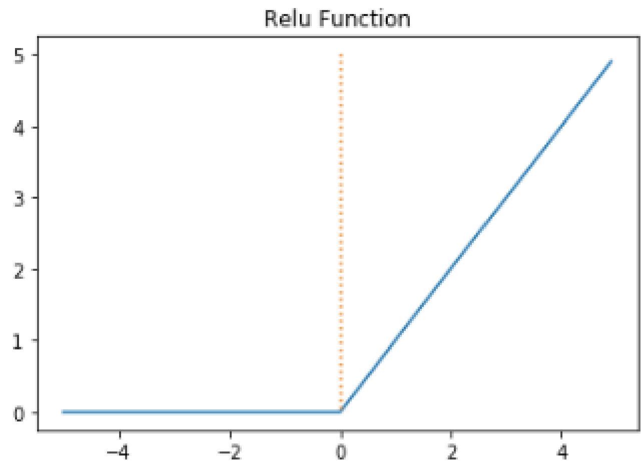
1주차

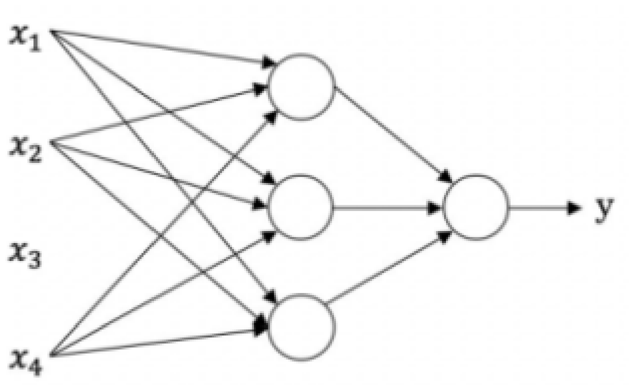
1. 딥러닝 소개

What is neural network?

- 신경망 : 함수
- x : 신경망의 입력 → O : 신경망에서의 하나의 뉴런 → y : 출력
- Relu 함수: 0으로 유지되다가 직선으로 올라가는 함수(Rectified linear unit)



- 뉴런 여러개를 쌓아 더 큰 신경망을 만들 수 있음



- O들은 은닉 유닛이고 모든 입력 특성들과 연결되어있음

Supervised Learning

지도학습

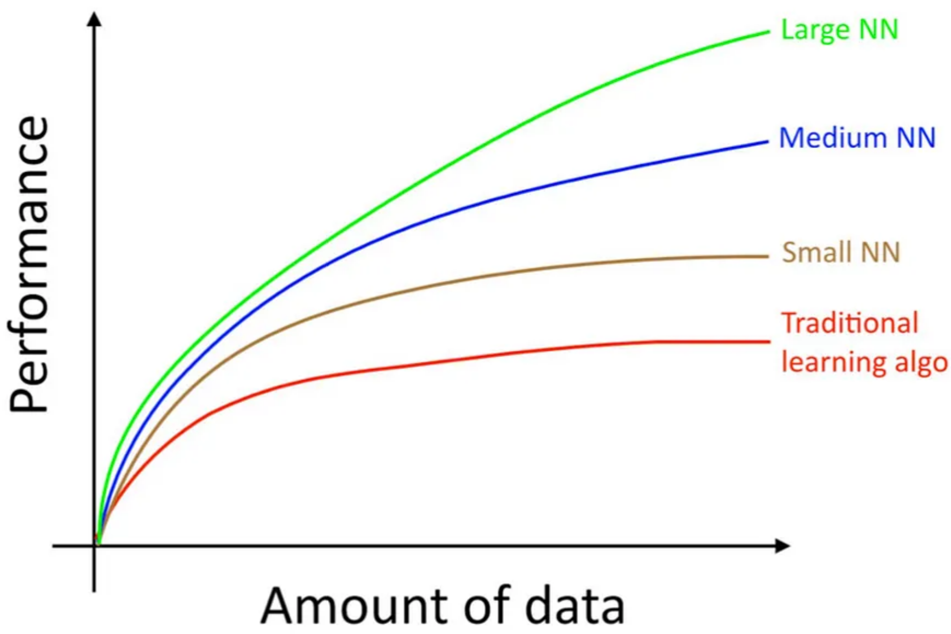
- 정답이 주어져있는 데이터를 사용해서 컴퓨터를 학습시키는 방법
- 입력 x와 출력 y에 매핑되는 함수를 학습하려 함
- 분야에 따라 적용되는 신경망이 다르다

구조적/비구조적 데이터

구조적 데이터	비구조적 데이터
특성들이 잘 정의된 데이터 특성들을 열로 가진 데이터베이스	이미지, 음성 파일, 텍스트 데이터들 특성은 이미지 픽셀값이나 텍스트의 각 단어

- 비구조적 데이터가 작업하기 훨씬 어려움
- 딥러닝 덕분에 비구조적 데이터 해석 발전 (음성 인식, 이미지 인식, 자연어처리 등)

Why is deep learning taking off?



x축의 데이터는 labeled data (입력값 x와 레이블 y가 같이 있는 훈련 세트)

x축 = Amount of data = m : 훈련 세트의 크기

- 높은 성능을 발휘하기 위해 충분히 큰 신경망, 많은 데이터가 필요하다
- 규모가 딥러닝의 발전을 주도했다

    - 규모 : 신경망 크기 (많은 은닉유닛, 많은 연결/파라미터, 데이터의 규모)
- 훈련할 데이터가 많지 않다면 구현방법에 따라 성능이 결정되는 경우가 많음

딥러닝 부상 3가지 요인

1. 데이터양 증가
2. 컴퓨터 성능 향상
    - Idea, Code, Experiment 과정을 빨리 반복해 아이디어 더 빨리 발전 가능
3. 알고리즘의 개선

    ex) sigmoid → ReLu 함수

    - sigmoid 함수 : 경사가 거의 0인 곳에서 학습이 굉장히 느려짐
    - ReLu 함수 : 입력값이 양수인 경우의 경사가 1로 모두 같으므로 경사가 서서히 0에 수렴할 가능성이 훨씬 적어짐

2. 신경망과 로지스틱회귀

Binary Classification (이진 분류)

이진분류

- 1 (그렇다) vs 0 (아니다) 2개로 분류하는 것
    - ex) 고양이가 맞다 / 고양이가 아니다
- x (입력 사진) → y (1 vs 0)

Logistic Regression

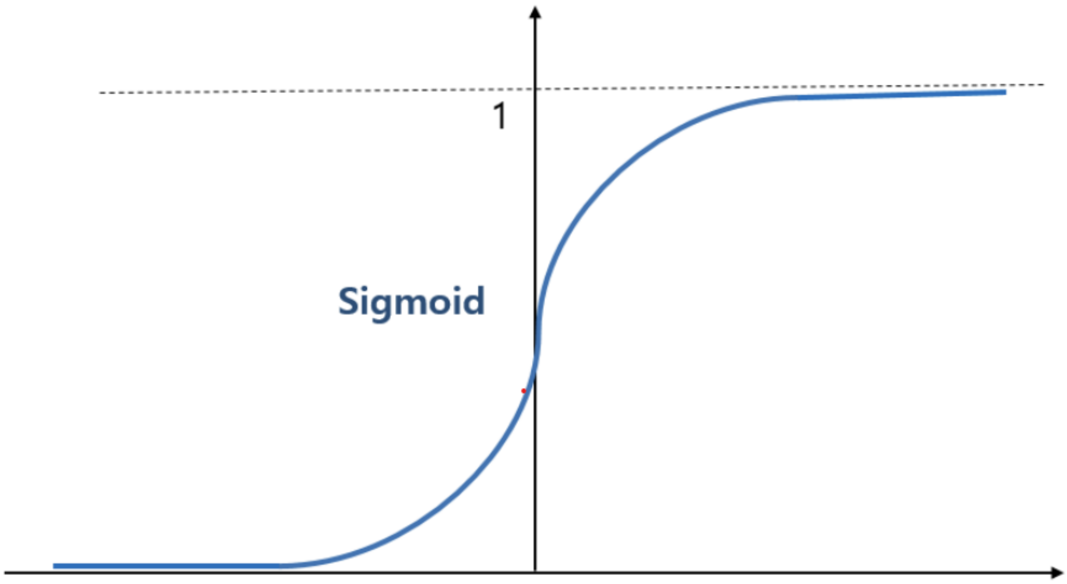
로지스틱 회귀

- 답이 0 또는 1로 정해져있는 이진 분류 문제에 사용되는 알고리즘
- x : 입력 특성 y : 실제 값
- $\hat{y}$: y의 예측값
    - $\hat{y} = P(y=1 \mid x)$ → y가 1일 확률
    - $0 \le \hat{y} \le 1$

- sigmoid 함수를 이용해 선형함수 $z = w^T x + b$ 를 0과 1 사이의 값으로 변환해 줌

$$\hat{y} = \sigma(w^T x + b)$$

- sigmoid 함수



$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

- z→∞ $\sigma(z) \approx \frac{1}{1+0} = 1$
- z→-∞ $\sigma(z) \approx \frac{1}{1+\infty} = 0$

Logistic Regression cost function

- i 번째 훈련 샘플에 관한 데이터 : 위첨자 (i) 사용

Loss function (손실 함수)

- 실제값(y)와 예측값(ŷ)의 오차를 계산하는 함수
- 하나의 입력에 대한 오차 계산
- 주로 $L(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$이지만 로지스틱 회귀에서는 사용 x
  - 경사하강법 사용 불가하기 때문
- 로지스틱 회귀에서의 손실 함수

$$L(\hat{y}, y) = -(ylog\hat{y} + (1-y)log(1-\hat{y}))$$

- y=1 : $L(\hat{y}, y) = -log\hat{y}$ → ŷ 1에 수렴
- y=0 : $L(\hat{y}, y) = -log(1-\hat{y})$ → ŷ 0에 수렴

Cost function (비용 함수)

- 모든 입력에 대한 오차 계산
- 모든 입력에 대해 계산한 손실 함수의 평균값
- 비용 함수

$$J(w,b) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)}log\hat{y}^{(i)} + (1-y^{(i)})log(1-\hat{y}^{(i)})$$

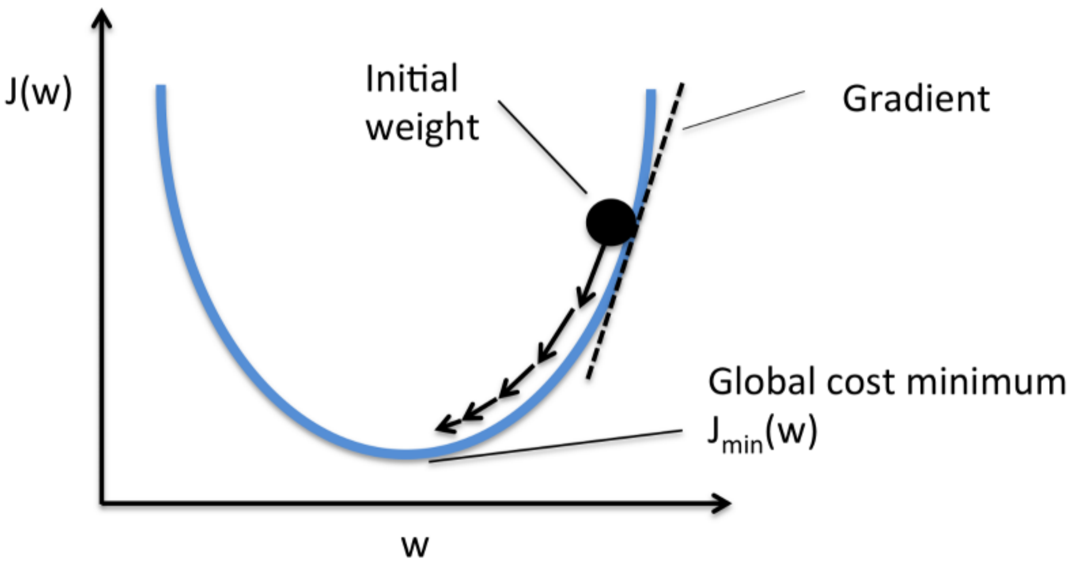
Gradient Descent (경사하강법)

- J(w,b)를 최소화시키는 파라미터 w,b를 찾아내는 방법
- 비용 함수 J는 볼록한 형태여야 함 (볼록하지 않은 함수는 지역 최적값이 여러개이므로 최적의 파라미터를 찾을 수 없음)
- w,b를 초기화해서 시작 → 가장 가파른 방향으로 한 스텝씩 이동
- 알고리즘

$$w := w - \alpha\frac{dJ(w,b)}{dw}$$

$$b := b - \alpha\frac{dJ(w,b)}{db}$$

  - α : learning rate
  - $\frac{dJ(w)}{dw}$ = dw : 기울기



  - dw>0이면 w가 더 작은 방향으로 업데이트
  - dw<0이면 w가 더 큰 방향으로 업데이트
- 2개 이상의 변수이면 d 대신 ∂(편미분 기호)를 사용

  ex) $b := b - \alpha\frac{\partial J(w,b)}{\partial b}$

Derivatives (미분)

도함수

- 함수의 기울기
- a를 아주 작게 변화시켰을 때 f(a)의 변화량
- $\frac{d}{da}f(a) = \frac{df(a)}{da}$
- 함수가 직선이면 기울기가 항상 동일
  - ex) f(x) = 3x

More derivatives examples

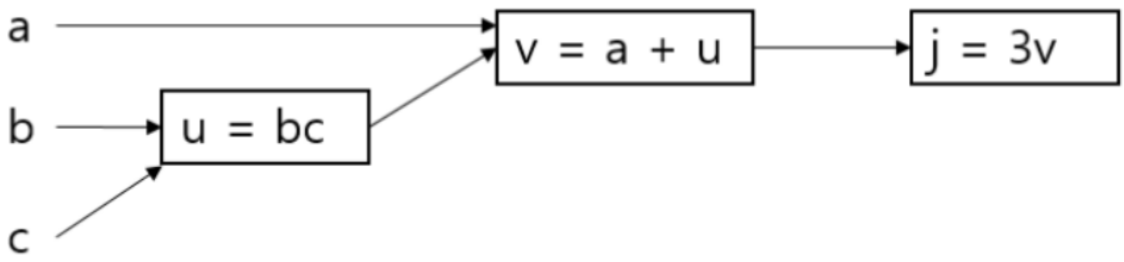
- 함수의 기울기는 함수의 위치에 따라 다른 값을 가질 수 있음
  - ex) f(x) = x^2

f(a)	$\frac{d}{da}f(a)$
a^2	$2a$
a^3	$3a^2$

$ln(a)$	$\frac{1}{a}$

Computation Graph (계산 그래프)

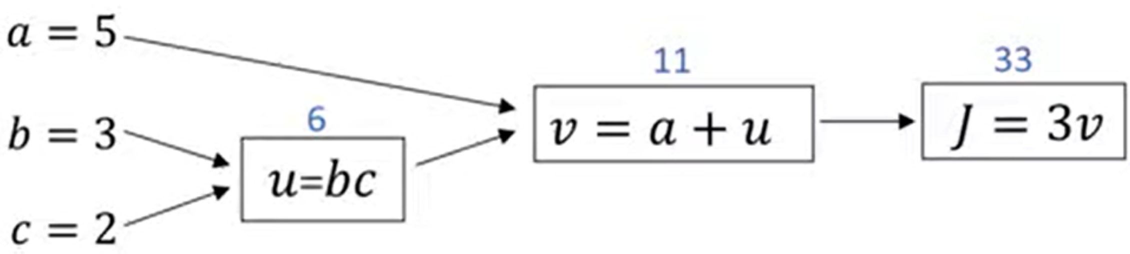
- 왼쪽에서 오른쪽으로 계산하는 화살표를 사용해 계산을 정리(정방향)
- J(a,b,c)=3(a+bc)
  1. u=bc
  2. v=a+u
  3. J=3v



Derivatives with a Computation Graph

미분의 연쇄법칙

- 합성함수의 도함수에 대한 공식 (역방향)
- 합성함수를 구성하는 함수의 미분을 곱함으로써 구할수 있음
  - ex) $\frac{dJ}{du} = \frac{dJ}{dv}\frac{dv}{du}$

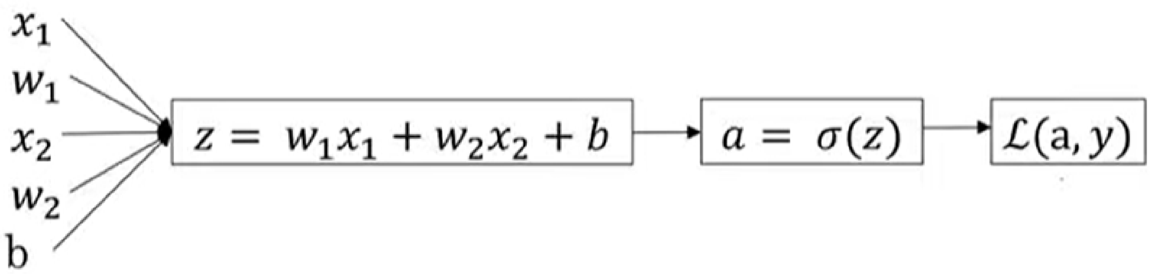


- 입력변수 a에서 출력변수 J 까지 a→b→J
  - $\frac{dJ}{dv} = 3$ $\frac{dv}{da} = 1$
    - v=11 → 11.001 일 때 J=33 → 33.003 이므로 $\frac{dJ}{dv} = 3$
    - a=5 → 5.001 일 때 v=11 → 11.001 이므로 $\frac{dv}{da} = 1$

- $\frac{dJ}{da} = 3 = \frac{dJ}{dv}\frac{dv}{da} = 3 * 1$
- $\frac{dJ}{db} = \frac{dJ}{du}\frac{du}{dv} = 3 * 2 = 6$

- 최종변수 : Final output var, 미분하려고 하는 변수 : var

Logistic Regression Gradient descent



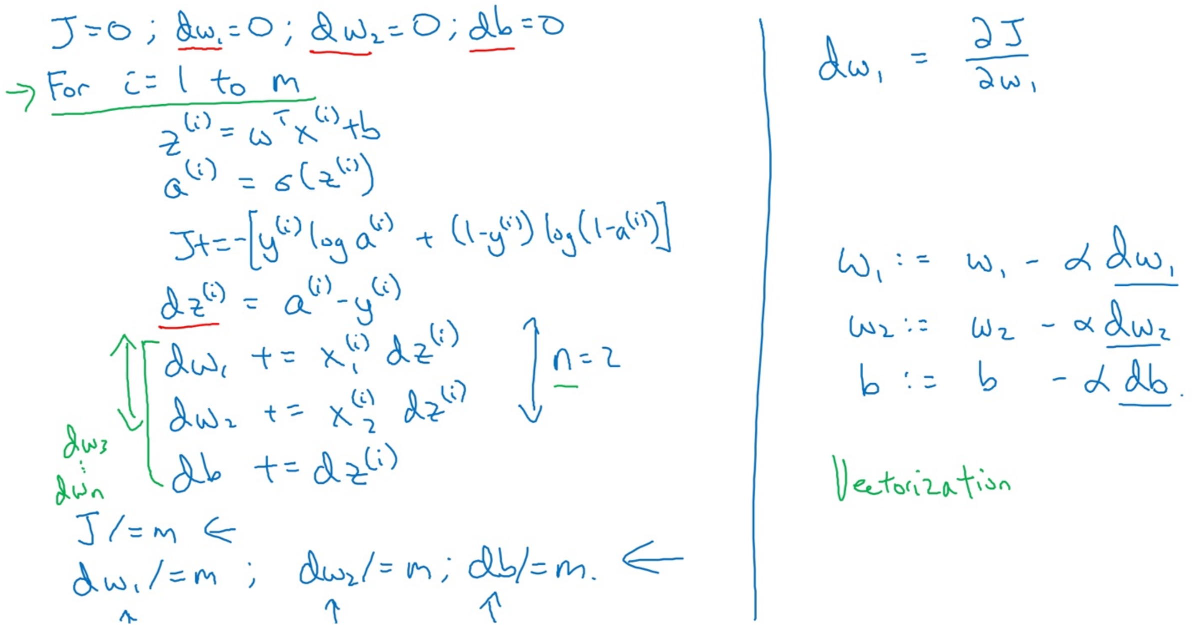
- $da = -\frac{y}{a} + \frac{1-y}{1-a}$
- $dz = \frac{dL}{dz} = \frac{dL(a,y)}{dz} = \frac{dL}{da}\frac{da}{dz} = a - y$
- $dw1 = \frac{dL}{dw1} = x1dz$
- $dw2 = x2dz$
- $db = dz$
- 값들을 구한 후 갱신
    - $w1 := w1 - \alpha dw1$
    - $w2 := w2 - \alpha dw2$
    - $b := b - \alpha db$

Gradient descent on m examples

- 비용 함수

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(a^{(i)}, y^{(i)})$$

- 로지스틱 회귀 경사하강법 구현 코드



- 이 과정은 경사하강법 한 단계에 사용
- dw1, dw2, db는 전체 비용 함수의 도함수
- 특성의 개수가 여러개면 특성의 개수도 for문을 이용해 처리해야 함 → 알고리즘 비효율적이므로 벡터화로 명시적 for문 제거