MACHINE LEARNING

By - Andrew Ng

Sun 10 Jul

Week 1

ML

Supervised

"right answers" are provided

- Regression
- Classification

Unsupervised

small no. of output

- Clustering

Terminology :

$x \rightarrow$ Input / Feature

$y \rightarrow$ output / Target

$x \longrightarrow \boxed{f} \longrightarrow \hat{y}$ → "y-hat"

f: model/funcⁿ

$\hat{y}$: estimated

$y \rightarrow$ target

$m \rightarrow$ Total no. of training set

$(x^{(i)}, y^{(i)}) \rightarrow i^{th}$ set

$$f_{w,b}(x^{(i)}) = wx + b$$

: w, b are parameters

Cost function :

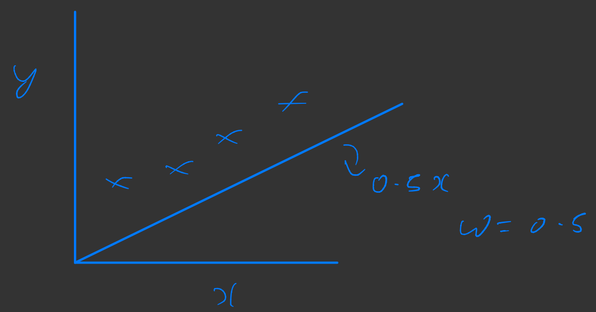
$$\boxed{J_{(w,b)} = \frac{1}{2m} \sum_{i=1}^{m} \left(\hat{y}^{(i)} - y^{(i)} \right)^2}$$

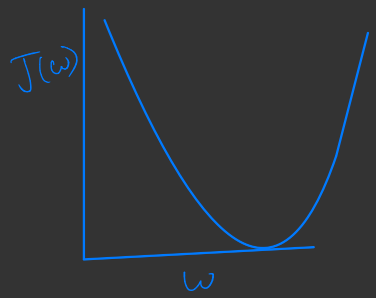
Or

$$\boxed{J_{(w,b)} = \frac{1}{2m} \sum_{i=1}^{m} \left(f_{(w,b)}\left(x^{(i)}\right) - y^{(i)} \right)^2}$$

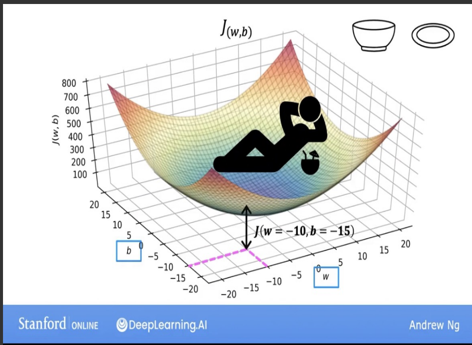
Let $b = 0 \Rightarrow f(x) = wx$

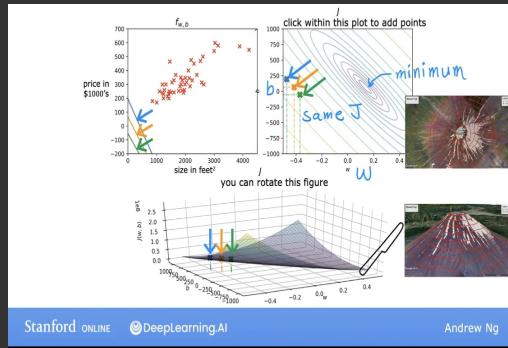
$\therefore$





Now, if $b \neq 0$



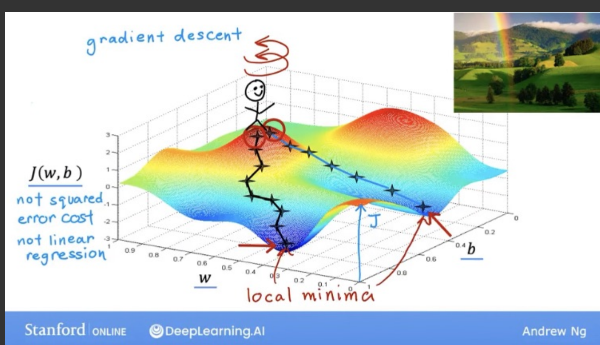


GRADIENT Descent

$$\min_{w,b} J(w,b) \qquad \min_{w_1, w_2, \ldots, w_n, b} J(w_1, w_2, w_3, \ldots, w_n, b)$$

Start $(w=0, b=0)$

may have one or more minimum.



Gradient Descent algo

→ Learning rate → α size of step

$$w = w - \alpha \frac{d}{dw} J(w,b)$$

Coding equal sign {assignment}

Simultaneously update w & b

$$b = b - \alpha \frac{d}{db} J(w,b)$$

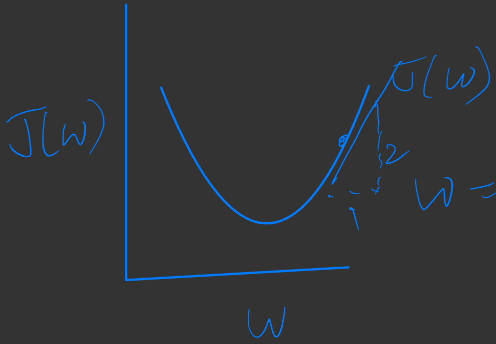
⇒

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

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

$$w = temp\_w$$

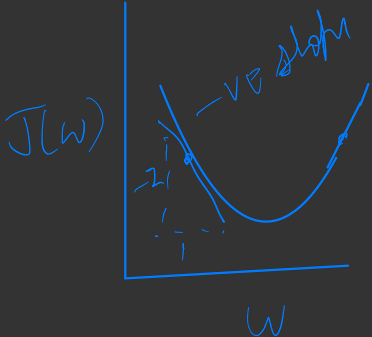
$$b = temp\_b$$



$$w = w - \alpha \boxed{\frac{d}{dw} J(w)}$$

> 0

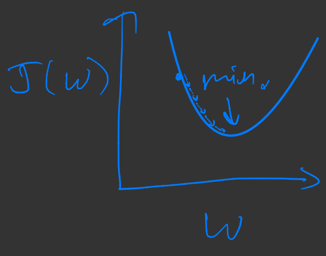
$$w = w - \alpha \cdot +ve\ no.$$



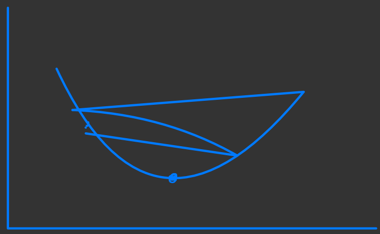
$$\Rightarrow w = w - \alpha \cdot (-ve\ no.)$$

If α is too small....

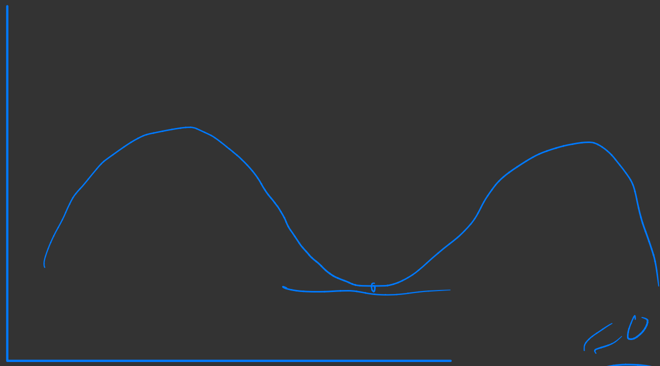
gradient descent may be too slow



If α is too large . . .

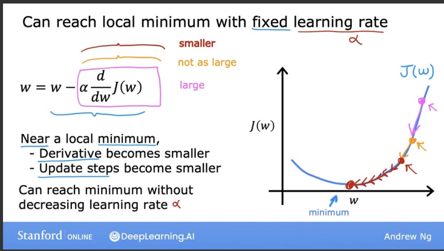


may overshoot & may never reach min.
Or never converge or even diverge.



$$w = w - \alpha \overbrace{\frac{dJ(w)}{dw}}^{<0}$$

$$w = w$$



Linear regression model

$$f_{w,b}(x) = wx + b$$

Cost funⁿ

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2$$

Gradient descent algo:

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

$$= w - \alpha \frac{\partial}{\partial w}\left(\frac{1}{2m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)^2\right)$$

$$= w - \alpha \frac{\partial}{\partial w}\left(\frac{1}{2m}\sum_{i=1}^{m}\left(wx^{(i)} + b - y^{(i)}\right)^2\right)$$

$$= w - \frac{\alpha}{\cancel{2}m}\left[\cancel{2}\sum_{i=1}^{m} x^{(i)}\left(wx^{(i)} + b - y^{(i)}\right)\right]$$

$$\boxed{w = w - \frac{\alpha}{m}\sum_{i=1}^{m} x^{(i)}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)}$$

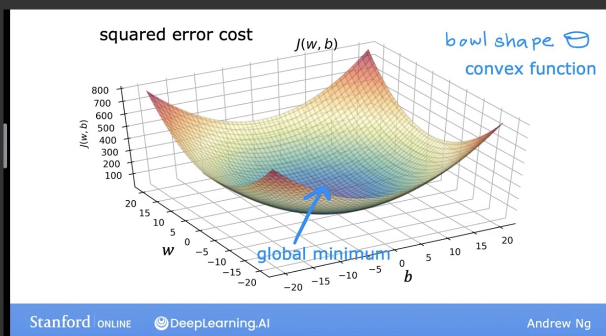
Similarly,

$$b = b - \alpha \frac{\delta(J_{w,b})}{\delta b}$$

$$\boxed{b = b - \frac{\alpha}{m}\sum_{i=1}^{m}\left(f_{w,b}(x^{(i)}) - y^{(i)}\right)}$$

→ convex fun^n.

Square error cost will have one single global minimum.



Batch gradient descent: uses all training examples.