⟨Gradient Descent Algorithm⟩

- Until now, manually try to read the graph for the best value of w, b

- Gradient Descent Algorithm : automatically finding the values of parameters w, b
  that makes best fit line that minimizes cost function

Gradient Descent algorithm
can be applied not just a cost func
for linear regression, but any func

⟨Gradient Descent Outline⟩

④ have some function $J(w,b)$,

goal: $\min\limits_{w,b} J(w,b)$

① Start with some w, b as the initial guess (e.g. $w=0, b=0$)

② keep changing w, b to reduce $J(w,b)$ ⇒ until hopefully settle at or near a minimum

⟨Implementing Gradient Descent Algorithm⟩

"Repeat until convergence" (= "local minimum" where w, b no longer change / reach the point at)

[updating new $\underbrace{w}_{\text{new}} \underset{\text{assignment}}{:=} \underbrace{w}_{\text{previous}} - \alpha \frac{\partial}{\partial w} J(w,b)$] || [updating new $\underbrace{b}_{\text{new}} := \underbrace{b}_{\text{previous}} - \alpha \frac{\partial}{\partial b} J(w,b)$]

- α : learning rate = basically controls how big of a step to downhill (size)

- $\frac{\partial}{\partial w} J(w,b)$: derivative of the cost function J = which direction to take a step to downhill (direction)
  (편미분)

* Important detail : Simultaneously update w, b = update both parameteres w, b at the same time

[Correct implementation : simultaneous update]

$temp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$

$temp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$ pre-updated w

$w = temp\_w$

$b = temp\_b$

[Incorrect implementation]

$temp\_w = w - \alpha \frac{\partial}{\partial w} J(w,b)$

$w = temp\_w$

$temp\_b = b - \alpha \frac{\partial}{\partial b} J(w,b)$ update w before calculating the new value for other parameter b

$b = temp\_b$