Machine Learning ~ CNN

조상연

순서

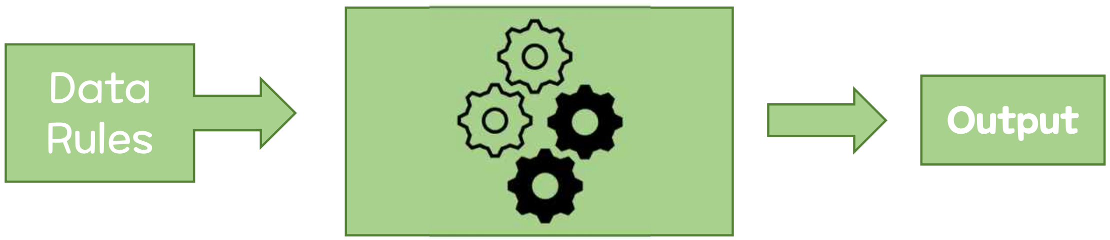
- Machine Learning
  - Linear Regression
  - Multi-variable Linear Regression
  - Logistic Regression/Classification
  - Softmax Regression/Classifier
- Deep Learning
  - XOR
- Convolution Neural Network

Machine Learning(기계학습)

- 어떠한 자료(데이터)에서 스스로 “학습”해서 작동하는 것
  1. Supervised Learning
     - Label이 정해져 있는 데이터
     - Label을 보고 학습 -> Training Data sets
     - 종류
       - Predicting(예측)
       - Classification(분류)
  2. Unsupervised Learning
     - Label이 정해지지 않은 데이터
     - 데이터를 보고 학습을 함
     - 종류
       - 군집화
       - 차원축소

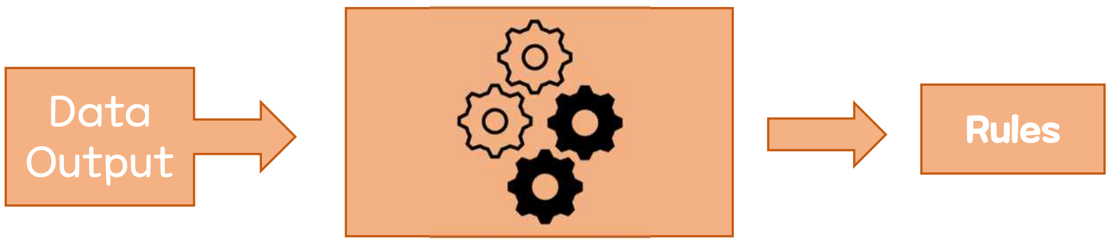
Machine Learning(기계학습)

기존



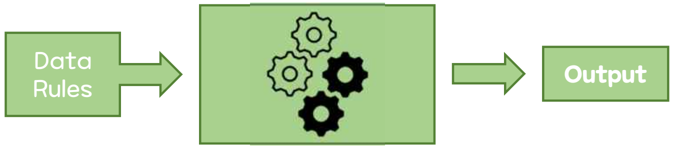
Rule을 정의하지 않으면, 문제를 풀지 못함

기계학습



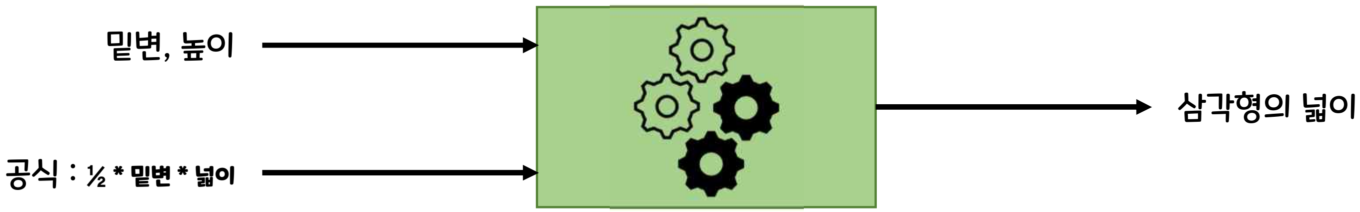
학습을 할수록, 응용되어도 스스로 문제를 해결

기존

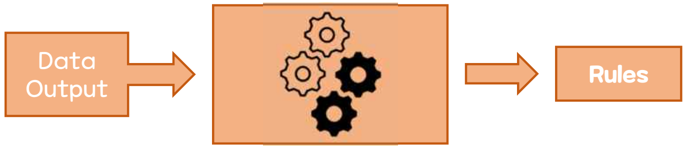


삼각형의 넓이 구하기

Data로 밑변과 높이
Rulu로는 삼각형 넓이를 구하는 공식이 들어가면,
삼각형의 넓이(Output)을 구할 수 있다.

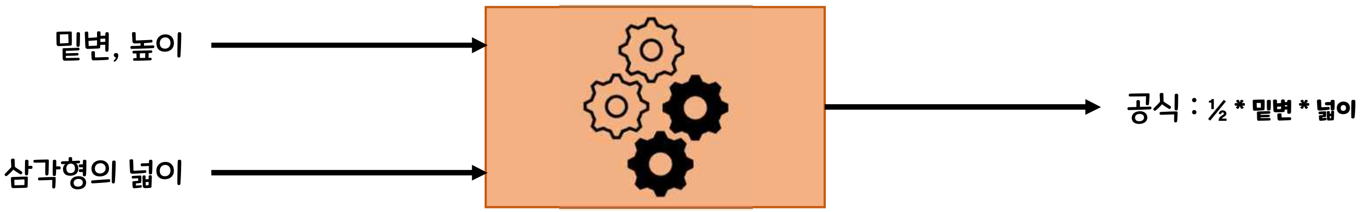


기존

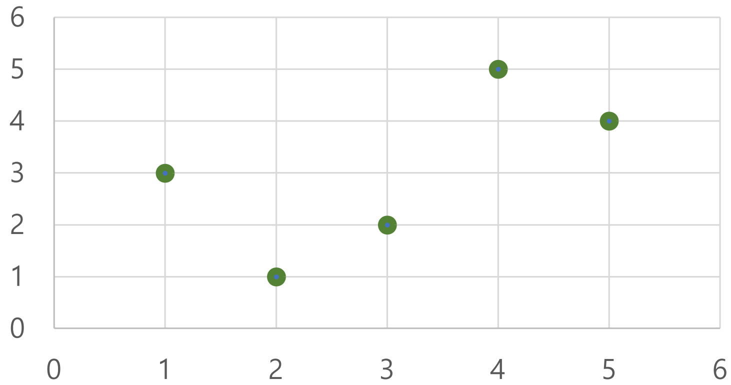


삼각형의 넓이 구하기

Data로 밑변과 높이
결과값인 삼각형의 넓이를 입력으로 넣어주면
모델은 Data와 Output사이에서 삼각형의 넓이를
구하는 공식을 스스로 학습하여 알아낸다.



Simple Linear Regression(선형회귀)

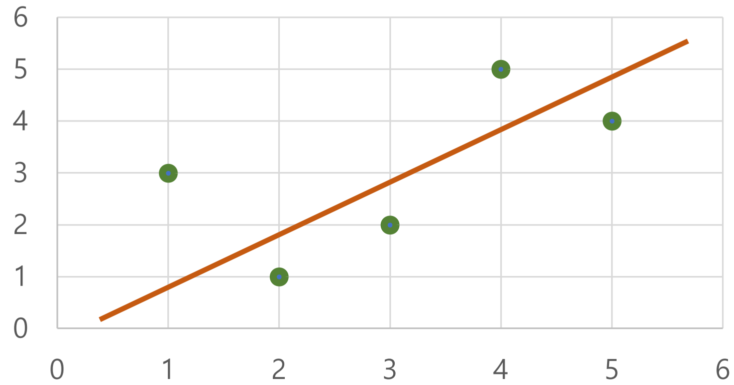


데이터

X	Y
1	3
2	1
3	2
4	5
5	4

가설
Hypothesis

데이터들을 가장 잘 대변하는
직선 방정식



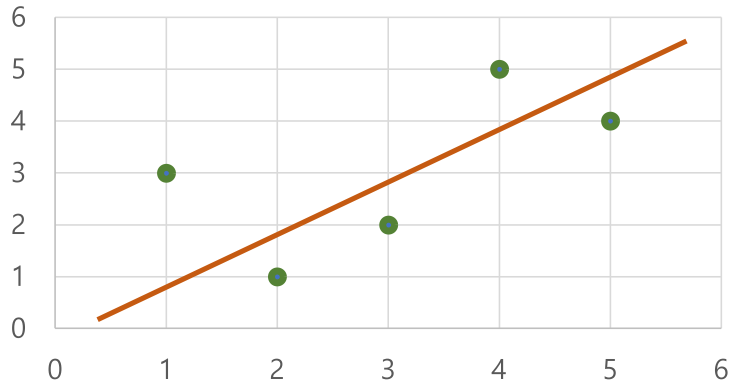
직선모델

Y = ax + b

H(x) = W*x + b

기울기(W)와 y절편(b)에 따라 선의 모양이 결정된다.

Simple Linear Regression(선형회귀)



직선모델

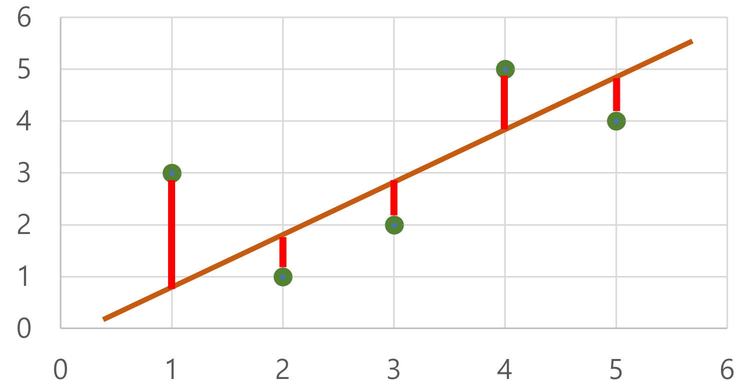
Y = ax + b

H(x) = W*x + b

기울기(W)와 y절편(b)에 따라 선의 모양이 결정된다.

비용
Cost

실제 데이터(y)와 가설의 값(H(x))의 차이



비용

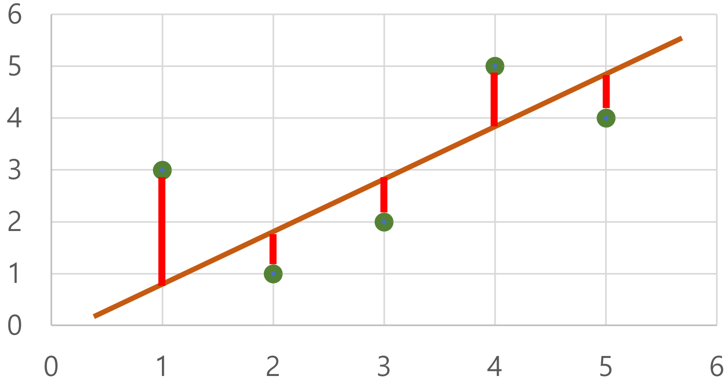
H(x) = W*x + b

Cost = H(x) – Y

Cost의 값이 음수가 될 수도 있기 때문에
제곱을 취한다. (=오차 제곱)

Cost = $(H(x) - Y)^2$

Simple Linear Regression(선형회귀)



비용

H(x) = W*x + b

Cost = H(x) – Y

Cost의 값이 음수가 될 수도 있기 때문에
제곱을 취한다. (=오차 제곱)

Cost = $(H(x) - Y)^2$

비용 최소화
Minimize Cost

Cost가 최소가 되는 W와 b값을 구한다.
평균 제곱 오차 (mean square error)

H(x) = W*x + b

Cost = $(H(x) - Y)^2$

MSE(W,b) = $\frac{1}{m}\left(\sum_{i=0}^{m}(H(x_i) - y_i)^2\right)$

Simple Linear Regression(선형회귀)

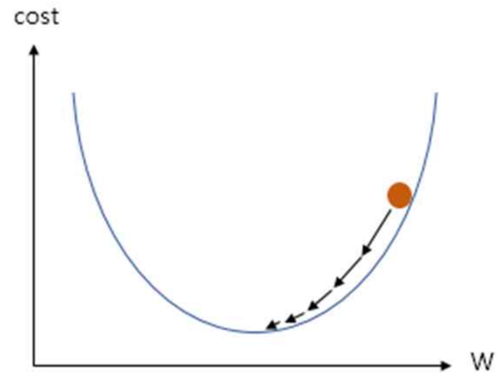
H(x) = W*x + b

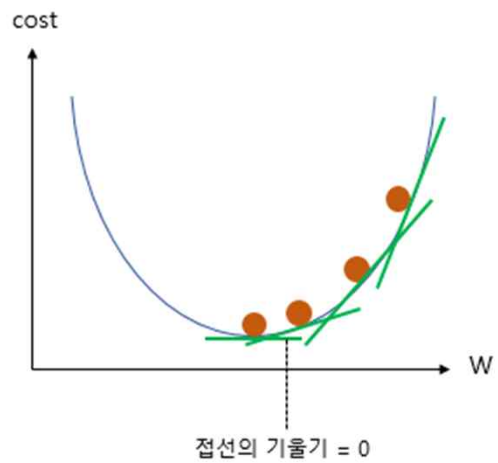
Cost = $(H(x) - Y)^2$

MSE(W,b) = $\frac{1}{m}\left(\sum_{i=0}^{m}(H(x_i) - y_i)^2\right)$

경사하강법
Gradient Descent

Cost가 최소화 되는 W,b값을 찾는 알고리즘
Mse(W,b)함수를 미분하여 나타냈을때(=기울기),
그 값이 0되는 W와 b값을 찾는다.





https://wikidocs.net/21670

Simple Linear Regression(선형회귀)

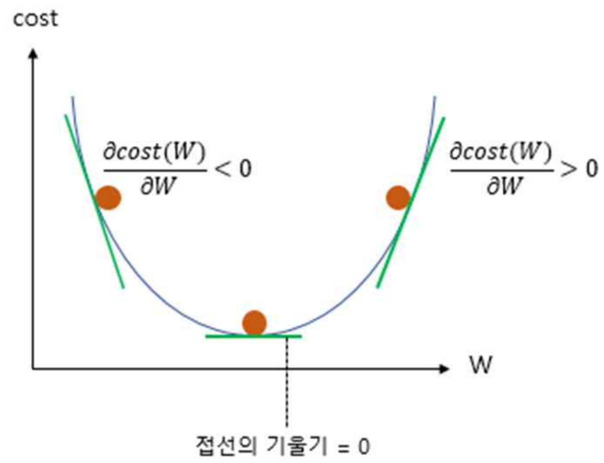
$H(x) = W*x + b$

$\text{Cost} = (H(x) - Y)^2$

$$\text{MSE(W,b)} = \frac{1}{m}\left(\sum_{i=0}^{m}(H(x_i) - Y_i)^2\right)$$

경사하강법
Gradient Descent

Cost가 최소화 되는 W,b값을 찾는 알고리즘
Mse(W,b)함수를 **미분**하여 나타냈을때,
기울기가 0되는 W와 b값을 찾는다.



Cost가 최소가 되는 W와 b를 구하기 위해 W와 b를 업데이트하는 방법을 취한다.
MSE함수를 미분하여 나타낸 접선의 기울기가 음수이거나 양수일때 0이 되게끔, α 값을 통해 적절하게 업데이트한다.

α = 학습률
Cost가 0이 되는 W와 b가 잘 찾아갈 수 있도록 결정하는 비율값
너무 높으면 0에 수렴하지 않고 발산하는 상황이 발생한다.

접선의 기울기 "$\frac{\Delta}{\Delta W}$MSE(W,b)"가 음수이면 기존의 W에 양수 값(-αx기존의 기울기 = 양수)을 더해 0으로 접근하게 하고
양수이면 기존의 W에 음수 값 (-αx기존의 기울기 = 음수) 을 더해 0으로 접근하게 한다.

$$W_{new} := W_{old} - \alpha \frac{\Delta}{\Delta W}\text{MSE(W,b)}$$

https://wikidocs.net/21670

Simple Linear Regression(선형회귀)

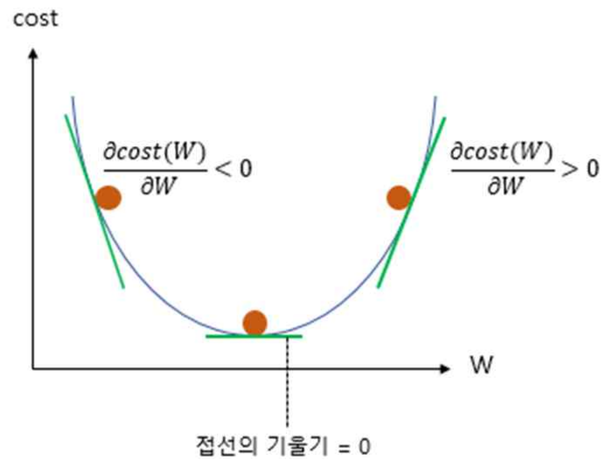
H(x) = W*x + b

Cost = $(H(x) - Y)^2$

MSE(W,b) = $\frac{1}{m}\left(\sum_{i=0}^{m}(H(x_i) - Y_i)^2\right)$

경사하강법
Gradient Descent

Cost가 최소화 되는 W,b값을 찾는 알고리즘
Mse(W,b)함수를 **미분**하여 나타냈을때,
기울기가 0되는 W와 b값을 찾는다.



편미분

$W_{new} := W_{old} - \alpha \frac{\Delta}{\Delta W} MSE(W,b)$

$b_{new} := b_{old} - \alpha \frac{\Delta}{\Delta b} MSE(W,b)$

$\frac{\Delta}{\Delta W} MSE(W,b) = \frac{2}{m}\sum_{i=0}^{m}(H(x_i) - Y_i)x_i$

$\frac{\Delta}{\Delta b} MSE(W,b) = \frac{2}{m}\sum_{i=0}^{m}(H(x_i) - Y_i)$

https://wikidocs.net/21670

Simple Linear Regression(선형회귀)

- 경사하강법
  - Cost함수를 최소화하는 알고리즘
  - 추정을 통해 랜덤하게 W_i 와 b를 결정
  - W_i 와 b를 지속적으로 갱신(Cost가 최소가 될 때 까지)
  - W_i 와 b를 아무 지점으로 결정하여도 최소점에 도달한다.
  - 최소점으로 도달하게하는 논리는 미분을 통해 기울기를 구하여 0이되는 지점을 구한다.
    - 하지만 사실 최소가 되는 W_i 와 b를 구하지 못할 수도 있다. = Local Minimum
    - 즉, Local Minimun이 Global Minimum인 Convex한 함수에서만 사용 가능하다.

Simple Linear Regression(선형회귀)

- 가설 설정 H(x)
  - 데이터들을 가장 잘 표현하는 직선
  - H(x) = Wx + b
- 비용 정의 : cost
  - 실제 데이터값과 가설의 값의 차이
  - Cost = H(x) - Y 에서
  - Cost = $(H(x) - Y)^2$
- 비용 최소화 : Minimize Cost
  - 평균제곱오차(MSE) : **MSE(W,b) =** $\frac{1}{m}\left(\sum_{i=0}^{m}(H(x_i) - y_i)^2\right)$
  - 이후 경사하강법을 이해 MSE(W,b)함수가 최소가 되는 W와 b 값을 결정

Multi-Variable Regression(다변수 회귀)

- X = Feature(input)
- Y = Label(output)
- Feature가 여러 개이면, Multi-variable(feature)

X_1	X_2	X_3	Y
73	80	75	152
93	88	93	185
89	91	90	180
96	98	100	196
73	66	70	142

X : Features

Y : Label

Multi-Variable Regression(다변수 회귀)

- X = Feature(input)
- Y = Label(output)
- Feature가 여러 개이면, Multi-variable(feature)

- Hypothesis = H(x)
  - H(x) = Wx + b
- Cost(W,b) = $\frac{1}{m}(\sum_{i=0}^{m}(H(x_i) - Y_i)^2)$

- Hypothesis = H(x_1, x_2, x_3⋯)
  - H(x) = W_1x_1 + W_2x_2 + W_3x_3 + b
- Cost(W,b) = $\frac{1}{m}(\sum_{i=0}^{m}(H(x_{1i}, x_{2i}, x_{3i} \dots) - y_i)^2)$

Multi-Variable Regression(다변수 회귀)

- Matrix (행렬)
  - 행렬의 곱셈 공식을 이용 (내적 = “dot product”)
  - $W_1 x_1 + W_2 x_2 + W_3 x_3 + \cdots + W_n x_n$ =

$$\begin{bmatrix} x_1 & x_2 & x_3 & \ldots & x_m \end{bmatrix} \odot \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ \cdots \\ W_n \end{bmatrix}$$

  - 복잡한 모양의 데이터(데이터의 개수 모양에 상관 x)에도 간략하게 표현이 가능
  - 따라서 H($x_1, x_2, x_3, \cdots , x_m$) = $X_m \times W_n + \mathrm{b}$

Multi-Variable Regression(다변수 회귀)

- 복잡한 모양의 데이터(데이터의 개수 모양에 상관 x)에도 간략하게 표현이 가능
- 따라서 H($x_1, x_2, x_3, \cdots, x_m$) = $X_m \times W_n + \mathrm{b}$

- 다변수 회귀 또한 기존의 가설을 그대로 사용할 수 있기 때문에
- 큰 변경점 없이 회귀를 적용할 수 있다!

Multi-Variable Regression(다변수 회귀)

- W의 모양은 Feature와 Label의 모양에 따라 결정된다.

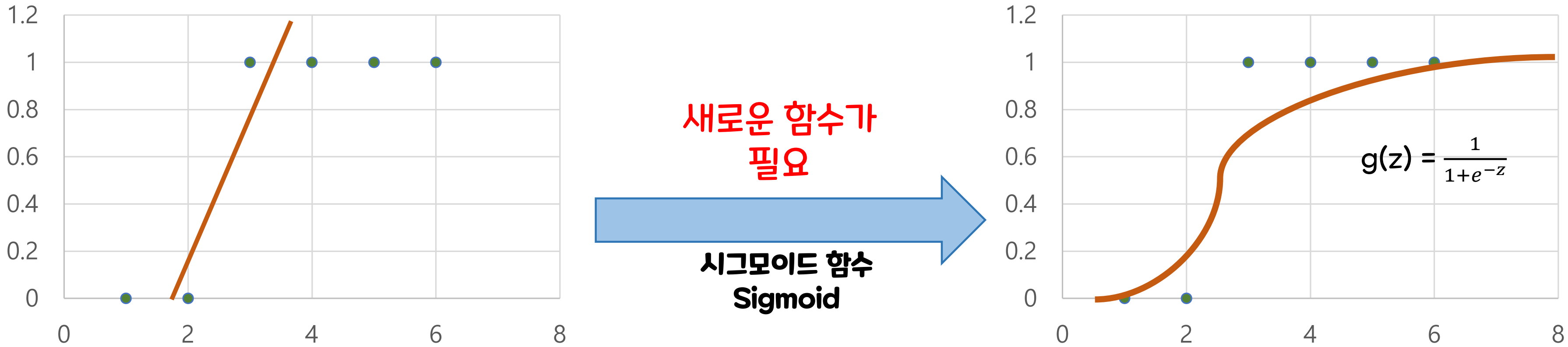
X_1	X_2	X_3	Y_1	Y_2
73	80	75	152	76
93	88	93	185	92
89	91	90	180	89
96	98	100	196	96
73	66	70	142	69

Features W Labels

- [n, 3] · [?, ?] = [n, 2]
- W = [3, 2]
  - Features의 열은 W의 행이되고
  - Labels의 열은 W의 열이된다.

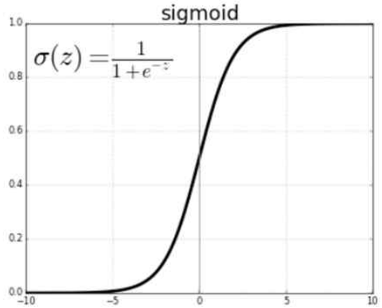
Logistic Regression/Classification

- 분류의 기초적인 알고리즘
- Logistic 회귀는 Y값(Label)이 “one hot” 데이터이다.
  - 연속된 데이터가 아니라 딱딱 끊어져서 표현된 데이터
  - Ex) [[1,0,0], [0,1,0], [0,0,1], [1,0,0], [0,1,0]]
- Label이 딱 정해지기 때문에 선형으로 표현하기에는 한계가 있다.
  - Y가 0과 1뿐인데 선형회귀의 경우 H(x)가 음과 양의 방향으로 무한하게 뻗어간다.

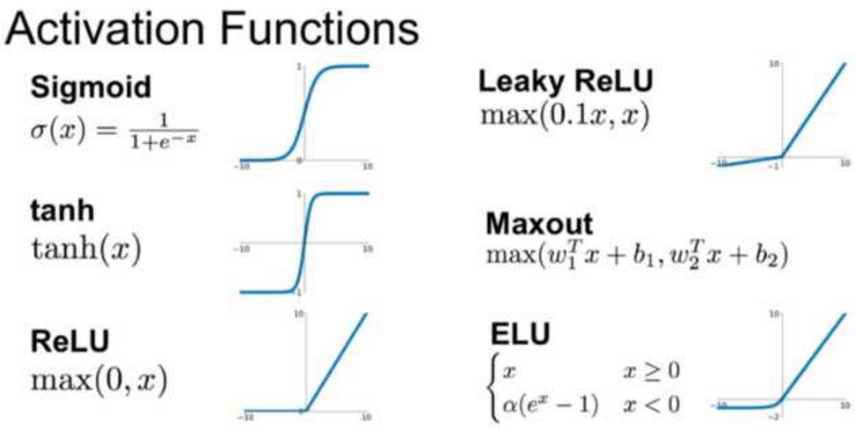


Logistic Regression/Classification

- **시그모이드 함수(Sigmoid)**
  - S자 곡선형 함수이다.
  - 0~1사이의 값을 가진다.



- 딥러닝에서는 **활성화 함수(Activation Function)**의 개념으로 사용됨
- 시그모이드 함수 외에도 사용되는 활성화 함수는 많음



Different Activation Functions and their Graphs

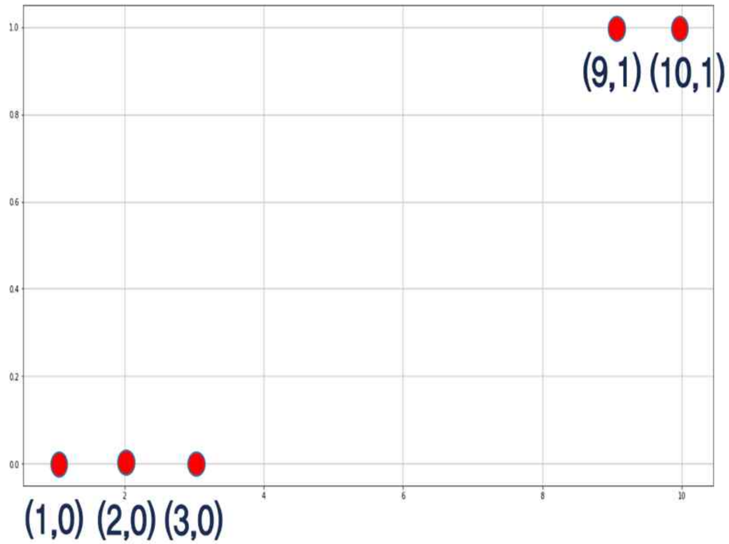
Logistic Regression/Classification

- 활성화 함수를 사용하는 이유
  - 선형적인 모델(가설)에 비선형(곡선)을 주기 위함이다.
  - 모델의 깊이가 깊어지는데, 단순한 선형회귀만 쌓는다면 모델은 단순한 구조가된다.
    - 즉 복잡한 구조의 모델을 설계하기 위해서 사용되는 기법이다.
- **머신러닝에서는 앞서 배운 선형회귀 가설에 활성화 함수를 씌워서 분류 문제를 해결한다.**

Logistic Regression/Classification

로지스틱회귀분석

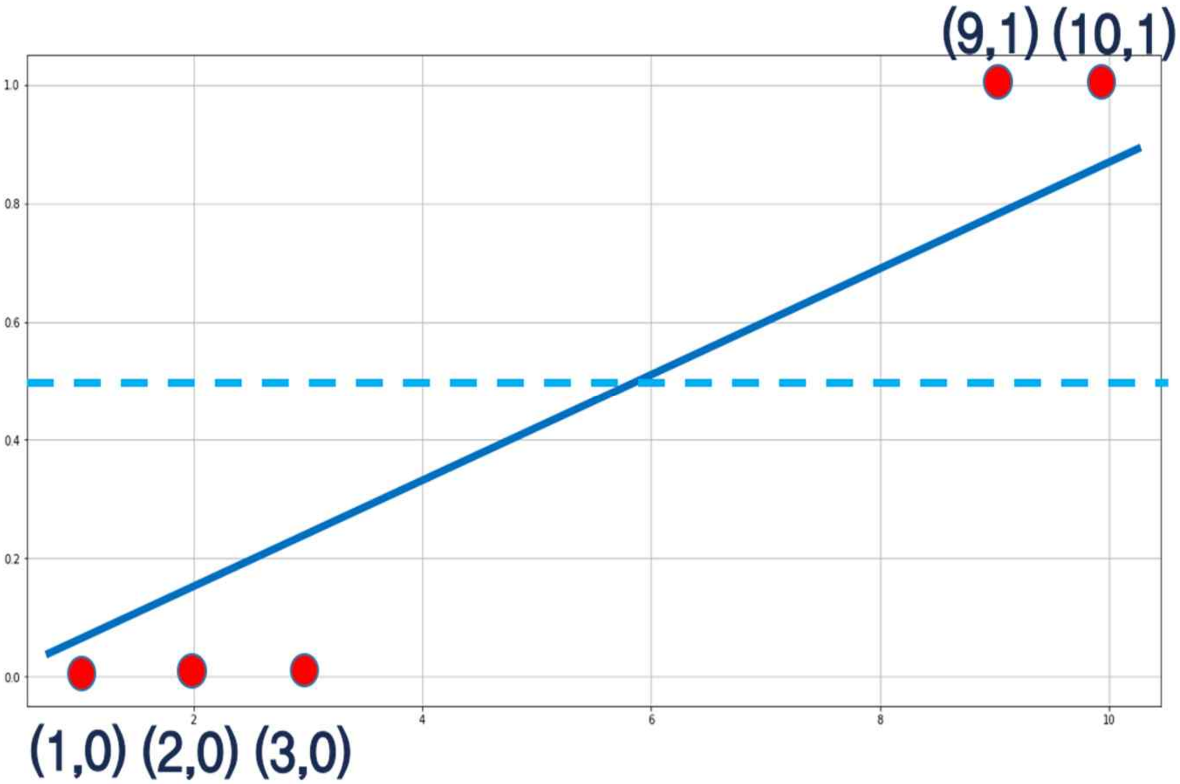
X 데이터	Y 데이터
10	Pass (1)
9	Pass (1)
3	Fail (0)
2	Fail (0)
1	Fail (0)
5	???



다음과 같은 데이터에 선형회귀를
적용한다고 가정한다면

https://icim.nims.re.kr/post/easyMath/64

Logistic Regression/Classification

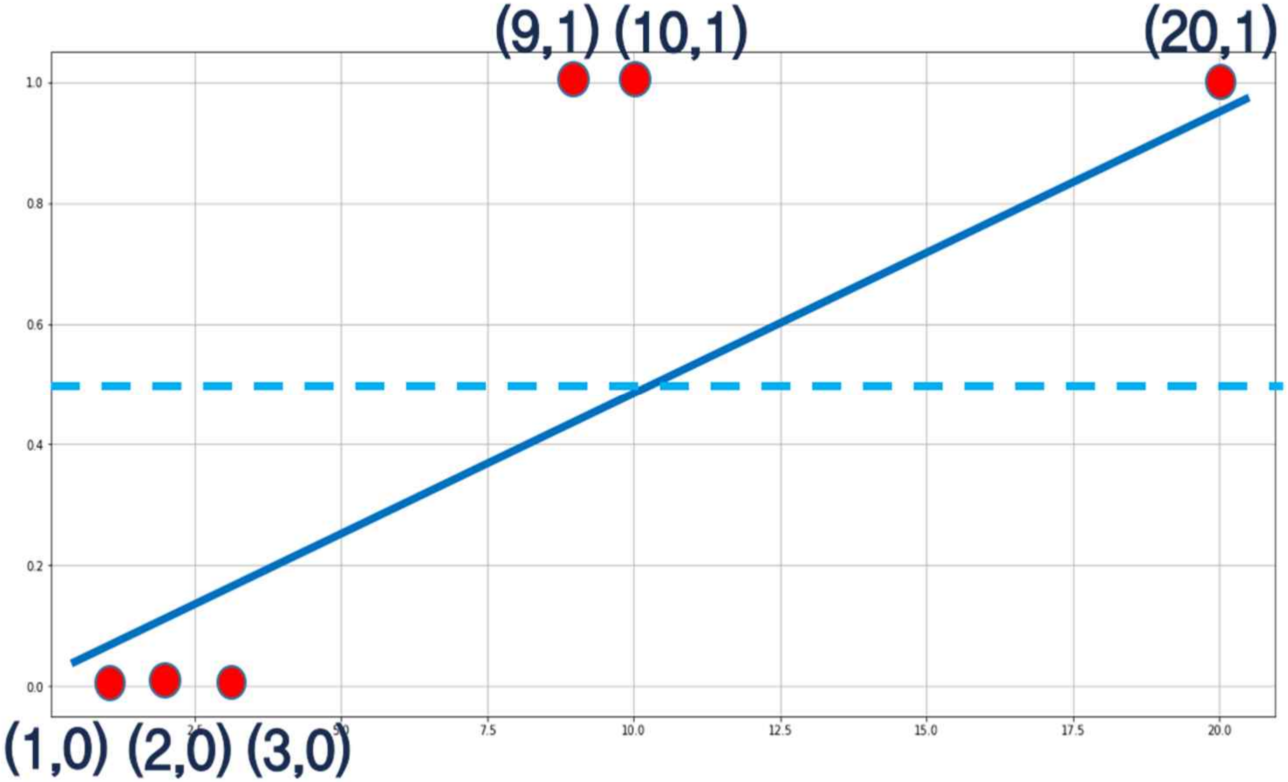


다음과 같은 데이터에 선형회귀를
적용한다고 가정한다면,
함수값이 ½가 되는 x=5를 기준으로
성공/실패를 구분할 것이다.

만약 여기에 새로운 데이터 (20,1)이 있다면,

https://icim.nims.re.kr/post/easyMath/64

Logistic Regression/Classification

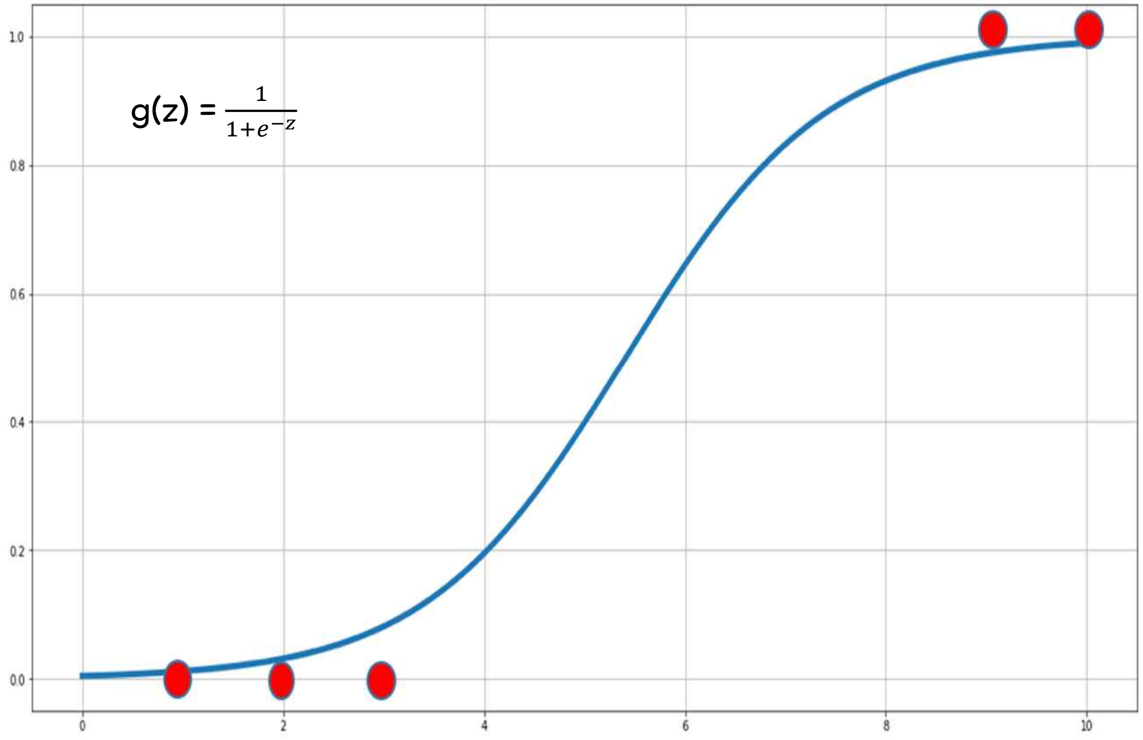


다음과 같은 데이터에 선형회귀를
적용한다고 가정한다면,
함수값이 ½가 되는 x=5를 기준으로
성공/실패를 구분할 것이다.

만약 여기에 새로운 데이터 (20,1)이 있다면,
함수값이 ½가 되는 x=10이므로 문제가 발생한다.
즉, (9,1), (10,1)에서 분류를 실패하게 된다.

https://icim.nims.re.kr/post/easyMath/64

Logistic Regression/Classification



시그모이드 함수는 y=1,0이 점근선이고
치역은 (0,1)이다.

시그모이드를 왜 사용하는가?
선형회귀의 목표는 실수값 예측이기 때문에
Y = Wx+b를 이용해 예측한다.
하지만 로지스틱 회귀에서는 실수값이 아닌 0 or 1의 값을
예측하기 때문에 y = Wx + b를 통한 예측은 의미가 없다.

따라서 Odds를 이용한다.

https://icim.nims.re.kr/post/easyMath/64

Logistic Regression/Classification

g(z) = $\frac{1}{1+e^{-z}}$

Odds(p) := $\frac{p}{1-p}$ 으로,
확률 p의 범위가 0~1이라면
Odds(p)함수의 범위는 0~∞ 이다.
이 Odds(p) 함수에 log를 취하면 범위는
− ∞ ~ + ∞이므로 실수전체이다.

따라서 log(Odds(p))함수를 선형회귀하는 것이 의미가 있는데,
Log(Odds(p)) = Wx + b

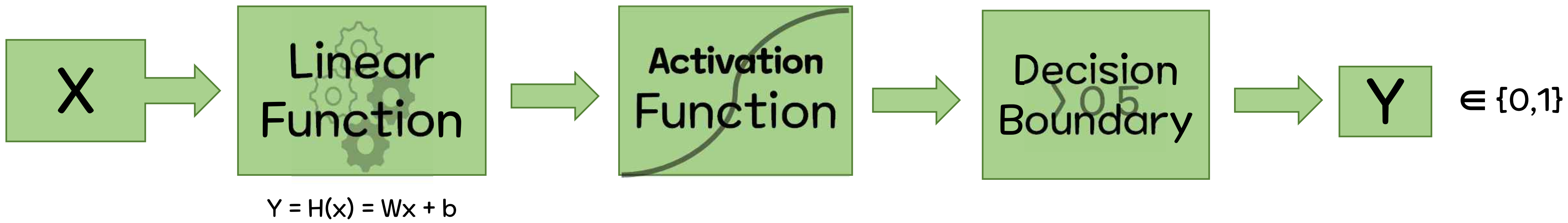
위 식을 p로 정리하면
P(x) = $\frac{1}{1+e^{-(Wx+b)}}$ 이고
이는 시그모이드 함수이다.

즉, 데이터 x가 주어졌을때, 0/1, 성공/실패를 예측하는
로지스틱 회귀는 시그모이드 함수의 W와 b를 찾는 문제이다.

https://icim.nims.re.kr/post/easyMath/64

Logistic Regression/Classification

Logistic Regression/Classification



Logistic Regression/Classification

Y = H(x) = Wx + b

$$g(z) = \frac{1}{1+e^{-z}}$$

z = H(x) = Wx + b

$$g(x) = \frac{1}{1+e^{-(Wx+b)}}$$

Logistic Regression/Classification

Y = H(x) = Wx + b

$g(z) = \frac{1}{1+e^{-z}}$

z = H(x) = Wx + b

$g(x) = \frac{1}{1+e^{-(Wx+b)}}$

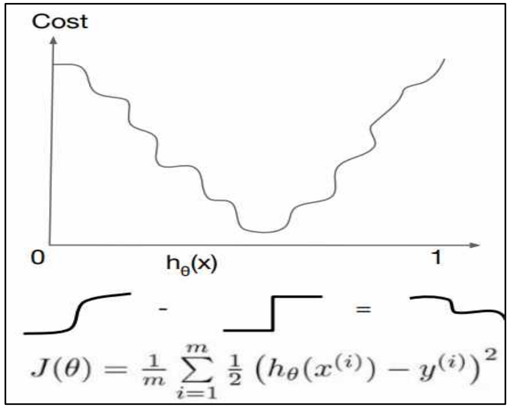
2진 분류
0.5를 기준으로 값을 분류

Logistic Regression/Classification

- 그렇다면 Logistic 회귀의 가설함수를 설계했으니,,,
- 회귀분석을 할 수 있겠다!
- Hypothesis
  - Z = H(x) = Wx + b
  - G(z) = $\frac{1}{1+e^{-z}}$
- Cost
- Minimize Cost

Logistic Regression/Classification

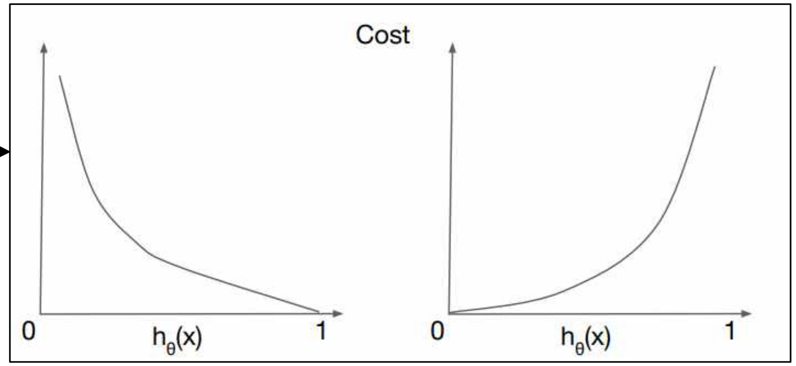
- Cost
  - 선형회귀때 처럼, cost = g(z) – Y으로 설계를 해야할까?
  - Cost(W,b) = $\frac{1}{m}\sum_{i=0}^{m}(H(x_i) - Y_i)^2$
  - Convex하지 못한 cost함수는 경사하강법을 적용할 수 없다.



- 따라서 새로운 cost함수를 정의해야한다.

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

cost(h_θ (x),y) = -ylog(h_θ(x)) - (1-y)log(1- h_θ(x))



Convex하게 정의가 됐다!

Logistic Regression/Classification

- Minimize Cost
  - 경사하강법을 이용해 Cost의 W와 b를 갱신한다.

Logistic Regression/Classification

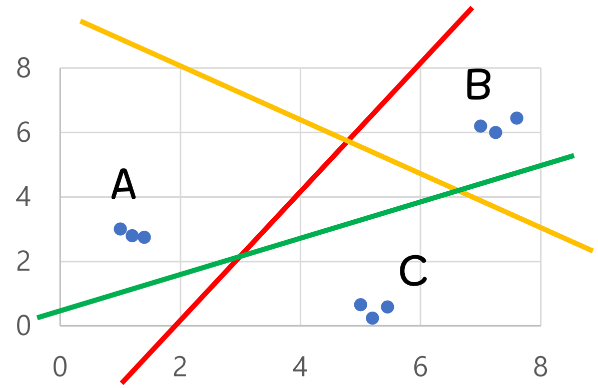
- Hypothesis
  - Z = H(x) = Wx + b
  - G(z) = $\frac{1}{1+e^{-z}}$
- Cost Function

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

$\mathrm{cost}(h_{\theta,}(x),y) = -y\log(h_\theta(x)) - (1-y)\log(1 - h_\theta(x))$

- Minimize Cost
  - 경사하강법을 통한 Cost function의 W,b 갱신

Softmax Regression/Classifier



- Multinomial Classification(다중 분류)
- Y의 값이 A,B,C 3개이고, 2진분류를 통해 분류한다면?
  - A or not
  - B or not
  - C or not
- 다변수 분류도 Matrix로 해결이 가능하다.

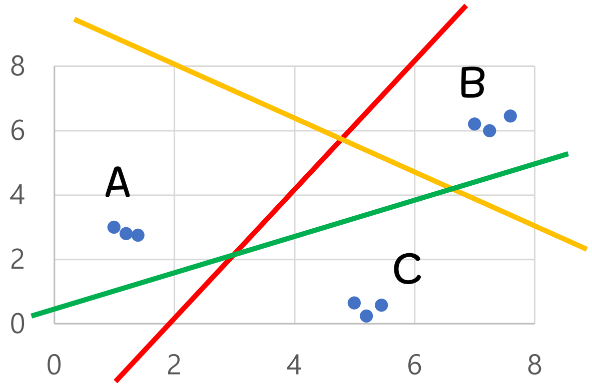
$H_1(x)$ → **Sigmoid A** → 0 ~ 1

$H_2(x)$ → **Sigmoid B** → 0 ~ 1

$H_3(x)$ → **Sigmoid C** → 0 ~ 1

Softmax Regression/Classifier

- 다변수 분류도 Matrix로 해결이 가능하다.



H1(x) → Sigmoid A → 0 ~ 1

H2(x) → Sigmoid B → 0 ~ 1

H3(x) → Sigmoid C → 0 ~ 1

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

Softmax Regression/Classifier

- 다변수 분류도 Matrix로 해결이 가능하다.

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

$$\begin{bmatrix} W_1, & W_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_1x_1 + W_2x_2 \end{bmatrix}$$

$$\begin{bmatrix} W_{A1}, & W_{A2} \\ W_{B1}, & W_{B2} \\ W_{C1}, & W_{C2} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} W_{A1}x_1 + W_{A2}x_2 \\ W_{B1}x_1 + W_{B2}x_2 \\ W_{C1}x_1 + W_{C2}x_2 \end{bmatrix}$$

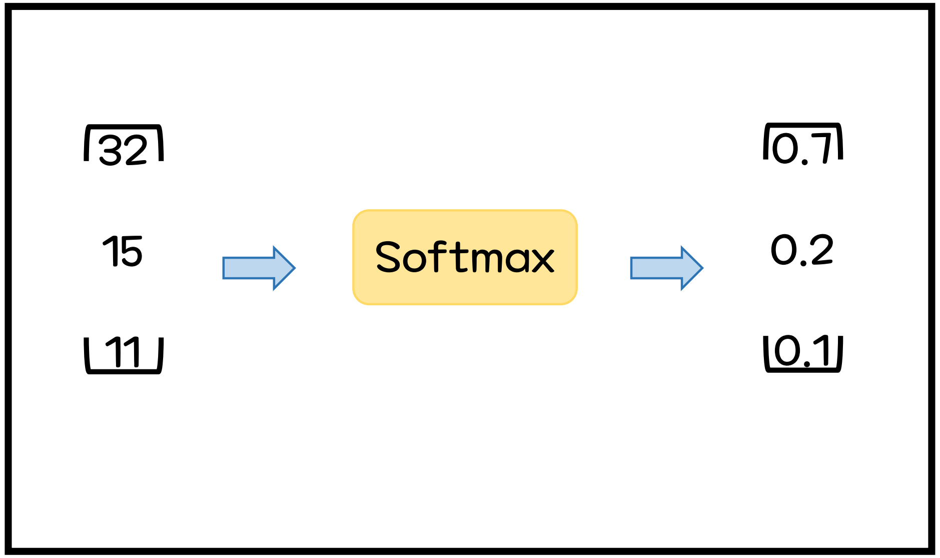
결국, $H(x) = Wx + b_n$ 처럼 표현이 가능하다.

Softmax Regression/Classifier

- 활성화 함수로 Softmax함수를 사용한다.
- Softmax함수는 입력값을 받아
- 0~1범위내로 압축한다.

$$\begin{bmatrix} W_{A1}x_1 + W_{A2}x_2 \\ W_{B1}x_1 + W_{B2}x_2 \\ W_{C1}x_1 + W_{C2}x_2 \end{bmatrix} \Rightarrow \begin{bmatrix} 32 \\ 15 \\ 11 \end{bmatrix}$$

결국, $H(x) = Wx + b_n$ 처럼 표현이 가능하다.



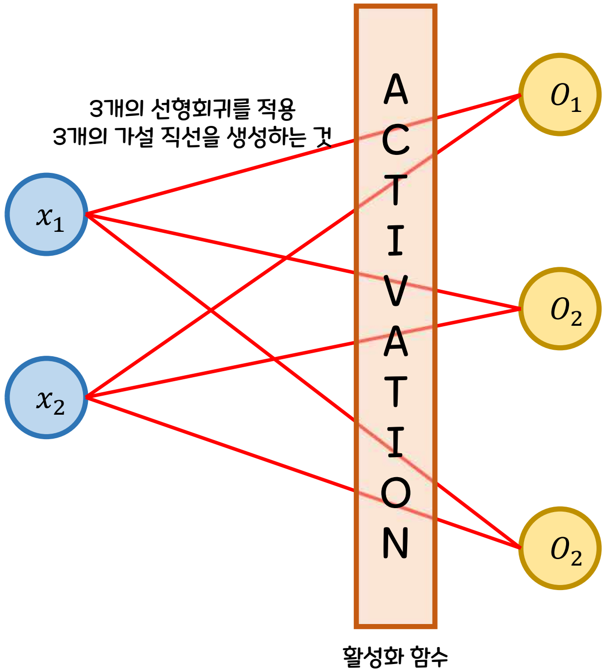
Softmax Regression/Classifier

- 0~1로 압축된 값들을 한번 더 가공을 한다.
- 대표적인 방법으로 one-hot encoding으로
- 가장 높은 값을 1로 나머지는 0으로 만드는 방법이다.

$$\begin{bmatrix}0.7\\0.2\\0.1\end{bmatrix} \Rightarrow \boxed{\text{One-hot encoding}} \Rightarrow \begin{bmatrix}1.0\\0.0\\0.0\end{bmatrix}$$

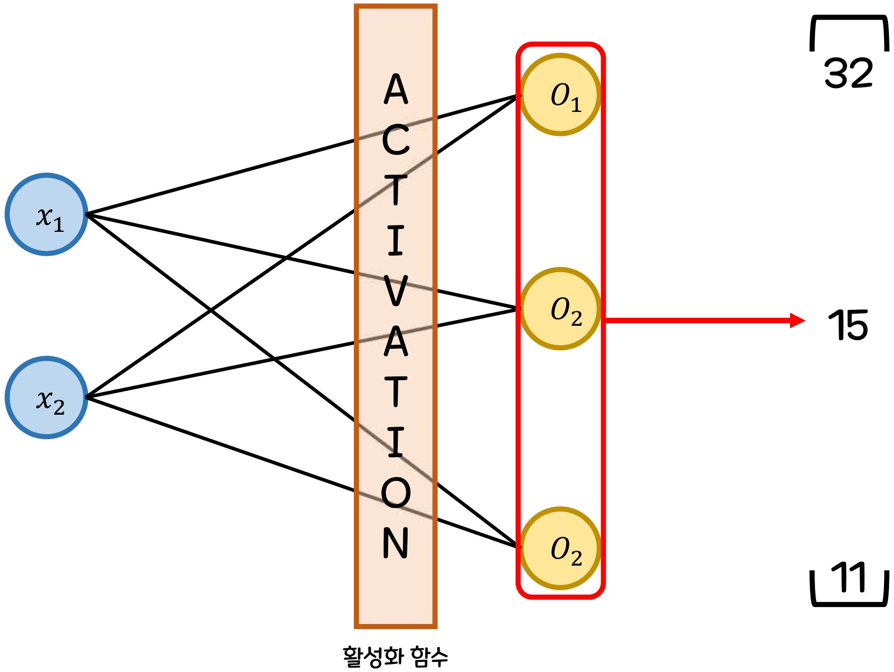
Softmax Regression/Classifier

- 정리해보면, (1개의 Layer를 가진 딥러닝 관점)



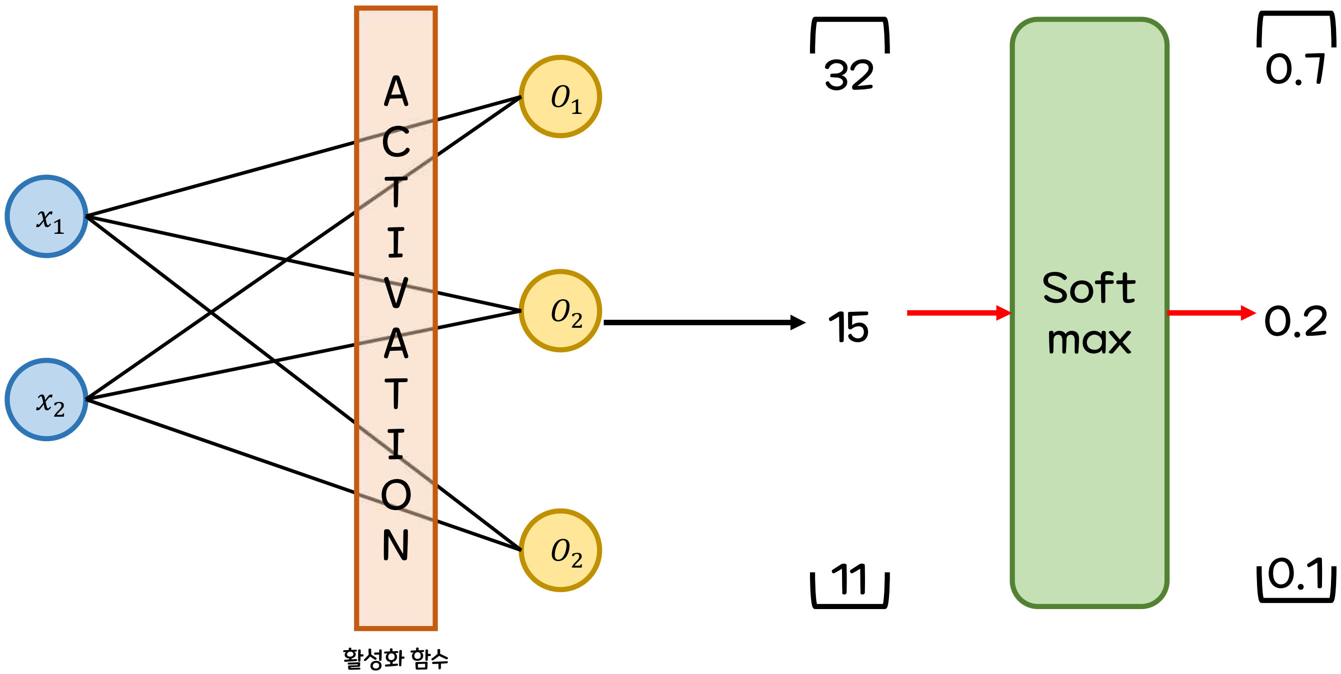
Softmax Regression/Classifier

- 정리해보면, (1개의 Layer를 가진 딥러닝 관점)



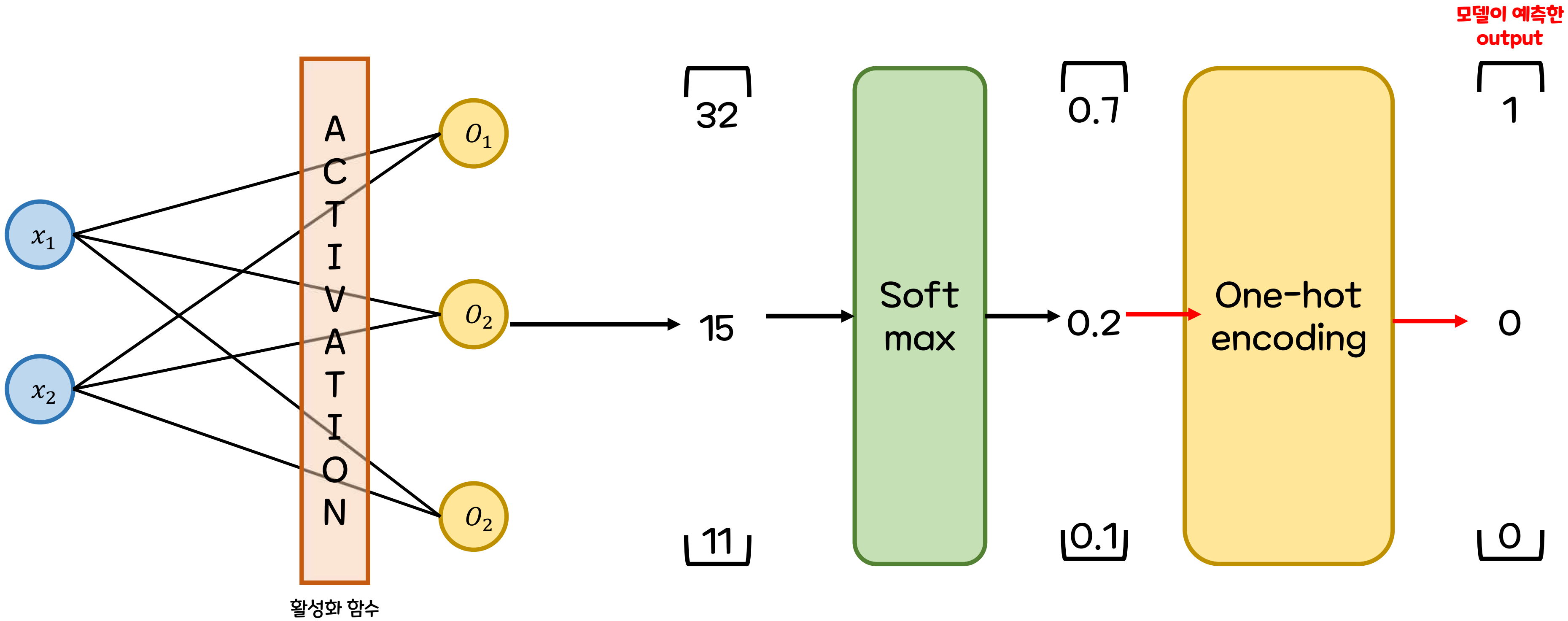
Softmax Regression/Classifier

- 정리해보면, (1개의 Layer를 가진 딥러닝 관점)



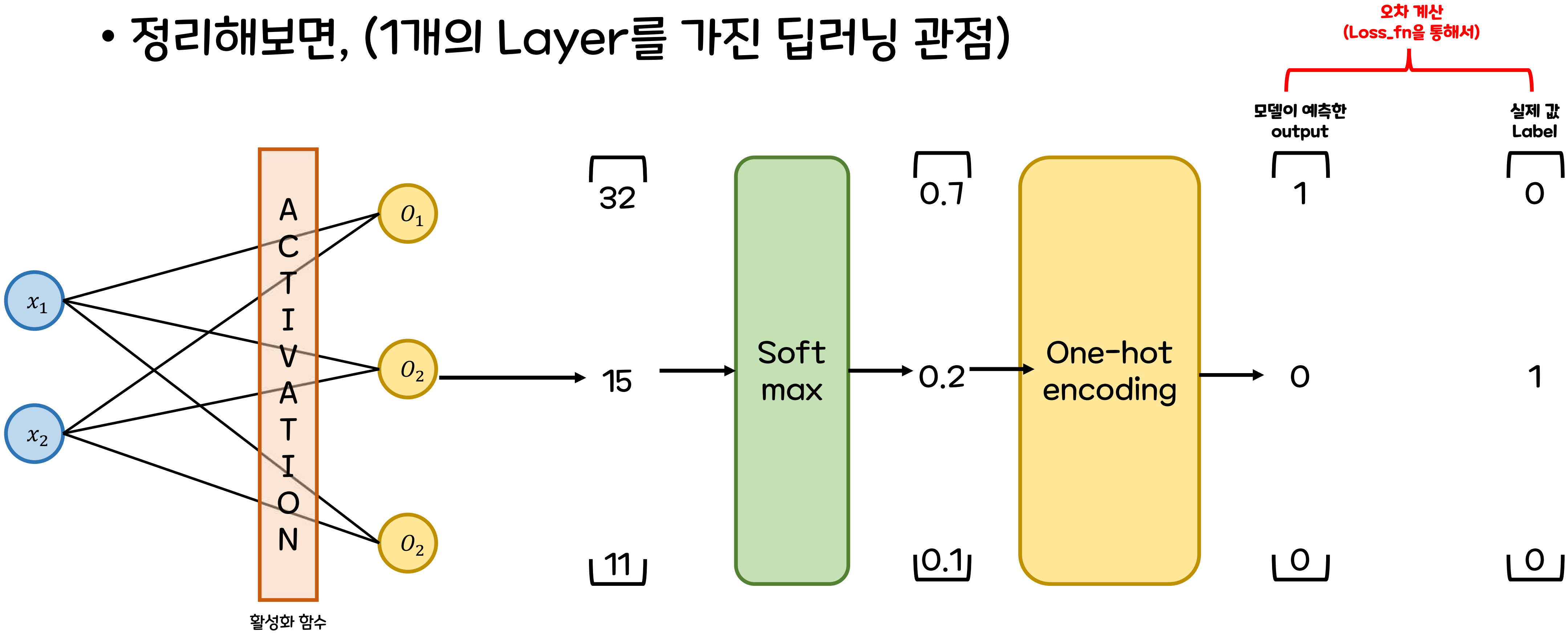
Softmax Regression/Classifier

- 정리해보면, (1개의 Layer를 가진 딥러닝 관점)



Softmax Regression/Classifier

- 정리해보면, (1개의 Layer를 가진 딥러닝 관점)



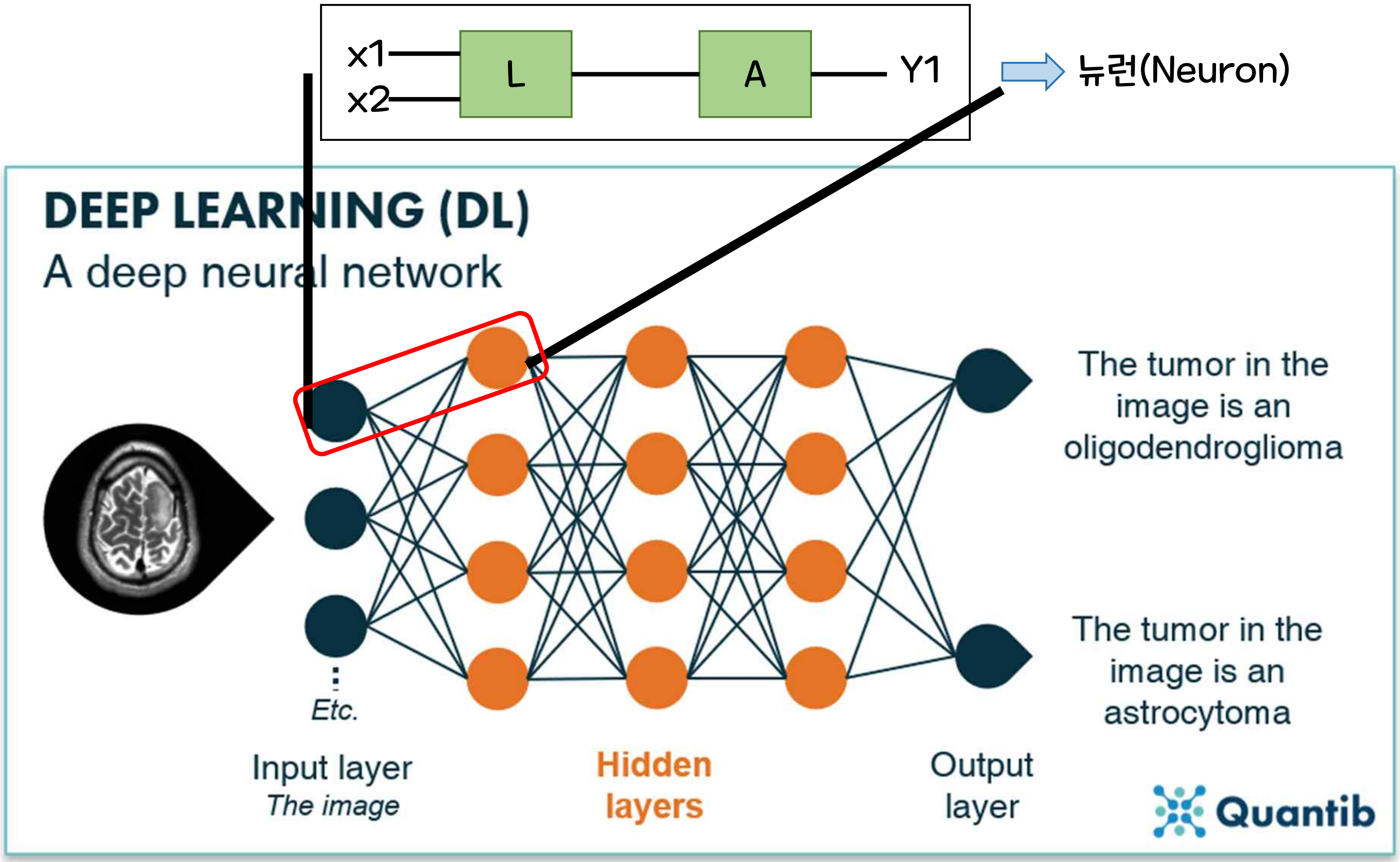
Softmax Regression/Classifier

- 가설을 설계했으니, 다음은...
  - Softmax(H(x)) = Softmax(Wx+b)
- 다중 분류에서의 Cost 함수는??
  - Cross – entropy
  - D(S,L) = $-\sum_{i=0}^{m} L_i \log(S_i)$
    - L_i : 실제 Y 값
    - S_i : 예측 값 = S(H(x))
  - 예측이 맞으면 cost함수는 0에 가까워지고, 틀리면 무한대로 수렴한다.
  - 즉, Convex function이므로 경사하강법을 적용할 수 있다.
- Logistic회귀의 Cost함수와, Softmax의 Cost함수는 같다는데…
  - 사실 이 부분은 이해하지 못했다.
  - 두 함수가 같은 이유는 두개의 예측 및 결과만 있기에 Logistic의 cost함수의 한쪽 부분이 0이되고 살아남은 부분이 Softmax의 그것과 같아진다.

Softmax Regression/Classifier

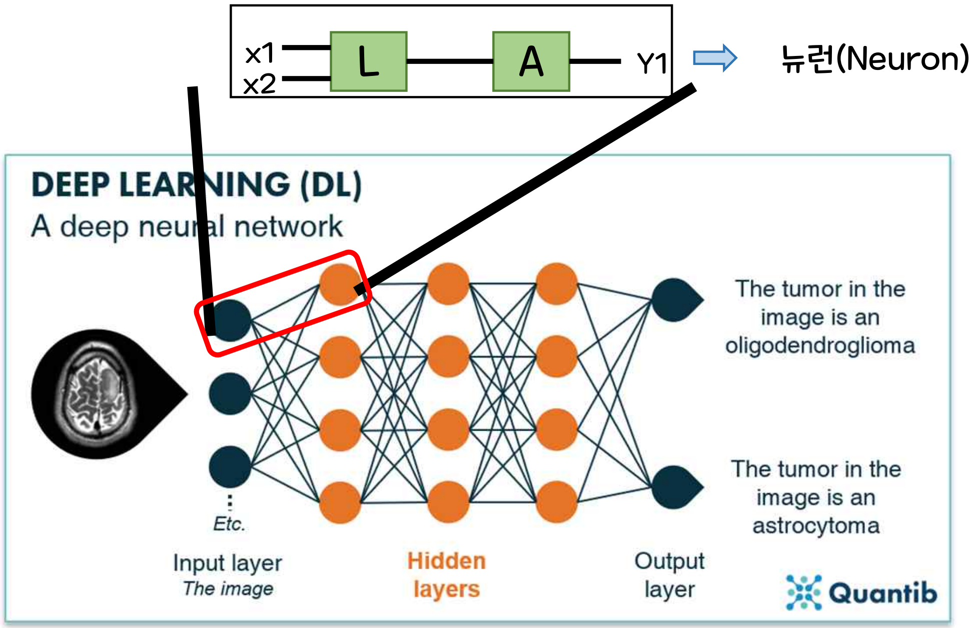
- Minimize Cost
  - Cost = $\frac{1}{N}\sum_{i}^{N} D(S_i, L_i)$
  - S_i = S(H(x)) = S(Wx+b)
  - 위 식을 이용해 경사하강법을 통해 cost를 최소화하는 W와 b 값을 찾는다.

Deep Learning



https://www.quantib.com/blog/how-does-deep-learning-work-in-radiology

Deep Learning



딥러닝은 연속된 층(Layer)에서 점진적으로
의미 있는 표현을 학습하는 새로운 방식이다.

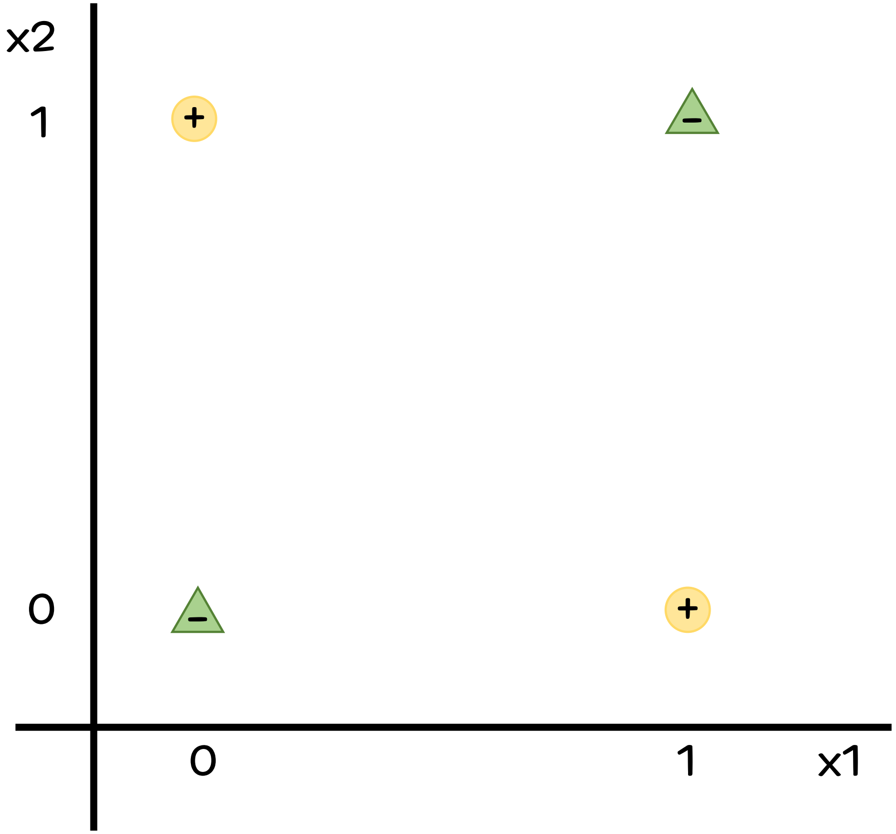
데이터로부터 모델을 만드는데 얼마나 많은 층을 사용했는지가
그 모델의 깊이(depth)가 된다.

딥러닝에서는 층을 겹겹히 쌓아 올려 구성한 신경망(Neural network)
이라는 모델을 사용하여 표현 층을 학습한다.

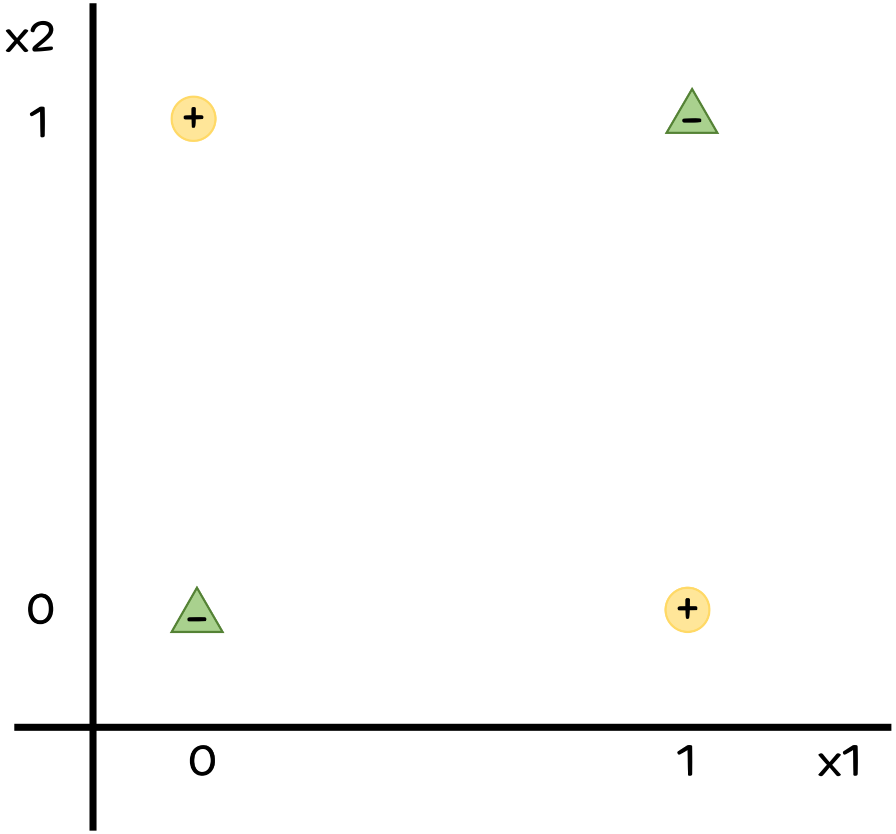
https://www.quantib.com/blog/how-does-deep-learning-work-in-radiology

Deep Learning - XOR

XOR	X1	X2	Output
input	0	0	0
	0	1	1
	1	0	1
	1	1	0

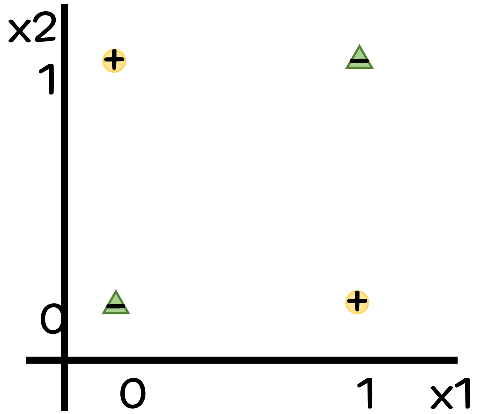


Deep Learning - XOR



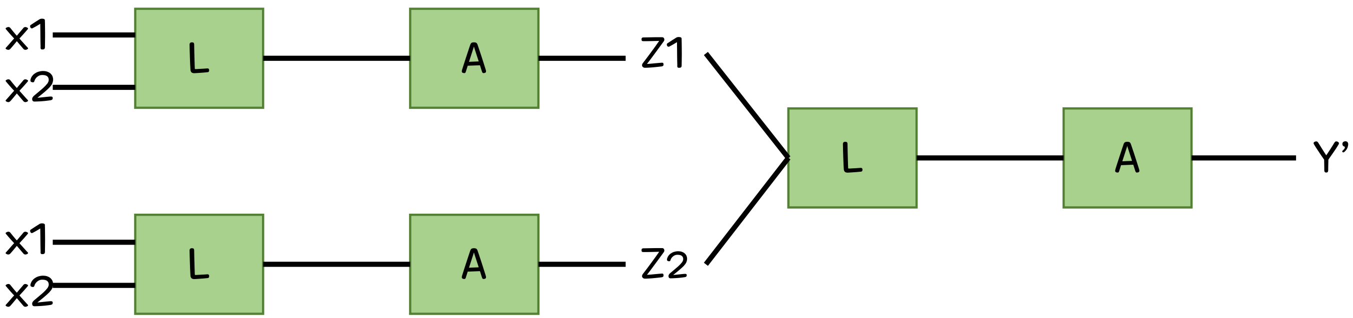
XOR은 기존의 분류방법으로는 분류가 안된다.

Deep Learning - XOR

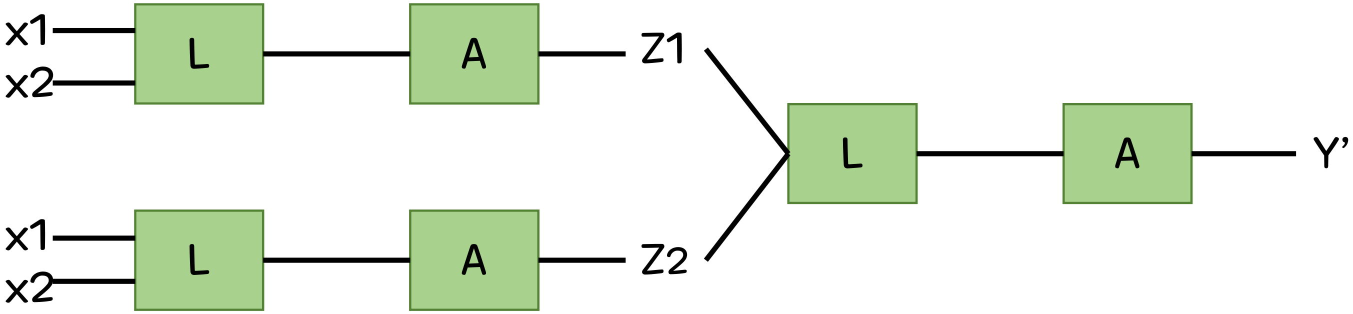


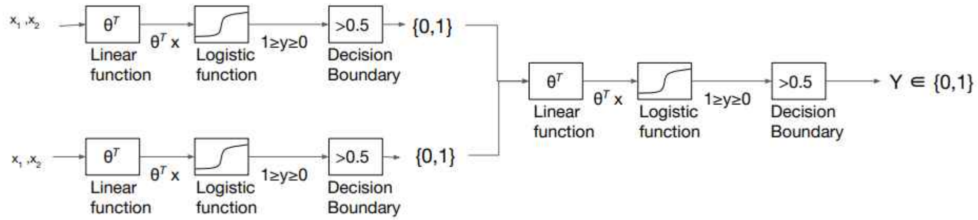
XOR은 기존의 분류방법으로는 분류가 안된다.

따라서, Multiple logistic model을 이용한다.
문제는 각 모델의 W,b를 어떻게 학습시키냐는 것이다.



Deep Learning - XOR





3개의 분류 모델을 사용함
Multi – layer
궁극적인 문제는 “어떻게 W와 b를 학습시킬 것이냐”

Gradient Descent 알고리즘??
⇒ 여러 모델이 중첩되어 있다보니 그 영향력도 고려해야므로
적절하지 않을 수 있다.

Deep Learning - XOR

- 가중치를 학습시키는 방법으로,,,
  - Back-Propagation이 이겠다.
  - 정방향으로 cost를 구하고 그 반대로 에러 값을 뒤로 전파시는 것
  - 역전파의 원리(?)는 미분이라고 할 수 있겠다...
  - 미분을 통해서 특정 Node의 가중치에 대해서 기여도를 구하는 것(?)