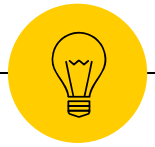
Chapter 4



김선
서울대학교 컴퓨터 공학부
생물정보 연구소

4

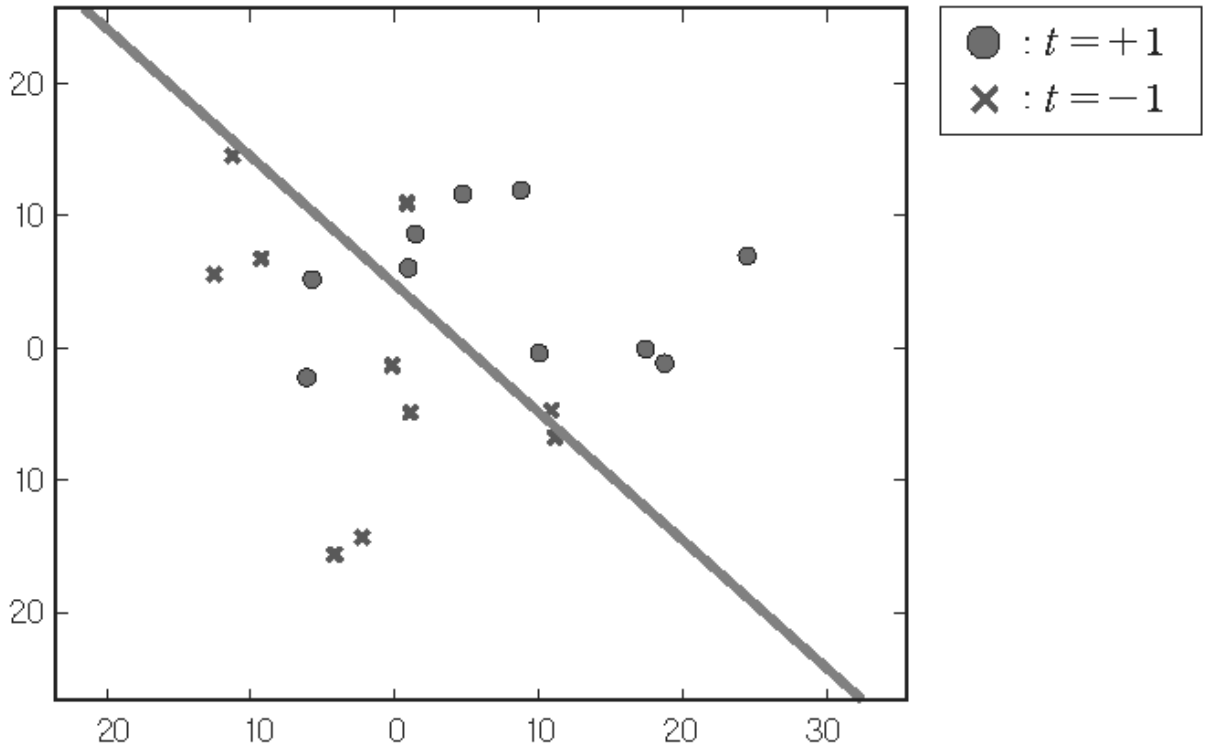
퍼셉트론 & 인공신경망
(Perceptron & Artificial Neural Network)

김선
서울대학교 컴퓨터 공학부
생물정보 연구소

Classification(분류)

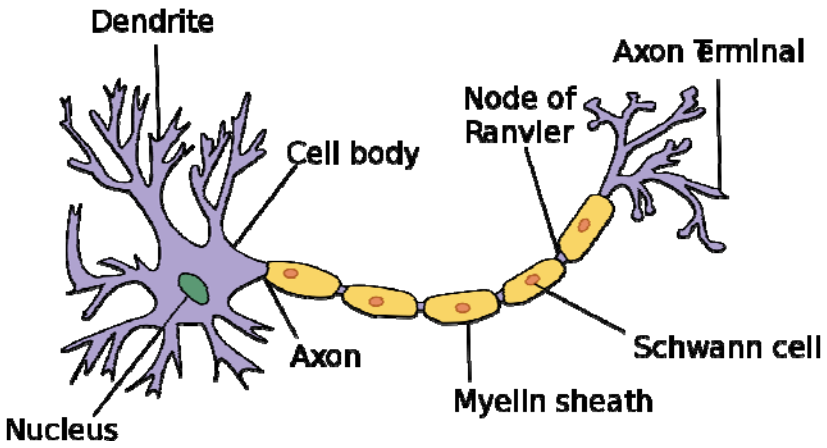
- When data has discrete target classes.
- # of classes (n)?
  - (n=2) Binary classification.
  - (n>2, allow single class) Multiclass classification.
  - (n>2, allow multiple classes) Multi-label classification.
- Example)
  - Given personal information, determine one is man or woman.
  - Given midterm score, predict the final grade.
  - Which objects can be found from given picture.

Binary Classification(분류)



Neuron(뉴런)

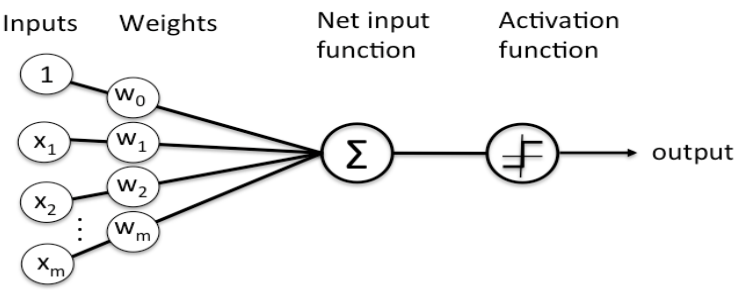
- A neuron receives several inputs and combines these in the cell body.
- If the input reaches a threshold, then the neuron may fire (produce an output).
- Some inputs are excitatory, while others are inhibitory.



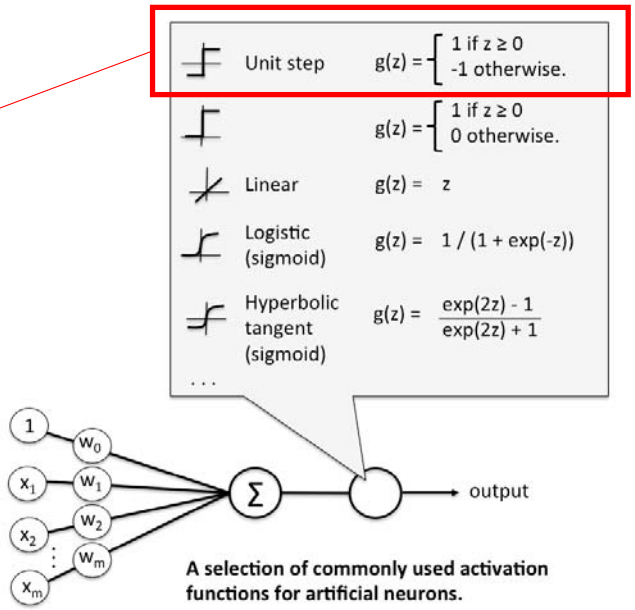
Perceptron(퍼셉트론)

- Simplest form of artificial neuron.
- Linear classification algorithm.
- Formulation.
  - Data: $D = \{(\mathrm{X}, t)^n\}_{n=1}^{N}$
  - Input features: $\mathrm{X} = (x_1, .., x_k)$
  - Output: $t \in \{-1, 1\}$
  - Model: $\hat{t} = g\big(f(\mathrm{X})\big)$

$$f(X) = \sum_{i=0}^{k} w_i x_i$$



Schematic of Rosenblatt's perceptron.



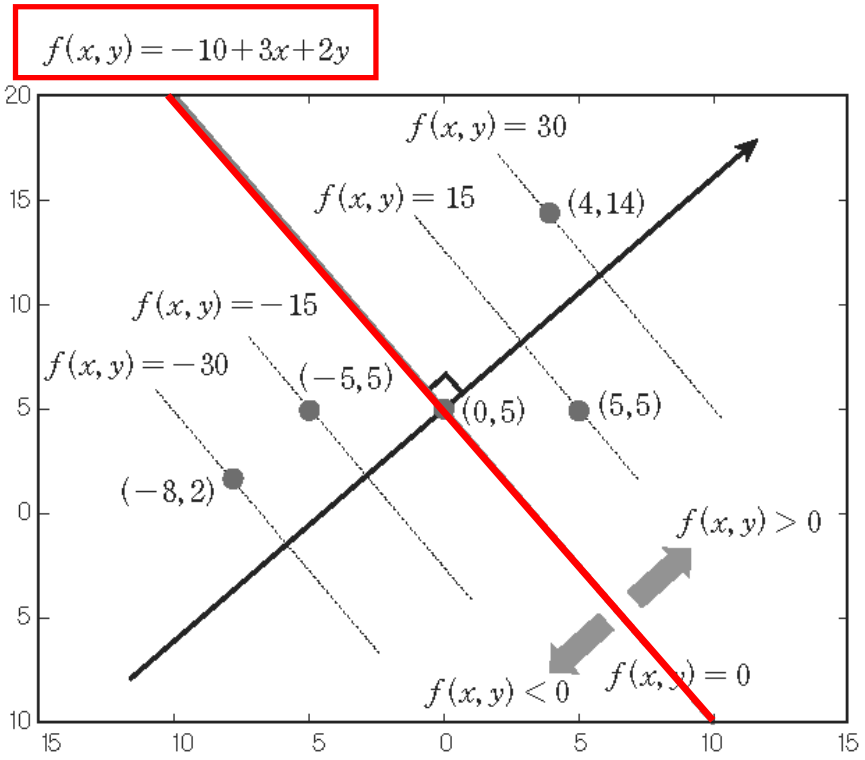
A selection of commonly used activation functions for artificial neurons.

Perceptron(퍼셉트론)

- Example)
  - Data: $D = \{(\mathrm{X}, t)^n\}_{n=1}^{N}$
  - Input features: $\mathrm{X} = (x_1, x_2) = (x, y)$
  - Output: $t \in \{-1, 1\}$
  - Model: $\hat{t} = g(f(\mathrm{X}))$

$$f(X) = \sum_{i=0}^{k} w_i x_i = w_0 + w_1 x + w_2 y$$

$$where, \quad w_0 = -10, w_1 = 3, w_2 = 2$$



Perceptron(퍼셉트론)

- Example) Make 'AND' function with perceptron.
  - Data: $D = \{(\mathrm{X}, t)^n\}_{n=1}^{N}$
  - Input features: $\mathrm{X} = (x_1, x_2)$
  - Output: $t \in \{0, 1\}$
  - Model: $\hat{t} = g(f(\mathrm{X}))$

$$f(X) = \sum_{i=0}^{k} w_i x_i = w_0 + w_1 x_1 + w_2 x_2$$

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

  - Find (w_0, w_1, w_2) that makes model as logical 'AND' function.
- There are multiple answers!
  - One possible answer is $(w_0, w_1, w_2) = (-0.8, 0.5, 0.5)$.

x_1	x_2	x_2 ***AND*** x_2
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

Perceptron(퍼셉트론)

- TODO) Make 'OR' function with perceptron.
  - Data: $D = \{(\mathrm{X}, t)^n\}_{n=1}^{N}$
  - Input features: $\mathrm{X} = (x_1, x_2)$
  - Output: $t \in \{0, 1\}$
  - Model: $\hat{t} = g(f(\mathrm{X}))$

$$f(X) = \sum_{i=0}^{k} w_i x_i = w_0 + w_1 x_1 + w_2 x_2$$

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

  - Find (w_0, w_1, w_2) that makes model as logical 'OR' function.
- There are also multiple answers!
  - One possible answer is $(w_0, w_1, w_2) = ???$.

x_1	x_2	x_2 **OR** x_2
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

Perceptron(퍼셉트론)

- TODO) Make 'OR' function with perceptron.
  - Data: $D = \{(\mathrm{X}, t)^n\}_{n=1}^{N}$
  - Input features: $\mathrm{X} = (x_1, x_2)$
  - Output: $t \in \{0, 1\}$
  - Model: $\hat{t} = g(f(\mathrm{X}))$

$$f(X) = \sum_{i=0}^{k} w_i x_i = w_0 + w_1 x_1 + w_2 x_2$$

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

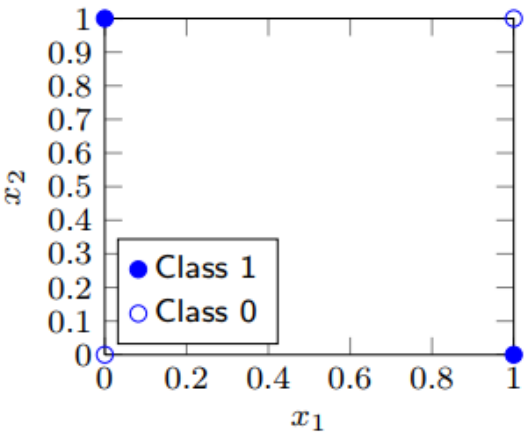
  - Find (w_0, w_1, w_2) that makes model as logical 'OR' function.
- There are also multiple answers!
  - One possible answer is $(w_0, w_1, w_2) = (-0.3, 0.5, 0.5)$.

x_1	x_2	x_2 **OR** x_2
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**1**

Perceptron(퍼셉트론)

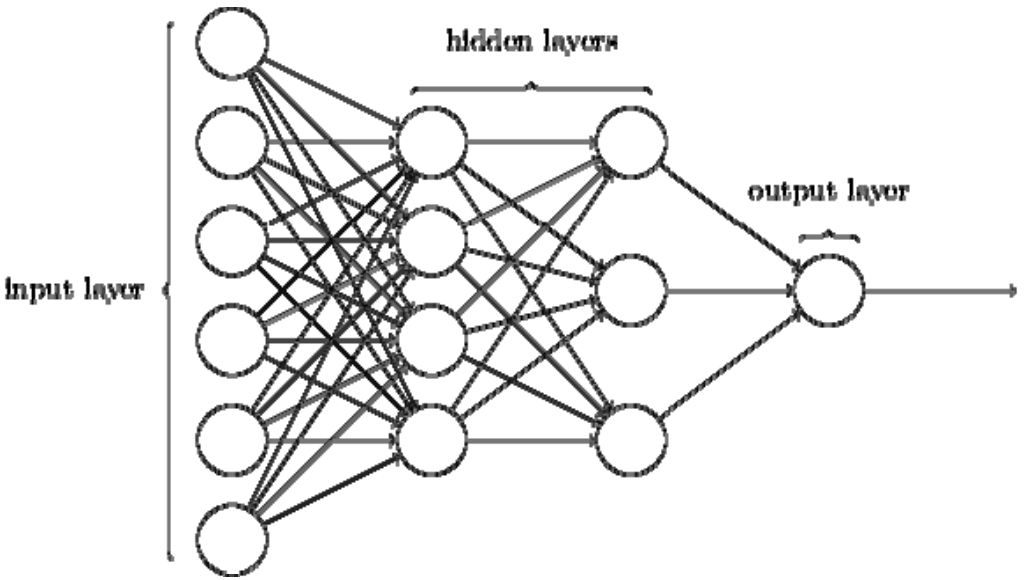
- Then, can perceptron model 'XOR' function?
  - No, it cannot.
  - To do so, we can separate two classes by drawing a single line on the plot. However, it is not possible.
- Perceptron cannot model non linearly separabl data!
- Multilayer perceptron can represent more complex models.

x_1	x_2	x_2 **XOR** x_2
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

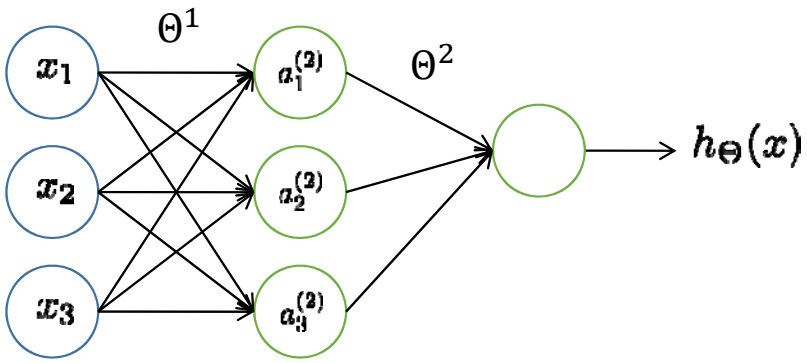


Multilayer perceptron(MLP)

- MLP consists of one input and output layer with multiple hidden layers.
  - Each layer is a perceptron.
  - Most widely used architecture of artificial neural network (feedforward neural network).



Multilayer perceptron(MLP)



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

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

x_0, a_0 = bias

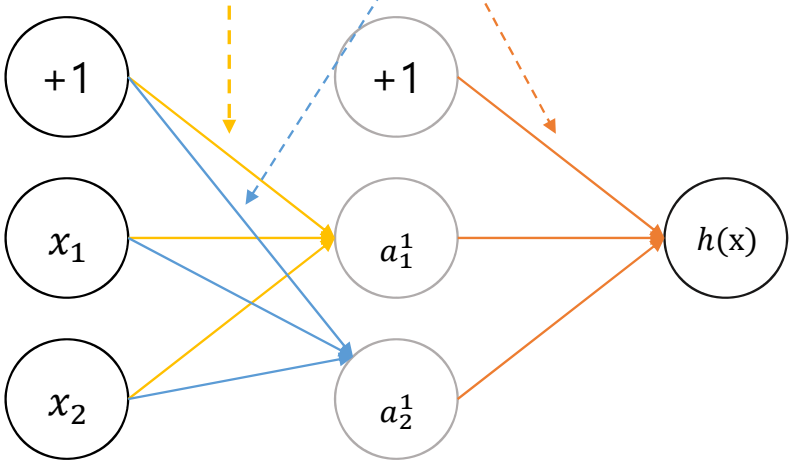
$g(x)$ = activation function

$$a_1^{(2)} = g(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$
$$a_2^{(2)} = g(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$
$$a_3^{(2)} = g(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\Theta(x) = g(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

Multilayer perceptron(MLP)

- Modeling ‘XOR’ function with MLP?
  - $XOR = (x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2)$
    - ∨: ‘AND’, ∧: ‘OR’, ¬: ‘NOT’
  - With two layers

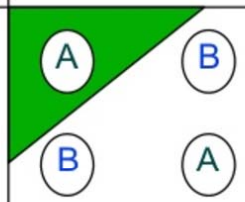
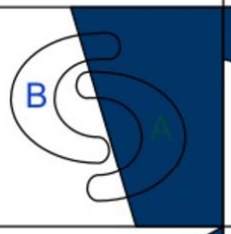
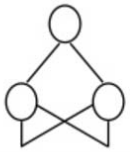
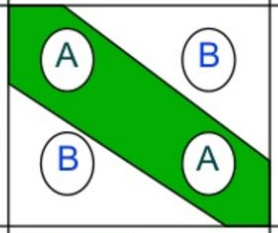
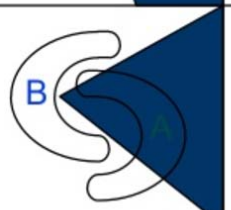
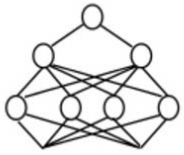
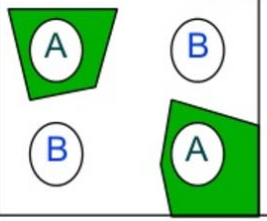
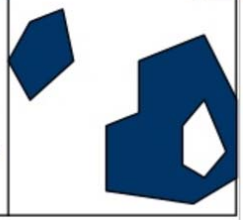


x_1	x_2	x_2 **XOR** x_2
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**

Multilayer perceptron(MLP)

Different Non-Linearly Separable Problems

http://www.zsolutions.com/light.htm

Structure	*Types of Decision Regions*	*Exclusive-OR Problem*	*Classes with Meshed regions*	*Most General Region Shapes*
Single-Layer 	*Half Plane Bounded By Hyperplane*			
Two-Layer 	*Convex Open Or Closed Regions*			
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)			

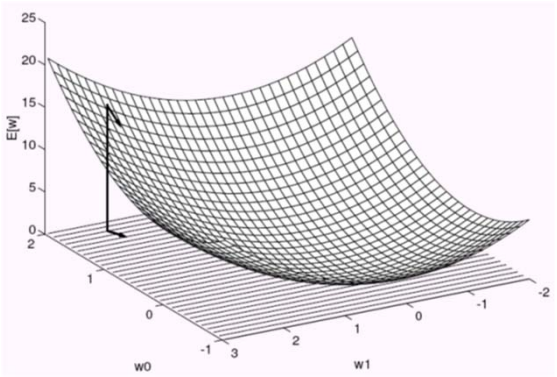
Gradient Descent(기울기 하강법)

- Find parameters of perceptron via 3 steps of parametric model.

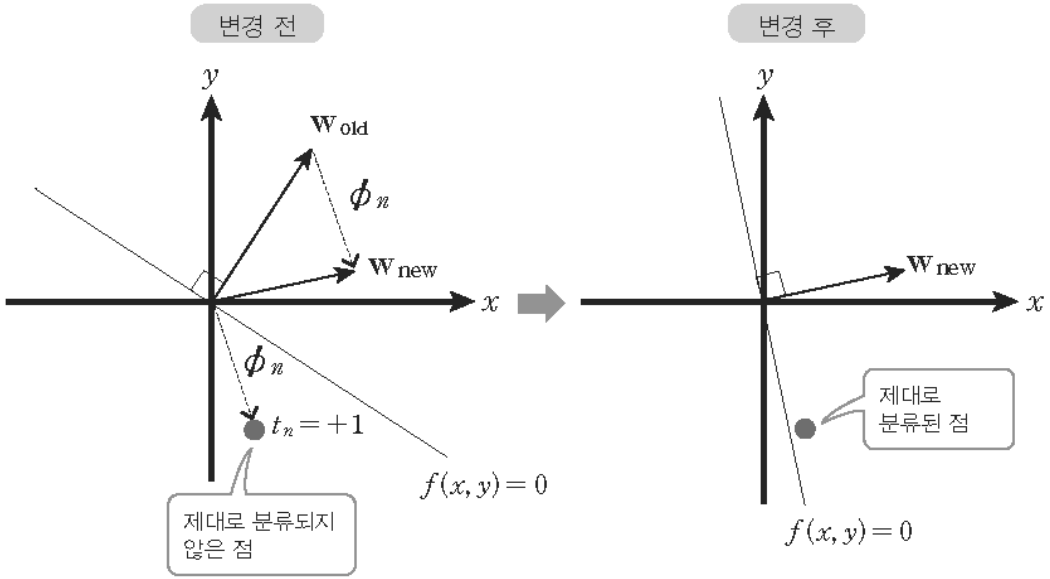
1. Define model.
   - Data: $\mathrm{D} = \{(X^n, t_n)\}_{n=1}^N, X^n = (x_1^n, \dots, x_k^n), t_n = \{-1, 1\}$
   - Parameters: $\theta = \{w_i\}_{i=0}^k$
   - Model: $f(X^n; \theta) = \sum_{i=0}^k w_i x_k^n, \widehat{t_n} = \begin{cases} -1, & f(X^n) < 0 \\ 1, & f(X^n) \geq 0 \end{cases}$
2. Define evaluation criterion.
   - $\begin{cases} f(X^n)t_n < 0 : wrong\ prediction \\ f(X^n)t_n > 0 : correct\ prediction \end{cases}$
   - $E_n = \max(0, -f(X^n)t_n)$*: if prediction is correct, E=0. if not, E signify how much far from the correct prediction.*
   - $E_D = \sum_{n=1}^N E_n$
   - Criterion: $\underset{\theta}{argmin}\, E_D$
3. Find parameters.

Gradient Descent(기울기 하강법)

- However, we cannot find the answer directly by differentiation as we did in least squares and maximum likelihood estimation.
- Gradient Descent
  - Given the way of defining E_D, find local minimum point of E_D (can also find global minimum if there exists only one minimum).
  - Starting with an arbitrary initial weights, repeatedly modifying weights in small steps.
    - How small = learning rate (η)
  - Algorithm
    - Repeat until convergence
      - $\nabla E_D = \left[\frac{\partial E_D}{\partial w_0}, \ldots, \frac{\partial E_D}{\partial w_k}\right]$: gradient
      - $[w_0, \ldots, w_k] = [w_0, \ldots, w_k] - \eta \nabla E_D$: update parameters



Gradient Descent(기울기 하강법)



Stochastic Gradient Descent (확률적 기울기 하강법)

- For gradient descent (batch gradient descent), we use all observed data to calculate E_D and ∇E_D.
- If data size is too big, it takes too much time to calculate gradient.
- Let's sampling train data -> stochastic!
  - Rather than calculate $E_D = \sum_{n=1}^{N} E_n$, use E_n while randomly sampling n from $\{1, .., \mathrm{N}\}$
- Stochastic Gradient Descent (SGD) algorithm.
  - Repeat until convergence
    - For n in $\mathbf{random\_sort}(\{1, .., \mathrm{N}\})$
      - $\nabla E_n = \left[\frac{\partial E_n}{\partial w_0}, \ldots, \frac{\partial E_n}{\partial w_k}\right]$: gradient
      - $[w_0, \ldots, w_k] = [w_0, \ldots, w_k] - \eta \nabla E_n$: update parameters

Stochastic Gradient Descent
(확률적 기울기 하강법)