<Checking Gradient Descent for Convergence>

- Gradient Descent : repeat {

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

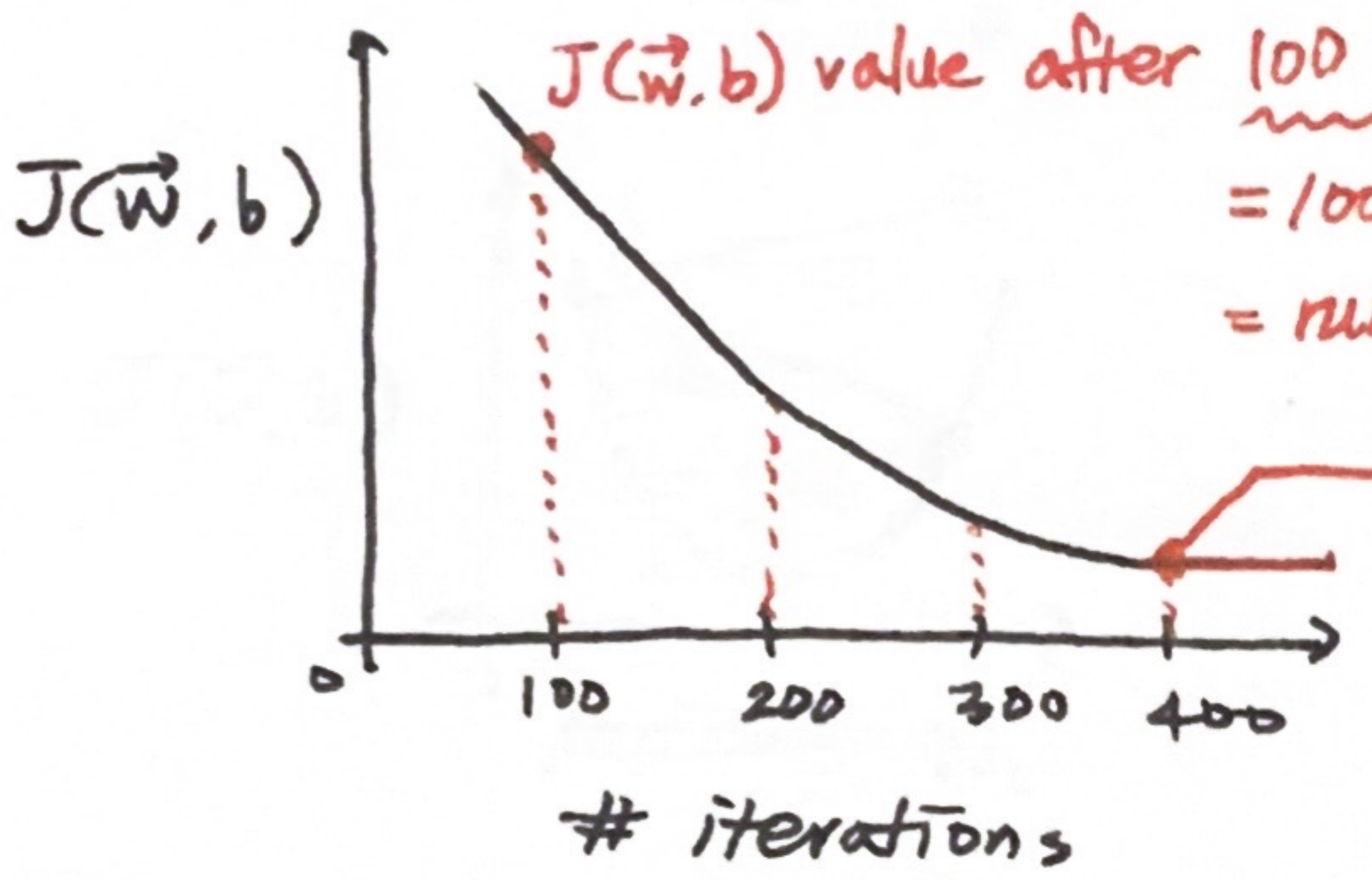
$$b := b - \alpha \frac{\partial}{\partial b} J(\vec{w}, b)$$

}

objective :

$$\underset{\vec{w}, b}{\text{minimize}}\ J(\vec{w}, b)$$

<learning curve>



$J(\vec{w}, b)$ value after 100 iterations

= 100 simultaneous update of w, b

= run 100 gradient descent calculation

→ "gradient descent has converged" = "training is done"

iterations = each simultaneous update of parameter w, b

like this

* If gradient descent is working properly ⇒ $J(\vec{w}, b)$ decrease after every iteration

∴ Looking at "learning curve" graph helps to see how cost J changes after each iteration of gradient descent

⊕ when the model's training is done

* If cost J increases after one iteration ⇒ ① alpha is chosen poorly (alpha is too large)

② there could be bug in the code

<Automatic Convergence Test : using "Epsilon">

- another way to decide when model is done training

ex)

Let ε (epsilon) $= 10^{-3}$ ⇒ if $J(\vec{w}, b)$ decreases by less than ϵ in one iteration

⇒ "declare convergence"

(특정 설정값 (epsilon) 보다 더 작은 값으로 cost J가 감소했다면, convergence에 도달했다고 판단하는 것)