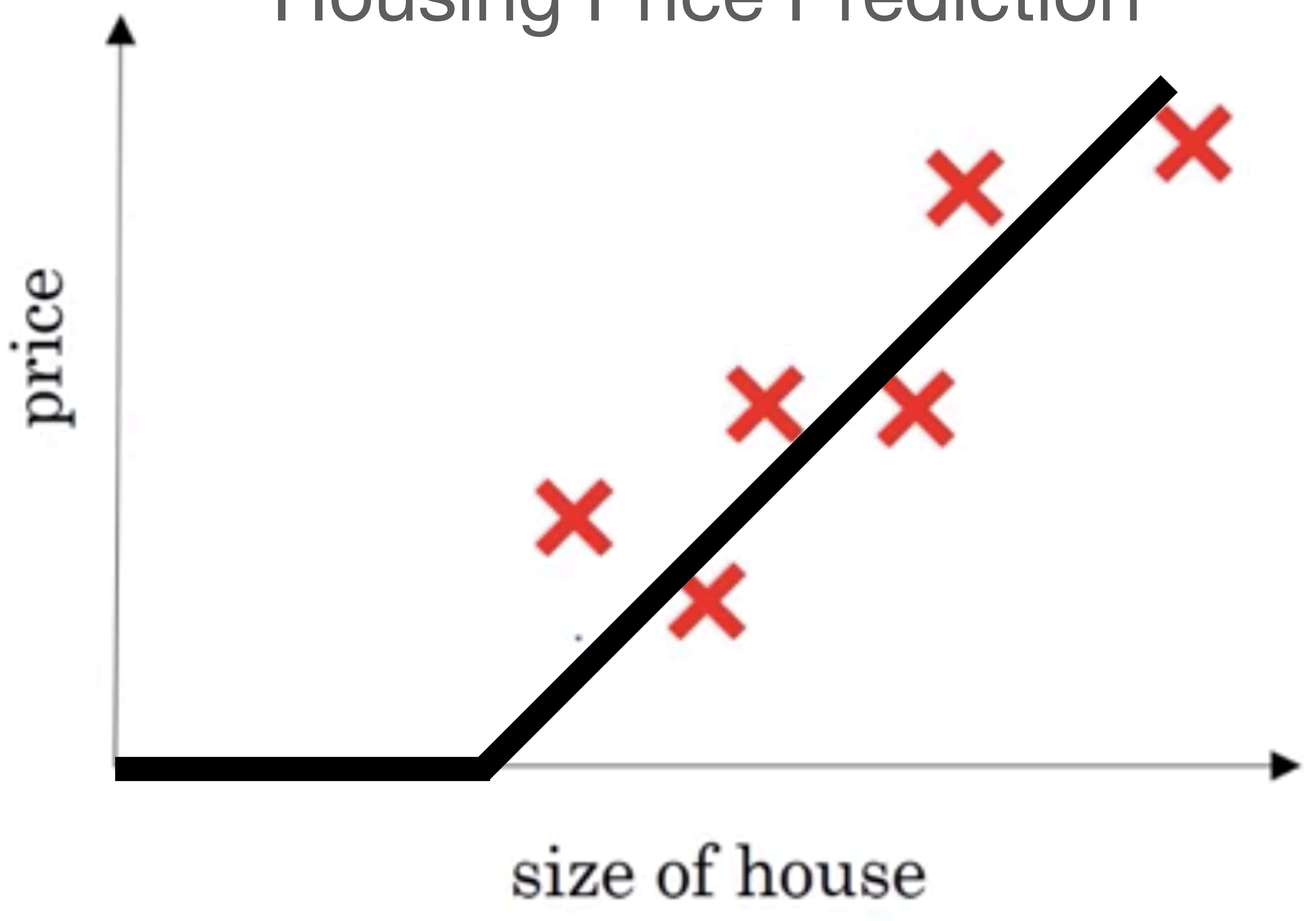
Neural Network and Deep Learning

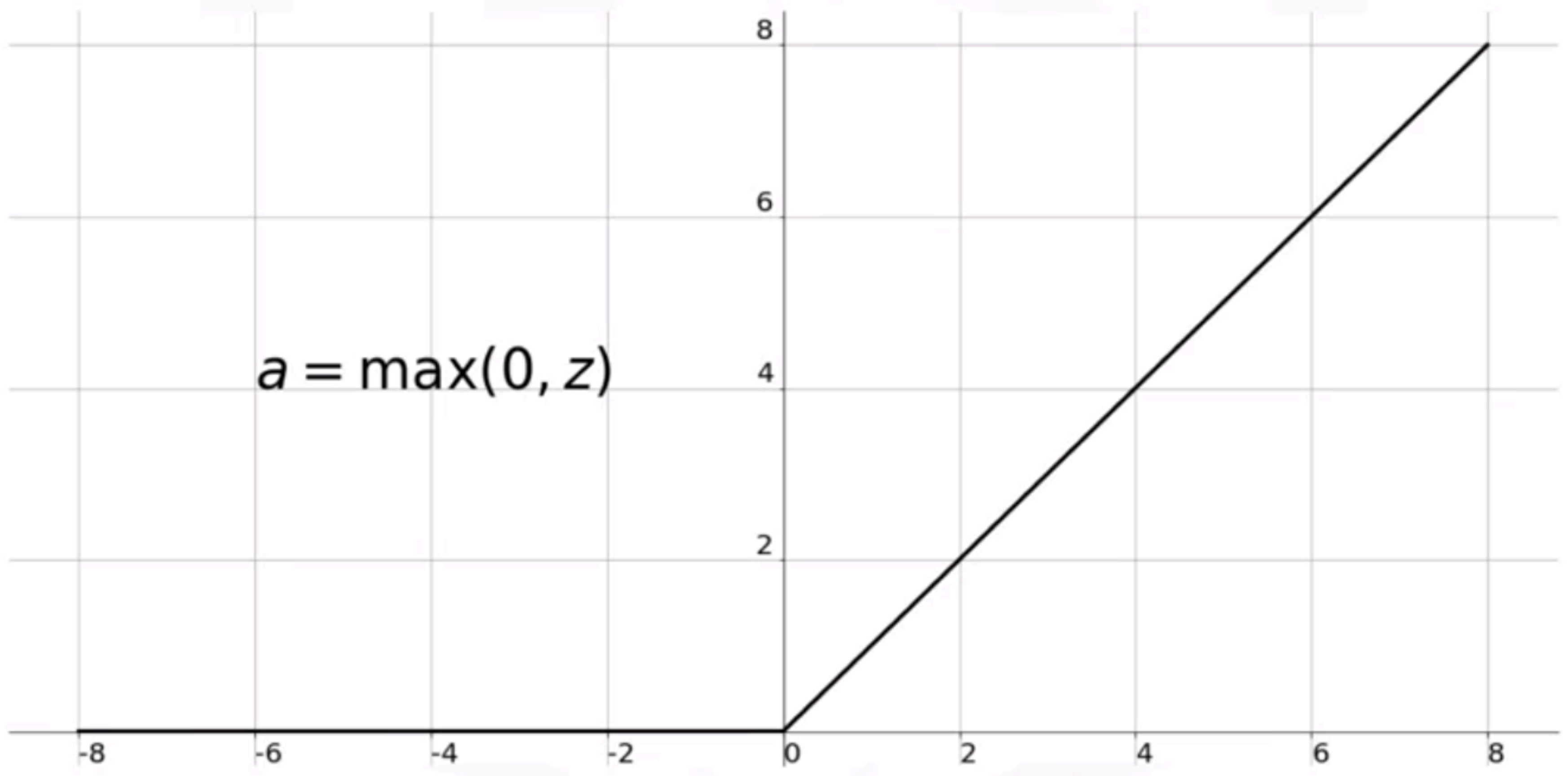
박달님 04.23.21

Neural Network란 무엇인가? (1)

- 딥러닝 - 신경망의 트레이닝
- 미래 집 값 예측
  - 가격은 마이너스가 될 수 없기 때문에 0이 되는 부분 생성
  - ReLU 함수 (Rectified Linear Units)
    - Rectified: 0을 최대값
- 신경 세포 - 크기를 입력값으로 받아 일차함수를 만든다.

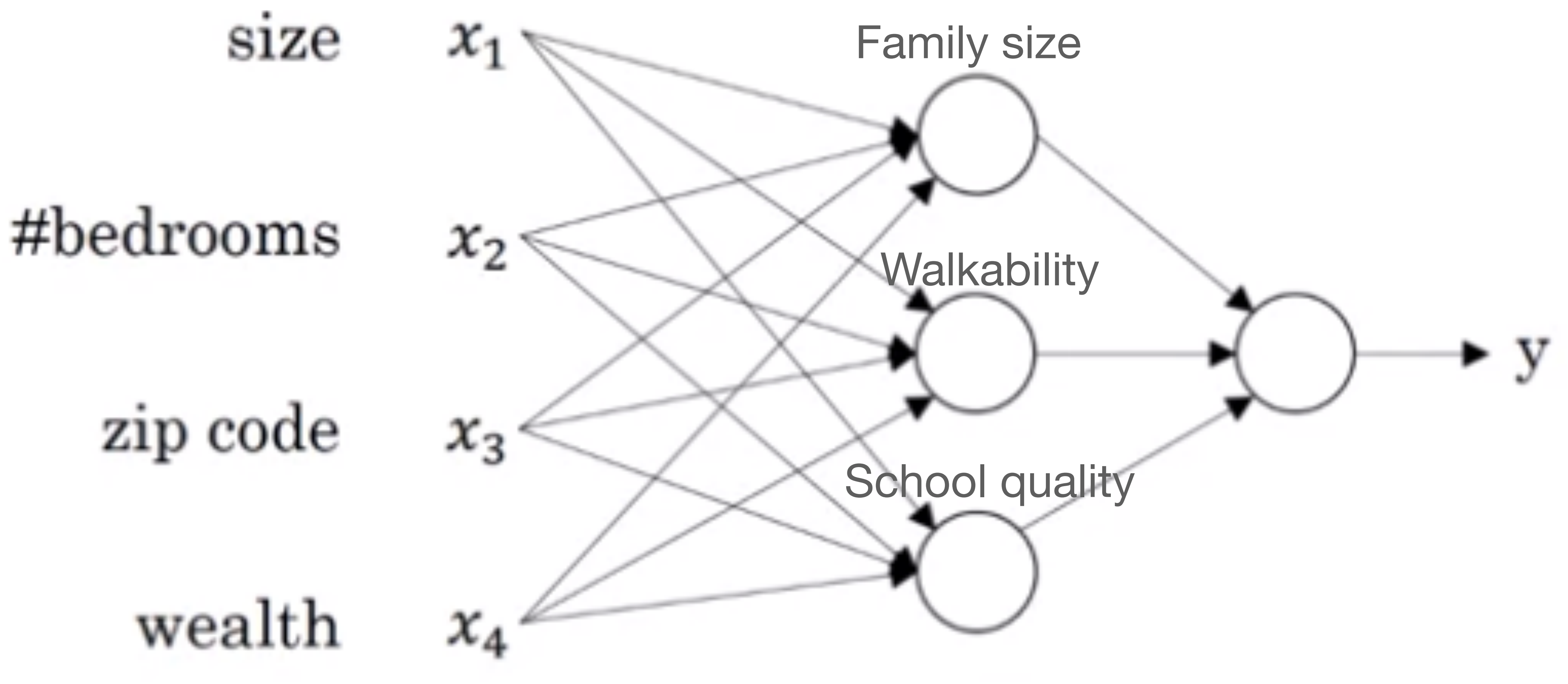
Housing Price Prediction





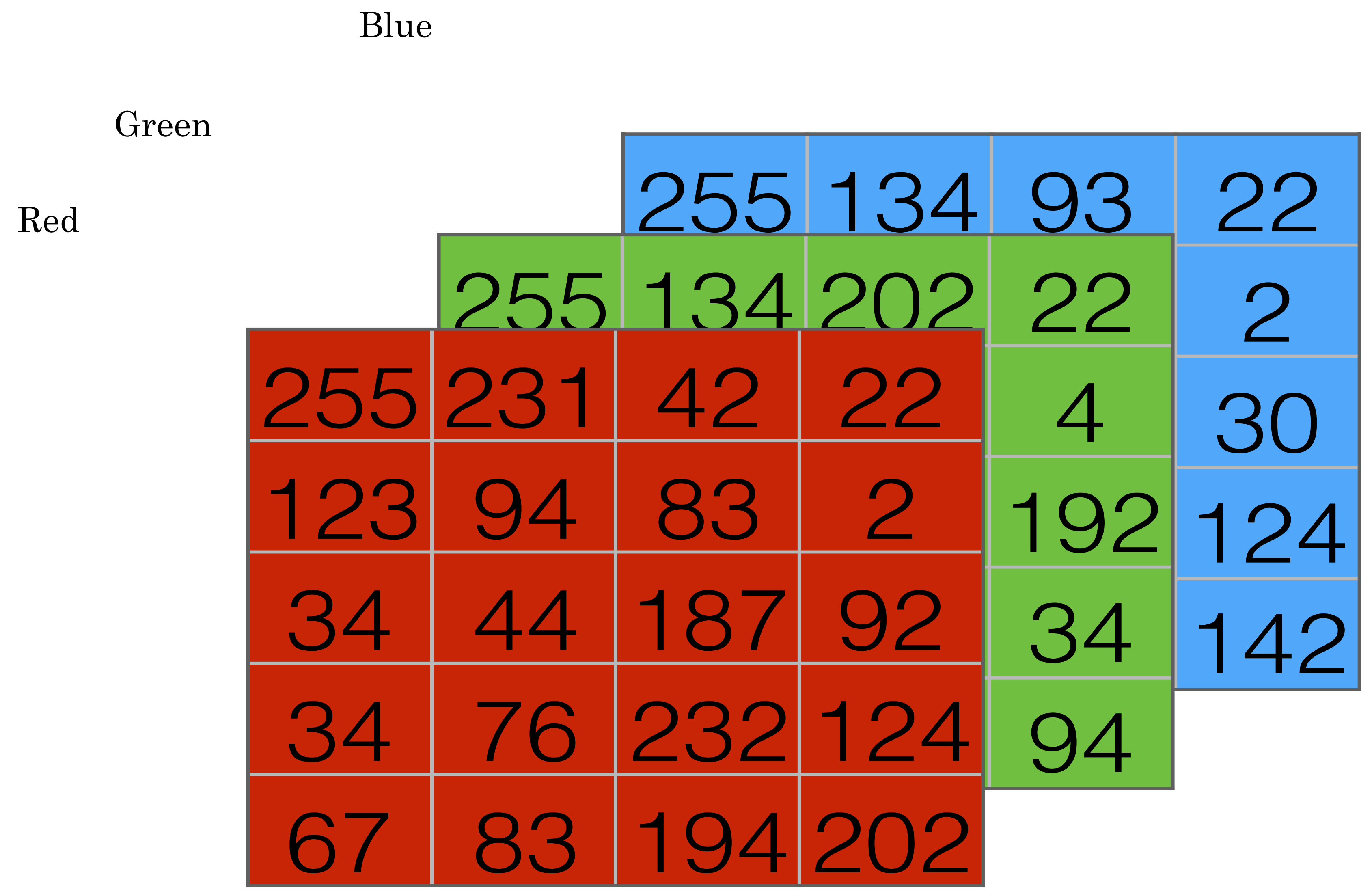
Neural Network란 무엇인가? (2)

- 다른 특성이 있다고 가정
- 원 - ReLU/다른 비선형 함수
- Hidden Units - 각각 특성을 반영
  - Ex) 첫번째 노드 - family size
  - X1과 X2에만 의존하기 보다는 신경망이 어떤 것이던 선택할 수 있도록 한다.



Binary Classification

신경망 프로그래밍의 기초 기술

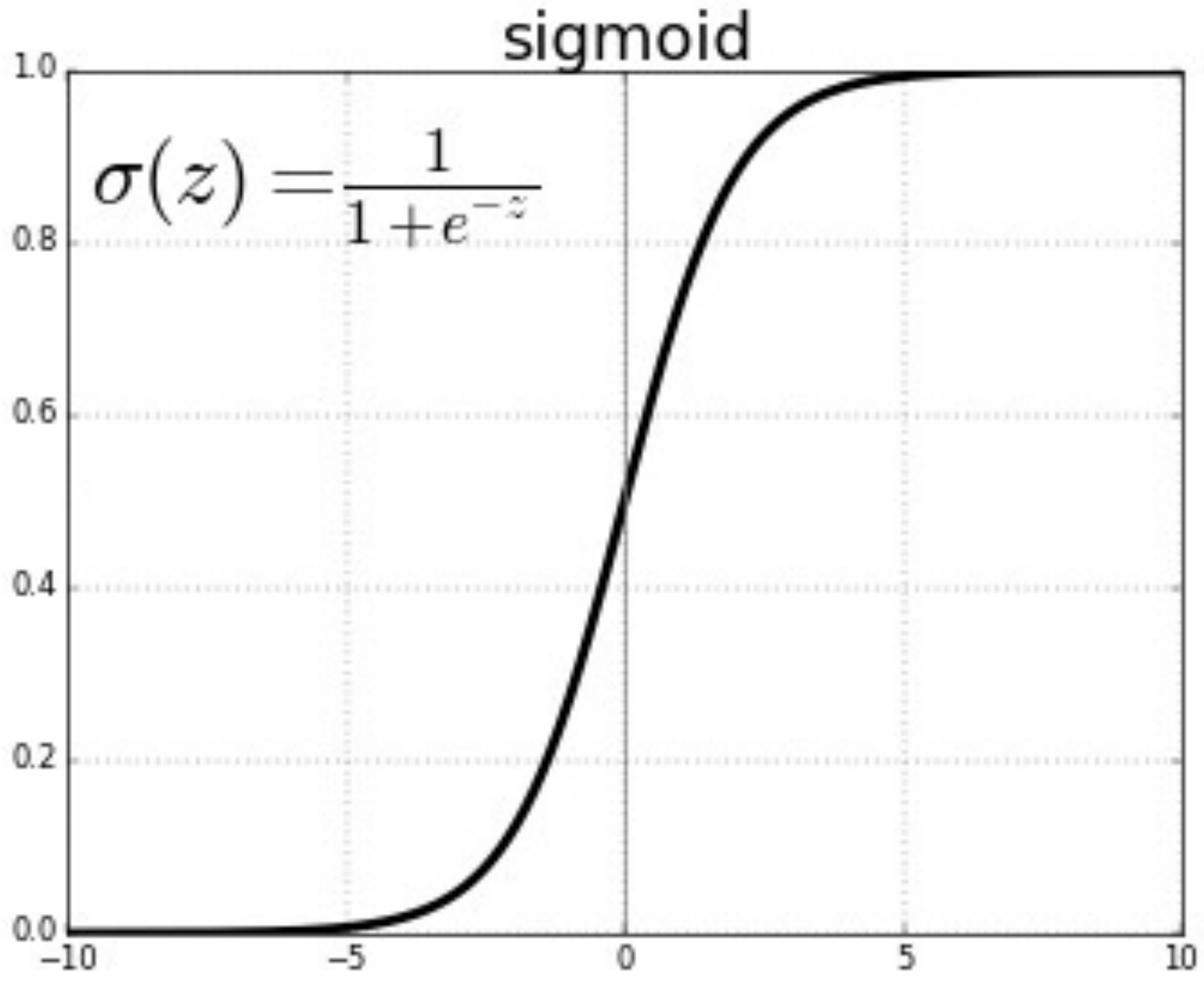


- Input of image. Yes (1) vs No (0)
- 컴퓨터의 이미지 표현 - RGB 채널에 대응하는 세 개로 분리된 행렬 사용
- 입력 이미지: 64 x 64 픽셀
  - RGB 픽셀의 채도에 해당하는 값을 가진 3개의 64 x 64 행렬이 있다
  - 픽셀들의 채도값을 특징 벡터로 바꾸기 위해 픽셀값 모두를 하나의 입력 특징벡터 x에 펼침
  - 특징벡터 x의 전체 차원: nx = 64 x 64 x 3 = 12,288
- 목표: 입력 벡터 x로 표현된 이미지를 입력으로 주어 분류자를 학습 시킴. 출력 레이블 y가 1인지 0인지 예측
- Single training example: (x, y)
  - x: x차원을 가진 특징벡터, y: 0/1 중 하나의 레이블, 학습 세트: m개의 학습 표본으로 구성
  - 행렬 X: 학습 표본의 수 m개의 세로줄, nx개의 가로줄. X.shape() -> (nx, m)
  - 행렬 Y: (1, m) matrix

Logistic Regression

Binary Classification에 사용

- Given x, want ŷ = P(y=1 | x)
- x ∈ R^nx, 매개 변수: w ∈ R^nx, b ∈ R
- Output ŷ = w^T x + b
  - Binary classification으로 그닥 좋지 않은 알고리즘
  - ŷ이 Y가 1일 확률이 되도록 만드는 것이 좋음. w^T x + b 값이 1보다 훨씬 크고 마이너스일 수도 있기 때문
- Logistic regression: ŷ = σ(w^T x + b)
  - Z = w^T x + b
  - 목표: 매개변수 w와 b를 배워서 ŷ이 Y=1이 되는 확률을 추정한 수치를 구하는 것



Logistic Regression Cost Function

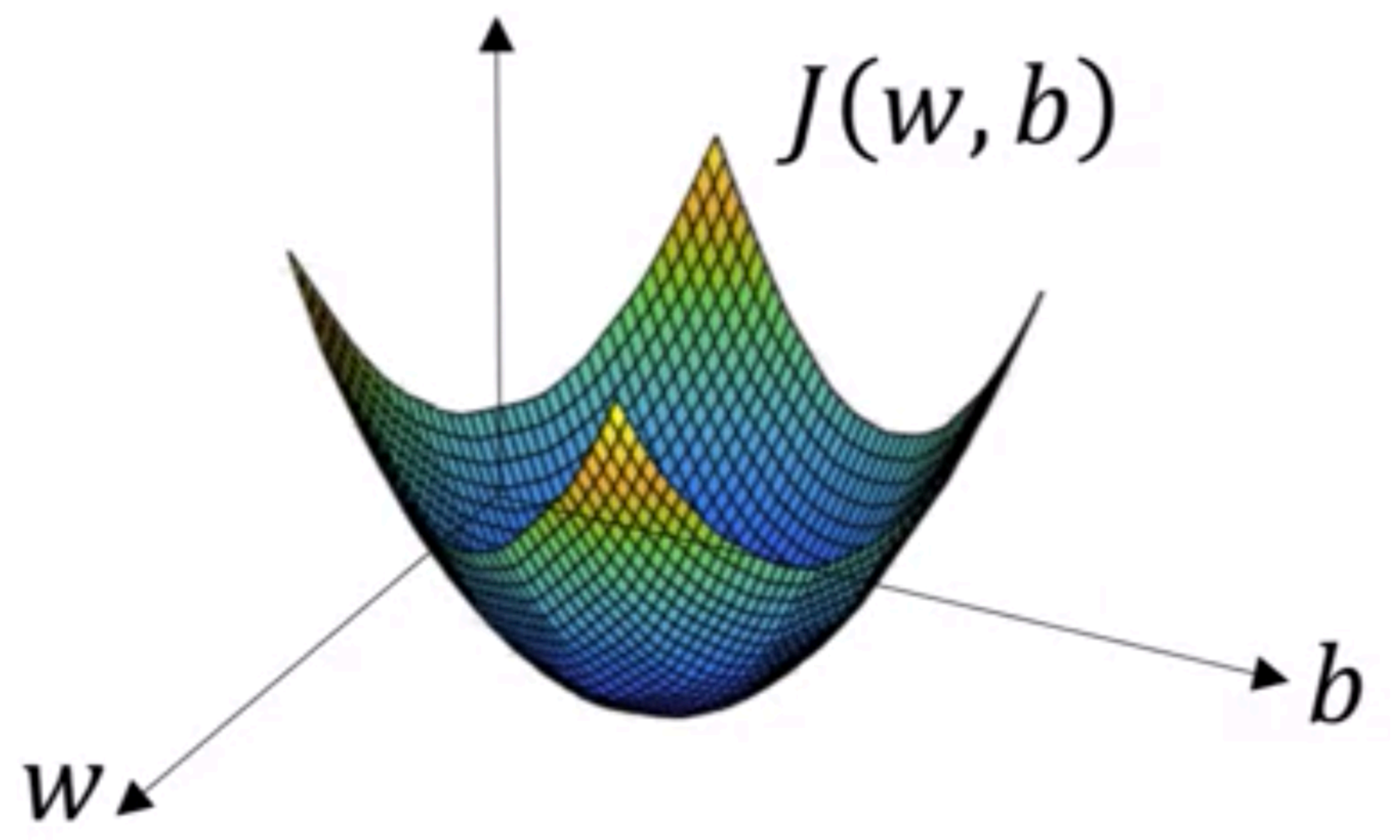
$\hat{y} = \sigma(w^T x + b)$, where $\sigma(z) = \frac{1}{1+e^{-z}}$

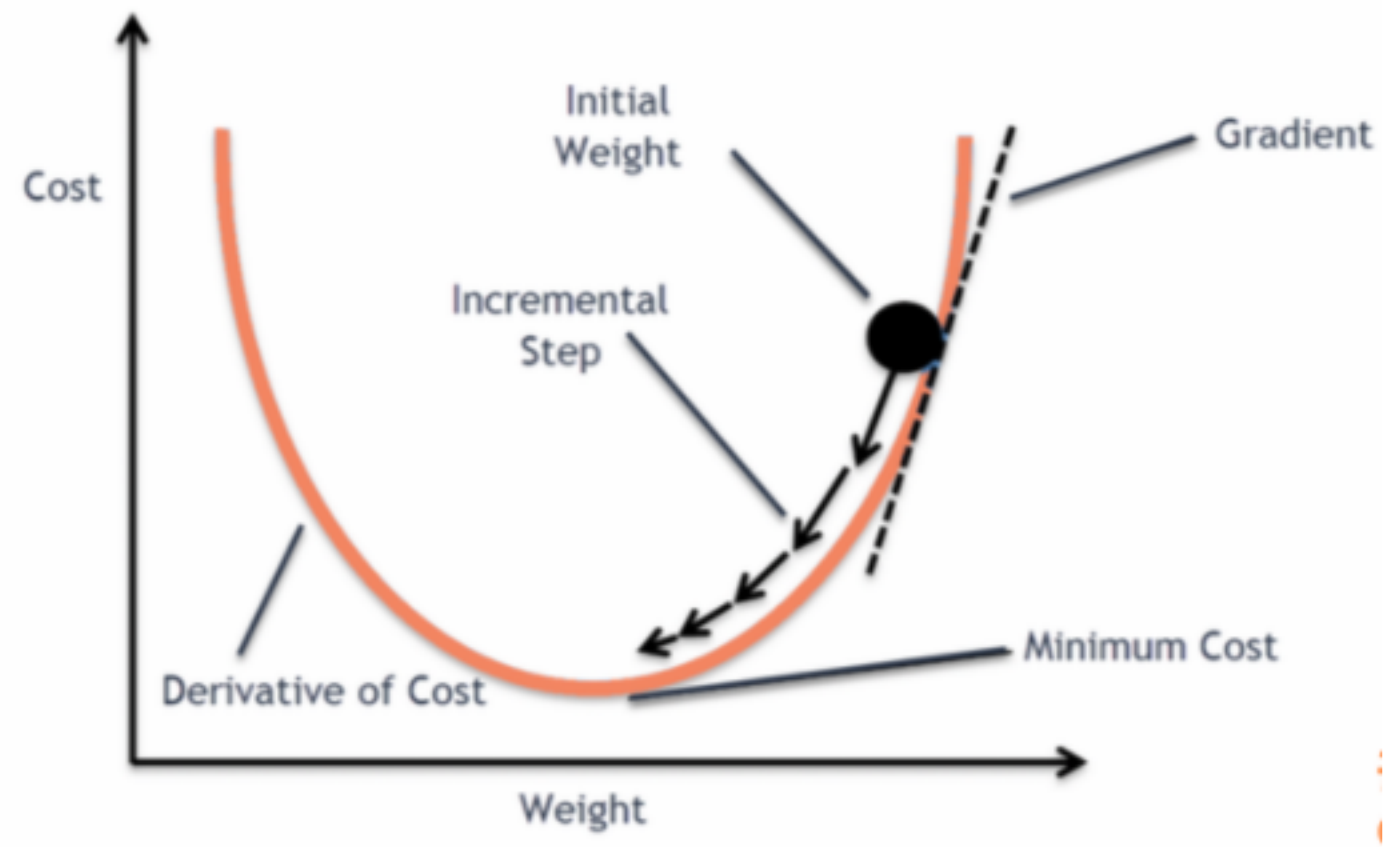
Given $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, want $\hat{y}^{(i)} \approx y^{(i)}$

- Loss function: L(ŷ, y) = ½ (ŷ-y)^2 -> logistic regression에서 잘 사용하지 않는다.
  - Optimization becomes non-convex -> Multiple local optima
    - Gradient descent가 전역 최적값을 못 찾을수도 있다
- 실제로 logistic regression에서 사용하는 Loss function
  - L(ŷ, y) = - (y-log ŷ + (1-y)log(1-ŷ))
    - If y = 1 : L(ŷ, y) = - log ŷ -> Want large ŷ -> ŷ이 1에 가까워짐 (sigmoid)
    - If y =0: L(ŷ, y) = - log(1-ŷ) -> Want small ŷ -> ŷ이 0에 가까워짐 (sigmoid)
- Cost Function -> 전체적인 training test에서 얼마나 잘 작동하는지의 여부 측정
- $J(w,b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) = -\frac{1}{m} \sum_{i=1}^{m} [y^{(i)} log \hat{y}^{(i)} + (1 - y^{(i)}) log(1 - \hat{y}^{(i)})]$

Gradient Descent

- J(w,b)를 최소화 시키는 w, b를 찾아야함
- Gradient Descent: 가로축이 공간 매개 변수 w와 b를 나타냄
- 볼록 함수 - 초기화해서 똑같은 지점에 도달함
- 기울기 강하 - 시작점에서 시작해서 가장 기울기가 높은 내리막길 방향으로 이동





$$X = X - lr * \frac{d}{dX} f(X)$$

Where,

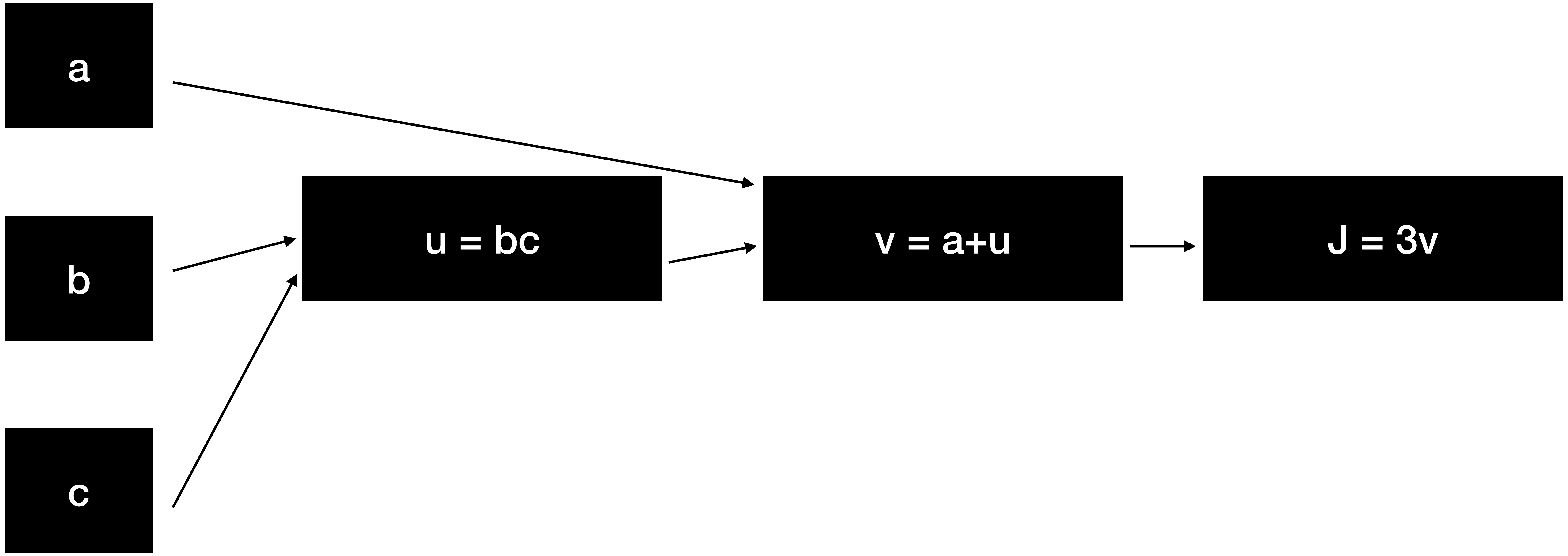
X = *input*

$F(X)$ = *output based on* X

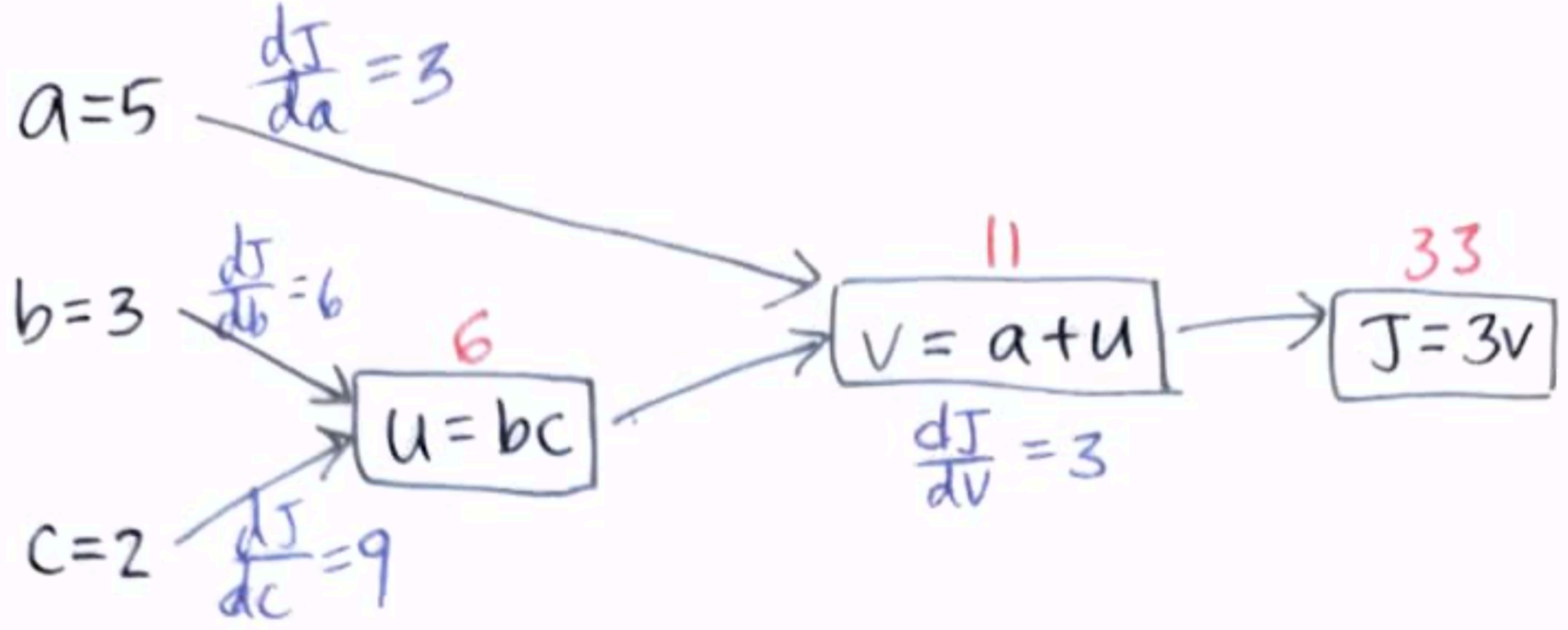
lr = *learning rate*

Computation Graph

- 신경망 계산법 - forward propagation step
- 신경망의 결과값 계산, backward propagation step을 통해 기울기나 derivative를 계산
- J -> 최소화시키고 싶은 비용함수
- Ex) J(a,b,c) = 3(a+bc)
- u=bc
- v=a+u
- J=3v



Derivatives with a Computation Graph



$$\frac{dJ}{dv} = 3$$

$$\frac{dJ}{da} = 3 = \frac{dJ}{dV} \cdot \frac{dv}{da}$$

$$3 \times 1$$

$$\frac{dJ}{du} = \frac{dJ}{dV} \cdot \frac{dv}{du} = 3 \times 1 = 3$$

$$\frac{dJ}{db} = \frac{dJ}{du} \cdot \frac{du}{db} = 3 \times 2 = 6$$

$$\frac{dJ}{dc} = \frac{dJ}{du} \cdot \frac{du}{dc} = 3 \times 3 = 9$$