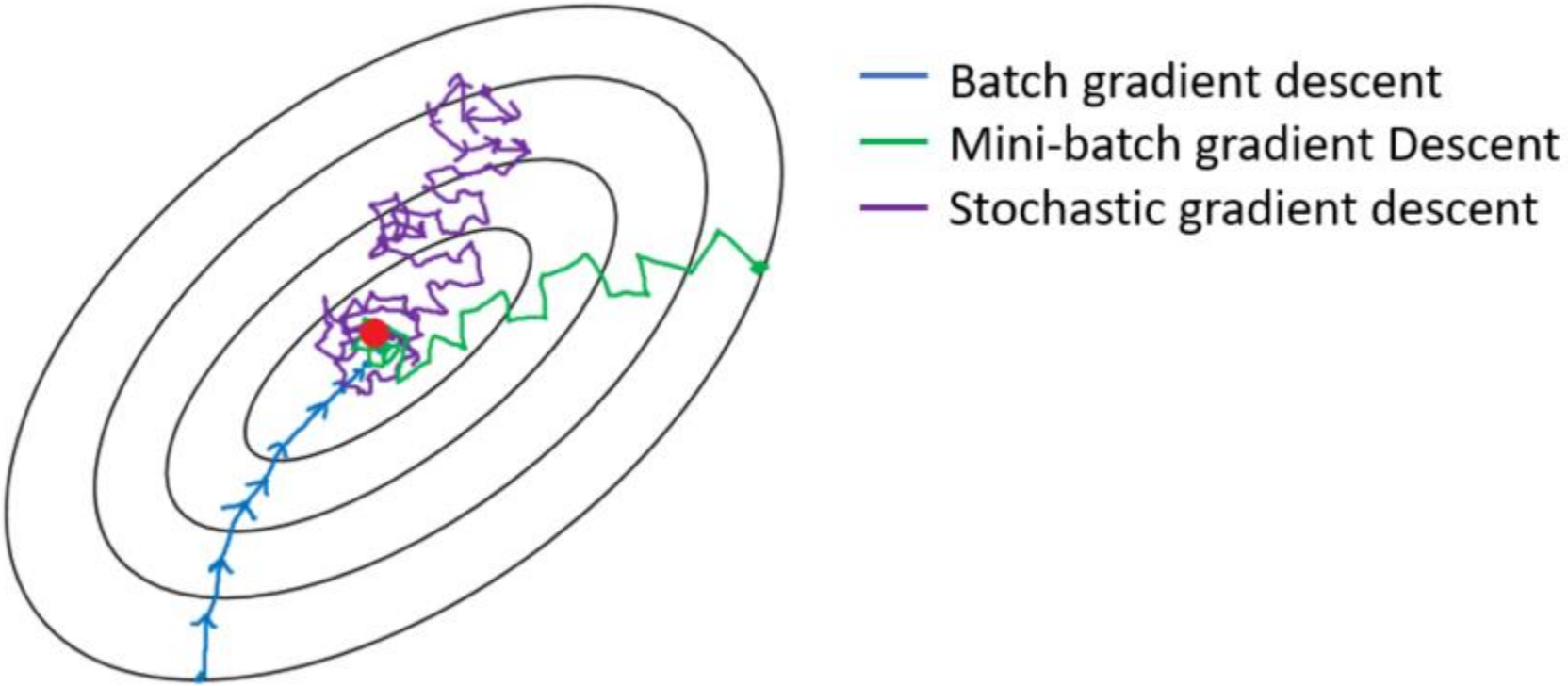
or

$$\frac{\partial L}{\partial \mathbf{w}_1} \approx \sum_{i\in\mathcal{R}} \frac{\partial L_i}{\partial \mathbf{w}_1} \qquad \frac{\partial L}{\partial \mathbf{w}_2} \approx \sum_{i\in\mathcal{R}} \frac{\partial L_i}{\partial \mathbf{w}_2}$$

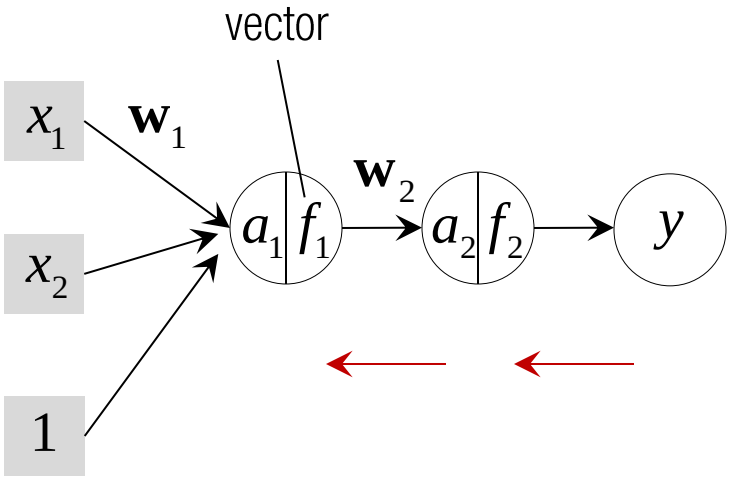
$$|\mathcal{R}| \ll n$$

$\mathcal{R}$: random subset of training data, a.k.a. mini-batch

STOCHASTIC GRADIENT DESCENT (SGD)



BACK-PROPAGATION ALGORITHM W/ GRADIENT DESCENT



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

While until converges:

For each data sample,

1. Prediction

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

2. Measure error

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

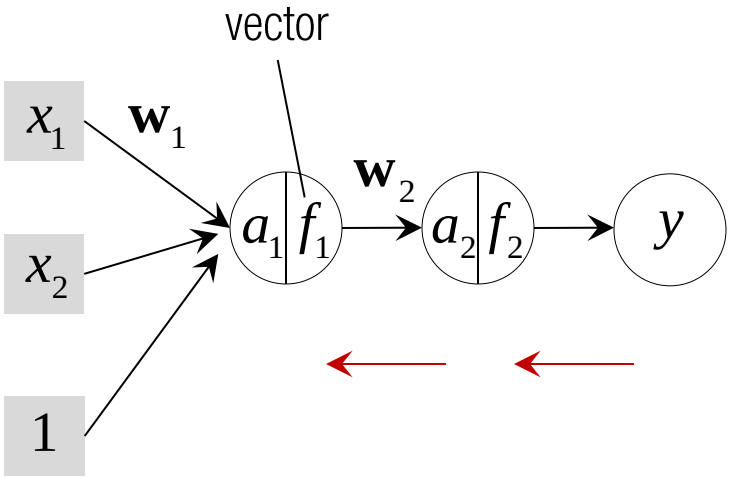
3. Back-propagation

$$\frac{\partial L}{\partial \mathbf{w}_2} \mathrel{+}= \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{w}_2} \qquad \frac{\partial L}{\partial \mathbf{w}_1} \mathrel{+}= \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

Update gradient

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1} \qquad \mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

BACK-PROPAGATION ALGORITHM W/ STOCHASTIC GRADIENT DESCENT



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

While until converges:

For each data sample,

1. Prediction

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

2. Measure error

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

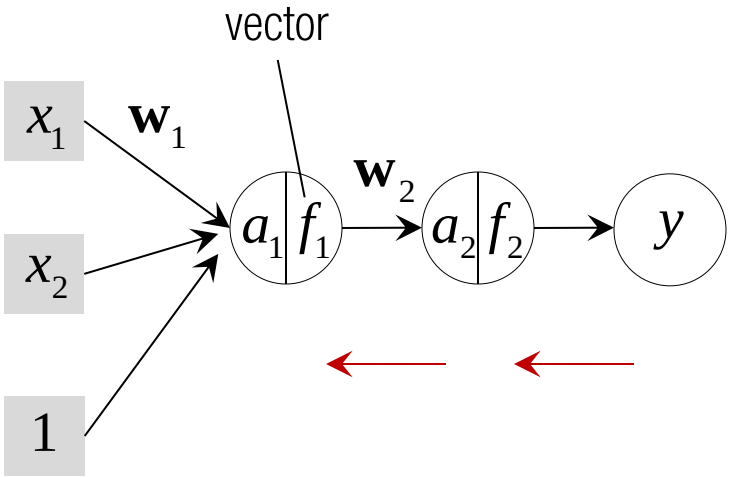
3. Back-propagation

$$\frac{\partial L}{\partial \mathbf{w}_2} += \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{w}_2} \qquad \frac{\partial L}{\partial \mathbf{w}_1} += \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

4. Update gradient

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1} \qquad \mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

Back-propagation Algorithm w/ Stochastic Gradient Descent



$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

While until converges:

Sample mini-batch

For each data sample in mini-batch,

1. Prediction

$$f_2 = \sigma\left(\mathbf{w}_2 \cdot \sigma\left(\mathbf{w}_1 \cdot \mathbf{x}\right)\right)$$

2. Measure error

$$L(\mathbf{w}_1, \mathbf{w}_2) = \sum_i \left(\tilde{y}^i - y^i\right)^2$$

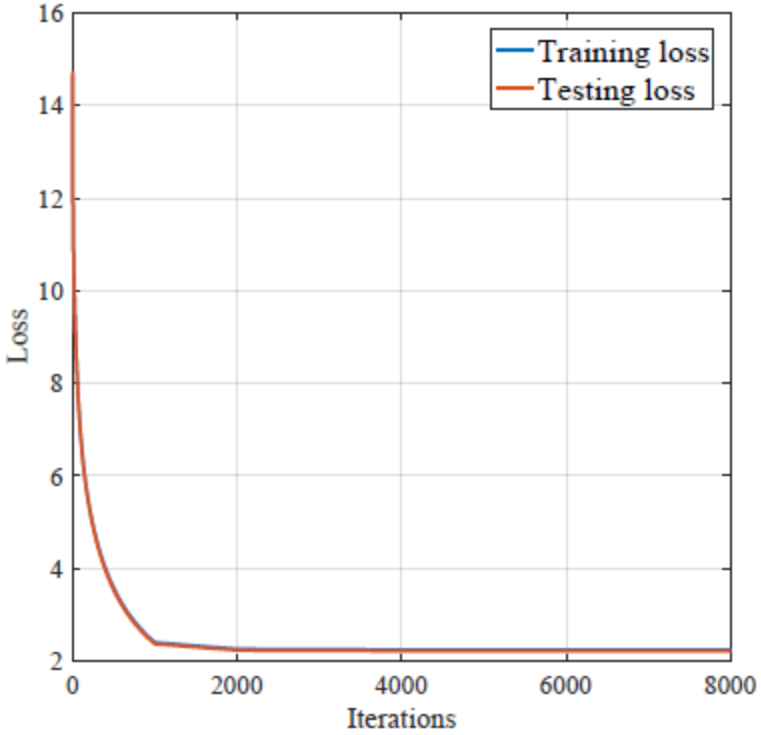
3. Back-propagation

$$\frac{\partial L}{\partial \mathbf{w}_2} += \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial \mathbf{w}_2} \qquad \frac{\partial L}{\partial \mathbf{w}_1} += \frac{\partial L}{\partial f_2}\frac{\partial f_2}{\partial a_2}\frac{\partial a_2}{\partial f_1}\frac{\partial f_1}{\partial a_1}\frac{\partial a_1}{\partial \mathbf{w}_1}$$

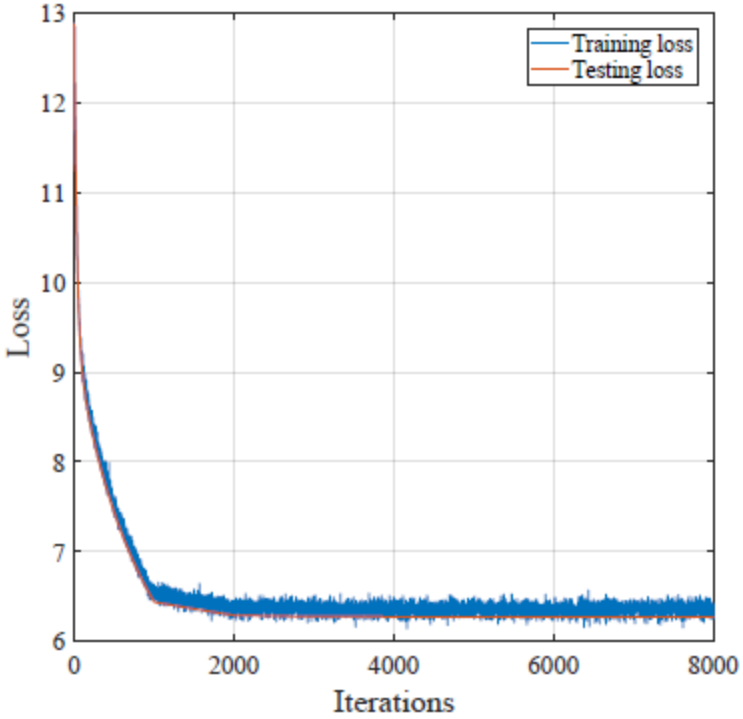
Update gradient

$$\mathbf{w}_1 = \mathbf{w}_1 - \gamma \frac{\partial L}{\partial \mathbf{w}_1} \qquad \mathbf{w}_2 = \mathbf{w}_2 - \gamma \frac{\partial L}{\partial \mathbf{w}_2}$$

Gradient descent



SGD with mini-batch



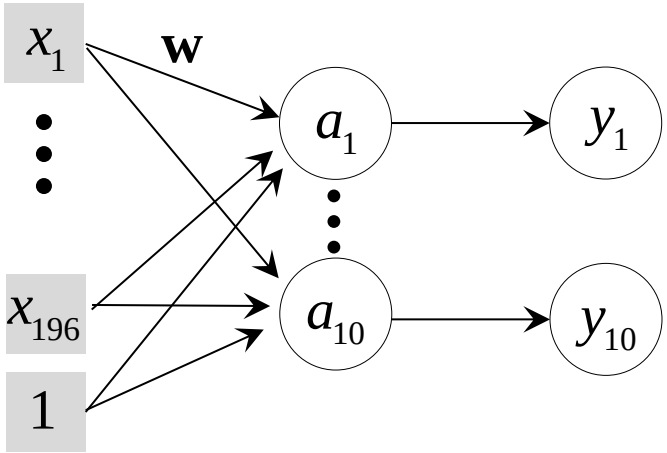
HW #4: Neural Network Implementation

Algorithm 2 Stochastic Gradient Descent based Training

Set the learning rate γ
Set the decay rate $\lambda \in (0, 1]$
Initialize the weights with a Gaussian noise $\mathbf{w} \in \mathcal{N}(0, 1)$
for iIter $= 1 :$ nIters **do**
    At every 100^{th} iteration, $\gamma \leftarrow \lambda\gamma$
    $\frac{\partial L}{\partial \mathbf{w}} \leftarrow 0$
    Sample R images randomly.
    for $i = 1 : R$ **do**
        Label prediction of $\mathbf{x}_i$
        Loss computation l
        Gradient back-propagation of $\mathbf{x}_i$, $\frac{\partial l}{\partial \mathbf{w}}$ using back-propagation.
        $\frac{\partial L}{\partial \mathbf{w}} = \frac{\partial L}{\partial \mathbf{w}} + \frac{\partial l}{\partial \mathbf{w}}$
    end for
    Update the weights, $\mathbf{w} \leftarrow \mathbf{w} - \frac{\gamma}{R}\frac{\partial L}{\partial \mathbf{w}}$
end for

HW #4: Neural Network Implementation

HW #4: Neural Network Implementation
Step 1: Single Linear Perceptron




```
function a = Multiply(x, w)
function dadw = Multiply_backward_w(x, w)
function l = Loss(y, y_tilde)
function dldy_tilde = Loss_backward(y, y_tilde)
```

One hot representation of label:

$$[y_1 \quad \cdots \quad y_{10}] = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]$$

Label: 0 1 2 3 4 5 7 6 8 9

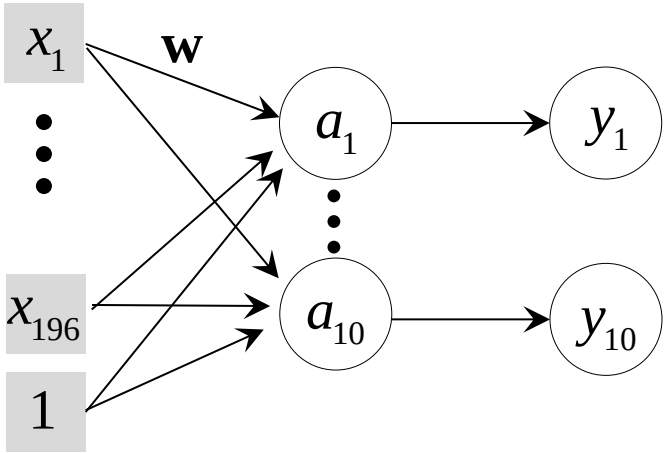
HW #4: Neural Network Implementation
Step 1: Single Linear Perceptron

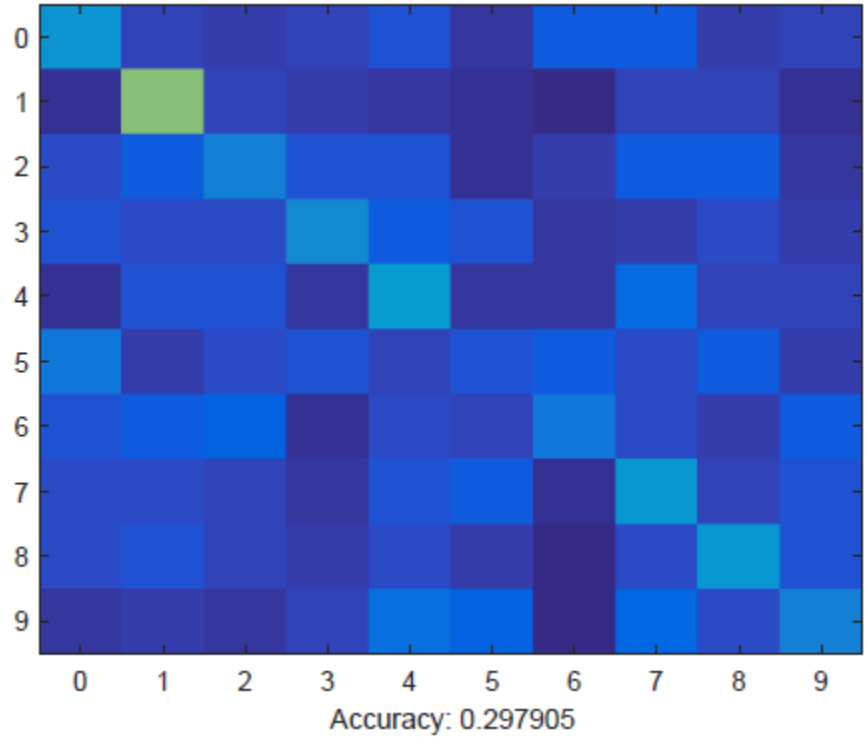
`function [w confusion_matrix] = SLP_linear`
Output: $w \in \mathbb{R}^{10\times 197}$ is the trained weights, and `confusion_matrix` $\in \mathbb{R}^{10\times 10}$ is confusion matrix (see HW#3 for detail).
Description: Using above functions, you will write a code to train the neural network by leveraging the gradient descent method:

- Label prediction: `Multiply`
- Loss computation: `Loss`
- Gradient back-propagation: `Multiply_backward_w`, `Loss_backward`

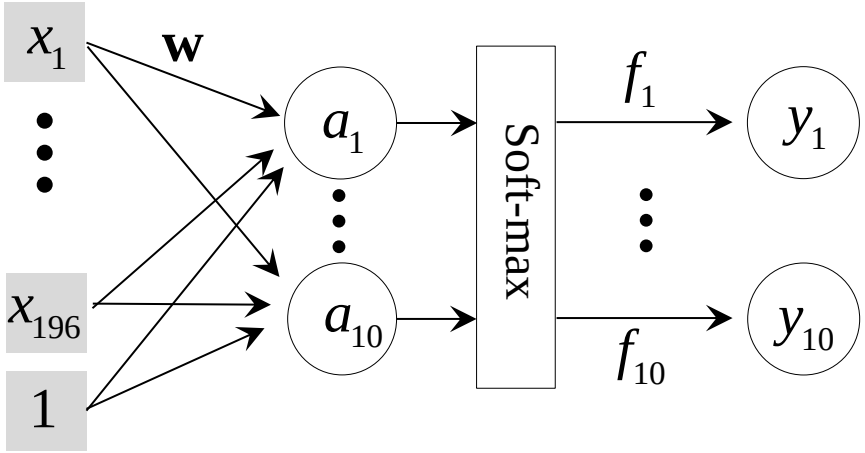
HW #4: Neural Network Implementation

Step 1: Single Linear Perceptron



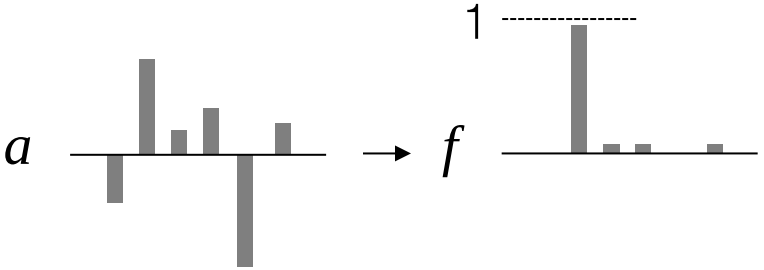


HW #4: Neural Network Implementation
Step 2: Single Perceptron

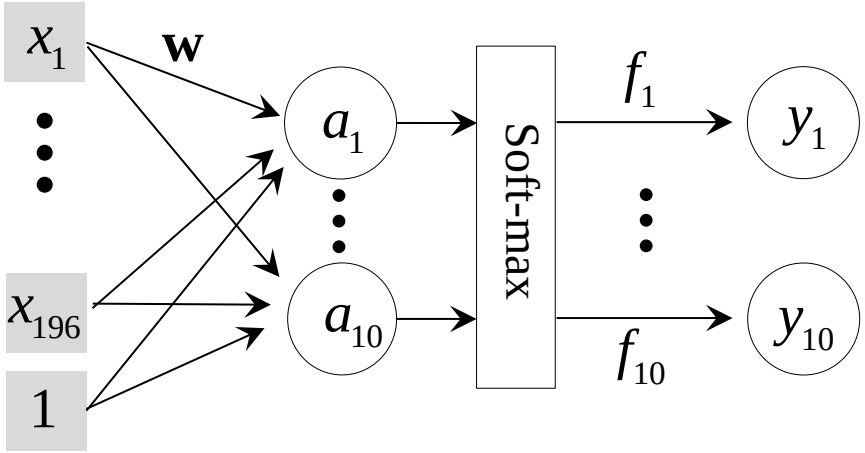


Soft-max (winners-take-all):

$$f_i = \frac{e^{a_i}}{\sum_i e^{a_i}}$$

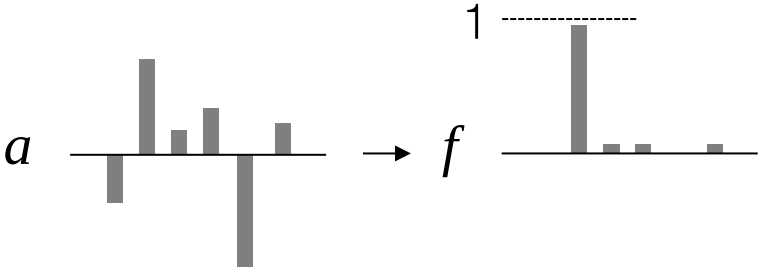


HW #4: Neural Network Implementation
Step 2: Single Perceptron



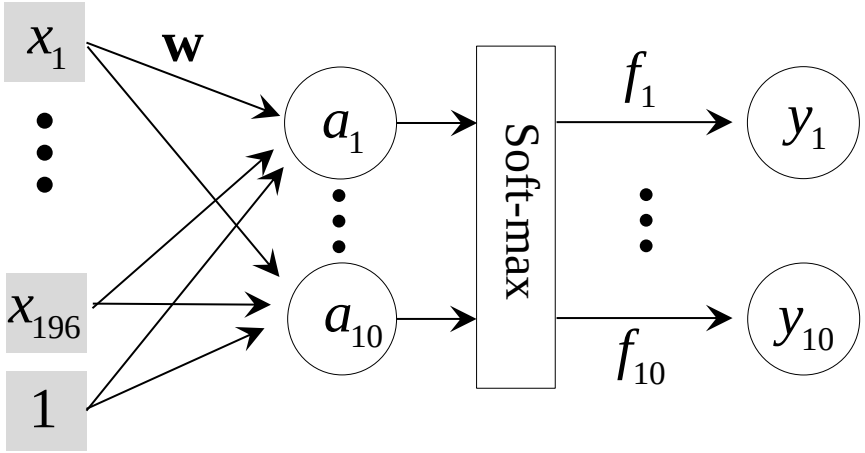
Soft-max (winners-take-all):

$$f_i = \frac{e^{a_i}}{\sum_i e^{a_i}}$$



$$\frac{\partial f_i}{\partial a_j} = \begin{cases} f_i(1-f_i) & i = j \\ -f_i f_j & i \neq j \end{cases}$$

HW #4: Neural Network Implementation
Step 2: Single Perceptron

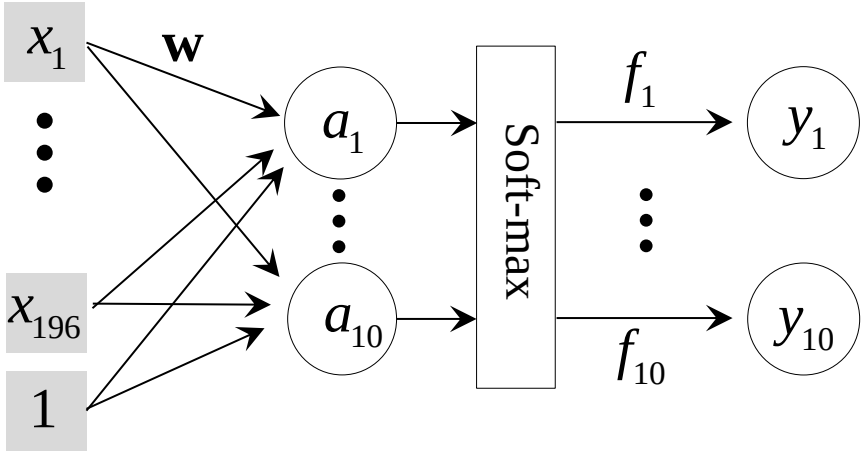


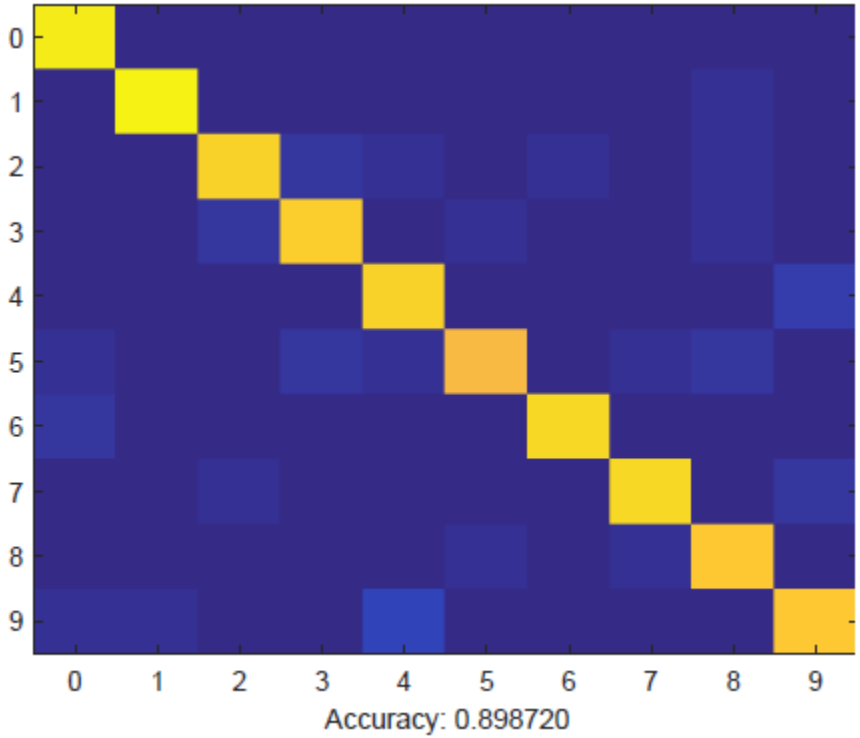

```
function f = Softmax(x)

function dfdx = Softmax_backward(f)
```

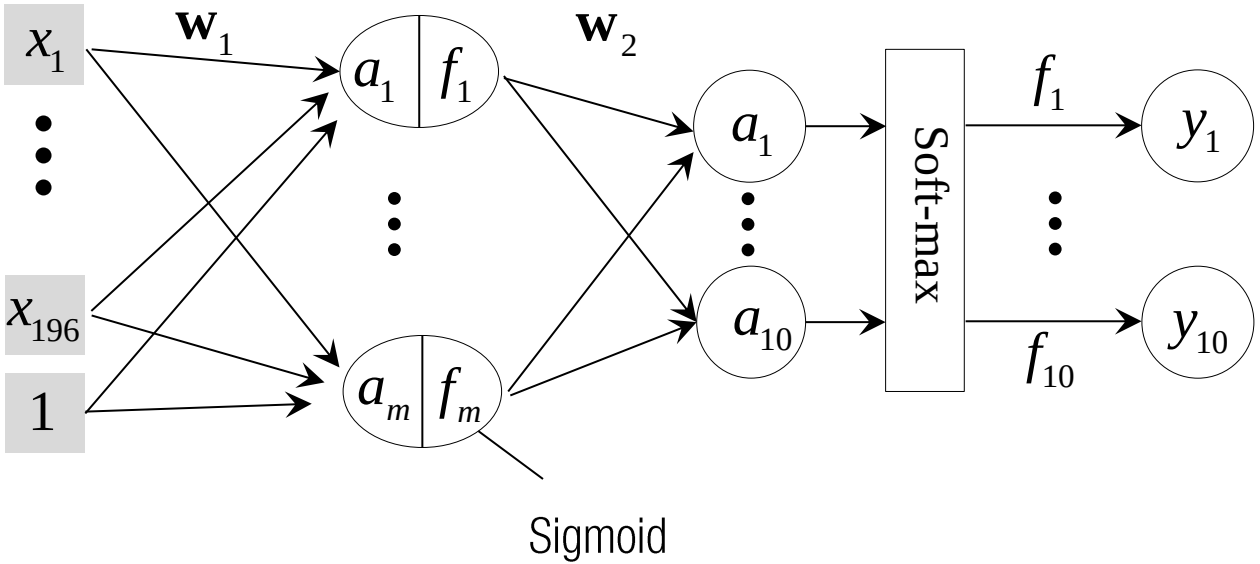
HW #4: Neural Network Implementation

Step 2: Single Perceptron





HW #4: Neural Network Implementation
Step 3: Multi-layer Perceptron



Dimensions?

HW #4: Neural Network Implementation

Step 3: Multi-layer Perceptron