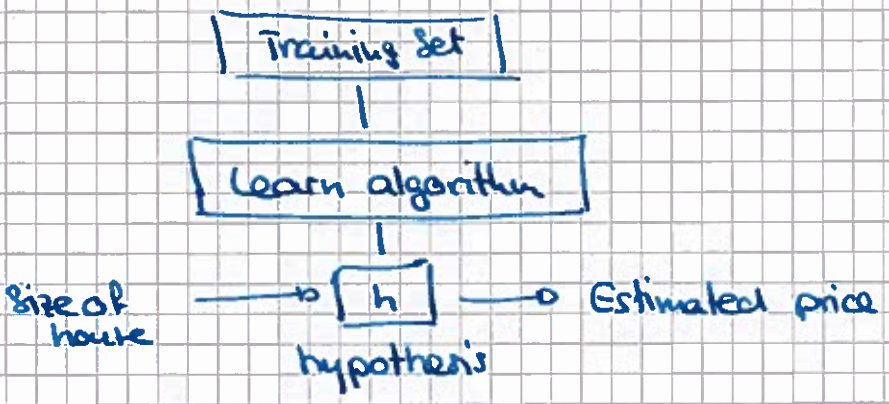
Machine learning
- Stanford University

2. Linear Regression

Notation:

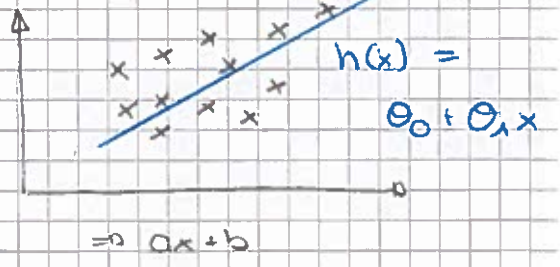
- m - number of training examples
- x - input variable / feature
- y - output variable
- (x, y) single training example
- $(x^{(i)}, y^{(i)})$ specific example



Representation of h

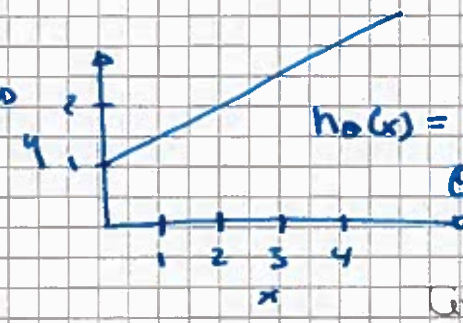
$\Rightarrow h_\theta(x) = \theta_0 + \theta_1 x$ $\Rightarrow$

to shorthand $h(x)$

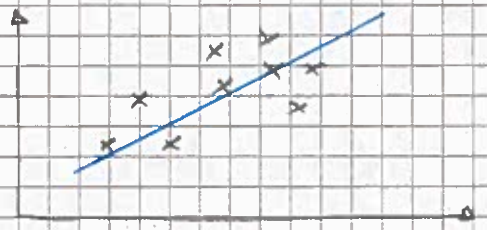


Linear Regression with one variable
Univariate Linear Regression

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



Linear function
$ax+b$



Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for training example (x,y)

$$\underset{\theta_0,\theta_1}{\text{minimize}} \quad \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

(m = # of training examples)

$h_\theta(x^{(i)})$: Predicted value

$y^{(i)}$: real value

$h(x^{(i)}) = \theta_0 + \theta_1 x$ ⇒ given by the data

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

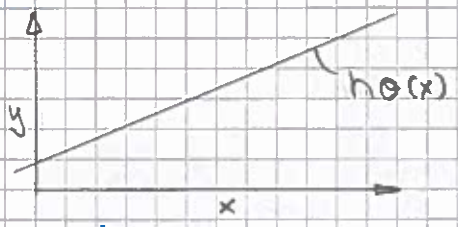
minimize $J(\theta_0,\theta_1)$ ⇒ Cost function

⇒ Squared error function

Summary:

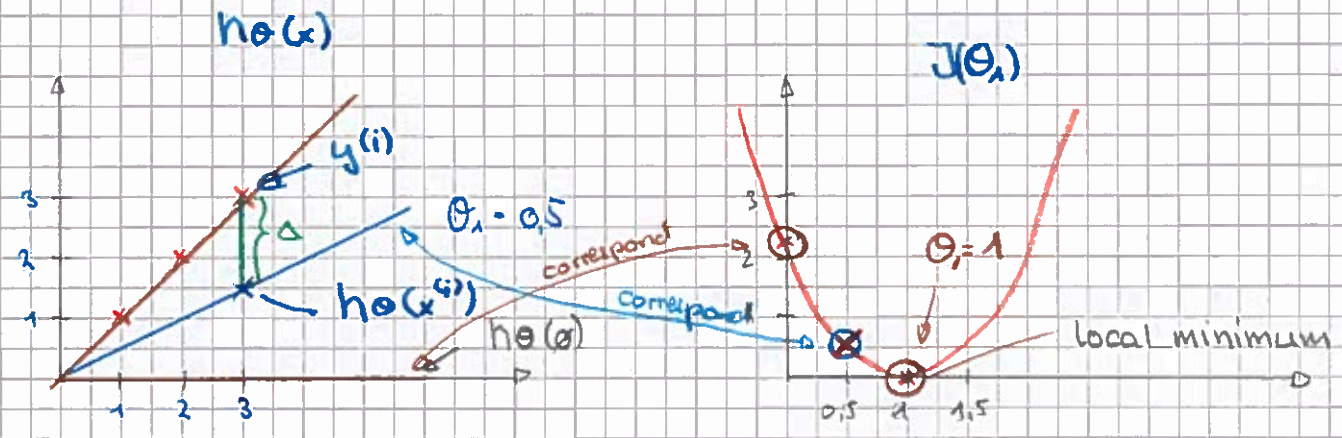
Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1



Cost function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$



$$J(0,5) = \frac{1}{2m} \sum_{i=1}^{m} = 0 \left[(0,5-1)^2 + (1-2)^2 + (1,5-3)^2 \right]$$

$$= 0 \quad \frac{1}{2 \cdot 3} \cdot 3,5 = 0,58$$

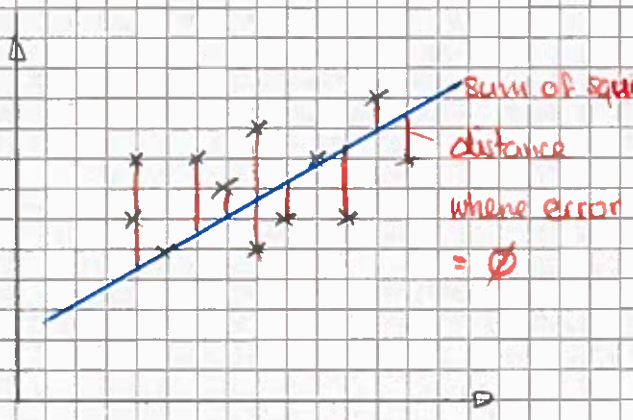
Δ difference between predicted and effective value

minimize $J(\theta_1)$

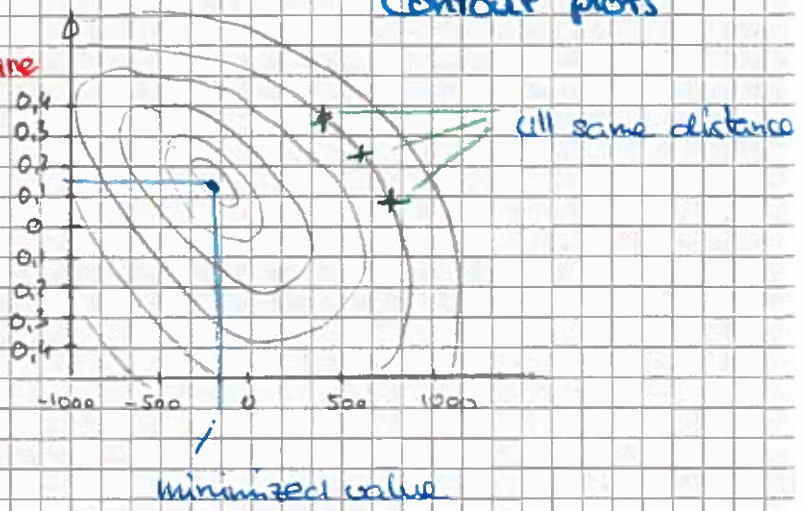
$$J(0) \Rightarrow \frac{1}{2m} \cdot \left[(0-1)^2 + (0-2)^2 + (0-3)^2 \right] = \frac{1 \cdot 14}{2 \cdot 3} = \sim 2,3$$

$$J(1) = 0 \quad \frac{1}{2m} \cdot \left[(1-1)^2 + (2-2)^2 + (3-3)^2 \right] = \frac{0}{6} = 0$$

Cost function intuition II



Contour plots



Gradient descent algorithm

⇒ want to get the local/global minimum

$$\min J(\theta_0, \theta_1)$$

⇒ Gradient descent applies to more general functions

$$J(\theta_0, \theta_1, \dots, \theta_n) \Rightarrow \min J(\theta_0, \theta_1, \dots, \theta_n)$$

Repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial_j} J(\theta_0, \theta_1) \quad (\text{for } j=0,\ j=1)$$

}

⇒ simultaniously update

$$\text{temp}0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

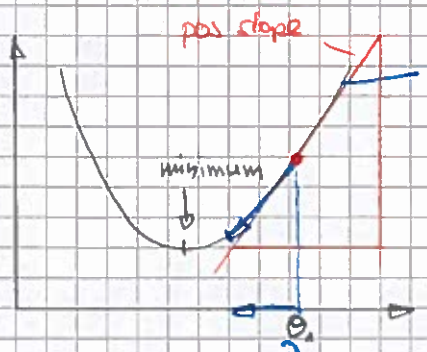
$$\text{temp}1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp}0$$

$$\theta_1 := \text{temp}1$$

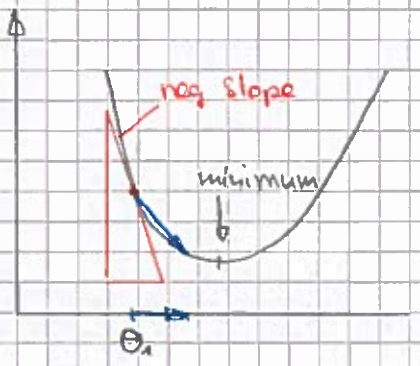
α = learning rate

derivative term



$J(\theta_1) = 0$

$\theta_1 \in \mathbb{R}$



$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} \cdot J(\theta_1)$$

$$\geq 0$$

$$\theta_1 := \theta_1 - \alpha \cdot (\text{pos. number})$$

always pos.

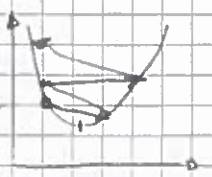
$$\frac{\partial}{\partial \theta_1} \cdot J(\theta_1)$$

$$\leq 0$$

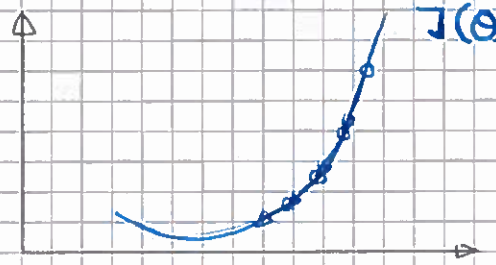
$$\theta_1 := \theta_1 - \alpha \cdot (\text{neg. number})$$

make sure that α is not too small or to large

- too small might take to many steps to converge
- too big might overshoot the min. and might never converge.



Gradient descent can converge to a local minimum, even with the learning rate α fixed.



$$\theta_1 := \theta_1 - \alpha \underbrace{\frac{\partial}{\partial \theta_1} \cdot J(\theta_1)}_{x}$$

term x will decrease therefore value will be smaller & smaller

As we approach local minimum, gradient descent will take smaller steps. No need to decrease α.

Linear Regression with One Variable

Apply gradient descent to linear regression function

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

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

$$\theta_0 := \quad j=0 : \frac{\partial}{\partial \theta_0} \cdot J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)$$

$$\theta_1 := \quad j=1 : \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}$$

Gradient descent algorithm

repeat until convergence {

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

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

}

always update simultaniously.