4. 얕은 신경망 네트워크

날짜	@September 23, 2023

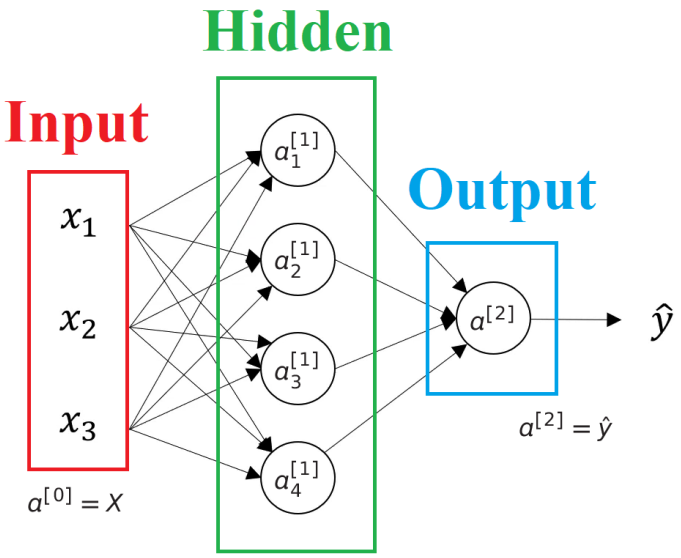
▼ 목차

신경망 네트워크의 구성

1️⃣ Neural Network Representation

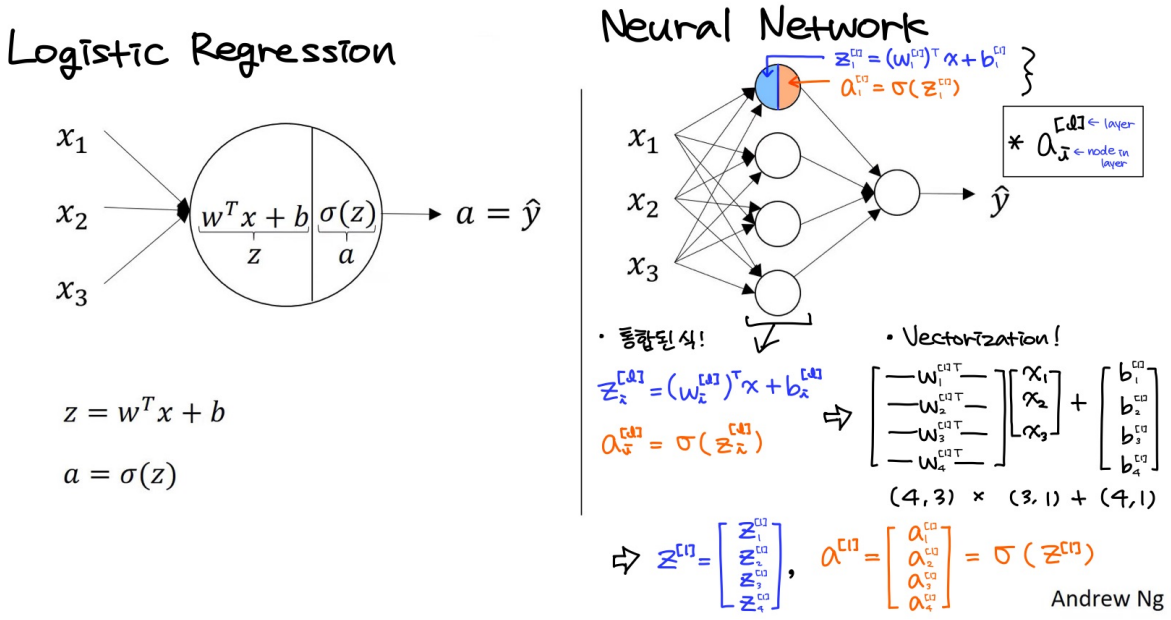
🔹 아래는 은닉층이 1개인 **2층 신경망** (관례적으로 입력층은 포함시키지 않음)

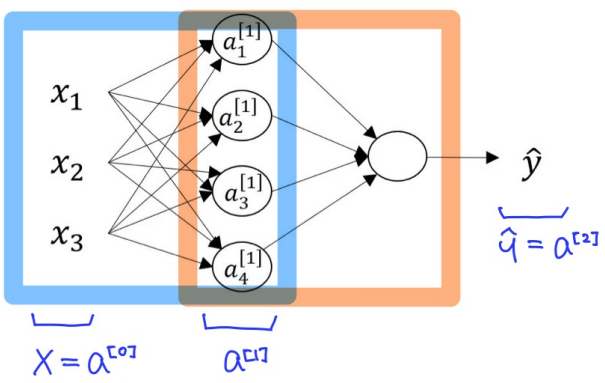
- **입력층**: 입력 feature x_1, x_2, x_3가 세로로 쌓여 있으며, 입력값 X를 은닉층으로 전달
- **은닉층**: 이 층에서 계산되는 값은 training set(입력값 X, 출력값 y만 있음)에 기록되어 있지 않음
- **출력층**: 예측값 $\hat{y}$를 계산함



- a는 활성값으로, 신경망의 층들이 다음 층으로 전달해주는 값
  - 각 층이 다음 층으로 전달하는 값을 각각 $a^{[0]}$, $a^{[1]}$, $a^{[2]}$로 다르게 표기할 수 있음
  - 은닉층의 노드가 4개이므로 $a^{[2]}$는 4차원 벡터가 됨
- 입력층을 제외한 각 층은 매개변수 $w^{[i]}$와 $b^{[i]}$에 관련되어 있음
  - $w^{[1]}$는 (4, 3) 행렬, $b^{[1]}$는 (4, 1) 벡터 (은닉 노드 4개, 전달되는 값 3개)
  - $w^{[2]}$는 (1, 4) 행렬, $b^{[2]}$는 (1, 1) 벡터 (출력 노드 1개, 전달되는 값 4개)

신경망 네트워크 출력의 계산





Given input x:

- $z^{[1]} = W^{[1]} a^{[0]} + b^{[1]}$ (4,1) (4,3) (3,1) (4,1)
- $a^{[1]} = \sigma(z^{[1]})$ (4,1) (4,1)
- $z^{[2]} = W^{[2]} a^{[1]} + b^{[2]}$ (1,1) (1,4) (4,1) (1,1)
- $a^{[2]} = \sigma(z^{[2]})$ (1,1) (1,1)

Andrew Ng

많은 샘플에 대한 벡터화

Non-vectorized pseudo code (2층 신경망)

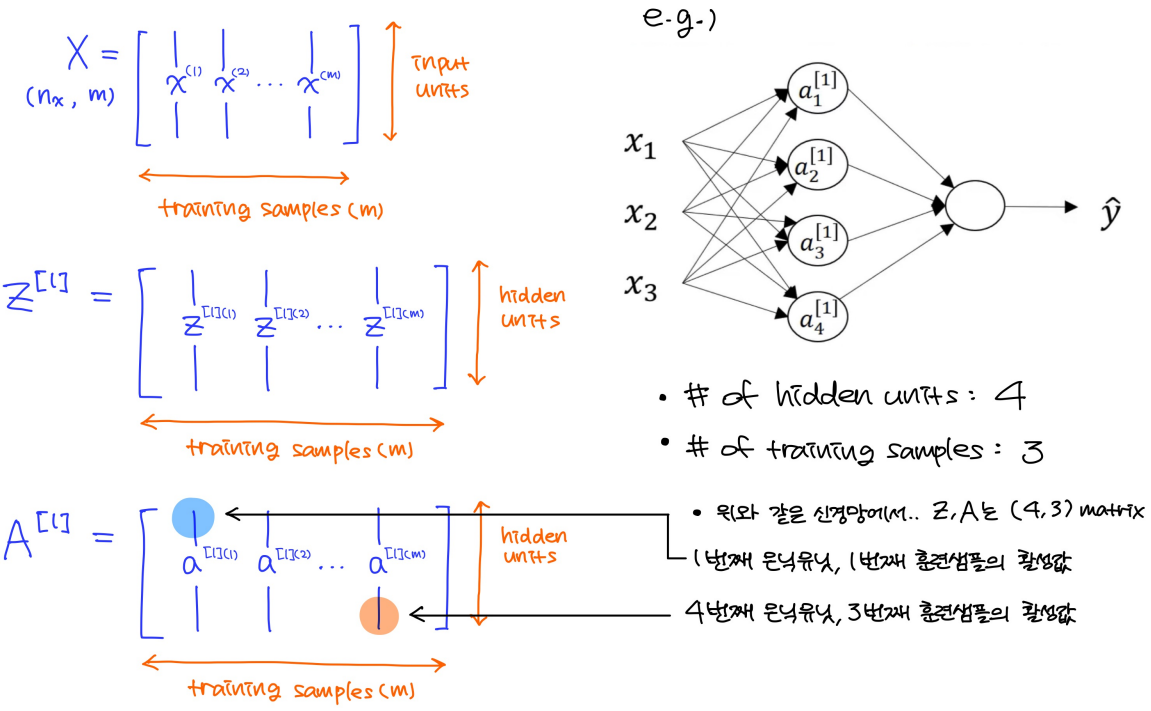
for i = 1 to m, (i-th training sample)

$$z^{[1](i)} = w^{[1]}x^{(i)} + b^{[1]}$$

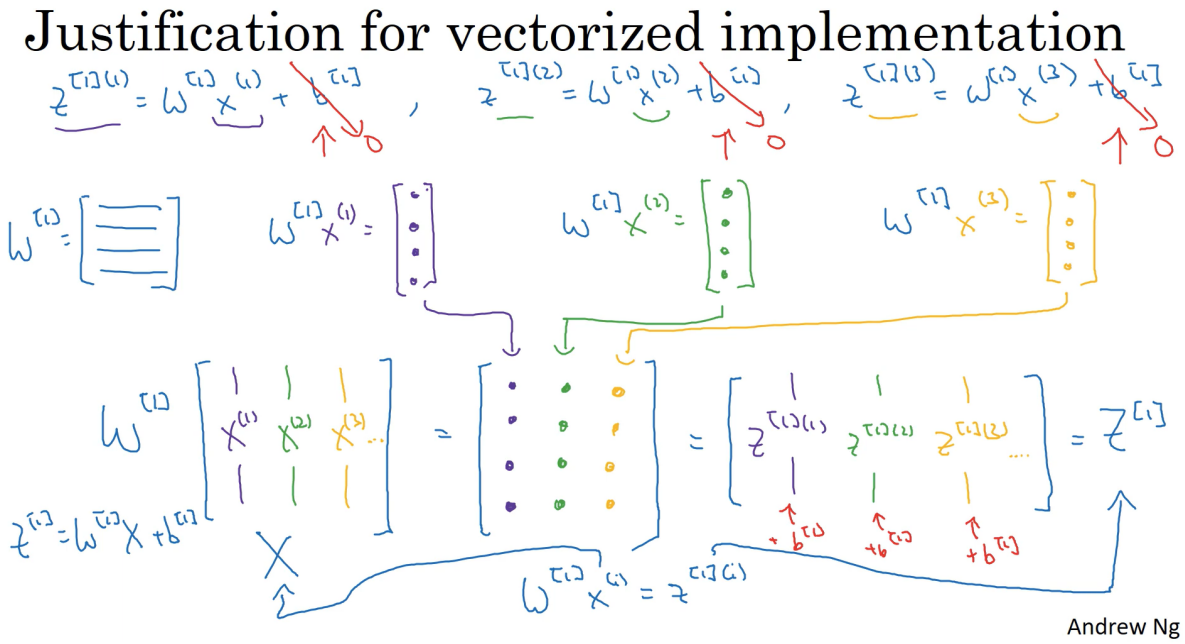
$$a^{[1](i)} = \sigma(z^{[1](i)})$$

$$z^{[2](i)} = w^{[2]}a^{[1](i)} + b^{[2]}$$

$$a^{[2](i)} = \sigma(z^{[2](i)})$$



벡터화 구현에 대한 설명

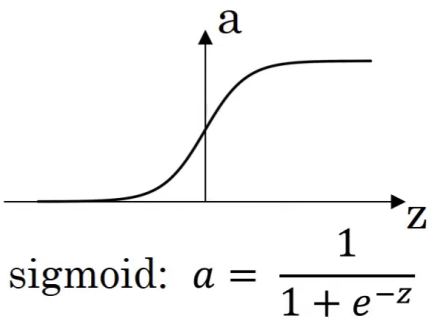


👾 **Vectorized pseudo code (2층 신경망)**

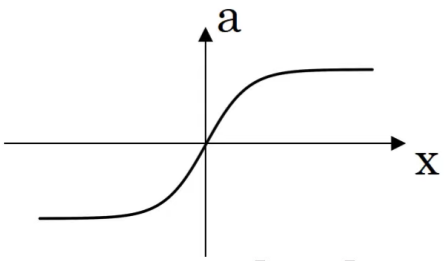
$$
\begin{aligned}
Z^{[1]} &= W^{[1]}X + b^{[1]} \\
A^{[1]} &= \sigma(Z^{[1]}) \\
Z^{[2]} &= W^{[2]}A^{[1]} + b^{[2]} \\
A^{[2]} &= \sigma(Z^{[2]})
\end{aligned}
$$

활성화 함수와 그 미분

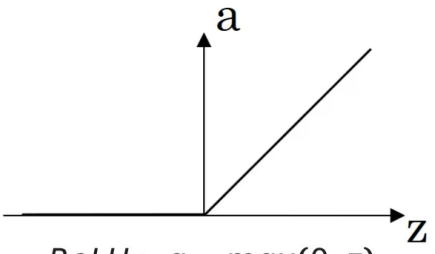
1️⃣ Activation Functions



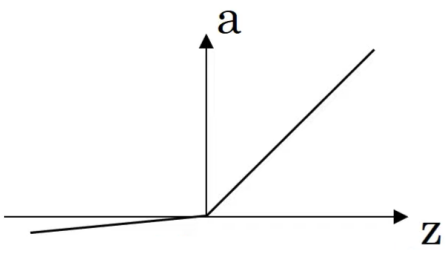
$$\text{sigmoid: } a = \frac{1}{1+e^{-z}}$$



$$\tanh: \ a = \frac{e^{z}-e^{-z}}{e^{z}+e^{-z}}$$



$$ReLU: \ a = \max(0, z)$$



$$leaky\ ReLU: \ a = \max(0.01z, z)$$

tanh

- 은닉층에서 시그모이드보다 대체로 좋음
  - 값이 1과 -1 사이로 평균값이 0에 더 가깝기 때문
  - 데이터의 중심을 0.5 대신 0으로 만들어, 다음 층의 학습을 더 쉽게 함 (최적화 알고리즘 관련 내용!)
- 이진 분류 시 출력층에서는 시그모이드를 쓰는 것이 더 좋음
  - y가 0 또는 1인 경우 출력층에서는 0과 1 사이의 값이 출력되는 것이 좋음
  - 각 층별로 다른 활성화 함수를 사용할 수 있음
- 시그모이드와 tanh 단점
  - z가 너무 크거나 작으면 도함수(기울기)가 굉장히 작아져 경사 하강법이 느려짐

ReLU

- 은닉층에서 시그모이드, tanh보다 주로 쓰임
- z가 0일 때의 도함수가 정의되지 않았지만(미분 불가능) 컴퓨터 구현 시 z가 정확히 0이 될 확률이 굉장히 낮아 문제되지 않음

- ReLU 단점
  - z가 음수일 때 도함수가 0임

leaky ReLU

- ReLU보다 대체로 결과가 좋지만 실제로 많이 쓰이지는 않음
- z가 음수일 때의 계수를 학습 알고리즘의 변수로 넣어 최적화할 수도 있음
- (leaky) ReLU의 장점
  - 대부분의 z에 대해 기울기가 0과 매우 달라 신경망의 학습이 빠름
  - z의 절반(음수)의 기울기가 0이기는 하지만 실제로 충분한 은닉 유닛의 z는 0보다 크므로 잘 동작함*

왜 비선형 활성화 함수를 써야 할까요?

선형 활성화 함수 (또는 입력값을 출력값으로 내보내는 항등함수) $g(z) = z$

- 이 경우 신경망은 입력의 선형식만을 출력하게 되며, 선형 은닉층을 쓴다는 것은 신경망이 아무리 깊어도 은닉층이 없는 것이나 다름없어짐
  - 여러 개의 선형 함수를 조합하면 결국 하나의 선형 함수가 되기 때문
- 선형 활성화 함수를 쓰는 경우가 간혹 존재하며, 주로 출력층에 이용함
  - y가 실수 값인 회귀 문제에서는 출력층에 선형 활성화 함수를 써도 괜찮음
    - 출력값인 $\hat{y}$이 $-\infty$부터 ∞까지의 실수 값이 되도록 해야 하기 때문이며, 은닉 유닛에는 앞서 언급한 비선형 활성화 함수를 써야 할 것임

신경망 네트워크와 경사 하강법

앞서 제시한 2층 신경망에 경사 하강법을 적용한 것을 살펴보자.

1️⃣ Forward Propagation

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

2️⃣ Back Propagation

$$
\begin{aligned}
dz^{[2]} &= a^{[2]} - y \\
dW^{[2]} &= dz^{[2]} a^{[1]T} \\
db^{[2]} &= dz^{[2]} \\
dz^{[1]} &= W^{[2]T} dz^{[2]} * g^{[1]\prime}(z^{[1]}) \\
dW^{[1]} &= dz^{[1]} x^T \\
db^{[1]} &= dz^{[1]}
\end{aligned}
$$

$$
\begin{aligned}
dZ^{[2]} &= A^{[2]} - Y \\
dW^{[2]} &= \frac{1}{m} dZ^{[2]} A^{[1]T} \\
db^{[2]} &= \frac{1}{m} np.sum(dZ^{[2]},\ axis = 1,\ keepdims = True) \\
dZ^{[1]} &= W^{[2]T} dZ^{[2]} * g^{[1]\prime}(Z^{[1]}) \\
dW^{[1]} &= \frac{1}{m} dZ^{[1]} X^T \\
db^{[1]} &= \frac{1}{m} np.sum(dZ^{[1]},\ axis = 1,\ keepdims = True)
\end{aligned}
$$

랜덤 초기화

- 신경망에서 w의 초깃값을 0으로 설정한 후 경사 하강법을 적용할 경우 올바르게 작동하지 않음
  - dw를 계산했을 때 모든 층이 같은 값을 갖게 되기 때문
- 따라서 `np.random.rand()`를 이용해 0이 아닌 랜덤한 값을 부여해줘야 함

출석퀴즈 오답노트

▼ 3. 다음 코드를 실행했을 때 결과

```
A = np.random.randn(4,3)
B = np.sum(A, axis=1)
C = np.sum(A, axis=0, keepdims = True)

print(B.shape, C.shape)
```

- B의 shape는 (4,)
  - `axis=1`: 열끼리 연산
  - `keepdims = True`가 포함되지 않음 → reshape되지 않음
- C의 shape는 (1,3)
  - `axis=0`: 행끼리 연산
  - `keepdims = True`가 포함됨 → reshape

▼ 5. 신경망의 각 값의 표기법

- X: 각 column(열)이 하나의 훈련 샘플인 행렬
- $a^{[2]}$: 2번째 레이어의 활성화 벡터
- $a^{[2](6)}$: 6번째 훈련 샘플에 대한 2번째 레이어의 활성화 벡터
- $a_4^{[2]}$: 2번째 레이어의 4번째 뉴런의 활성화 함수의 출력

▼ 7. 랜덤 초기화 TF 문제

로지스틱 회귀의 가중치 w를 모두 0으로 설정하면 "break symmetry"할 수 없어 적절한 decision boundary를 학습할 수 없기 때문에 랜덤하게 초기화해야 한다. (F)