9월 23일 (수요일)

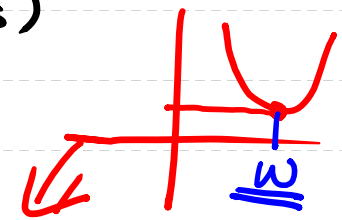
- machine learning을 구현하기 위해 Regression을 이용함

→ Classical Linear Regression Model

$\hat{y} = \beta_0 + \sum_{i=1}^{p} \beta_i x_i$ (• 독립변수 1개 : 단순선형회귀 (simple linear regression)
• 독립변수 2개이상 : 다중선형회귀 (Multiple " "))

$\hat{y} = \beta_0 + \beta_1 x_1 \rightarrow y = ax + b \Rightarrow y = wx + b$ (weight, bias)

Hypothesis



- 오차(error)를 줄이는 방향으로 w, b 찾아가면 될거 같아!!

→ 최소제곱법을 이용해서 오차를 계산. loss function, $E(w,b) = \frac{1}{n}\sum_{i=1}^{n}[t_i - (wx_i+b)]^2$

→ Gradient Descent algorithm을 이용해서 최적의 w, b를 찾아가는 과정을 거쳐야 되요!!

$$w' = w - \alpha \frac{\partial E(w,b)}{\partial w}, \quad b' = b - \alpha \frac{\partial E(w,b)}{\partial b}$$

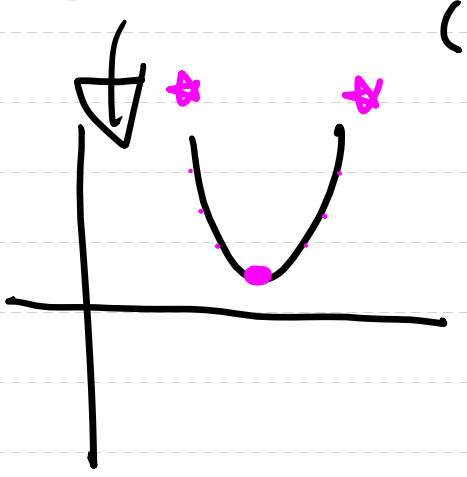
⇒ 계산된 loss function의 값이 최소가 될때까지 반복.

α: learning rate (학습률) $\alpha = 0.001$

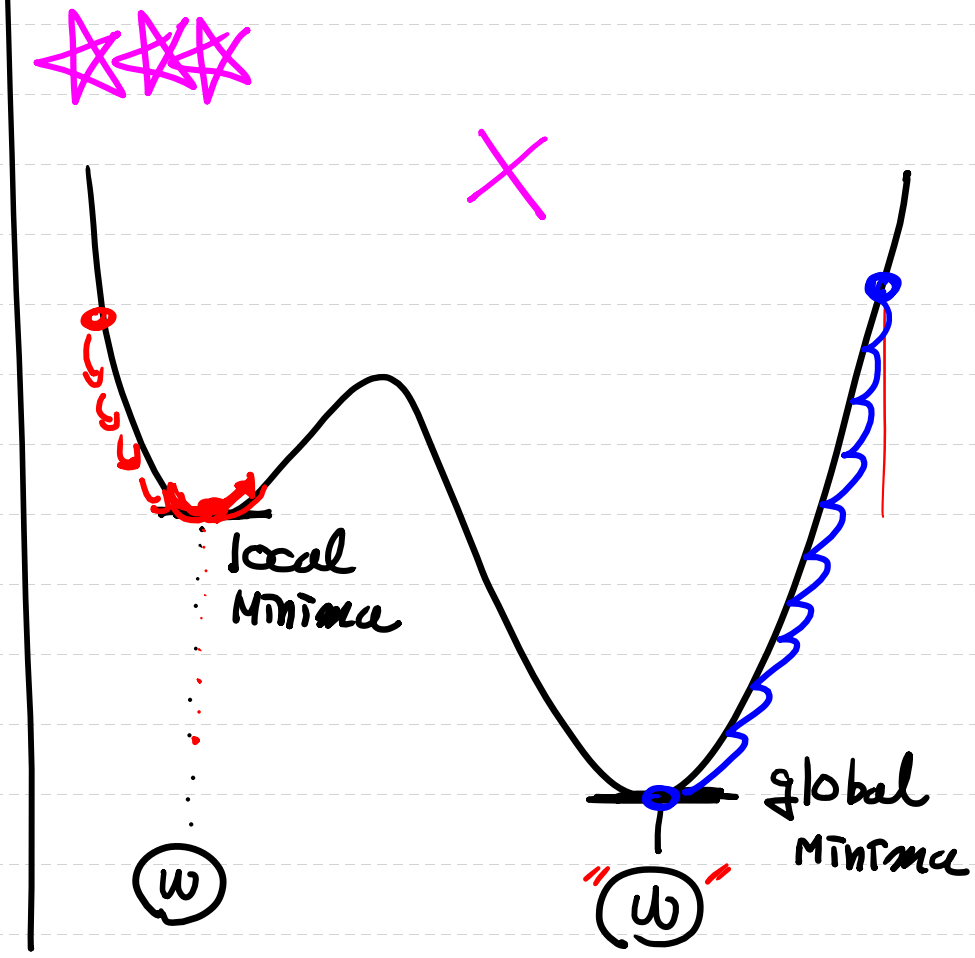
Gradient Descent Algorithm
(경사하강법)

이 알고리즘으로 최적의 w, b 찾으려면

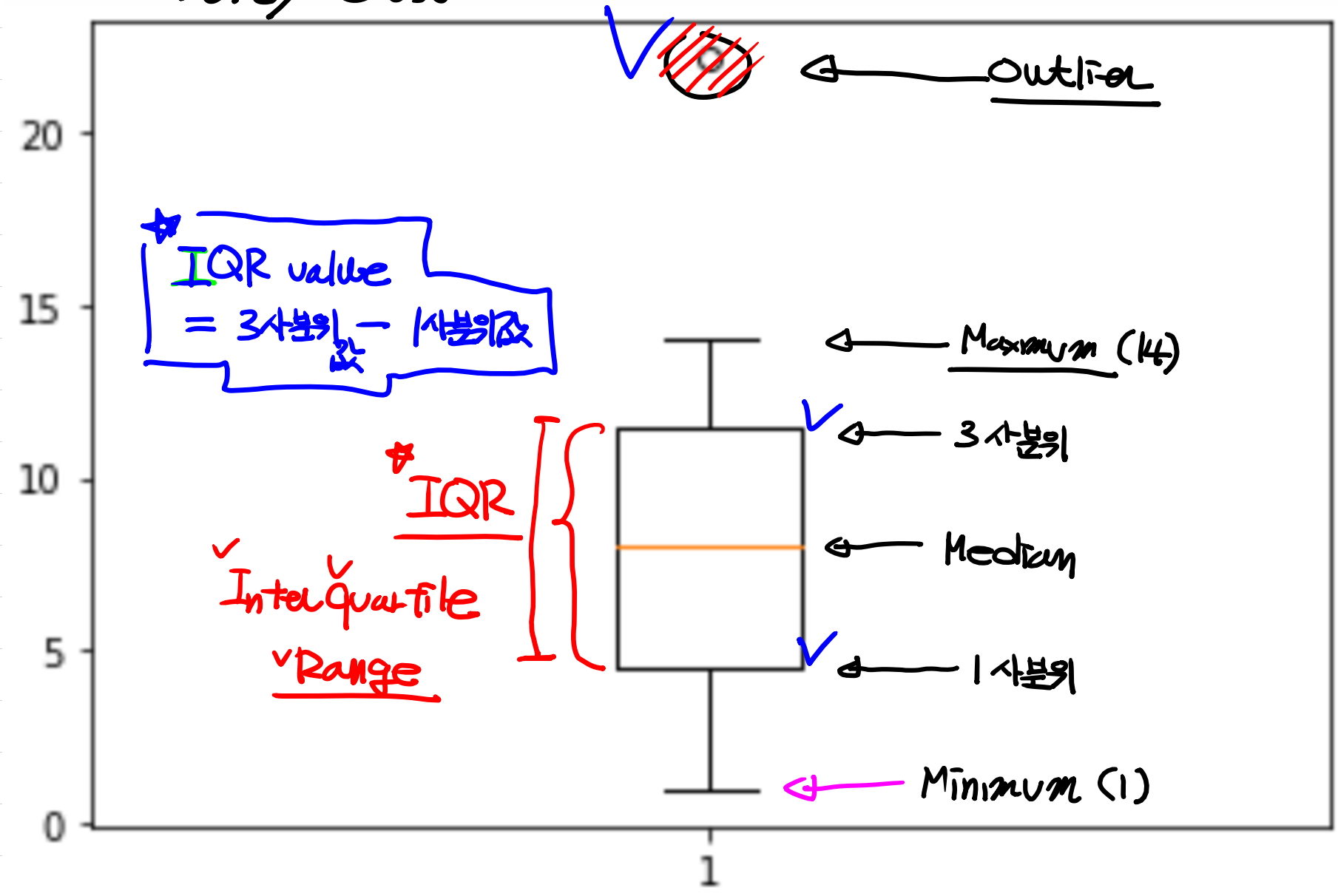
loss function이 "convex function" 형태를 (볼록) 가져야 해요.



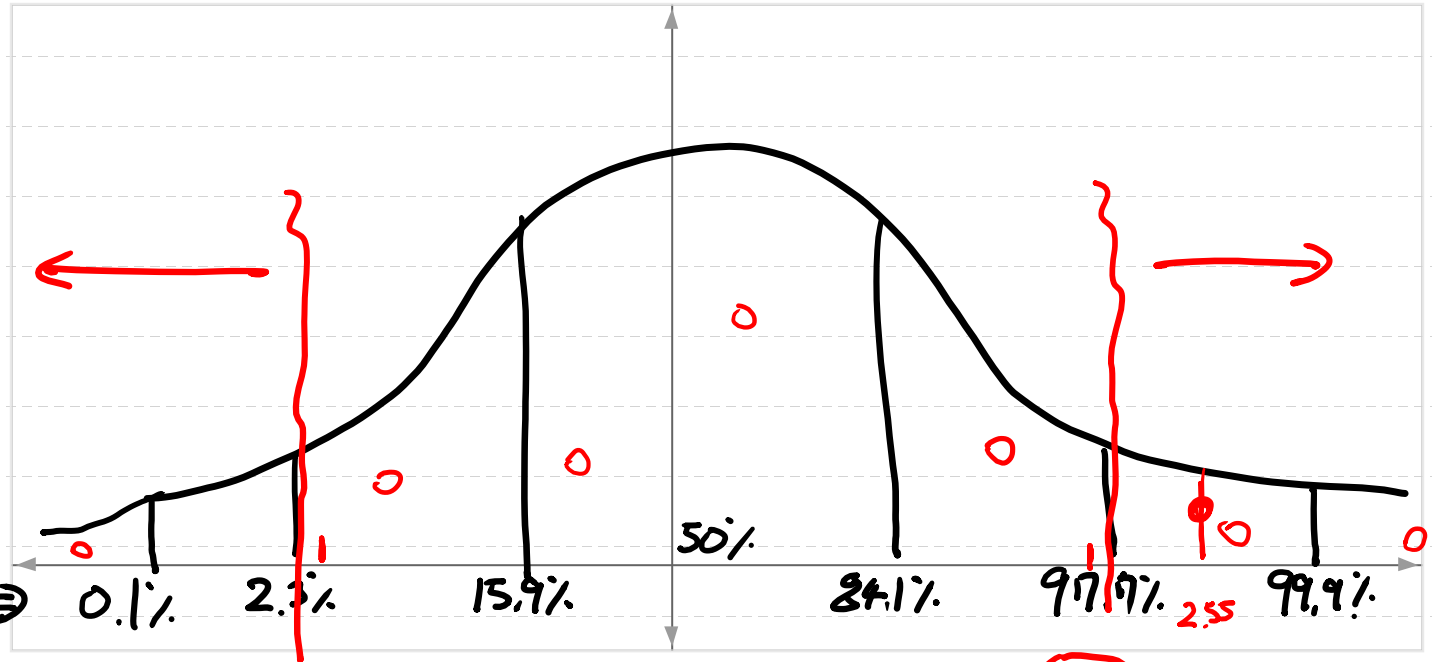
만약 loss function의 모양이



Tukey Outlier



- Z-score (Z점수) : 정규분포 이용 ⇒ 두개 사용되는 데이터를 정규분포로 만들고 각각의 데이터가 표준 편차상 어느곳에 위치하는지 알려주는 값..



Percentile ⇒

Z-Score ⇒ -3 -2 -1 0 +1 +2 +3

1.8 "2" 이하에서 optimal