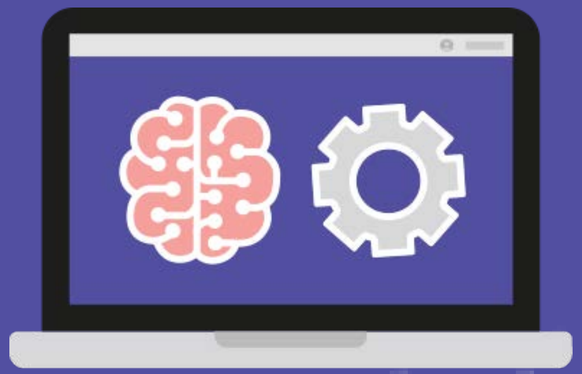
/* elice */

양재 AI School 인공지능 캠프

Lecture 13

인공신경망(ANN)과 손실함수, 역전파법



김도경 선생님

수업 목표

1. 인공신경망(ANN)과 손실함수
   1. 인공신경망(ANN)
   2. 손실함수(Loss function)
2. 학습 알고리즘

   GD, SGD, Momentum,

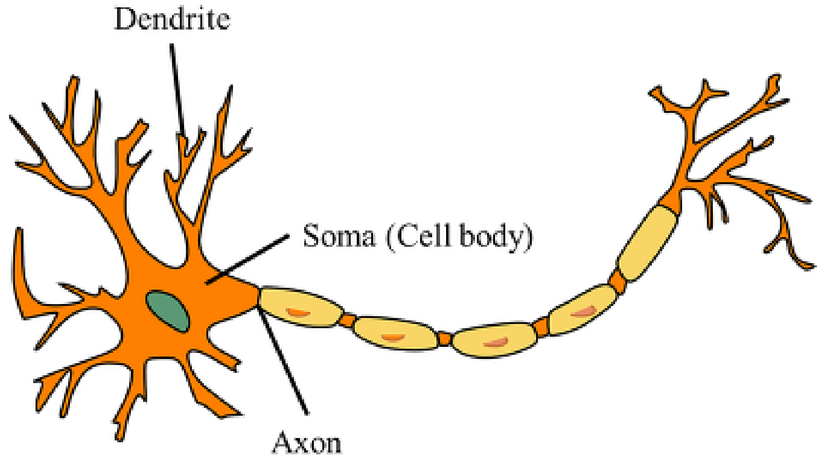
   AdaGrad, RMSProp, Adam
3. 역전파법(Back Propagation)
   1. 역전파법
   2. 역전파 구현

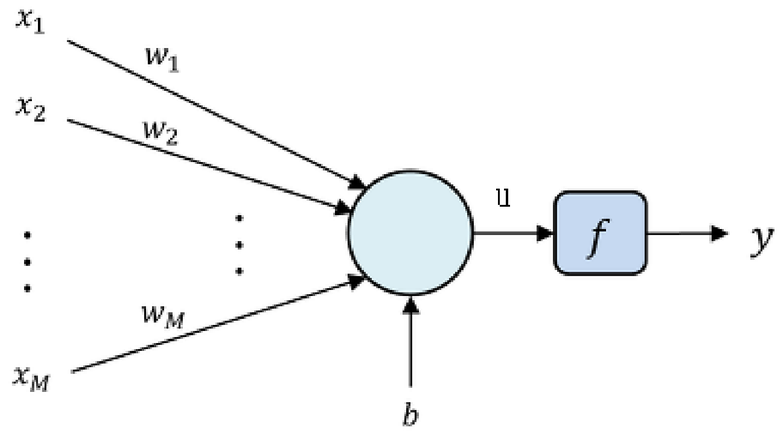
1. 인공신경망(ANN)과 손실함수

1-1. 인공신경망(ANN)

- 뇌는 신경세포(neuron)와 신경세포를 연결하는 시냅스(synapse)를 통해서 신호를 주고 받음으로써 정보를 저장하고 학습
- **인공신경망**(Artificial Neural Network)

  뇌의 학습방법을 수학적으로 모델링한 기계학습 알고리즘





$$u = \sum_{i=1}^{M} w_i x_i + b$$

$$y = f(u)$$

1-1. 인공신경망(ANN)

- 기본 용어

  x_i: 입력(Input)

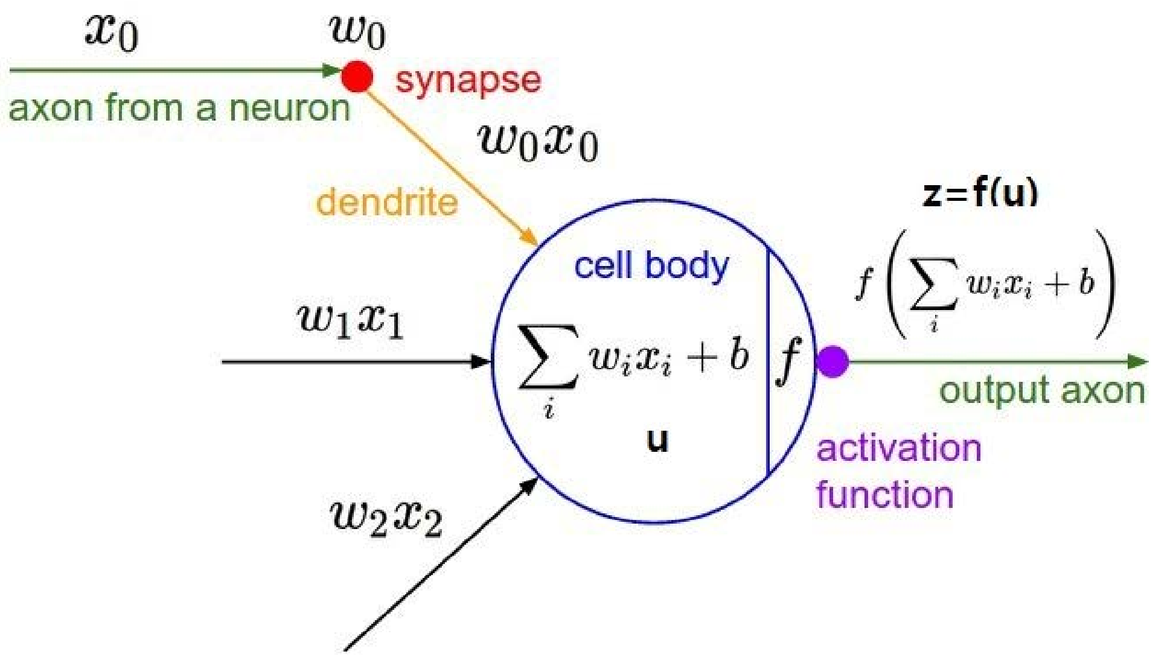
  w_i: 가중치(Weight)

  b: 편향(Bias)

  f: 활성화(Activation) 함수

  u: 선형결합(Net)

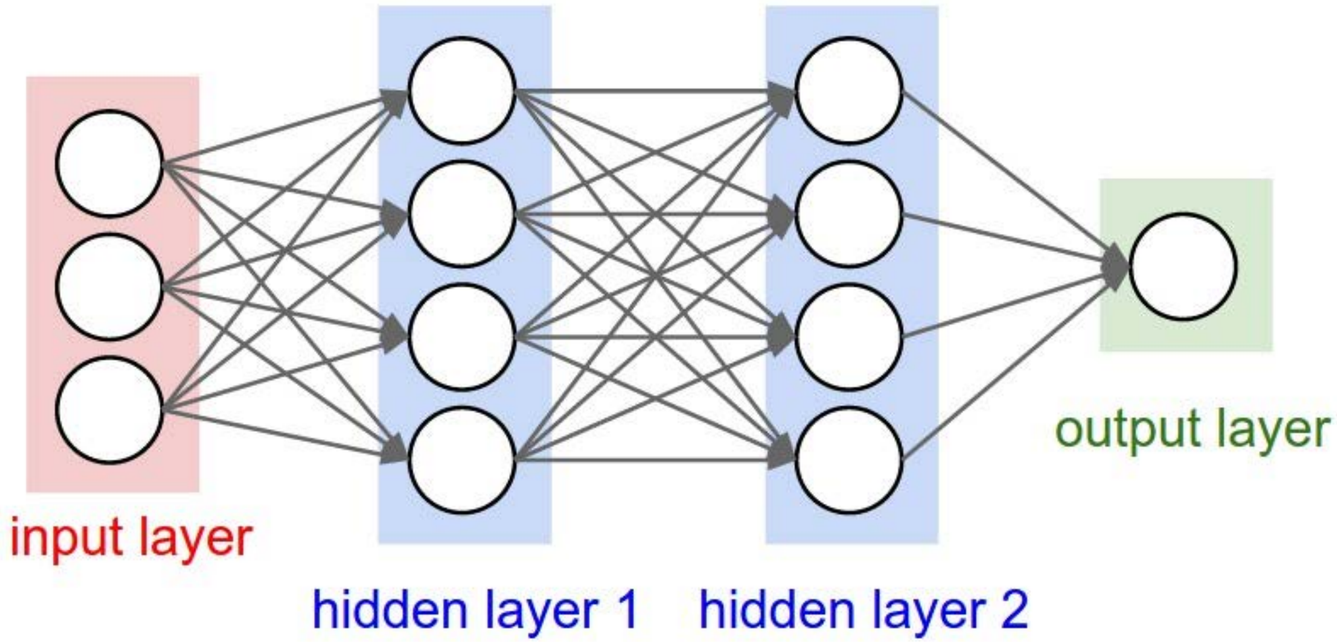
  z: 출력(Output)



- 뉴런에는 선형 결합과 활성화 함수 기능이 존재
- 시냅스는 뉴런과 뉴런을 연결해주는 가중치 역할을 담당

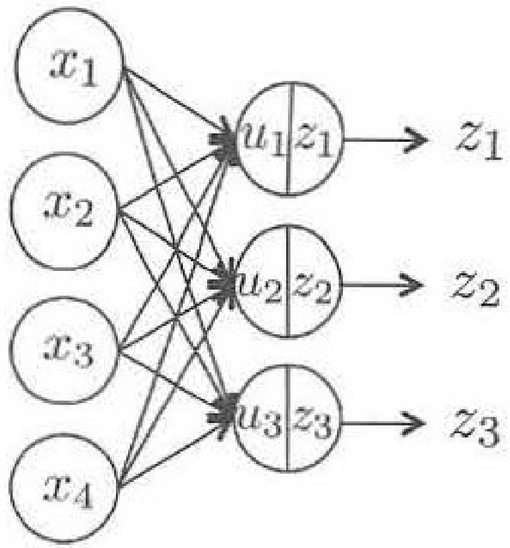
1-1. 인공신경망(ANN)

- 인공신경망은 **입력층**, **히든층**, **출력층**으로 구성
- 각 뉴런의 출력은 직접 전달되는 정보에만 의존할 뿐, 다른 정보들과는 무관
- 이 때문에 병렬처리가 가능하므로 연산속도가 매우 빠름



1-1. 인공신경망(ANN)

- 예) 1개의 레이어로 이루어진 MLP



$$u_1 = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + b_1$$

$$u_2 = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + b_2$$

$$u_3 = w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + b_3$$

1-1. 인공신경망(ANN)

- 벡터 및 행렬 기호 도입

$$\mathbf{u} = \begin{bmatrix} u_1 \\ \vdots \\ u_J \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_I \end{bmatrix}, \ \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_J \end{bmatrix}, \ \mathbf{z} = \begin{bmatrix} z_1 \\ \vdots \\ z_J \end{bmatrix}$$

$$\mathbf{W} = \begin{bmatrix} w_{11} & \cdots & w_{1I} \\ \vdots & \ddots & \vdots \\ w_{JI} & \cdots & w_{JI} \end{bmatrix}, \ \mathbf{f}(\mathbf{u}) = \begin{bmatrix} f(u_1) \\ \vdots \\ f(u_J) \end{bmatrix}$$

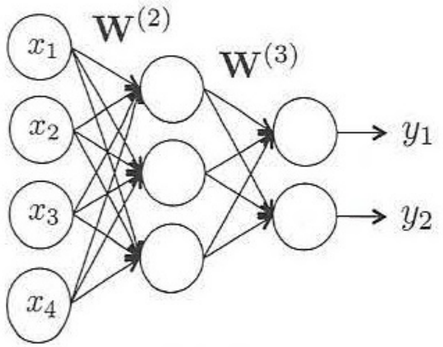
- 앞페이지의 식은 다음과 같이 간단하게 표현

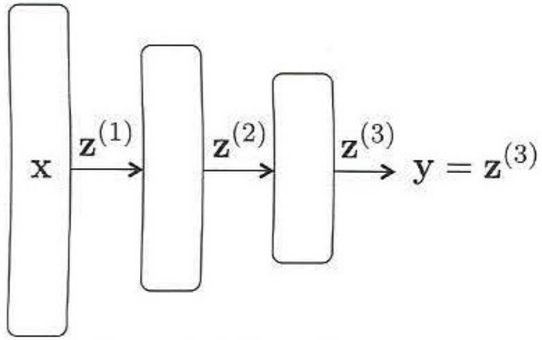
$$\mathbf{u} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

$$\mathbf{z} = \mathbf{f}(\mathbf{u})$$

1-1. 인공신경망(ANN)

- 예) 2개의 레이어로 이루어진 MLP에서의 식





$$\mathbf{u}^{(2)} = \mathbf{W}^{(2)}\mathbf{x} + \mathbf{b}^{(2)}$$

$$\mathbf{z}^{(2)} = \mathbf{f}(\mathbf{u}^{(2)})$$

$$\mathbf{u}^{(3)} = \mathbf{W}^{(3)}\mathbf{z}^{(2)} + \mathbf{b}^{(3)}$$

$$\mathbf{z}^{(3)} = \mathbf{f}(\mathbf{u}^{(3)})$$

- 임의의 개수의 레이어로 이루어진 신경망에서는 $\mathbf{z}^{(1)} = \mathbf{x}$라 두면

$$\mathbf{u}^{(l+1)} = \mathbf{W}^{(l+1)}\mathbf{z}^{(l)} + \mathbf{b}^{(l+1)}$$

$$\mathbf{z}^{(l+1)} = \mathbf{f}(\mathbf{u}^{(l+1)})$$

1-2. 손실함수(Loss function)

- **손실함수(Loss or Cost function)**

  신경망에서 내놓는 결과값과 실제 결과값 사이의 차이를 정의하는 함수

- **신경망 학습의 목표**

  손실함수를 최소화하는 것

  이를 위해 SGD 등의 학습 알고리즘 사용

1-2. 손실함수(Loss function)

- 신경망의 최종출력을 $\mathbf{y} = \mathbf{z}^{(L)}$로 표기하고

  Train set을 $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{d}_1), (\mathbf{x}_2, \mathbf{d}_2), \cdots, (\mathbf{x}_N, \mathbf{d}_N)\}$라 하자.

- **회귀(Regression)**

  손실함수로 제곱오차(Mean-squared error)를 사용

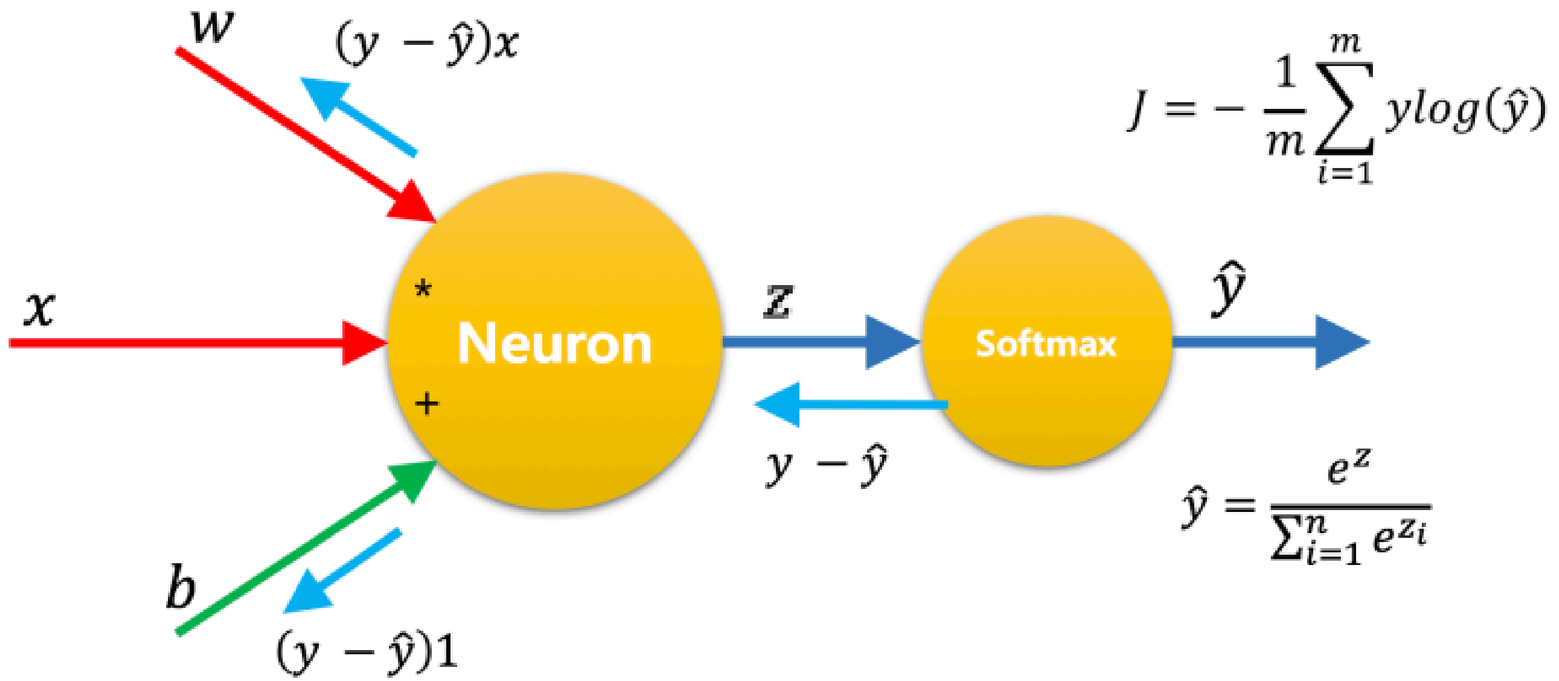
$$E(\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N} E_n(\mathbf{w}) = \frac{1}{2N}\sum_{n=1}^{N} ||\mathbf{d}_n - \mathbf{y}(\mathbf{x}_n; \mathbf{w})||^2$$

1-2. 손실함수(Loss function)

- **분류(Classification)**

  활성화 함수로 소프트맥스(Softmax) 함수,

  손실함수로 크로스 엔트로피(Cross entropy)를 사용



2. 학습 알고리즘

2. 학습 알고리즘

- **경사하강법(Gradient Descent)**

  1. 네트워크의 parameter들을 θ 라고 했을 때, 손실함수 J(θ)의 값을 최소화하기 위해 기울기(gradient) ∇J(θ)를 이용하는 방법

  2. GD에서는 gradient의 반대 방향으로 일정 크기만큼 이동하는 것을 반복하여 손실함수의 값을 최소화하는 θ의 값을 찾음

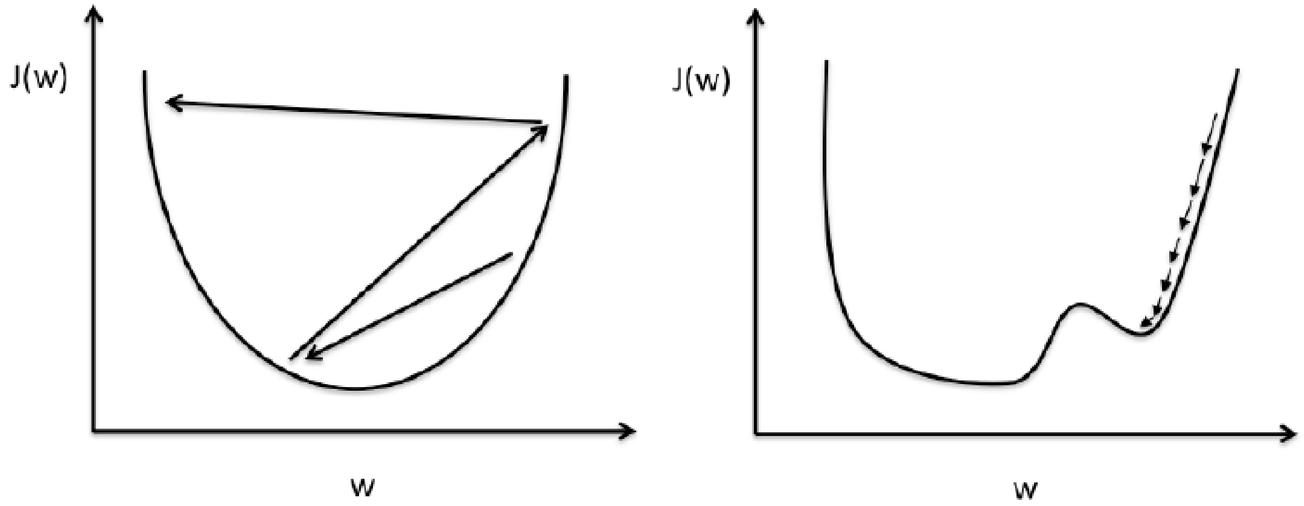
$$\theta = \theta - \eta \nabla_\theta J(\theta)$$

  3. 이 때 η는 미리 정해진 걸음의 크기(step size)로, 학습률(learning rate)이라고 함. 보통 0.01~0.001 정도를 사용

2. 학습 알고리즘

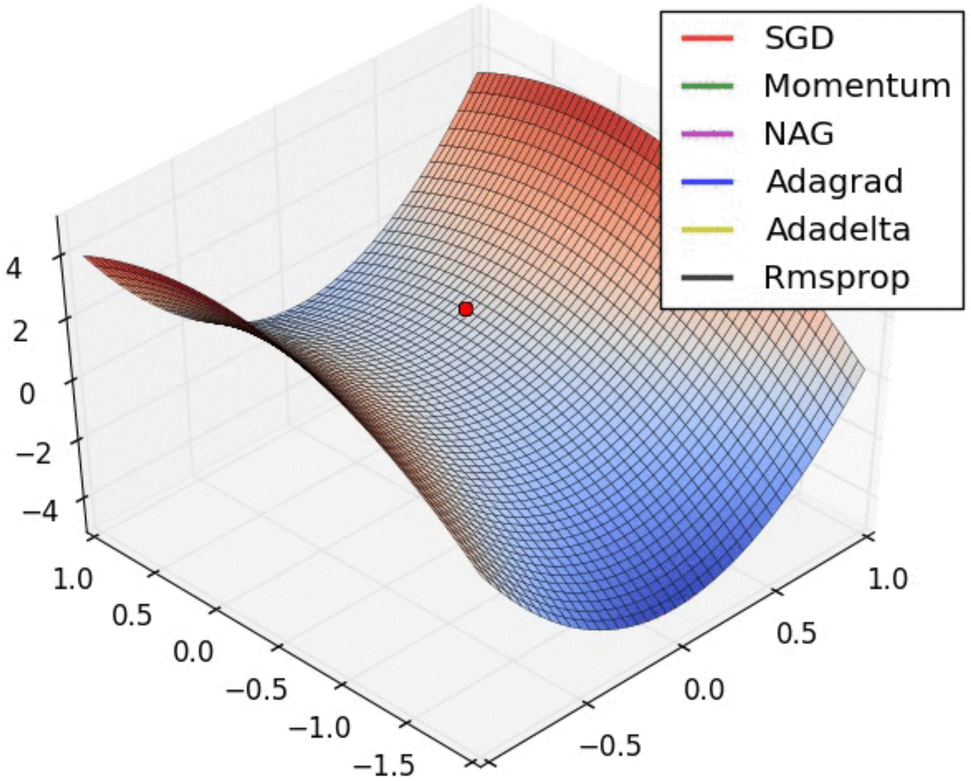
학습률(Learning rate)

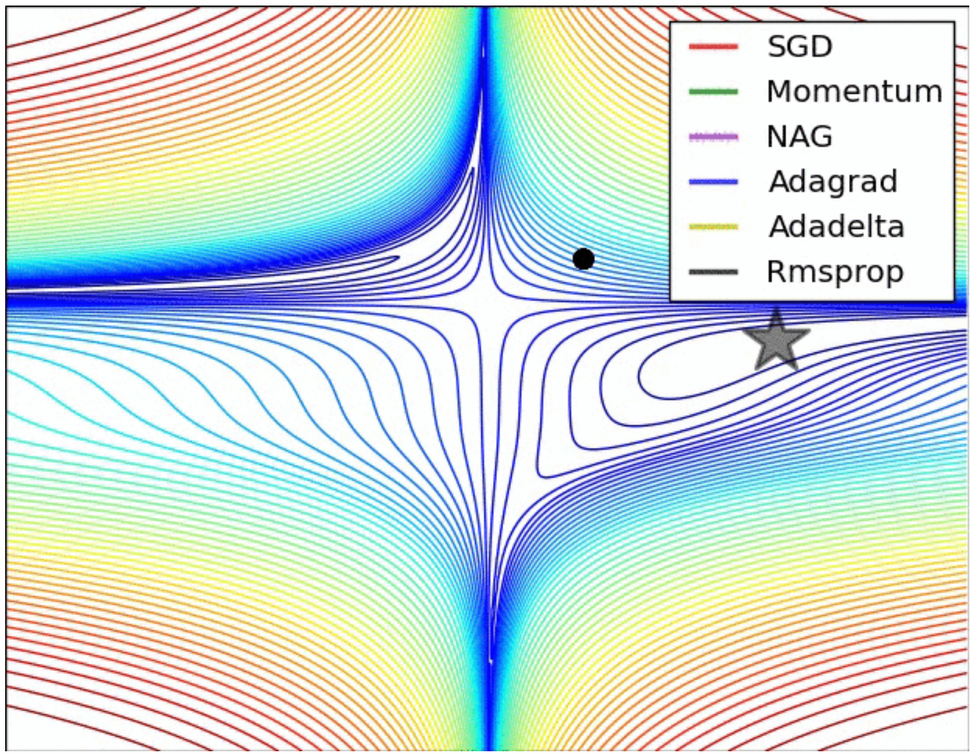
- 학습률이 너무 크면 듬성듬성하고 최소값(global minimum)을 지나쳐 갈수 있음
- 학습률이 너무 작으면 학습을 촘촘히 해서 학습속도가 느려지고 극소값(Local minimum)에 빠질 수 있음



2. 학습 알고리즘

- 여러 학습 알고리즘간의 **수렴속도 비교**



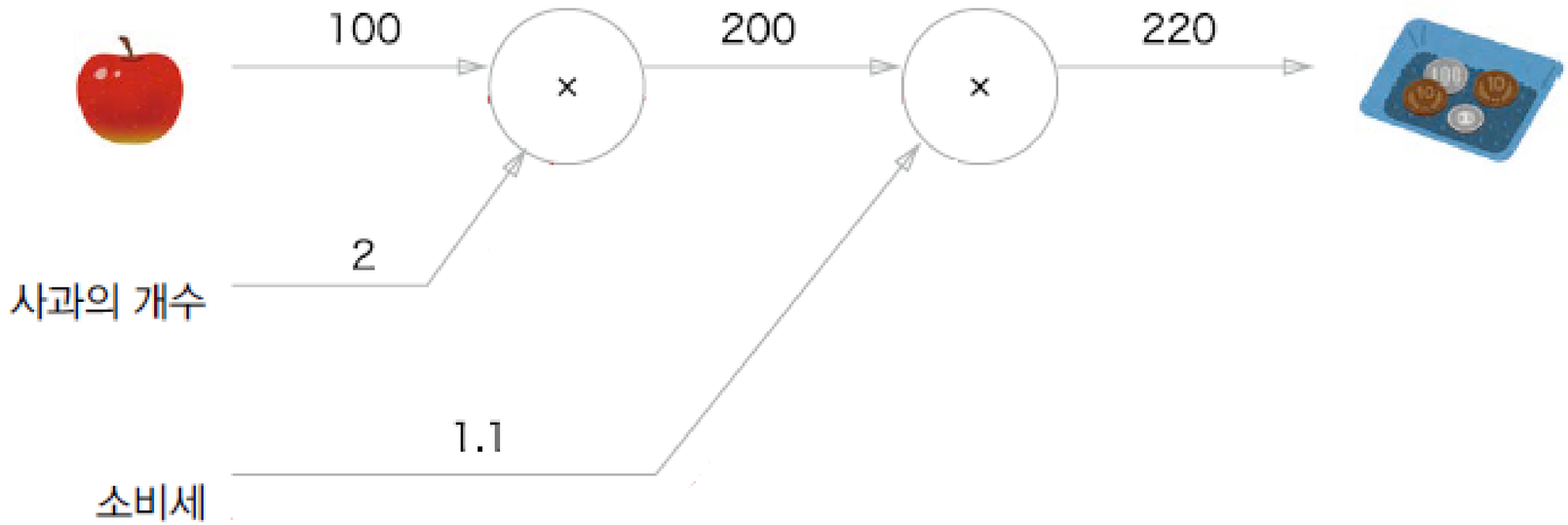


3. 역전파법(Back Propagation)

3-1. 역전파법(Back Propagation)

계산 그래프(computational graph)

- 계산 과정을 그래프로 나타낸 것
- 노드(node)와 엣지(edge)로 표현
- 노드는 연산을, 엣지는 데이터가 흘러가는 방향을 나타냄



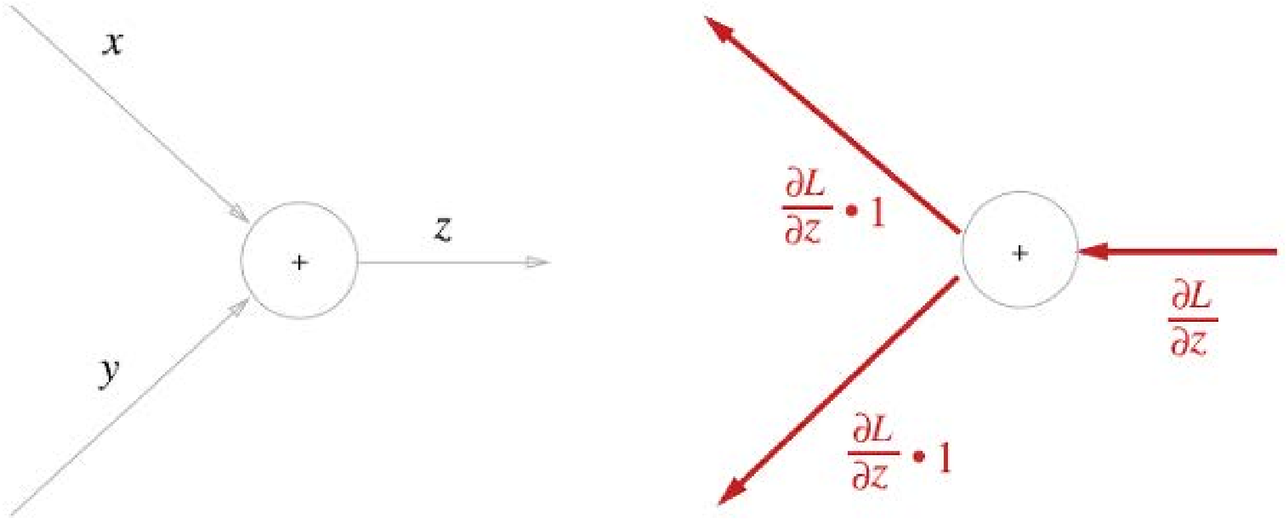
3-1. 역전파법(Back Propagation)

- 덧셈노드

$$z = x + y$$

$$\frac{\partial z}{\partial x} = 1$$

$$\frac{\partial z}{\partial y} = 1$$



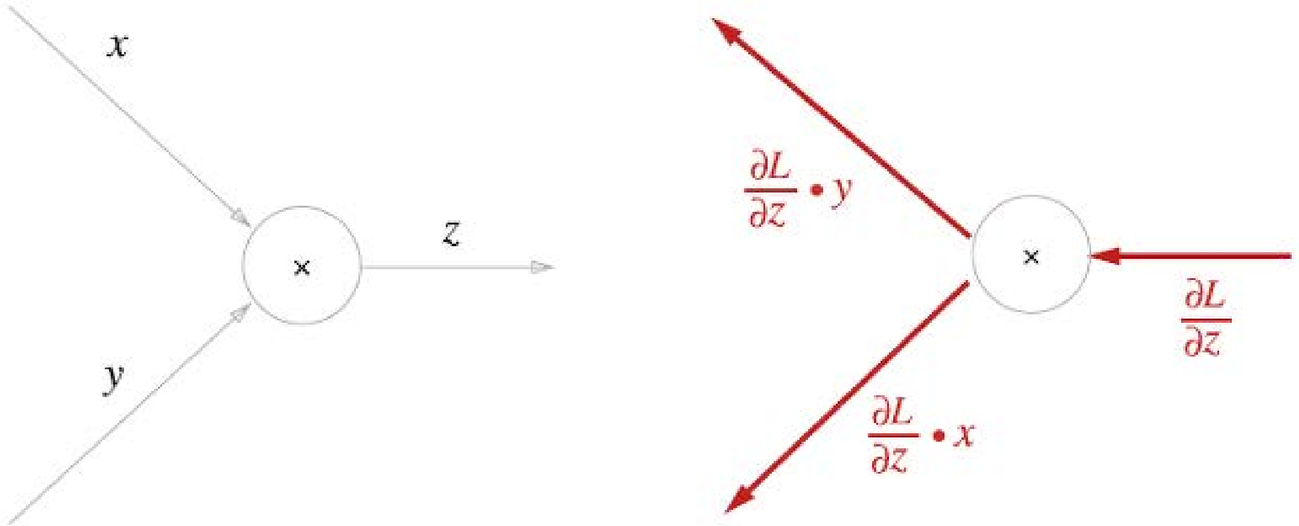
3-1. 역전파법(Back Propagation)

- 곱셈노드

$$z = xy$$

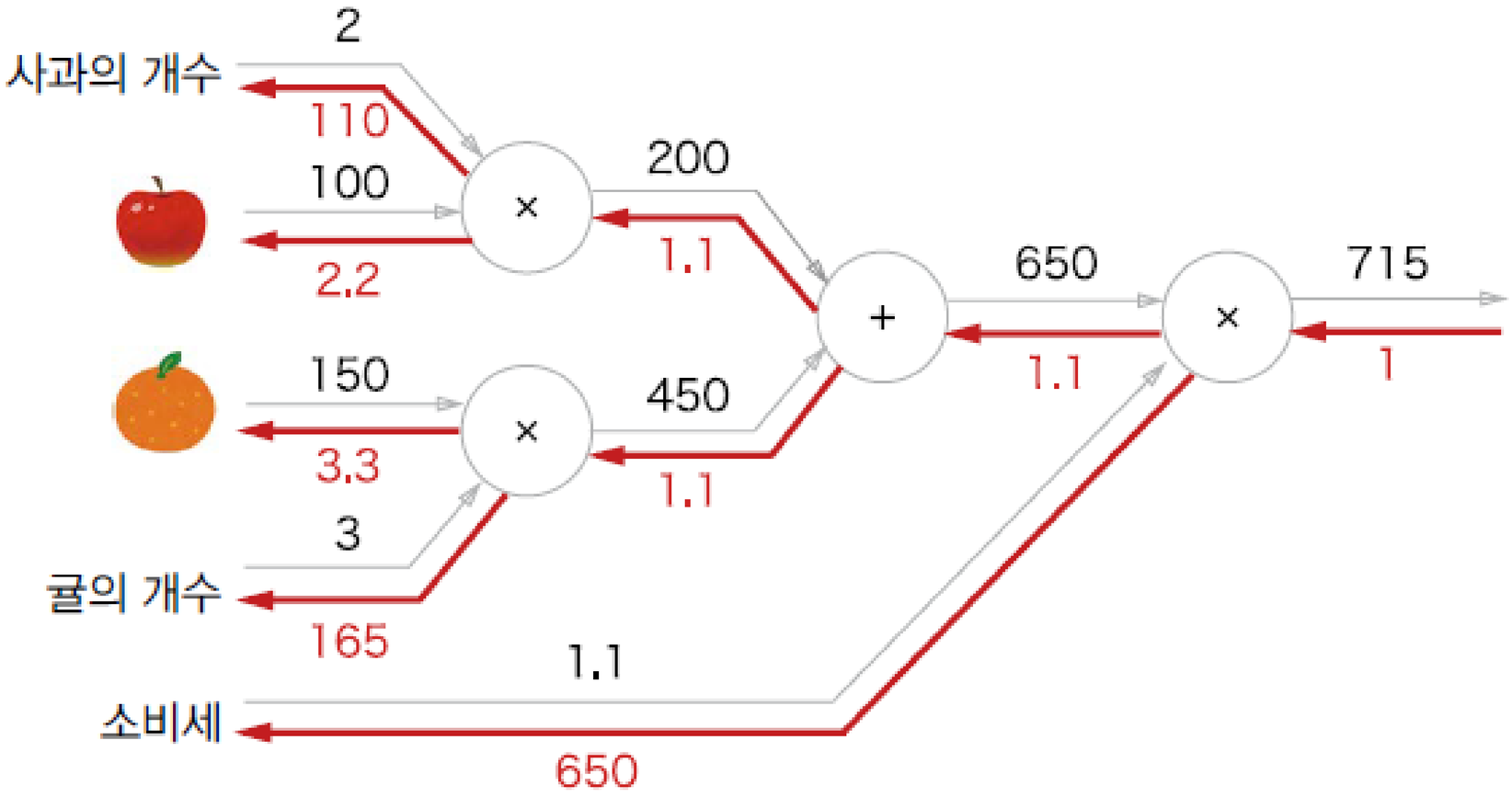
$$\frac{\partial z}{\partial x} = y$$

$$\frac{\partial z}{\partial y} = x$$



3-1. 역전파법(Back Propagation)

- **계산그래프에서의 역전파**



3-1. 역전파법(Back Propagation)

- **합성함수 미분법(Chain rule) 1**

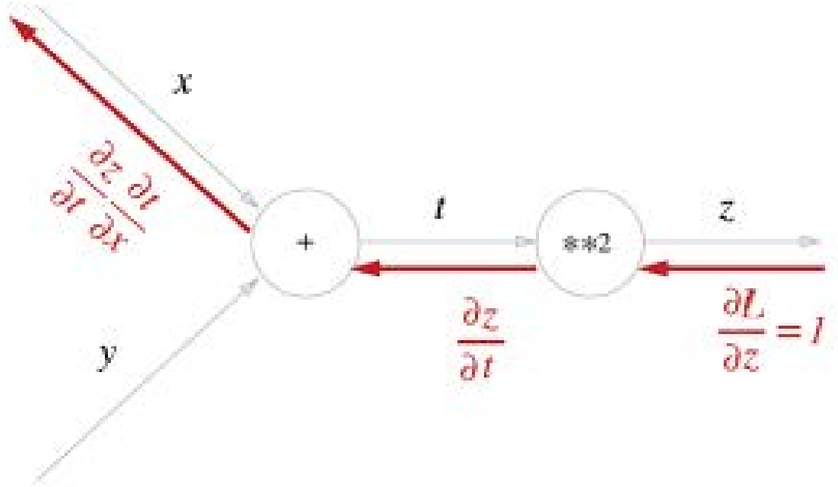
$$t = x + y$$

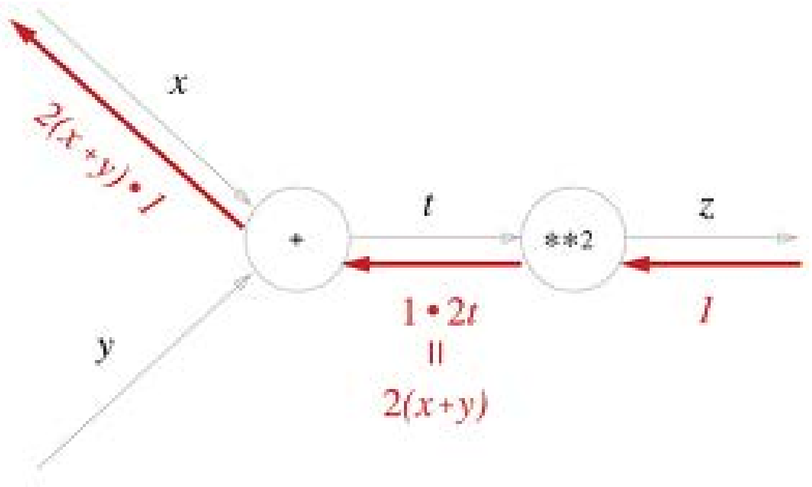
$$z = t^2$$

$$\frac{\partial z}{\partial t} = 2t$$

$$\frac{\partial t}{\partial x} = 1$$

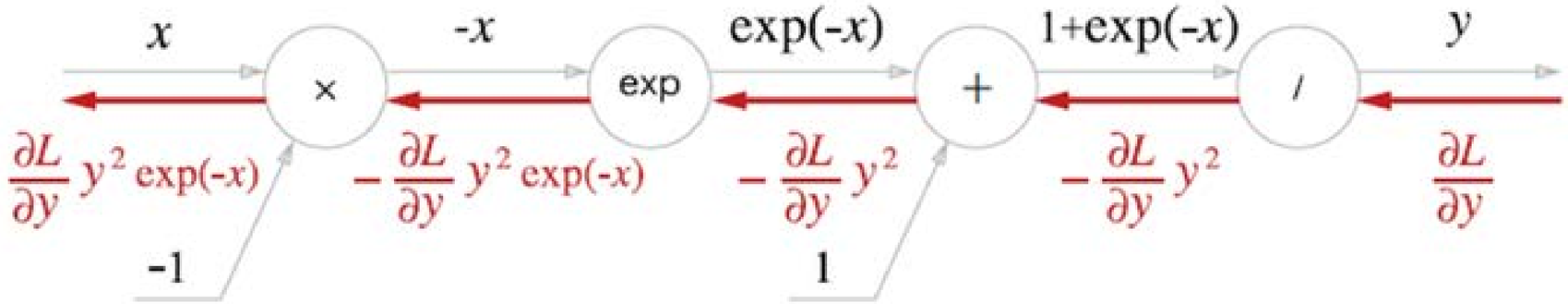
$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial t}\frac{\partial t}{\partial x} = 2t \cdot 1 = 2(x+y)$$

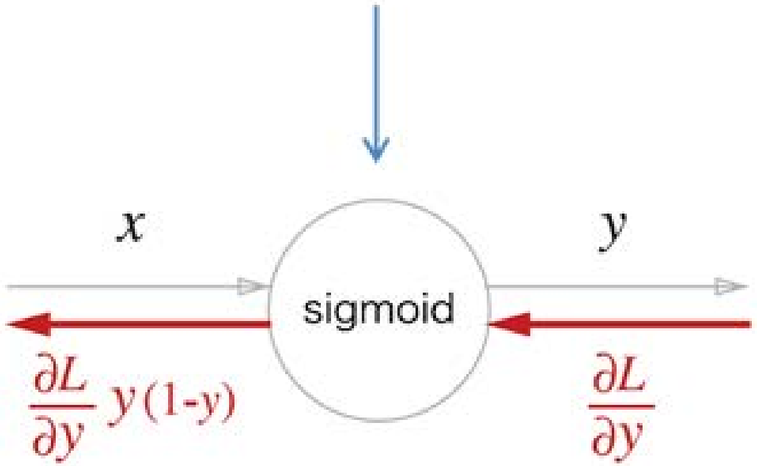




3-1. 역전파법(Back Propagation)

- 시그모이드 함수의 역전파



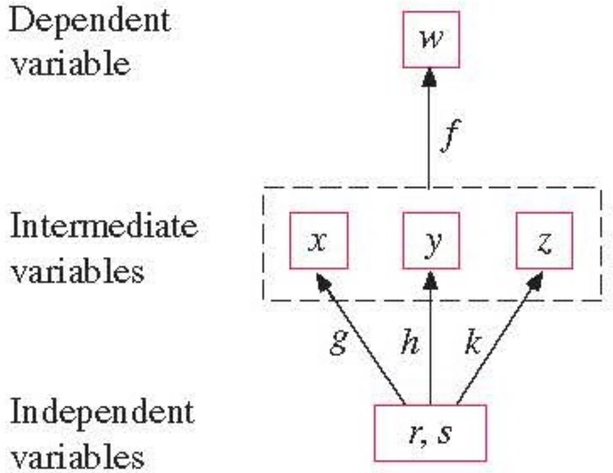


$$y = \frac{1}{1 + \exp(-x)}$$

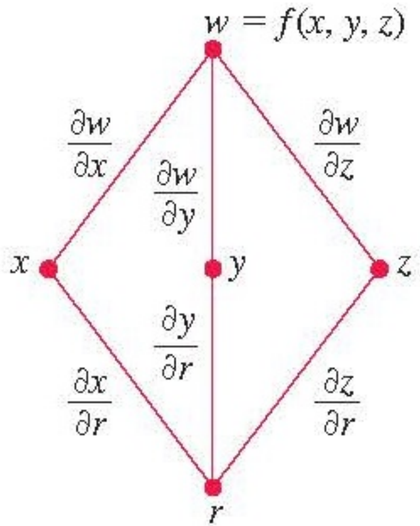
$$\begin{aligned}\frac{\partial L}{\partial y} y^2 exp(-x) &= \frac{\partial L}{\partial y} \frac{1}{(1 + exp(-x))^2} \exp(-x) \\ &= \frac{\partial L}{\partial y} \frac{1}{1 + exp(-x)} \frac{exp(-x)}{1 + exp(-x)} \\ &= \frac{\partial L}{\partial y} y(1 - y)\end{aligned}$$

3-1. 역전파법(Back Propagation)

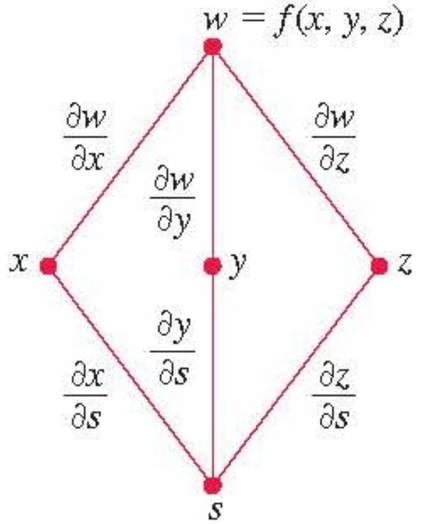
- **합성함수 미분법(Chain rule) 2**



$$w = f(g(r, s), h(r, s), k(r, s))$$



$$\frac{\partial w}{\partial r} = \frac{\partial w}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial w}{\delta y}\frac{\partial y}{\partial r} + \frac{\partial w}{\partial z}\frac{\partial z}{\partial r}$$



$$\frac{\partial w}{\partial s} = \frac{\partial w}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial w}{\partial y}\frac{\partial y}{\partial s} + \frac{\partial w}{\partial z}\frac{\partial z}{\partial s}$$

3-1. 역전파법(Back Propagation)

- **연습문제**

$$u = x^2 + 2y$$

$$x = r\sin t, \qquad y = \sin^2 t$$

일 때 $\frac{\partial u}{\partial r}, \frac{\partial u}{\partial t}$를 구하여라.

3-1. 역전파법(Back Propagation)

- **정답**

$$\frac{\partial u}{\partial r} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial r} = (2x)(\sin(t)) + (2)(0) = 2r\sin^2(t),$$

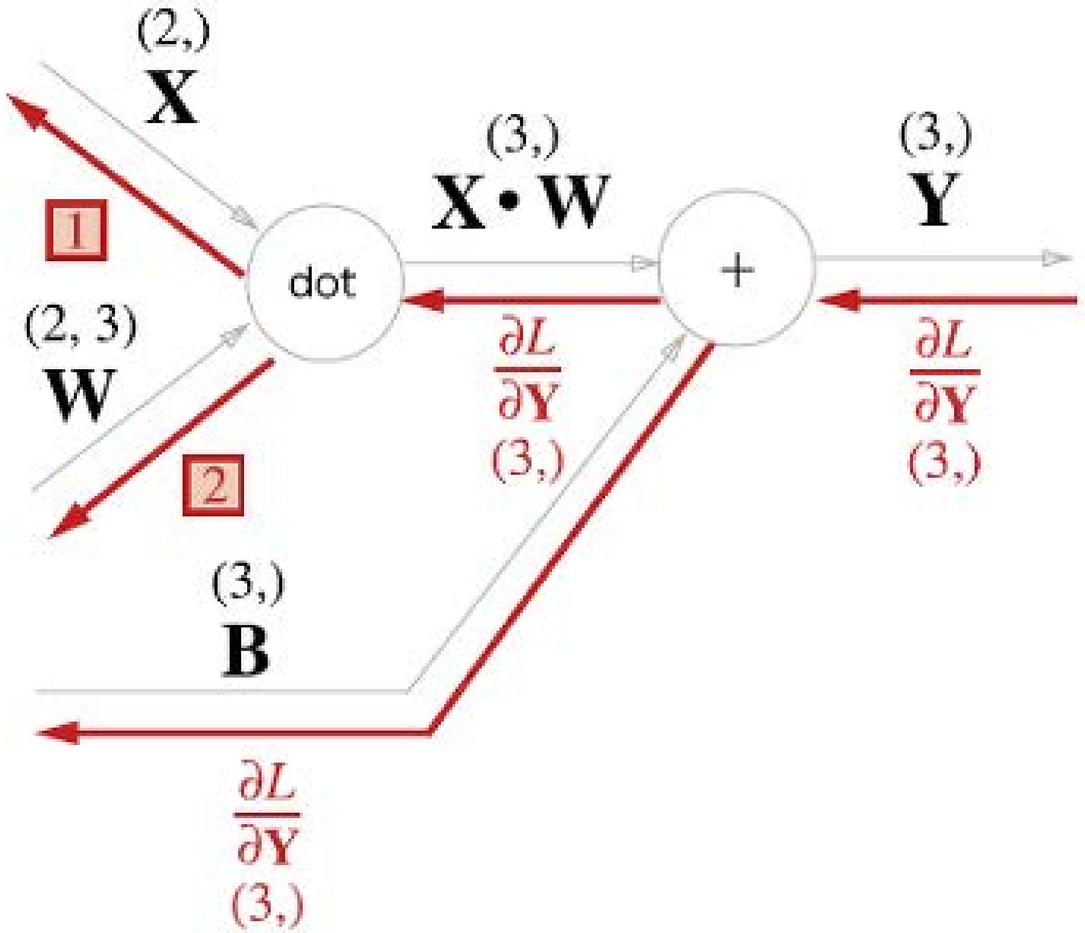
$$\begin{aligned}\frac{\partial u}{\partial t} &= \frac{\partial u}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial t} \\ &= (2x)(r\cos(t)) + (2)(2\sin(t)\cos(t)) \\ &= (2r\sin(t))(r\cos(t)) + 4\sin(t)\cos(t) \\ &= 2(r^2+2)\sin(t)\cos(t) \\ &= (r^2+2)\sin(2t).\end{aligned}$$

3-1. 역전파법(Back Propagation)

- **행렬연산과 역전파**

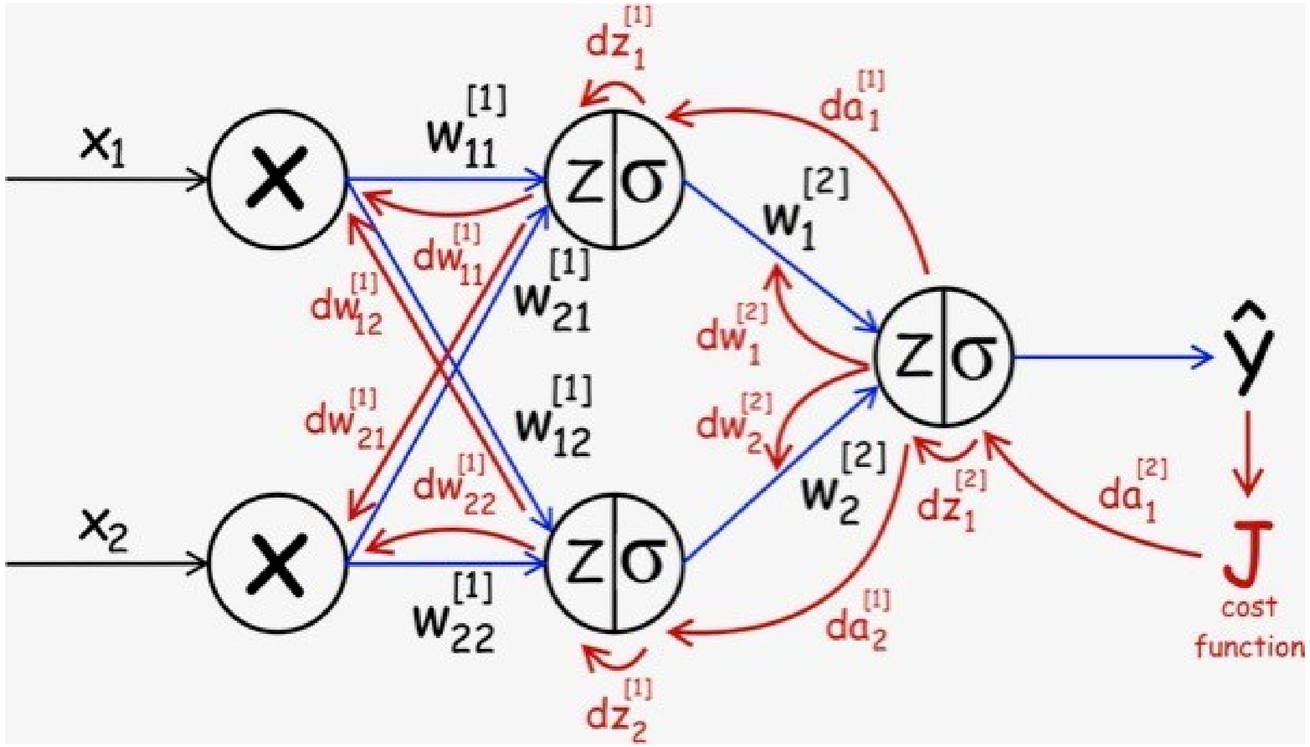
$$\boxed{1}\quad \underset{(2,)}{\frac{\partial L}{\partial \mathbf{X}}} = \underset{(3,)}{\frac{\partial L}{\partial \mathbf{Y}}}\ \underset{(3,2)}{\mathbf{W}^{\mathrm{T}}}$$

$$\boxed{2}\quad \underset{(2,3)}{\frac{\partial L}{\partial \mathbf{W}}} = \underset{(2,1)}{\mathbf{X}^{\mathrm{T}}}\ \underset{(1,3)}{\frac{\partial L}{\partial \mathbf{Y}}}$$



3-2. 역전파 구현

- binary 분류용 2-layer Neural Net에 대한 역전파



3-2. 역전파 구현

- binary 분류용 2-layer Neural Net에 대한 역전파

$$
\begin{aligned}
Z^{[1]} &= W^{[1]} X + b^{[1]} \\
A^{[1]} &= \sigma(Z^{[1]}) \\
Z^{[2]} &= W^{[2]} A^{[1]} + b^{[2]} \\
\hat{y} &= A^{[2]} = \sigma(Z^{[2]})
\end{aligned}
$$

$$
\begin{aligned}
W^{[1]} &=: W^{[1]} - \alpha \frac{dJ}{dW^{[1]}} \\
b^{[1]} &=: b^{[1]} - \alpha \frac{dJ}{db^{[1]}} \\
W^{[2]} &=: W^{[2]} - \alpha \frac{dJ}{dW^{[2]}} \\
b^{[2]} &=: b^{[2]} - \alpha \frac{dJ}{db^{[2]}}
\end{aligned}
$$

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

$$J = -\frac{1}{n}\left(Y log\left(A^{[2]}\right) - (1-Y)\, log\left(1 - A^{[2]}\right)\right)$$

$$\frac{dJ}{dW^{[2]}} = \left[-\frac{Y}{A^{[2]}} + \frac{1-Y}{1-A^{[2]}}\right]\left[A^{[2]}(1-A^{[2]})\right]\left[A^{[2]}\right]$$

$$\begin{aligned}\frac{dJ}{db^{[2]}} &= \frac{dJ}{dA^{[2]}}\frac{dA^{[2]}}{dZ^{[2]}}\frac{dZ^{[2]}}{db^{[2]}} \\ &= \left[A^{[2]} - Y\right]\left[1\right] \\ &= \left[A^{[2]} - Y\right] \\ &= dZ^{[2]}\end{aligned}$$

3-2. 역전파 구현

- binary 분류용 2-layer Neural Net에 대한 역전파

$$
\begin{aligned}
\frac{dJ}{dW^{[1]}} &= \frac{dJ}{dA^{[2]}}\frac{dA^{[2]}}{dZ^{[2]}}\frac{dZ^{[2]}}{dA^{[1]}}\frac{dA^{[1]}}{dZ^{[1]}}\frac{dZ^{[1]}}{dW^{[1]}} \\
&= \frac{dJ}{dZ^{[2]}}\frac{dZ^{[2]}}{dA^{[1]}}\frac{dA^{[1]}}{dZ^{[1]}}\frac{dZ^{[1]}}{dW^{[1]}} \\
&= \left[A^{[2]}-Y\right]\left[W^{[2]}\right]\left[g'\left(Z^{[1]}\right)\right]\left[A^{[0]}\right] \\
&= dZ^{[2]}W^{[2]}g'\left(Z^{[1]}\right)A^{[0]} \\
&= dZ^{[1]}A^{[0]}
\end{aligned}
$$

$$
\begin{aligned}
\frac{dJ}{db^{[1]}} &= \frac{dJ}{dA^{[2]}}\frac{dA^{[2]}}{dZ^{[2]}}\frac{dZ^{[2]}}{dA^{[1]}}\frac{dA^{[1]}}{dZ^{[1]}}\frac{dZ^{[1]}}{db^{[1]}} \\
&= \frac{dJ}{dZ^{[2]}}\frac{dZ^{[2]}}{dA^{[1]}}\frac{dA^{[1]}}{dZ^{[1]}}\frac{dZ^{[1]}}{db^{[1]}} \\
&= \left[A^{[2]}-Y\right]\left[W^{[2]}\right]\left[g'\left(Z^{[1]}\right)\right]\left[1\right] \\
&= dZ^{[2]}W^{[2]}g'\left(Z^{[1]}\right) \\
&= dZ^{[1]}
\end{aligned}
$$

- 즉, 다음 layer의 gradient 값들을 이용해 이전 layer의 gradient 값을 구할수 있다. (병렬연산 가능)

3-2. 역전파 구현

- **binary 분류용 L-layer Neural Net에 대한 역전파**
  - Initialize $W^{[1]}..W^{[L]}, b^{[1]}...b^{[L]}$
  - Set $A^{[0]} = X$ (Input) , $L =$ Total Layers
  - Loop epoch = 1 to max iteration
    - Forward Propagation
      - Loop $l = 1$ to $L - 1$
        - $Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$
        - $A^{[l]} = g\left(b^{[l]}\right)$
        - Save $A^{[l]}, W^{[l]}$ in memory for later use
      - $Z^{[L]} = W^{[L]} A^{[L-1]} + b^{[L]}$
      - $A^{[L]} = \sigma\left(Z^{[L]}\right)$
    - Cost $J = -\frac{1}{n}\left(Y log\left(A^{[2]}\right) - (1 - Y)\, log\left(1 - A^{[2]}\right)\right)$

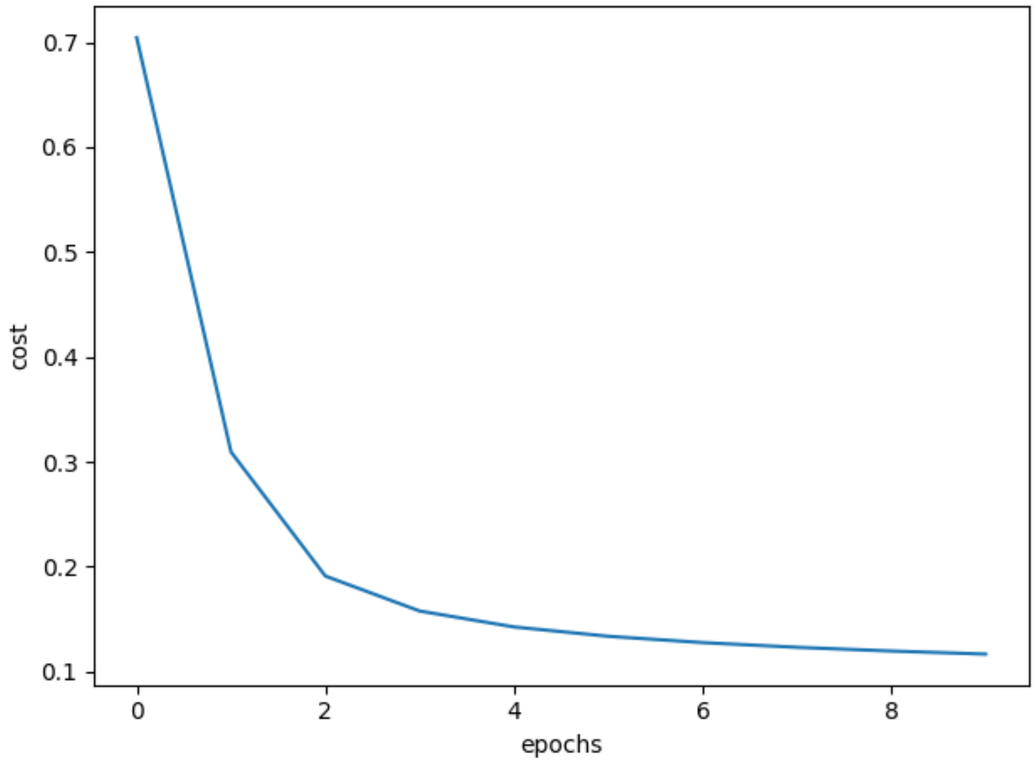
3-2. 역전파 구현

- **binary 분류용 L-layer Neural Net에 대한 역전파**
  - Backward Propagation
    - $dA^{[L]} = -\frac{Y}{A^{[L]}} + \frac{1-Y}{1-A^{[L]}}$
    - $dZ^{[L]} = dA^{[L]}\sigma'\left(dA^{[L]}\right)$
    - $dW^{[L]} = dZ^{[L]}dA^{[L-1]}$
    - $db^{[L]} = dZ^{[L]}$
    - $dA^{[L-1]} = dZ^{[L]}W^{[L]}$
    - Loop $l = L-1$ to 1
      - $dZ^{[l]} = dA^{[l]}g'\left(dA^{[l]}\right)$
      - $dW^{[l]} = dZ^{[l]}dA^{[l-1]}$
      - $db^{[l]} = dZ^{[l]}$
      - $dA^{[l-1]} = dZ^{[l]}W^{[l]}$
  - Update W and b
    - Loop $l = 1$ to L
      - $W^{[l]} = W^{[l]} - \alpha . dW^{[l]}$
      - $b^{[l]} = b^{[l]} - \alpha . db^{[l]}$

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

  MNIST의 5와 8을 이진분류로 1,000회 돌렸을 때의 손실함수



3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

```
def backward(self, X, Y, store):
        derivatives = {}
        store["A0"] = X.T
        A = store["A" + str(self.L)]
        dA = -np.divide(Y, A) + np.divide(1 - Y, 1 - A)
        dZ = dA * self.sigmoid_derivative(store["Z" + str(self.L)])
        dW = dZ.dot(store["A" + str(self.L - 1)].T) / self.n
        db = np.sum(dZ, axis=1, keepdims=True) / self.n
        dAPrev = store["W" + str(self.L)].T.dot(dZ)
        derivatives["dW" + str(self.L)] = dW
        derivatives["db" + str(self.L)] = db
            ...
        return derivatives
```

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

Forward Propagation – Layer 1:

$$
\begin{aligned}
X &= (11272, 784) \\
W^{[1]} &= (196, 784) \\
b^{[1]} &= (196, 1) \\
A^{[0]} &= X^T \\
&= (784, 11272) \\
Z^{[1]} &= W^{[1]} A^{[0]} + b^{[1]} \\
&= (196, 784) * (784, 11272) + (196, 1) \\
&= (196, 11272) + (196, 1) \\
&= (196, 11272) \\
A^{[1]} &= g\left(Z^{[1]}\right) \\
&= (196, 11272)
\end{aligned}
$$

MNIST 데이터셋의 Train set 6만개 중 5와 8만을 뽑아낸 것의 개수: 11,272

히든 레이어에 존재하는 노드의 수: 196

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

Forward Propagation - Layer 2:

$$
\begin{aligned}
W^{[2]} &= (1, 196) \\
b^{[2]} &= (1, 1) \\
Z^{[2]} &= W^{[2]} A^{[1]} + b^{[2]} \\
&= (1, 196) * (196, 11272) + (1, 1) \\
&= (1, 11272) + (1, 1) \\
&= (1, 11272) \\
A^{[2]} &= g\left(Z^{[2]}\right) \\
&= (1, 11272)
\end{aligned}
$$

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

Backward Propagation - Layer 2:

$$
\begin{aligned}
Y^T &= (1, 11272) \\
dA^{[2]} &= -\frac{Y^T}{A^{[2]}} + \frac{1 - Y^T}{1 - A^{[2]}} \\
&= (1, 11272) \\
dZ^{[2]} &= dA^{[2]} g'(Z^{[2]}) \\
&= (1, 11272) * (1, 11272) \\
&= (1, 11272) \\
dW^{[2]} &= dZ^{[2]} \left(A^{[1]}\right)^T \\
&= (1, 11272) * (11272, 196) \\
&= (1, 196) \\
db^{[2]} &= dZ^{[2]} \\
&= (1, 1) \\
dA^{[1]} &= \left(W^{[2]}\right)^T dZ^{[2]} \\
&= (196, 1) * (1, 11272) \\
&= (196, 11272)
\end{aligned}
$$

3-2. 역전파 구현

- **binary 분류용 2-layer Neural Net에 대한 역전파**

Backward Propagation - Layer 1:

$$
\begin{aligned}
dZ^{[1]} &= dA^{[1]} g'(Z^{[1]}) \\
&= (196, 11272) * (196, 11272) \\
&= (196, 11272) \\
dW^{[1]} &= dZ^{[1]} \left(A^{[0]} \right)^T \\
&= (196, 11272) * (11272, 784) \\
&= (196, 784) \\
db^{[1]} &= dZ^{[1]} \\
&= (196, 1)
\end{aligned}
$$

참고자료

1. 인공신경망
   https://untitledtblog.tistory.com/141

2. 학습 알고리즘
   http://shuuki4.github.io/deep%20learning/2016/05/20/Gradient-Descent-Algorithm-Overview.html

3. 역전파
   1. https://excelsior-cjh.tistory.com/171
   2. http://www.adeveloperdiary.com/data-science/machine-learning/understand-and-implement-the-backpropagation-algorithm-from-scratch-in-python/

/* elice */

문의 및 연락처

academy.elice.io

contact@elice.io

facebook.com/elice.io

medium.com/elice