# 인공지능_HW2

인공지능학부 215001 서가연

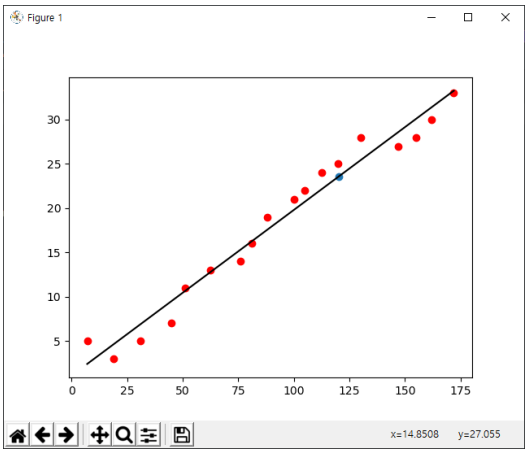
01. 사용자가 아파트 면적을 입력하면 아파트의 가격이 출력되는 시스템을 만들어보자. (선형회귀 사용)

소스 코드:

```python
# 아파트 면적을 입력하면 아파트 가격이 출력되는 시스템 : 머신러닝 > 선형 회귀 사용

# 필요한 라이브러리 불러오기
from sklearn import linear_model
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# 선형 회귀 모델 생성
lr = linear_model.LinearRegression()

# 아파트 면적과 가격 데이터
X = [[7], [19], [31], [45], [51], [62.5], [76], [81], [88], [100],
     [105], [112.5], [120], [130], [147], [155], [162], [172]]
y = [5, 3, 5, 7, 11, 13, 14, 16, 19, 21, 22, 24, 25, 28, 27, 28, 30, 33]

# 학습데이터로 모델 학습
lr.fit(X,y)

# 사용자의 입력값을 받아, 모델로 예측하기
x_new = float(input("아파트 면적을 입력하세요: "))
pred = lr.predict([[x_new]])

# 결과 화면 출력
print(f'아파트 가격은 {float(pred):0.2f}입니다')

# 예측 결과 시각화
plt.scatter(x_new, pred)
plt.scatter(X, y, color='red')
plt.plot(X, lr.coef_*X+lr.intercept_, color='black')
plt.show()
```

출력 결과:

```
(base) C:\Users\AI06\Desktop\AI\hw2>C:/Users/AI06/anaconda3/python.exe c:/Users/AI06/Desktop/AI/hw2/01.py
아파트 면적을 입력하세요: 120.3
아파트 가격은 23.60입니다
```

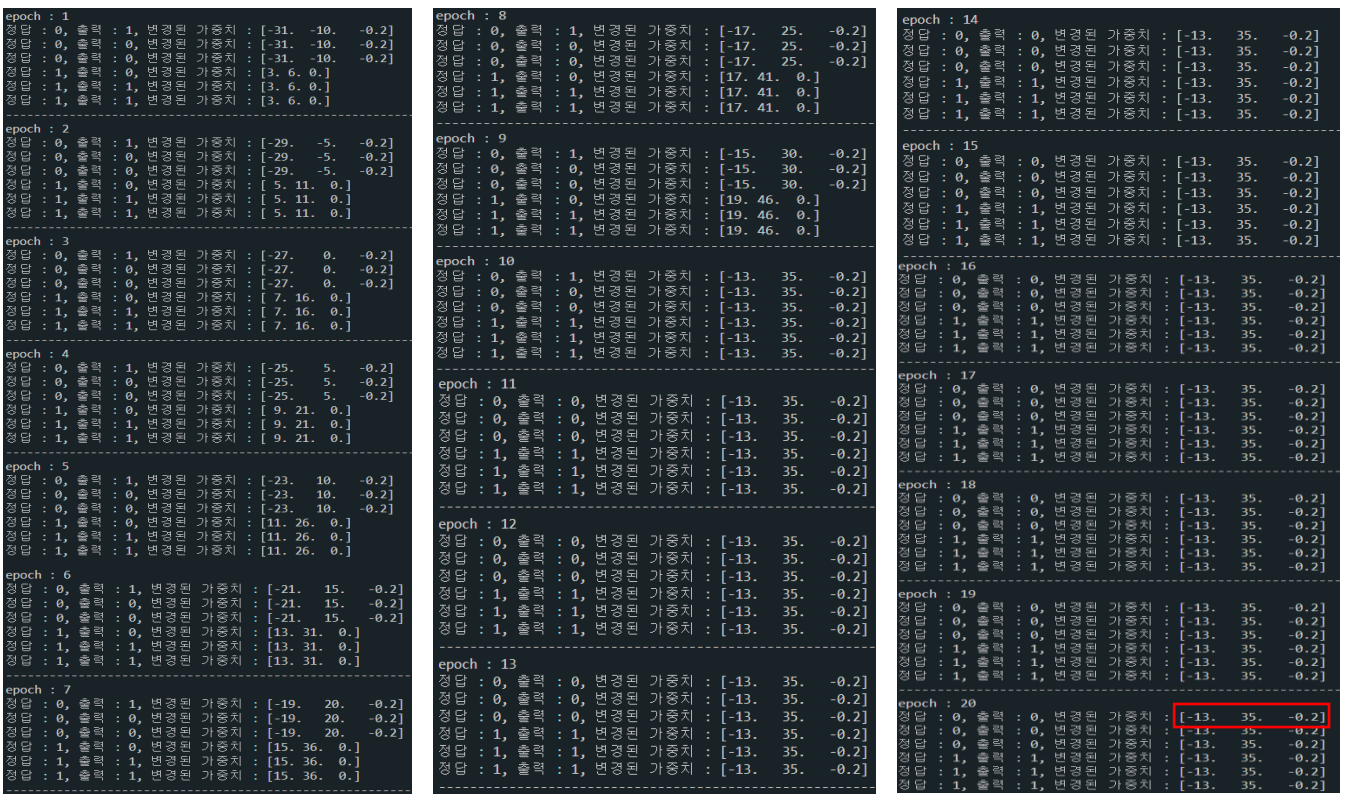


02. 대학생들의 신장과 체중을 받아서 성별을 출력하는 퍼셉트론을 만들어보자. 가중치와 바이어스가 어떻게 결정되는지 관찰해보자.

소스 코드:

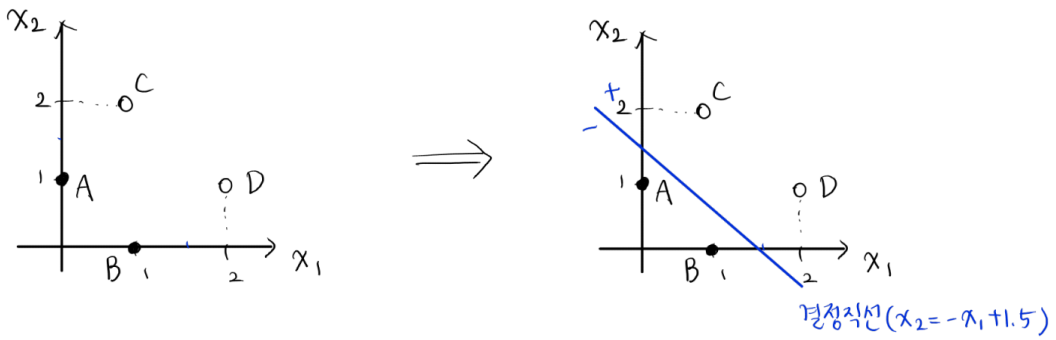
```python
# 신장과 체중데이터로 성별을 출력하는 퍼셉트론 => x1(학생의 신장) , x2(학생의 체중), y (성별_남/여)
from matplotlib import test
import numpy as np
import matplotlib.pyplot as plt

# 특징 데이터 : [x1, x2, x0] (x0은 바이어스 입력을 위함)
X = np.array([[160, 55, 1], [163, 43, 1], [165, 48, 1],   # -46  -35
              [170, 80, 1], [175, 76, 1], [180, 70, 1]])  # 43.8

y = np.array([0, 0, 0, 1, 1, 1])  # 정답 데이터 : 여자는 0, 남자는 1
W = np.array([1.0, 1.0, 0])  # 가중치는 [-1.0, 1.0], 바이어스 0으로 초기화

# 활성화 함수 : 계단함수 사용
def step_func(t):
    if t > 0: return 1
    else : return 0

# 퍼셉트론 학습 알고리즘
def perceptron_fit(X, y, epochs=20):
    global W
    eta = 0.2 # 학습률
    for i in range(1, epochs+1):
        print("-"*80)
        print("epoch :", i)
        for i in range(len(X)):
            predict = step_func(np.dot(X[i], W))
            error = y[i] - predict # 오차(예측값과 정답의 차이)
            W += eta*error*X[i] # 가중치 업데이트
            print(f'정답 : {y[i]}, 출력 : {predict}, 변경된 가중치 : {W}')

perceptron_fit(X, y)
```

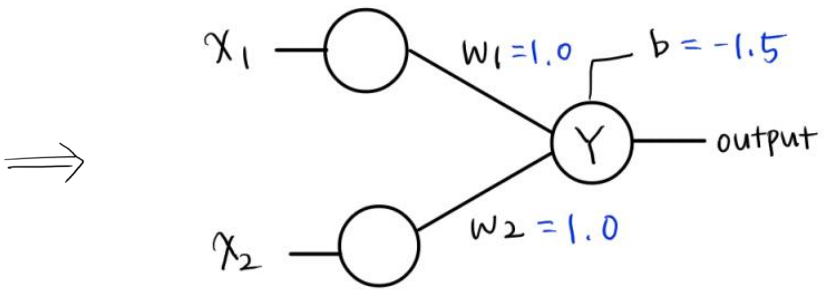
출력 결과:



03 - (a). 다음 표와 같이 입력데이터를 분류하는 퍼셉트론을 만들어보자. 가중치들과 바이어스 값을 수동으로 찾아보자.

| 훈련샘플 | X1 | X2 | 출력 |
|---|---|---|---|
| A | 0 | 1 | 0 |
| B | 1 | 0 | 0 |
| C | 1 | 2 | 1 |
| D | 2 | 1 | 1 |




$W = [w_1 \ \ w_2] = [1.0 \ \ 1.0]$, $b = -1.5$, $\tau(s) = \begin{cases} 0 & (s \le 0) \\ 1 & (s > 0) \end{cases}$

$z_y = x_1 + x_2 - 1.5$

$y = \tau(z_y)$

| $x_1$ | $x_2$ | $z_y$ | $y$ |
|---|---|---|---|
| 0 | 1 | -0.5 | 0 |
| 1 | 0 | -0.5 | 0 |
| 1 | 2 | 1.5 | 1 |
| 2 | 1 | 1.5 | 1 |

주어진 훈련샘플은 선형분리가 가능하기 때문에 퍼셉트론을 사용해 분리할 수 있다.

가중치 $W = [1.0 \ \ 1.0]$, $b = -1.5$인 퍼셉트론으로 해결가능하다.

03 - (b). 위의 샘플을 사용하여, 퍼셉트론 학습 알고리즘 진행 과정을 보여라. 학습률은 1.0, 가중치와 바이어스의 초기값은 w1=0, w2=0, b=-1.0이다. (각 단계에서 가중치와 바이어스 표시하기)

소스 코드:

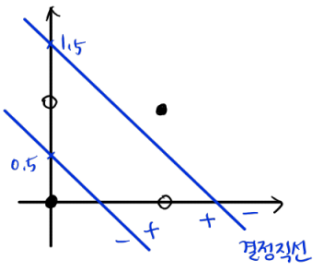
03.py > ...

```python
import numpy as np

# 특징 데이터 : [x1, x2, x0]
X = np.array([[0, 1, 1],
              [1, 0, 1],
              [1, 2, 1],
              [2, 1, 1]])
y = np.array([0, 0, 1, 1])  # 정답 데이터
W = np.array([0, 0, -1.0])  # 가중치

# 활성화 함수 : 계단함수 사용
def step_func(t):
    if t > 0: return 1
    else : return 0

# 퍼셉트론 학습 알고리즘
def perceptron_fit(X, y, epochs=5):
    global W
    eta = 1.0 # 학습률

    for i in range(1, epochs+1):
        print("-"*60)
        print("epoch :", i)
        for i in range(len(X)):
            predict = step_func(np.dot(X[i], W))
            error = y[i] - predict # 오차(예측값과 정답의 차이)
            W += eta*error*X[i] # 가중치 업데이트
            print(f'입력 : {X[i]}, 정답 : {y[i]}, 출력 : {predict}, 변경된 가중치와 바이어스 : {W}')


perceptron_fit(X, y)
```

출력 결과:

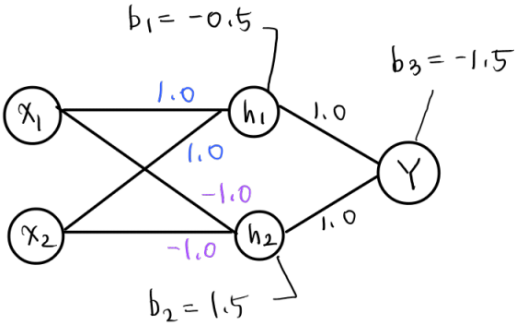
```
(base) C:\Users\AI06\Desktop\AI\hw2>C:/Users/AI06/anaconda3/python.exe c:/Users/AI06/Desktop/AI/hw2/03.py
------------------------------------------------------------
epoch : 1
입력 : [0 1 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 0.  0. -1.]
입력 : [1 0 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 0.  0. -1.]
입력 : [1 2 1], 정답 : 1, 출력 : 0, 변경된 가중치와 바이어스 : [1. 2. 0.]
입력 : [2 1 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [1. 2. 0.]
------------------------------------------------------------
epoch : 2
입력 : [0 1 1], 정답 : 0, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 0 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 2 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [2 1 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
------------------------------------------------------------
epoch : 3
입력 : [0 1 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 0 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 2 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [2 1 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
------------------------------------------------------------
epoch : 4
입력 : [0 1 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 0 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 2 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [2 1 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
------------------------------------------------------------
epoch : 5
입력 : [0 1 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 0 1], 정답 : 0, 출력 : 0, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [1 2 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
입력 : [2 1 1], 정답 : 1, 출력 : 1, 변경된 가중치와 바이어스 : [ 1.  1. -1.]
```

04. XOR을 처리할 수 있는, 은닉층을 가지는 신경망의 가중치와 바이어스값을 제시하라.




XOR 문제

⇒ 멀티퍼셉트론




i) $z_1 = x_1 + x_2 - 0.5$
$z_2 = -x_1 - x_2 + 1.5$
$\Leftrightarrow [z_1\ z_2] = [x_1\ x_2] \begin{bmatrix} 1.0 & -1.0 \\ 1.0 & -1.0 \end{bmatrix} + [-0.5\ \ 1.5]$

ii) $\tau(s) = \begin{cases} 1 & (s \geq 0) \\ 0 & (s < 0) \end{cases} \Rightarrow a_1 = \tau(z_1),\ a_2 = \tau(z_2)$

iii) $z_y = a_1 + a_2 - 1.5 \Leftrightarrow [a_1\ a_2] \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix} + [-1.5]$

iv) $y = \tau(z_y)$

⇒

| $x_1$ | $x_2$ | $z_1$ | $z_2$ | $a_1$ | $a_2$ | $z_y$ | $y$ |
|---|---|---|---|---|---|---|---|
| 0 | 0 | -0.5 | 1.5 | 0 | 1 | -0.5 | 0 |
| 0 | 1 | 0.5 | 0.5 | 1 | 1 | 0.5 | 1 |
| 1 | 0 | 0.5 | 0.5 | 1 | 1 | 0.5 | 1 |
| 1 | 1 | 0.5 | -0.5 | 1 | 0 | -0.5 | 0 |

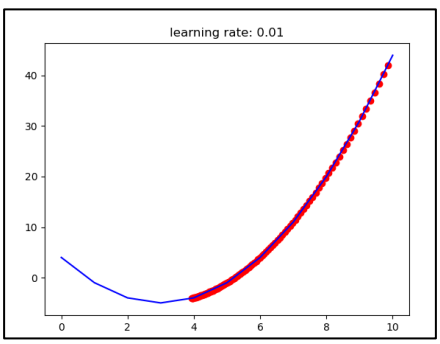
답: $W_1 = \begin{bmatrix} 1.0 & -1.0 \\ 1.0 & -1.0 \end{bmatrix}$, $W_2 = \begin{bmatrix} 1.0 \\ 1.0 \end{bmatrix}$, $B = [-0.5\ \ 1.5\ \ -1.5]$

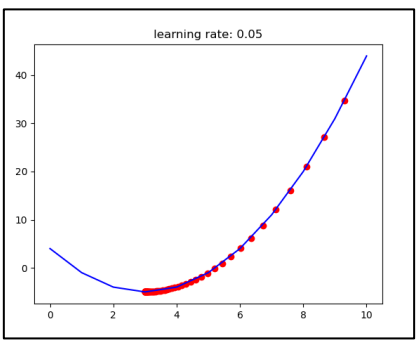
05. 오차함수가 $x^2 - 6x + 4$ 일 때, 경사하강법을 사용하여 오차함수의 최저값을 계산하는 절차를 그래프로 시각화하라.

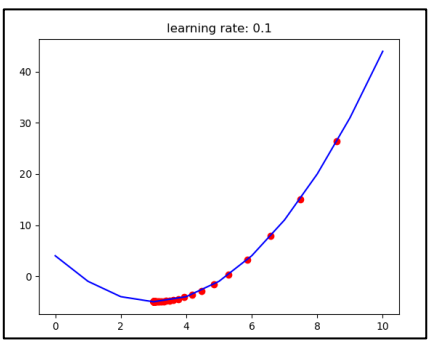
소스 코드:

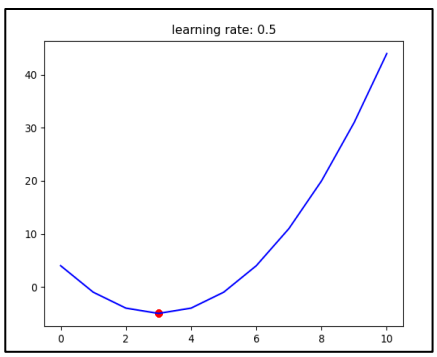
```python
# 05.py > ...
import matplotlib.pyplot as plt
import numpy as np

eta = [0.01, 0.05, 0.1, 0.5, 1.0]  # 학습률
max_iterations = 100  # 반복횟수

# 오차함수와 1차 도함수
def cost_func(x): return x*x-6*x+4
def gradient(x): return 2*x - 6

point = []

# 가중치 업데이트
for t in eta:
    x = 10
    for i in range(max_iterations):
        x = x - t*gradient(x)
        point.append(x)  # 변경된 가중치 저장

arr = np.array(point)

# 그래프 시각화
for i in range(1, len(eta) + 1):
    plt.plot(cost_func(np.arange(11)), 'b')
    plt.title(f'learning rate: {eta[i-1]}')

    plt.scatter(arr[(i-1)*max_iterations:i*max_iterations],
                cost_func(arr[(i-1)*max_iterations:i*max_iterations]), color='r')
    plt.show()
```

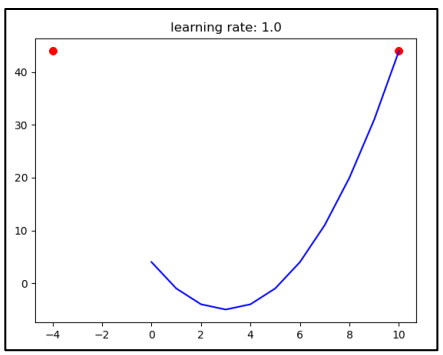
출력 결과:



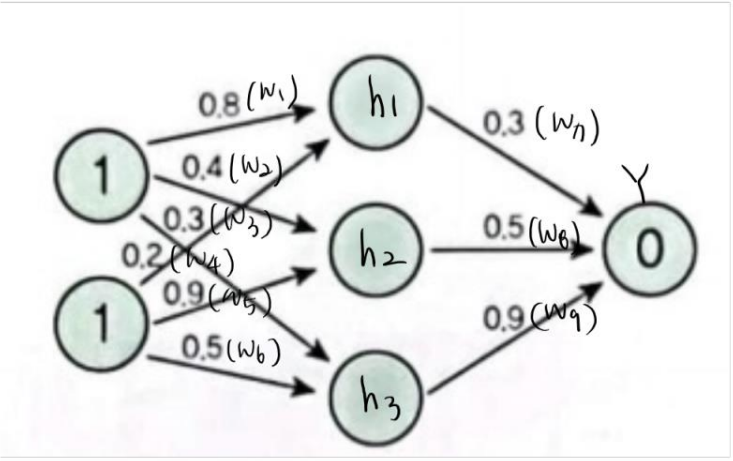



06-(a). 그림과 같은 MLP에서 각 뉴론의 출력값과 오차를 계산해보자. 활성화함수는 ReLU 함수라고 가정하며, 순방향 패스와 오차값만 계산해보자. (바이어스는 무시)




각 뉴런을 $h_1, h_2, h_3, Y$라 하고 뉴런의 가중합을 $(z_1, z_2, z_3, z_y)$, 출력값을 $(a_1, a_2, a_3, a_y)$라고 하자.

가중치 $(w_1 \sim w_7)$는 그림의 표기와 같으며, 활성함수는 $\tau(s) = \max(s, 0)$를 사용한다.

① 순방향 패스

$z_1 = 1 \times 0.8 + 1 \times 0.2 = 1.0$, $a_1 = \tau(z_1) = 1$

$z_2 = 1 \times 0.4 + 1 \times 0.9 = 1.3$, $a_2 = \tau(z_2) = 1.3$

$z_3 = 1 \times 0.3 + 1 \times 0.5 = 0.8$, $a_3 = \tau(z_3) = 0.8$

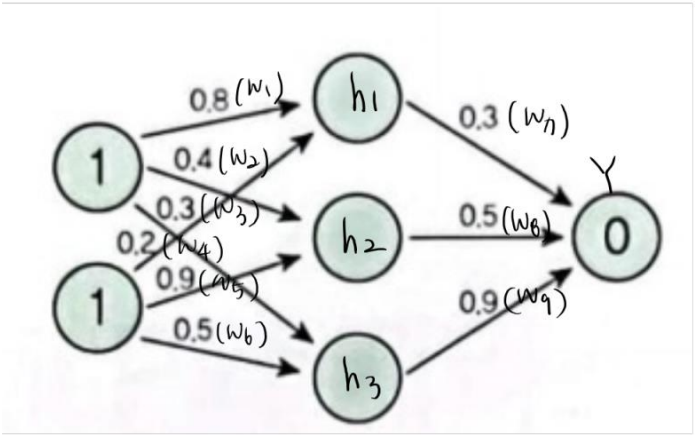
$z_y = 1.0 \times 0.3 + 1.3 \times 0.5 + 0.8 \times 0.9 = 1.67$, $a_y = \tau(z_y) = 1.67$

② 오차값 (손실함수로 MSE 사용)

$$E(w) = \frac{1}{2} \sum_i (y_i - t_i)^2$$

$$= \frac{1}{2}(y - a_y)^2 = \frac{1}{2}(0 - 1.67)^2 = 1.39445$$

06-(b). 출력노드(o)와 연결된 가중치 초기값 0.3에 대하여 오차가 역전파 되어 학습률(0.2)를 가지고 가중치를 업데이트하는 전체 과정을 한 단계만 계산하시오.




06-(a)과 동일한 표기를 사용

| 뉴런 | $h_1$ | $h_2$ | $h_3$ | Y |
|---|---|---|---|---|
| 가중합 | $z_1$ | $z_2$ | $z_3$ | $z_y$ |
| 출력값 | $a_1$ | $a_2$ | $a_3$ | $a_y$ |

$$\tau'(s) = \begin{cases} 0 & (s<0) \\ 1 & (s>0) \\ \text{불가능} & (s=0) \end{cases}$$

① gradient 구하기

i) $\frac{\partial E}{\partial a_y} = 2 \times \frac{1}{2}(y - a_y) \times (-1) = (a_y - y) = 1.67$

ii) $\frac{\partial a_y}{\partial z_y} = \tau'(z_y) = 1 \quad (\because z_y > 0)$

iii) $z_y = w_7 \times a_1 + w_8 \times a_2 + w_9 \times a_3$

$\frac{\partial z_y}{\partial w_7} = a_1 = 1$

i), ii), iii) 에 의해 $\frac{\partial E}{\partial w_7} = \frac{\partial E}{\partial a_y} \frac{\partial a_y}{\partial z_y} \frac{\partial z_y}{\partial w_7} = 1.67$

② 가중치 업데이트

$w_7 = w_7 - \eta \times \frac{\partial E}{\partial w_7} = 0.3 - 0.2 \times 1.67 = -0.034$