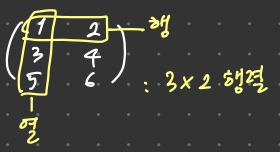
Ch1. 신경망 복습

1. 수학과 파이썬 복습

- 벡터 : 크기와 방향은 가진 양 (1차원 배열) ex) $\begin{pmatrix}1\\2\\3\end{pmatrix}$: 열 벡터 (1 2 3) : 행 벡터
- 행렬 : 숫자가 2차원 형태로 늘어선 것 ex)



: 3 x 2 행렬

  - ↳ 파이썬 라이브러리 Numpy 사용
    : np.array() 메서드로 생성

- 벡터의 내적

$$x \cdot y = x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

$y = (y_1, \cdots, y_n)$, $x = (x_1, \cdots, x_n)$

"두 벡터가 얼마나 같은 방향을 향하고 있는가"

- 행렬의 곱셈 "왼쪽 행렬의 행벡터와 오른쪽 행렬의 열벡터의 내적"

ex) $\begin{pmatrix}1 & 2\\3 & 4\end{pmatrix}\begin{pmatrix}5 & 6\\7 & 8\end{pmatrix} = \begin{pmatrix}19 & 22\\43 & 50\end{pmatrix}$

A, B

1·5 + 2·7 = 19

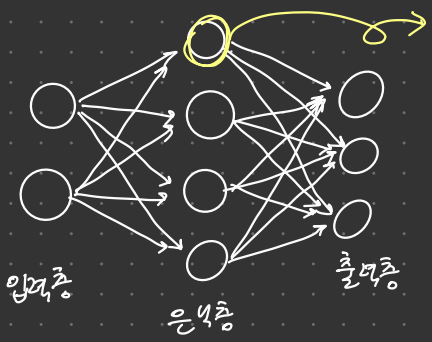
3·5 + 4·7 = 43

→ np.dot(), np.matmul() 등

★ 행렬 형상 확인 : 행렬의 곱에서는 대응하는 차원의 원소 수를 일치시킨다.

	A		B	=	C
shape:	3 × (2)	=	(2) × 4		3 × 4

2. 신경망의 추론



$h_1 = x_1 w_1 + x_2 w_2 + b_1$ (가중치, 편향)

$\therefore h = x W + b$ (입력, 가중치, 편향)

은닉층의 뉴런

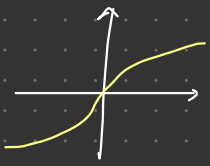
ex) N개의 샘플 데이터 미니배치 :

x	W	$=$	h
$N \times 2$	2×4		$N \times 4$

* 완전 연결 계층 ⇒ '선형' 변환 ⌟ Affine 계층

* 활성화 함수 ⇒ '비선형' 효과 부여

ex) sigmoid function : $\sigma(x) = \frac{1}{1 + \exp(-x)}$



$\therefore a = \text{sigmoid}(h)$ # activation

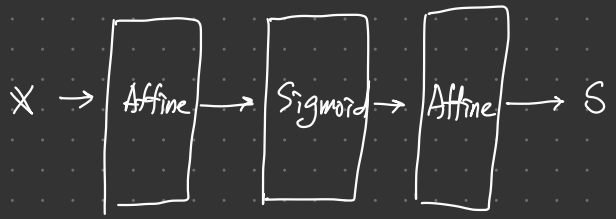
a	W_2	$=$	Score	→ softmax 함수에 입력하면 확률을 얻을 수 있다.
$N \times 4$	4×3		$N \times 3$	

Score : '확률'이 되기 전 값

3 : 3개의 각 클래스에 대응하는 점수 (for 분류)

⌟ Sigmoid 계층

// 여기까지 신경망 계층 구성



3. 신경망의 학습 : 최적의 매개변수를 찾는 작업.

· 손실함수 (Loss function) : 신경망의 성능을 나타내는 척도.

Softmax function (k번째 출력) : $y_k = \frac{\exp(s_k)}{\sum_{i=1}^{n} \exp(s_i)}$ → '확률'로 해석 (0 ~ 1)

↳ Cross Entropy Error : $L = -\sum_k t_k \log y_k$ * 정답 레이블 t는 원-핫 벡터로 표시

→ 미니배치 고려 (N개 데이터) : $L = -\frac{1}{N}\sum_n \sum_k t_{nk} \log y_{nk}$ * N으로 나눈 1개당 평균 손실 함수