* Gradient Descent

Have some function $J(w,b)$

Want $\min_{w,b} J(w,b)$

Outline:

(Random initialization)

- Start with some w, b (Set $w=0$, $b=0$)
- Keep changing w, b to reduce $J(w,b)$
- until we settle at or near a minimum

→ **Repeat till Convergence**

α → learning rate (usually $0 \to 1$)

↳ represents the control of steps of weights update

$$w = w - \alpha \frac{\partial J}{\partial w}$$

$$b = b - \alpha \frac{\partial J}{\partial b}$$

(Simultaneously update w and b)

* Gradient Descent:

→ general algorithm to optimize nearly any function (find Min.)

→ It is an algorithm to find the parameters θ_0, θ_1 of the cost function $J(\theta_0, \theta_1)$ that makes the cost function as minimum as possible.

→ The idea behind this algorithm is by getting the derivative of the cost function (tangent line to the function) and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent.

The gradient descent algorithm: → Repeat until convergence

$$\{ \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

where $j = 0, 1$ represents the feature index number

$\}$ α = learning rate (the size of each step)

At each iteration, (θ_0, θ_1) must be updated simulataneously as:

$$temp0 = \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$temp1 = \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

then:

$\theta_0 = temp0$

$\theta_1 = temp1$

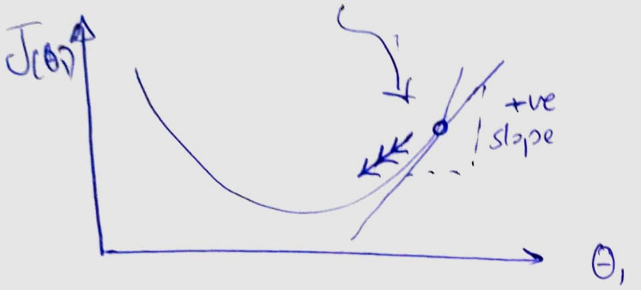
Note

- If α is very small, it causes small steps
- If α is very large, it causes Large steps

Let $\theta_0 = 0$ for simplicity :

$$\theta_1 = \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$ [α is always positive]

If we started at θ_1 larger than that makes $J(\theta_1)$ minimum



the derivative $\frac{\partial}{\partial \theta_1} J(\theta_1)$, which is the slope of tangent line at this point, will be positive

So, $\theta_1 = \theta_1 -$ (positive value)

So, θ_1 will decrease

If we started at θ_1 smaller than that make $J(\theta_1)$ minimum

the derivative $\frac{\partial}{\partial \theta_1}$ will be negative

So, $\theta_1 = \theta_1 -$ (negative value)

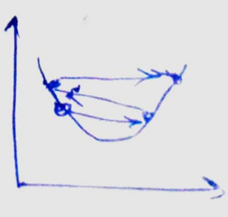
So, θ_1 will increase

So, θ_1 will eventually converges to the minimum value of $J(\theta_1)$

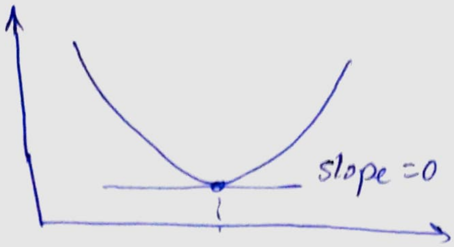
Note we should adjust our parameter (α) to ensure that the gradient descent will converge in a reasonable time.

- If α is too small : gradient descent can be slow

- If α is too large : gradient descent can overshoot the minimum. It may fail to converge, or may even diverge



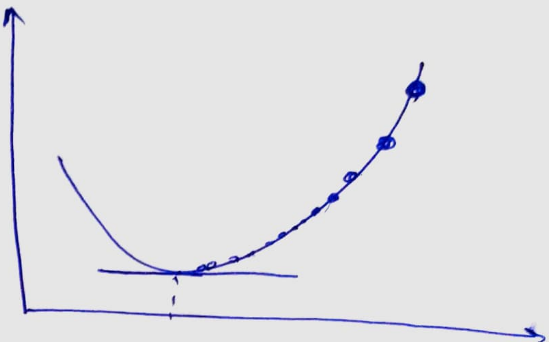
* What if we started at $J(\theta_1)$ is minimum already?



the $\dfrac{d}{d\theta_1} J(\theta_1) = 0$

so, θ_1 will be unchanged

* The steps will automatically be smaller because the slope decreases till it equals to zero



• Gradient Descent for Linear Regression:

Gradient Descent Algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

for $j = 0, 1$

}

Linear Regression Model

$$h_\theta(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$= \frac{1}{2m} \sum_{i=1}^{m} \left[(\theta_0 + \theta_1 x^{(i)}) - y^{(i)} \right]^2$$

$j = 0$: $\dfrac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \dfrac{1}{m} \displaystyle\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$ b

$j = 1$: $\dfrac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \dfrac{1}{m} \displaystyle\sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$ w

So, Gradient descent for linear regression:

repeat until convergence{

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)}$$

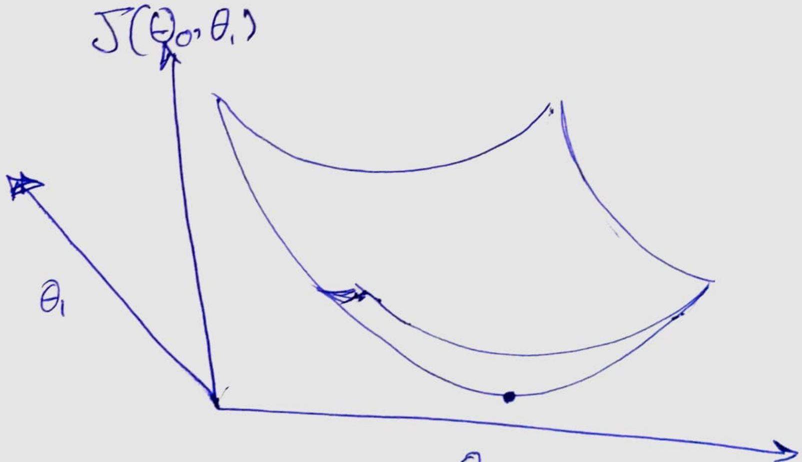
} update θ_0, θ_1 Simulataneously

- The point of all this that if we start with a guess for our hypothesis (a guess of θ_0, θ_1) and then we repeatedly apply these gradient descent equations, our hypothesis will become more and more accurate.

Convex function

↓

one global minimum



→ This gradient descent called "Batch" Gradient Descent as Each step of gradient descent uses all the training examples

→ There are other versions of gradient descent (stochastic, — mini-batch)