

3층 신경망

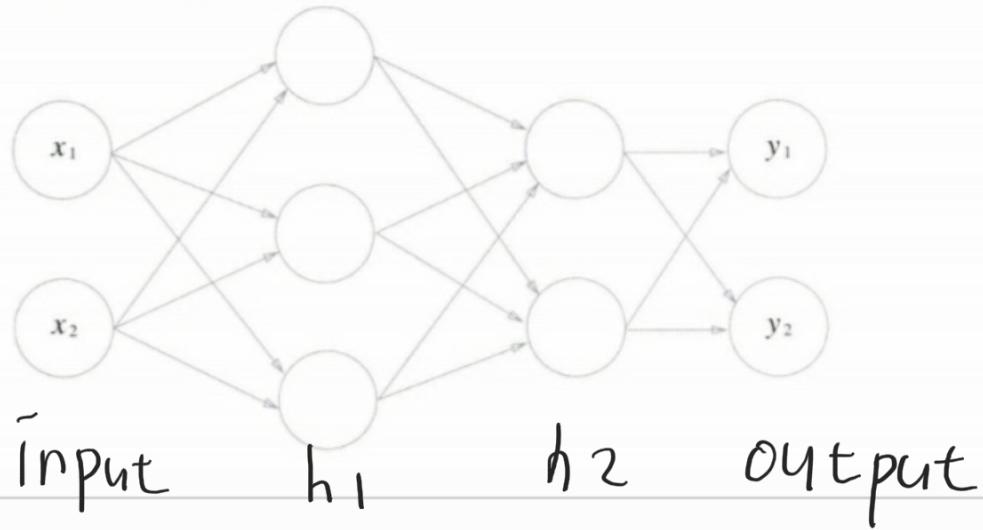
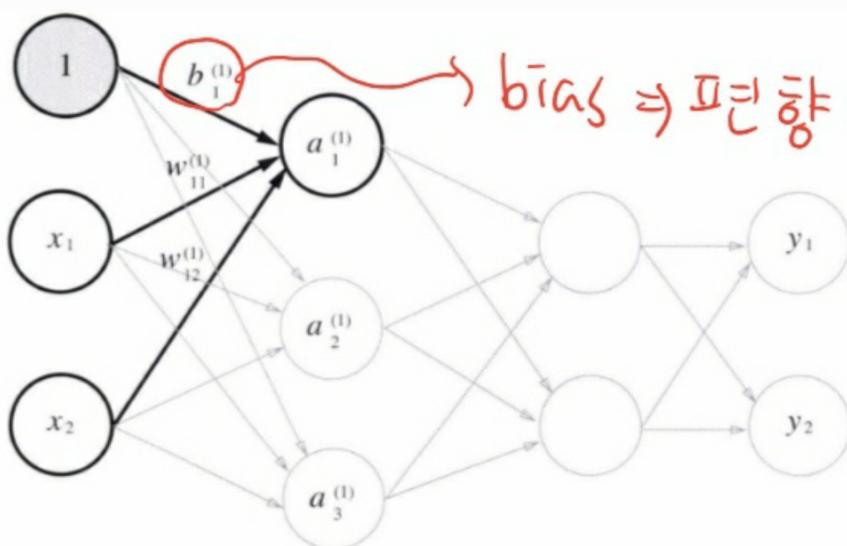
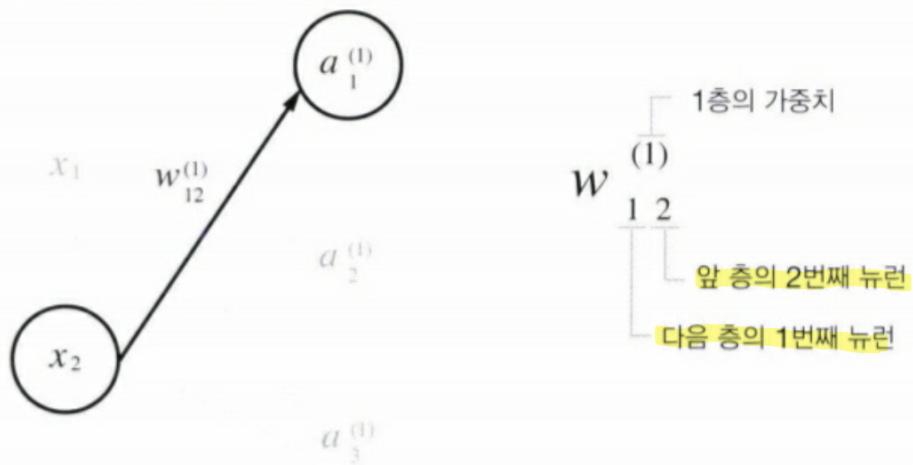


그림 3-16 중요한 표기



$$q_1^{(1)} = w_{11}^{(1)}x_1 + w_{12}^{(1)}x_2 + b_1^{(1)}$$

↓

$$A^{(1)} = XW^{(1)} + B^{(1)}$$

$$A^{(1)} = (a_1^{(1)} \ a_2^{(1)} \ a_3^{(1)}) , X = (x_1 \ x_2) , B^{(1)} = (b_1^{(1)} \ b_2^{(1)} \ b_3^{(1)})$$

$$W^{(1)} = \begin{pmatrix} W_{11}^{(1)} & W_{21}^{(1)} & W_{31}^{(1)} \\ W_{12}^{(1)} & W_{22}^{(1)} & W_{32}^{(1)} \end{pmatrix}$$

그림 3-18 입력층에서 1층으로의 신호 전달

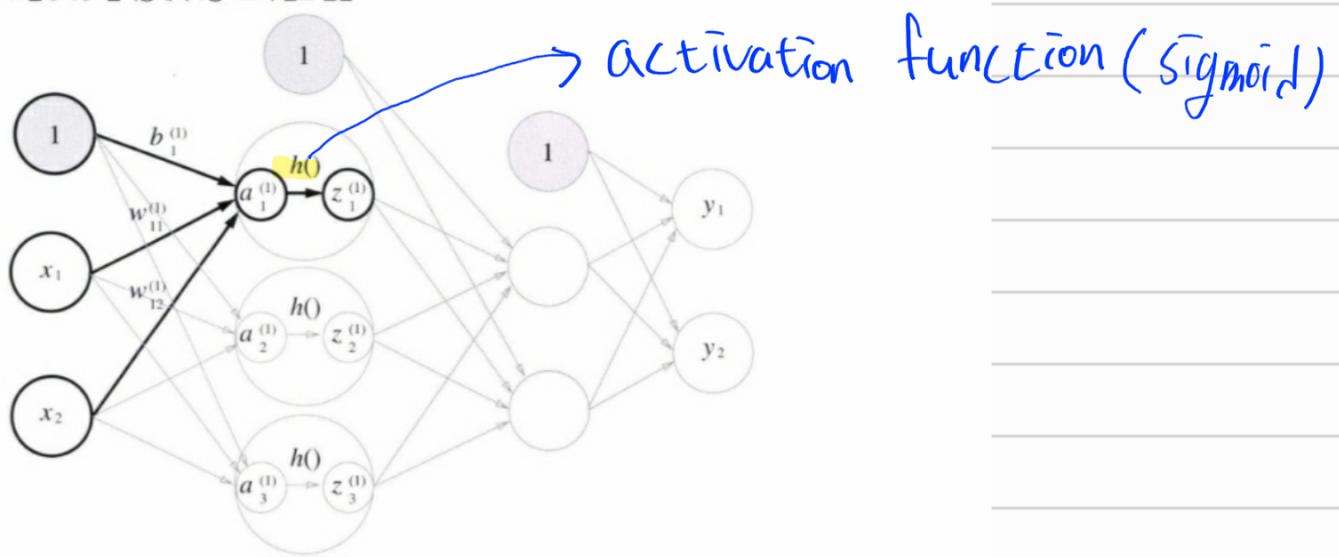


그림 3-19 1층에서 2층으로의 신호 전달

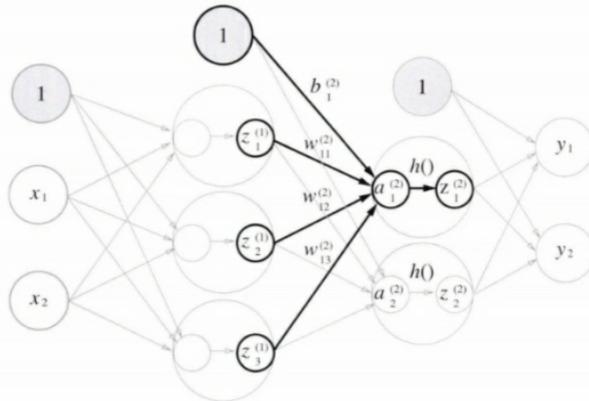
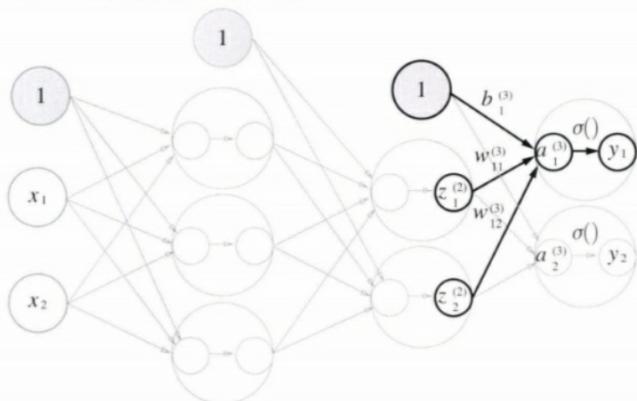


그림 3-20 2층에서 출력층으로의 신호 전달



출력층의 $f(\cdot)$ \Rightarrow activation function

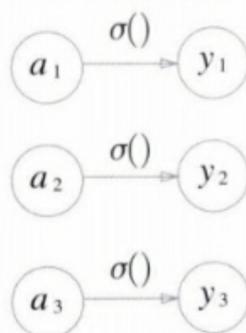
출력층 설계

분류 \Rightarrow Softmax function

회귀 \Rightarrow Identity function

항등함수 : 입력과 출력이 같다.

그림 3-21 항등 함수



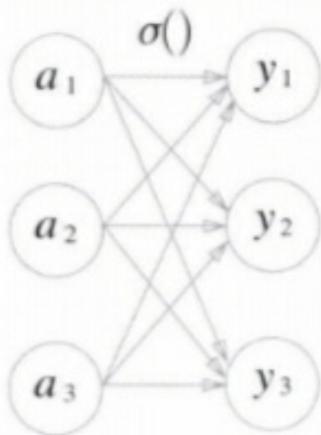
소프트맥스 함수

$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^n \exp(a_i)}$$

$n =$ 출력 뉴런의 수
 $a_k =$ 입력 신호

↳ 출력 층의 각 뉴런이 모든 입력 신호에 영향 받음

그림 3-22 소프트맥스 함수



* 소프트 맥스 함수 구현시 주의점

↳ Overflow!! ↳ 표현할 수 있는 수의 범위가 한정

$$\begin{aligned}
 y_k &= \frac{\exp(a_k)}{\sum_{i=1}^n \exp(a_i)} = \frac{C \exp(a_k)}{C \sum_{i=1}^n \exp(a_i)} \\
 &= \frac{\exp(a_k + \log C)}{\sum_{i=1}^n \exp(a_i + \log C)} \\
 &= \frac{\exp(a_k + C')}{\sum_{i=1}^n \exp(a_i + C')}
 \end{aligned}$$

← 개선한 수식

[식 3.11]

소프트맥스 함수 특징

Softmax output range

\Rightarrow 0\sim 1 \Rightarrow 확률로 해석 가능

```
>>> a = np.array([0.3, 2.9, 4.0])
>>> y = softmax(a)
>>> print(y)
[ 0.01821127  0.24519181  0.73659691]
1) _____ 2) _____ 3) _____
>>> np.sum(y) 1.0
```

①: $y[0]$ 을 확률, ②: $y[1]$ 을 확률로 만든다.

∴ 2번 클래스의 확률이 가장 높다

이때 어떻게 1, 2, 3 번 클래스의 순서가
유지가 되는가??



$$\hookrightarrow \because y = \exp(x) \Rightarrow \text{단조 증가함수}$$
$$a < b \rightarrow f(a) < f(b)$$

신경망으로 분류할 때 softmax 가능

train 예선 output 층에서 softmax 사용

But, inference 예선 사용 X

정규화 : 0~255를 0~1 사이로 변경(데이터를 특정 범위로 변환)

전처리 : 신경망의 입력 데이터에 특정 변환을 가하는 것

백색화 : 전체 데이터를 균일하게 분포 시키는 것

정규화 ⊂ 전처리

- 신경망 각 층의 배열 형상
 X W_1 W_2 W_3 $\rightarrow Y$

Shape 784 784x50 50x100 100x10 10

- 배치 처리를 위한 배열 형상 추가

X W_1 W_2 W_3 $\rightarrow Y$

Shape 100x784 784x50 50x100 100x10 100x10

↳ Batch

배치 : 입력데이터를 하나로 묶음(위의 예시에선
 $batch_size = 100$) -> 이미지가 지폐처럼 묶여있다.

배치를 사용하는 이유

- 수치 계산 라이브러리가 큰 배열을 효율적으로 처리할 수 있도록 고도로 최적화 되어 있음
- 커다란 신경망에선 데이터 전송이 병목으로 작용하는 경우가 있는데 배치 처리를 함으로써 버스에 주는 부하를 줄인다.