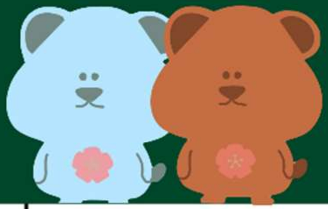
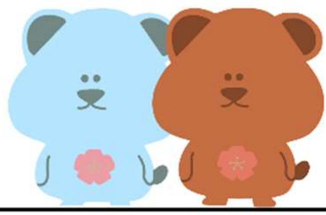


# 3. 최적화 문제 설정

유런 6기 중급팀 문가을

# 목차



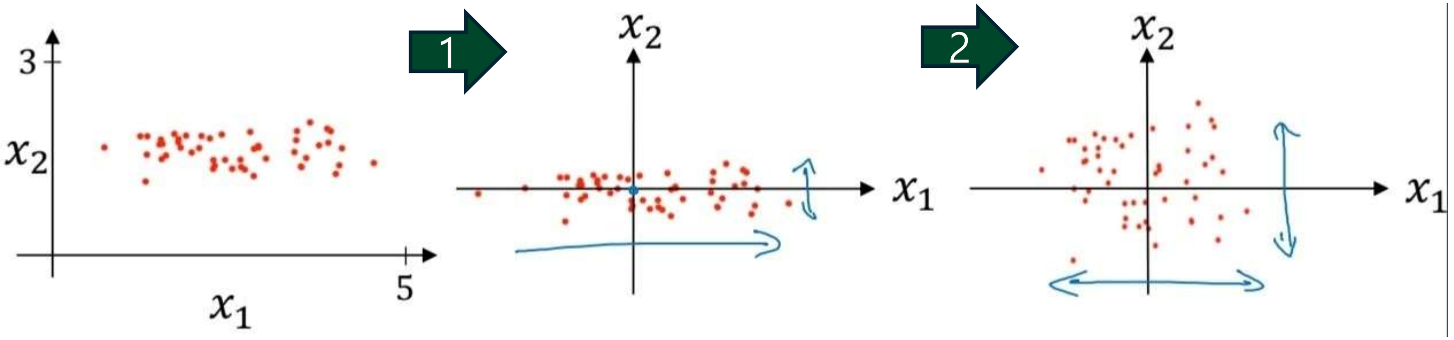


# 1. 입력 값의 정규화

# 1. 입력 값의 정규화

<u>스케일이 다른 입력값을 평균과 분산을 같게 만든다.</u>

$$X := \frac{X - \mu}{\sigma}$$



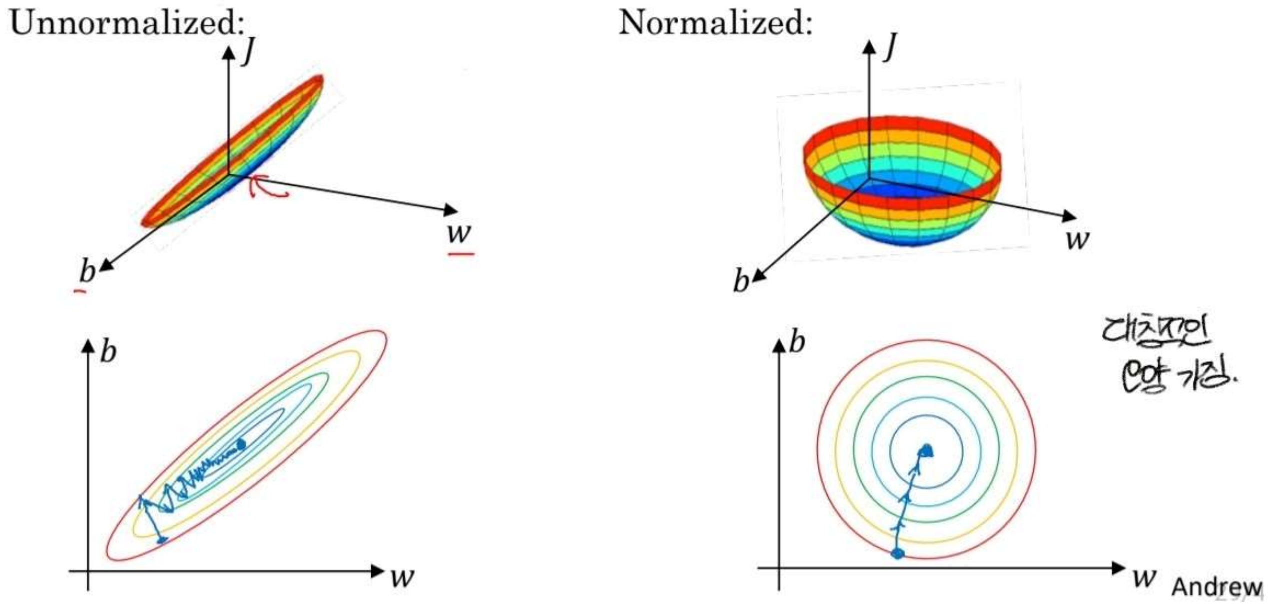

1. 평균 빼기 x := x – mean
   -> 평균을 0으로 만듦.

2. 분산 정규화하기 x := x / sigma
   -> 분산을 1으로 만듦.

Test set를 정규화할 때도 입력값을 정규화할 때 사용했던 입력값의 평균과 분산을 사용해야 함.

# 1. 입력 값의 정규화

## 비용함수 비교




정규화하지 않은 데이터셋으로 훈련할 경우

➔ 특성 값의 범위가 달라 매개 변수 값들이 매우 다름

➔ 비용 함수에서 앞뒤로 왔다갔다 하기 위해 많은 단계가 필요하기 때문에,
경사 하강법에서 매우 작은 학습률을 사용하게됨.

학습에 오랜 시간이 걸림

# 2. 경사 소실 / 폭발

EWHA EURON

# 2. 경사 소실 / 폭발

<u>미분값이 매우 작아지거나 매우 커지는 문제</u>

활성화 값이 매우 작아지거나 매우 커지는 예시로 설명

$g(z) = z$ , $b^{[l]} = 0$ 으로 가정

$$\hat{y} = w^{[L]} w^{[L-1]} w^{[L-2]} \cdots w^{[2]} w^{[1]} X$$

$$z^{[1]} = w^{[1]} X + 0$$

$$a^{[1]} = z^{[1]}$$

$$a^{[2]} = w^{[2]} w^{[1]} X$$

$$\vdots$$

---

$w^{[L]}$을 제외한 행렬은 같다고 가정.

$$w^{[l]} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \text{ 가정.} \qquad \hat{y} = w^{[L]} \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix}^{L-1} X = 1.5^{L-1} X$$

[1]~[L-1] 행렬수만큼 제곱

마지막 행렬은 다른 차원 가질 것임.

L의 값이 크면

y의 예측값이 매우 커짐

$$w^{[l]} = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \text{ 가정} \qquad \hat{y} = 0.5^{L-1} X$$

↳ y 예측값이 매우 작아짐

# 2. 경사 소실 / 폭발

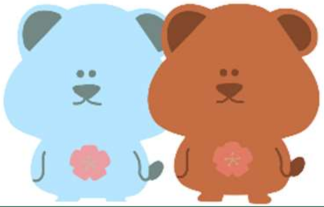
$W^{[l]} < I$ ➡ 활성화 값이 기하급수적으로 감소

$W^{[l]} > I$ ➡ 활성화 값이 기하급수적으로 폭발

해결하기 위해서는 가중치를 어떻게 초기화하는지가 중요함.

미분값이 소실/폭발하는 원리는 활성화 값이 소실/폭발하는 원리와 같음

# 3. 심층 신경망의 가중치 초기화



EWHA EURON

# 3. 심층 신경망의 가중치 초기화

Z 값이 너무 크거나 작아지지 않도록 해야 한다.

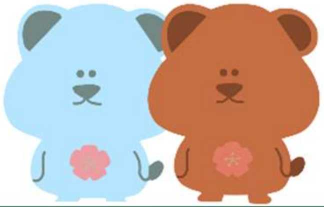
-> 가중치의 분산을 같게 만든다.

$$W^{[l]} = np.random.randn(shape) * np.sqrt(\frac{1}{n^{[l-1]}})$$

ReLU 활성화 함수를 사용하는 경우 $w_i$ 의 분산을 $\frac{2}{n^{[l-1]}}$ 으로 설정합니다.

tanh 활성화 함수를 사용하는 경우 $w_i$ 의 분산을 $\frac{1}{n^{[l-1]}}$ 또는 $\frac{2}{n^{[l-1]} + n^{[l]}}$ 으로 설정합니다.

# 4. 기울기의 수치 근사

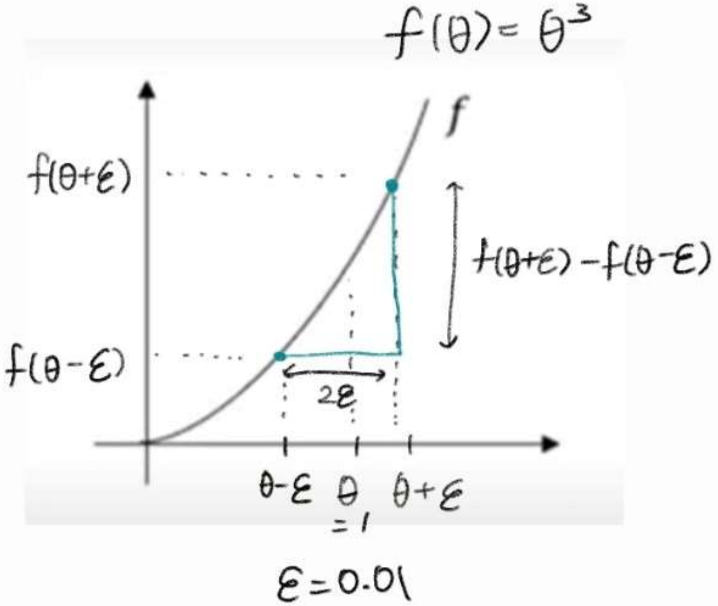


# 4. 기울기의 수치 근사

<u>역전파를 맞게 구현했는지 확인하기 위해 경사 검사를 진행한다.</u>

-> 먼저 경사의 계산을 수치적으로 근사하는 방법이 필요하다.

근사 미분값 계산 방법




$$\frac{f(\theta+\varepsilon)-f(\theta-\varepsilon)}{2\varepsilon} \approx g(\theta) = 3\theta^2$$

$$\frac{(1.01)^3-(0.99)^3}{2(0.01)} = 3.0001 \qquad g(1)=3$$

approx error : 0.0001

(one side) : 0.03

# 4. 기울기의 수치 근사

왜 일반적으로 하는 미분 계산과 다른 방법을 사용하는가?

$$f'(\theta) = \lim_{\varepsilon \to 0} \frac{f(\theta+\varepsilon) - f(\theta-\varepsilon)}{2\varepsilon}$$

이 근사의 오차는

$O(\varepsilon^2)$

오차가 작아지는 것을 나타냄

$\varepsilon = 0.01$

오차 → 0.0001 더 작음.

$$f'(\theta) = \lim_{\varepsilon \to 0} \frac{f(\theta+\varepsilon) - f(\theta)}{\varepsilon}$$

$O(\varepsilon)$

$\varepsilon = 0.01$

오차 → 0.01

-> 오차가 작아지기 때문이다.

# 5. 경사 검사

# 5. 경사 검사

<u>경사 검사를 하는 방법</u>

1. W[1], b[1], W[2], b[2], ⋯ ,W[L], b[L]을 하나의 큰 벡터 θ로 바꿈.
2. dW[1], db[1], dW[2], db[2], ⋯ ,dW[L], db[L]을 하나의 큰 벡터 dθ로 바꿈.
3. 근사적인 미분값을 구하고 미분값과 비슷한지 확인함.

# 5. 경사 검사

근사 미분값 구하기

for each i :

$$d\theta_{approx}^{[i]} = \frac{J(\theta_1, \theta_2, \cdots \theta_i + \varepsilon, \cdots) - J(\theta_1, \theta_2, \cdots, \theta_i - \varepsilon, \cdots)}{2\varepsilon}$$

$$\approx d\theta^{[i]} = \frac{\partial J}{\partial \theta_i}$$

결론 : $d\theta_{approx}$ (vector) 나옴 $\longrightarrow$ $d\theta$와 가까워야 함.

근사 미분값과 미분값이 근사적으로 같은지 확인하기
-> 유클리드 거리를 구해 확인함.

$$\frac{\| d\theta_{approx} - d\theta \|_2}{\| d\theta_{approx} \|_2 + \| d\theta \|_2} \approx 10^{-7} \rightarrow \text{great!}$$

$\varepsilon = 10^{-7}$로 하면

큰 값이 나오면 버그의 가능성 有

# 5. 경사 검사 시 주의할 점

# 5. 경사 검사 시 주의할 점

1. 디버깅을 위해서만 경사 검사를 사용하기
   -> 근사 미분값을 구하는 데에 시간이 오래 걸리기 때문임.

2. 경사 검사의 알고리즘이 실패하면 ( 근사 미분값과 미분값의 차이가 크면 )
   -> 개별적인 컴포넌트를 확인해 버그를 확인하기
   -> 각각 i에 대해 dθapprox[i]와 dθ[i] 확인

3. 비용함수에 정규화 term이 있다는 것을 기억하기

4. 드롭아웃에서는 경사 검사가 작동하지 않는다.
   -> 드롭아웃에서 비용함수를 계산하기 어렵기 때문.

5. 무작위적 초기화에서 w와 b가 0에 가까울 때 경사 검사가 잘 되는 경우
   -> 훈련을 조금 시켜 w와 b가 0에서 멀어지게 한 다음 경사 검사를 다시 해보기

- $J(w, b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m}\|w\|_2^2$
- L2 regularization: $\|w\|_2^2 = \sum_{i=1}^{n_x} w_j^2 = w^T w$
- L1 regularization : $\frac{\lambda}{2m}\|w_1\| = \frac{\lambda}{2m}\sum_{i=1}^{n_x}|w|$

# 퀴즈 리뷰

# 퀴즈 리뷰

✗ 1. 데이터 정규화(normalizing)에 대해 맞는 설명을 모두 골라주세요 * 0/1

- [x] 테스트 세트를 정규화할 때 테스트 데이터의 μ,σ를 사용한다. ✗
- [x] 정규화를 통해 비용함수의 모양은 더 둥글고 대칭에 가까워진다 ✓
- [x] 정규화는 학습을 빠르고 안정적으로 수행할 수 있도록 돕는다 ✓
- [x] 훈련 시 경사 폭발과 같은 문제가 발생할 수 있다 ✓

1. 테스트 세트를 정규화할 때는 훈련 데이터의 평균과 분산을 사용함.

2. 매개 변수의 값의 범위가 같아지기 때문에 둥글고 대칭에 가까워짐.

3. 정규화를 할 경우, 비용함수가 둥글고 대칭에 가까워져 최적화하기 좋은 형태가 됨
-> 학습 알고리즘이 빠르고 안정적으로 수행할 수 있도록 도움.

4. 경사 폭발은 데이터의 정규화와 관련이 있지 않고, 가중치의 값이 매우 크거나 작을 때에 생기는 문제임.

# 퀴즈 리뷰

✓ 6. J(θ) = θx라고 할 때, 이 모델은 하나의 파라미터 θ와 x를 입력으로 가집니다. *1/1
순전파를 구현하는 코드를 완성해 주세요.

In [2]:
```
# GRADED FUNCTION: forward_propagation

def forward_propagation(x, theta):
    """
    Implement the linear forward propagation (compute J) presented in Figure 1 (J(theta) =

    Arguments:
    x -- a real-valued input
    theta -- our parameter, a real number as well

    Returns:
    J -- the value of function J, computed using the formula J(theta) = theta * x
    """

    ### START CODE HERE ### (approx. 1 line)

    ### END CODE HERE ###

    return J
```

In [3]:
```
x, theta = 2, 4
J = forward_propagation(x, theta)
print ("J = " + str(J))
```

```
J = 8
```

- ○ J = theta*x
- ◉ J = np.dot(theta, x)
- ○ J = np.concatenate((theta*x))

순전파는 비용함수를 계산하는 과정
-> Θ와 X는 vector이므로 θX를 계산하기 위해서는 행렬 곱을 통해 계산하여야 함.

감사합니다.