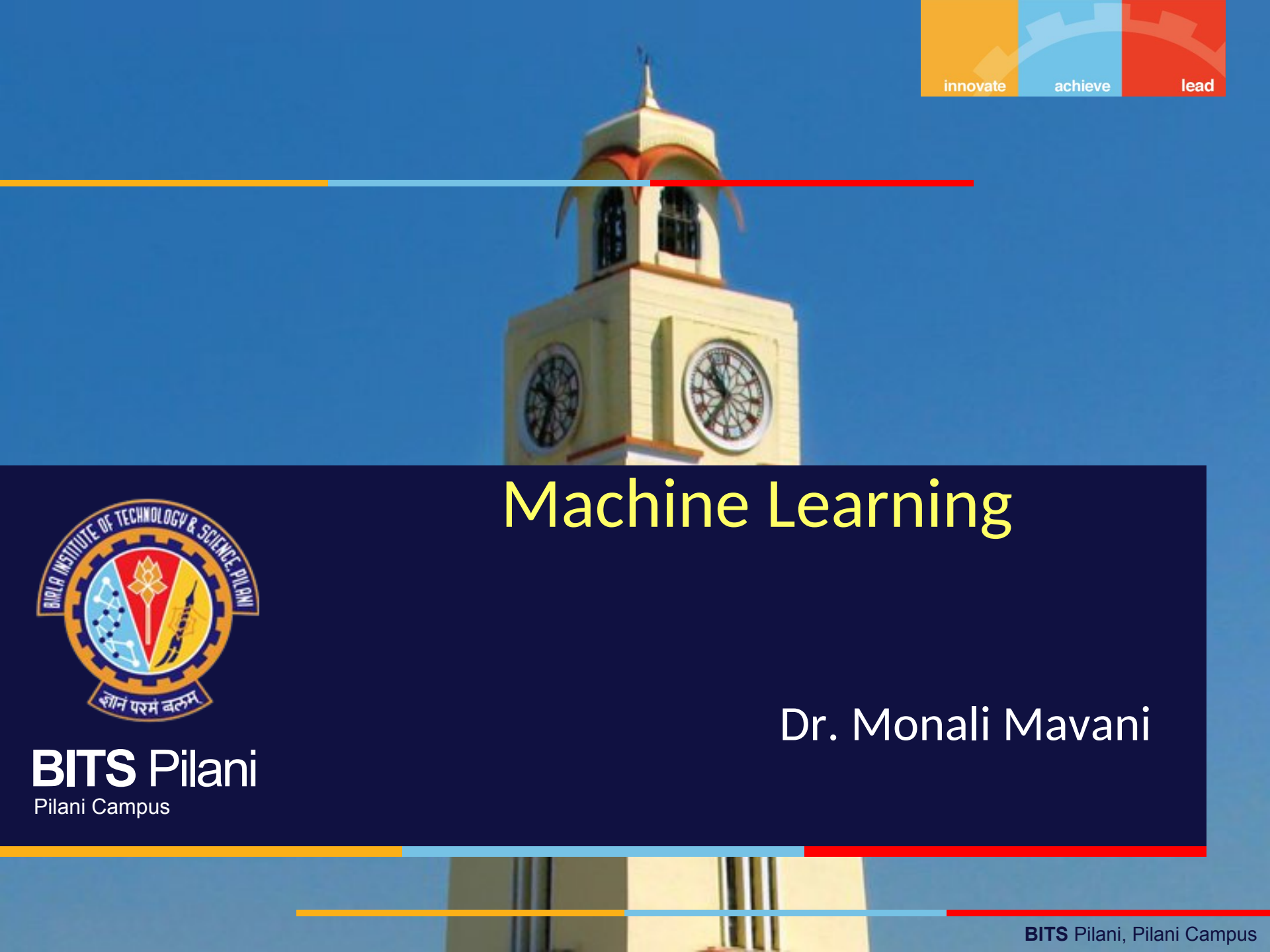


Machine Learning

Dr. Monali Mavani

BITS Pilani
Pilani Campus

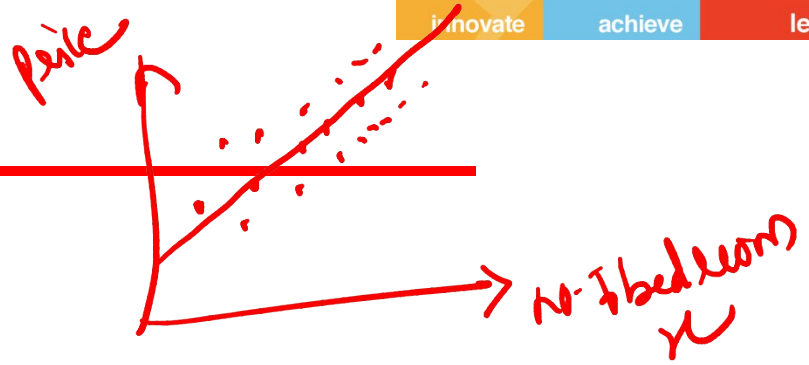
Machine Learning

innovate achieve lead

Disclaimer and Acknowledgement

- The content for these slides has been obtained from books and various other source on the Internet
- I here by acknowledge all the contributors for their material and inputs.
- I have provided source information wherever necessary
- I have added and modified the content to suit the requirements of the course

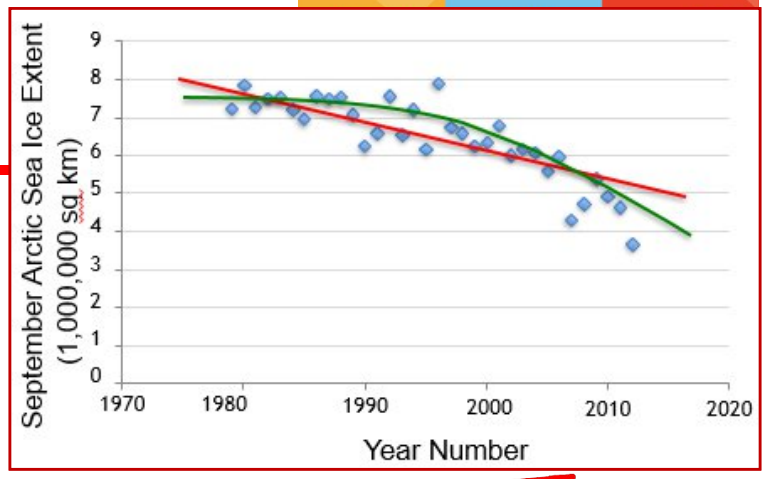
Session Content



Linear models for regression

- Linear Regression
- Linear Regression Cost Function
- Learning Model Parameters - Gradient Descent Optimization
- Learning Model Parameters – Closed Solution
- Linear Basis Function Models
- Quality of fit
- Evaluation metrics
-

Linear Regression



d = dimension of dataset
d = no. of features.

Given:

- Data $\boldsymbol{X} = \left\{ \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n)} \right\}$ where $\boldsymbol{x}^{(i)} \in \mathbb{R}^d$
- Corresponding labels $\boldsymbol{y} = \left\{ y^{(1)}, \ldots, y^{(n)} \right\}$ where $y^{(i)} \in \mathbb{R}$

Wish to learn f : X -> Y,

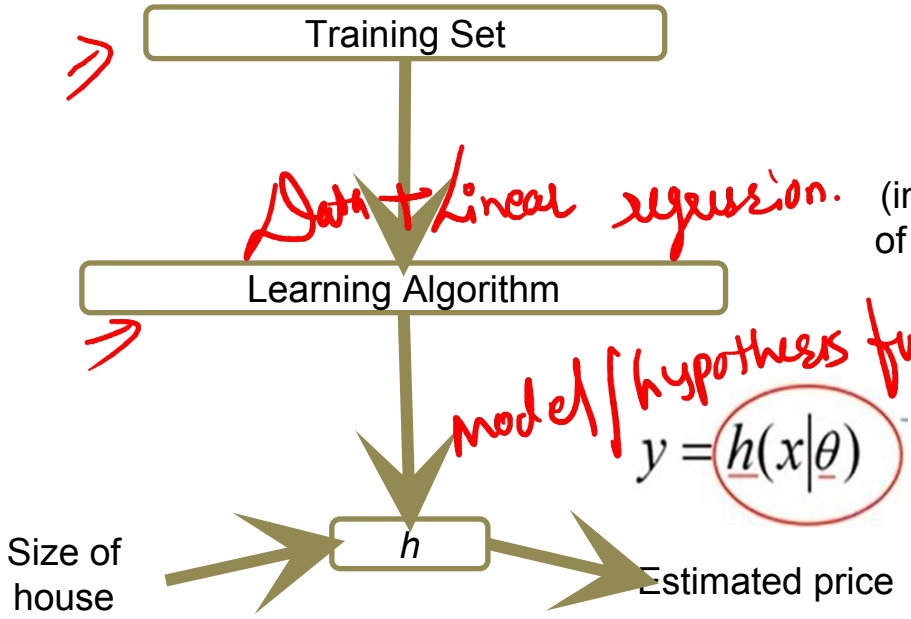
y = f(x) -> true function

x_1 x_2

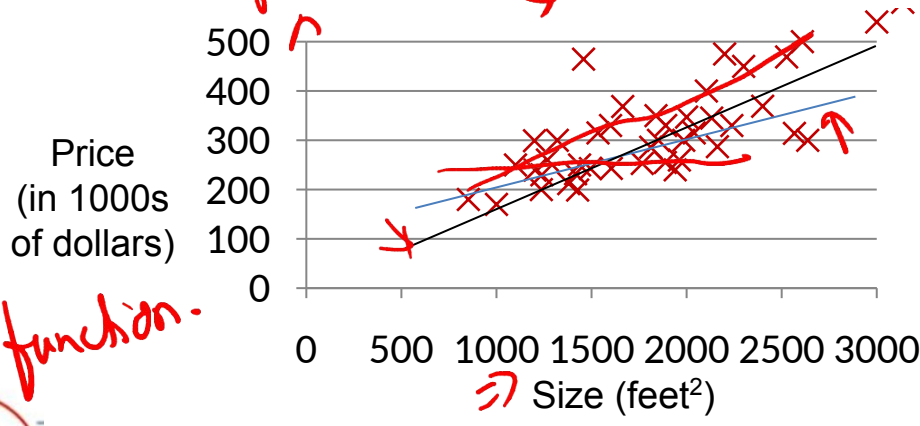
For each input observation $x^{(i)}$, a vector of features $[x_1, x_2, \ldots, x_n]$.

j^{th} feature of i^{th} training example: $x_j^{(i)}$ -> training example, feature

Machine Learning Process : Interpretation



$y = h(x|\theta)$



Size in feet² (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

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

θ_i's : Parameters

h(.) : Model

How to choose θ_i's ?

Types of Regression Models

Simple Linear regression model

(Education) x → y (Income)

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

Multiple Linear regression model

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

(Education) x_1

(Soft Skills) x_2

(Experience) x_3

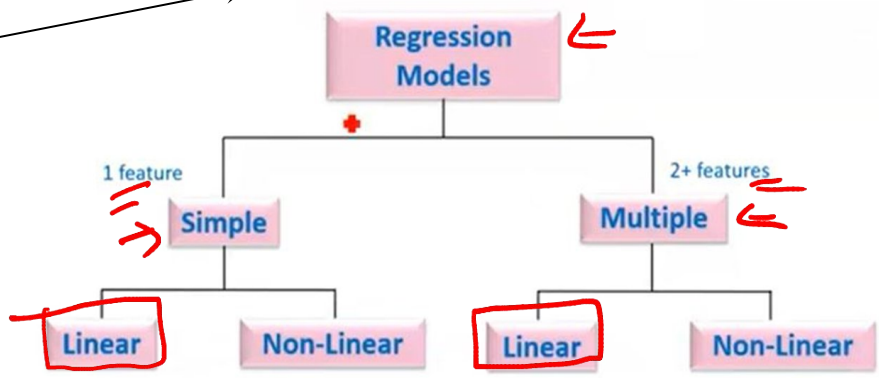
(Age) x_4

→ y (Income)

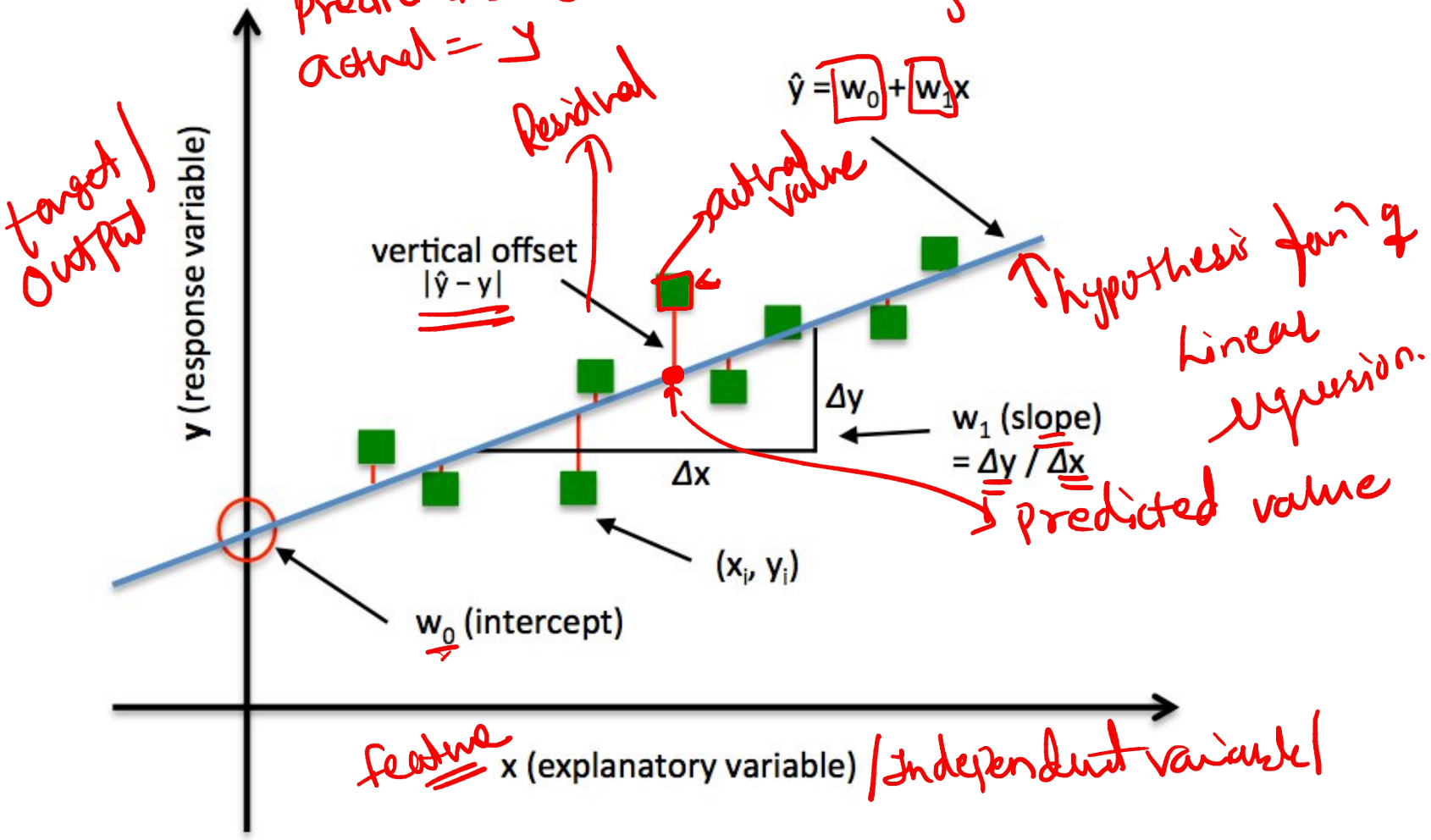
- Hypothesis:

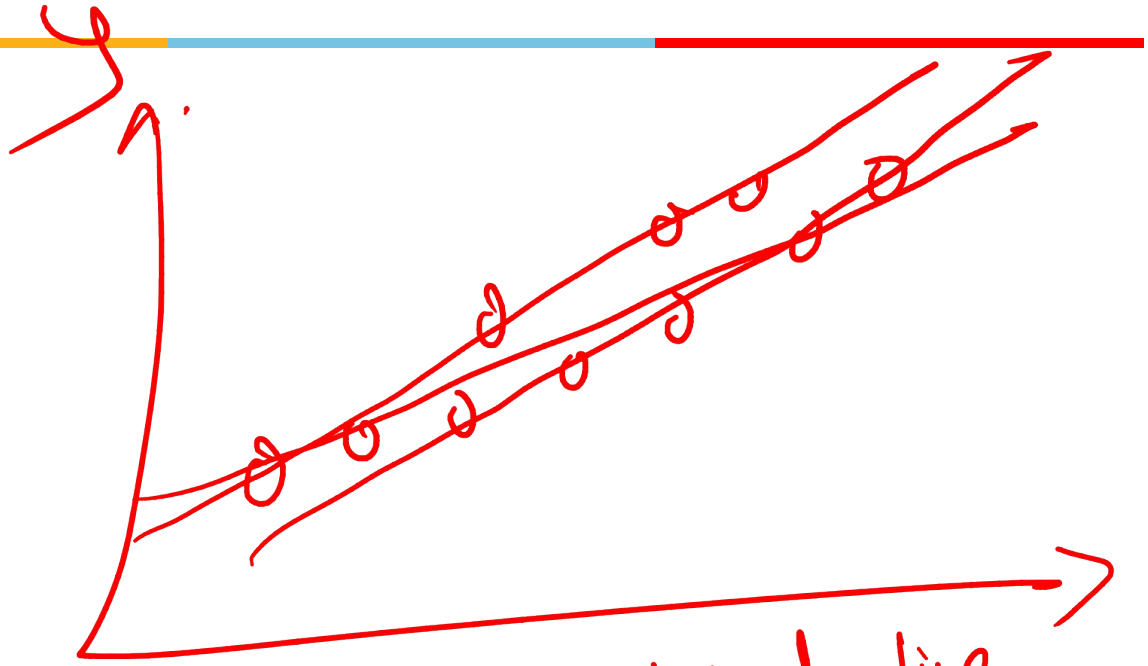
$$y = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \sum_{j=0}^{d} \theta_j x_j$$

Assume $x_0 = 1$



Least square regression line





best fitted line
least square regression line
$\sum (Pred - Act)^2$

Linear Regression – Hypothesis function

- Hypothesis:

$$y = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_d x_d = \sum_{j=0}^{d} \theta_j x_j$$

Assume $x_0 = 1$

bias term

features / dimensions

$\theta_1 \ldots \theta_d$ => weights

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

weights and bias => parameters

Learning the Model Parameters

Closed Form Solution Approach

Gradient Descent Approach

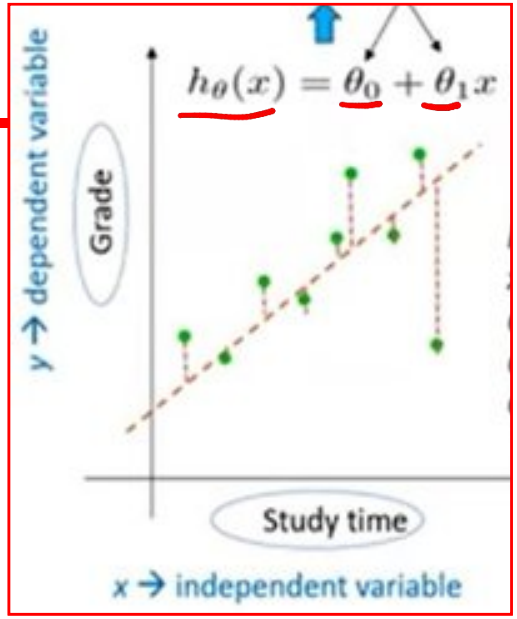
How to choose θ_i's ?

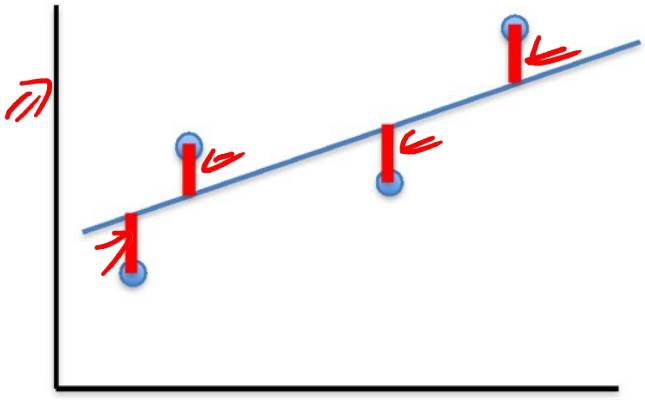
Linear Regression : Least Squares

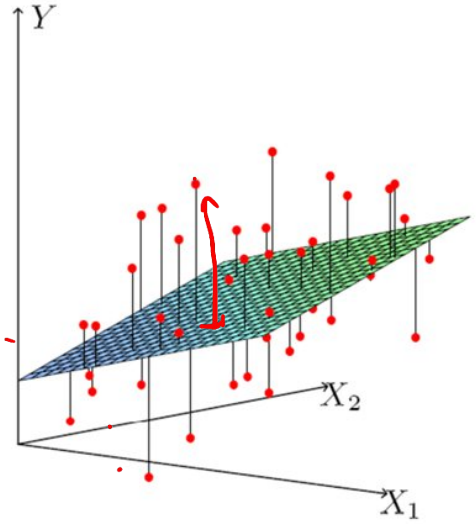
- Cost Function

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

- Fit by solving $\min_{\theta} J(\theta)$



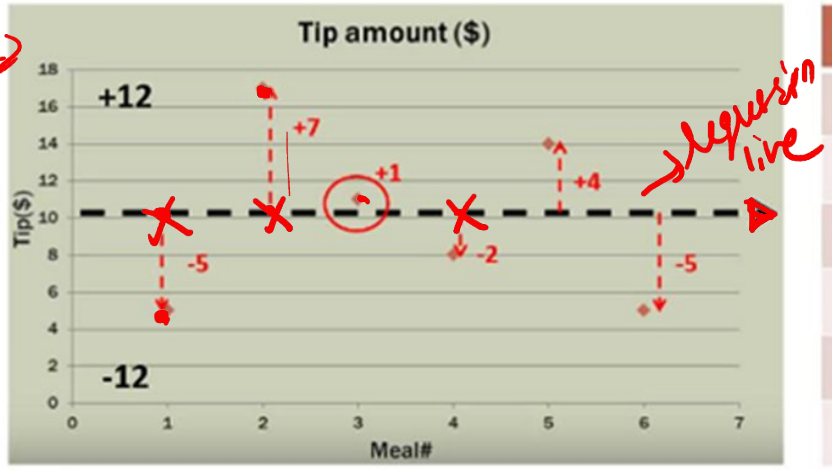




Notion of Cost Function

Meal #	Tip amount ($)
1	5.00
2	17.00
3	11.00
4	8.00
5	14.00
6	5.00

mean = $10



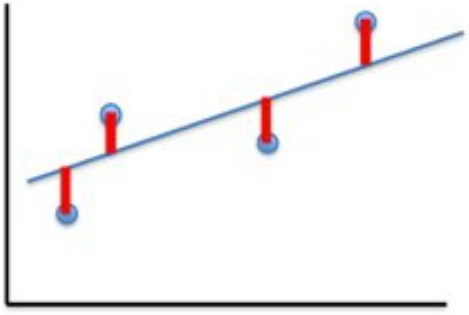
RESIDULAS (error): Distance from best fit line and the actual values

Meal#	Residual	Residual²
1	−5	25
2	+7	49
3	+1	1
4	−2	4
5	+4	16
6	−5	25

Sum of Squared Errors (SSE) = 120

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2$$

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2$$



Notion of Cost Function

Meal #	Tip amount ($)
1	5.00
2	17.00
3	11.00
4	8.00
5	14.00
6	5.00

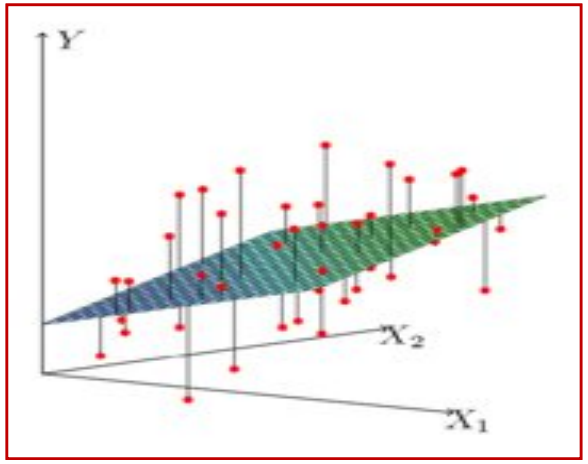
mean = $10

Meal	Total bill ($)	Tip amount ($)	$\hat{y}_i = 0.1462x - 0.8188$	$\hat{y}_i$ (predicted tip amount)
	x	y		
1	34	5	$\hat{y}_i = 0.1462(34) - 0.8188$	4.1505
2	108	17	$\hat{y}_i = 0.1462(108) - 0.8188$	14.9693
3	64	11	$\hat{y}_i = 0.1462(64) - 0.8188$	8.5365
4	88	8	$\hat{y}_i = 0.1462(88) - 0.8188$	12.0453
5	99	14	$\hat{y}_i = 0.1462(99) - 0.8188$	13.6535
6	51	5	$\hat{y}_i = 0.1462(51) - 0.8188$	6.6359
	$\bar{x} = 74$	$\bar{y} = 10$	Observed vs. Predicted	

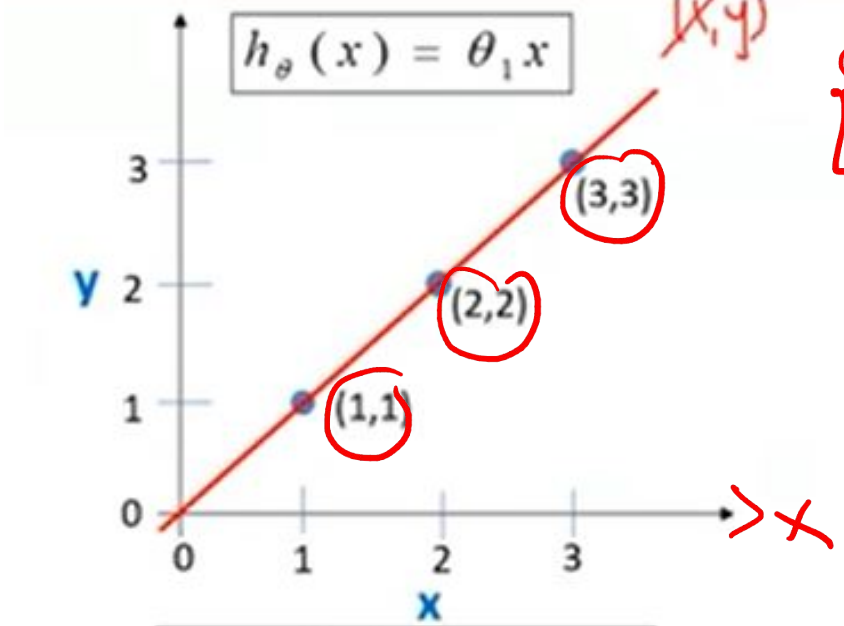
$y = \theta_0 + \theta_1 x$

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(\hat{y}^{(i)} - y^{(i)})^2$$

$$J(\theta) = \frac{1}{2n}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2$$

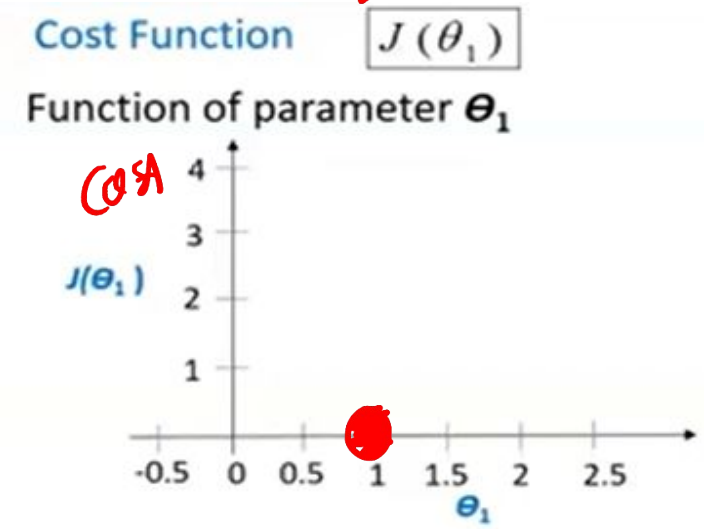


Understanding Cost Function



$\theta_0 = 0$

$\theta_1 = 1$



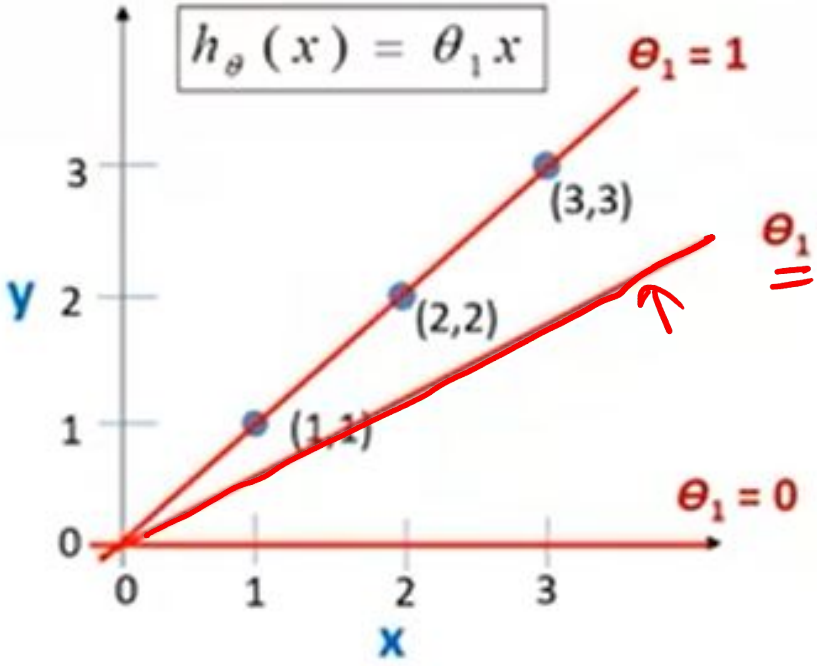
$$J(\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

$$= \frac{1}{2m}\sum_{i=1}^{m}(\theta_1 x^{(i)} - y^{(i)})^2$$

$$J(\theta_1) = \frac{1}{2*3}((1*1-1)^2 + (1*2-2)^2 + (1*3-3)^2)$$

$$= 0$$

Understanding Cost Function



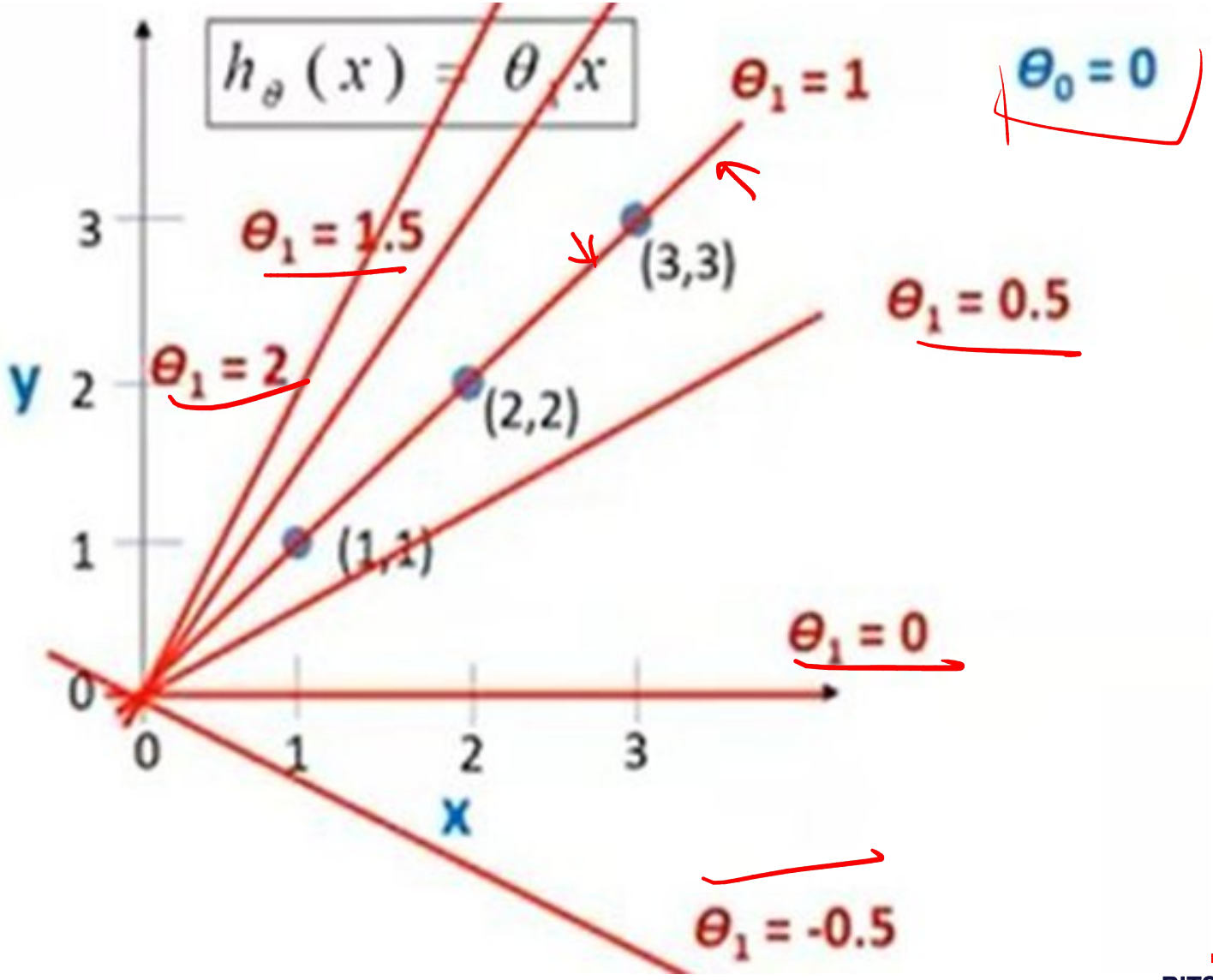
$\theta_0 = 0$

$h_\theta(x) = 0.5 \cdot x + 0$

$J(\theta_1) = J(0.5) = 0.58$

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

Understanding Cost Function



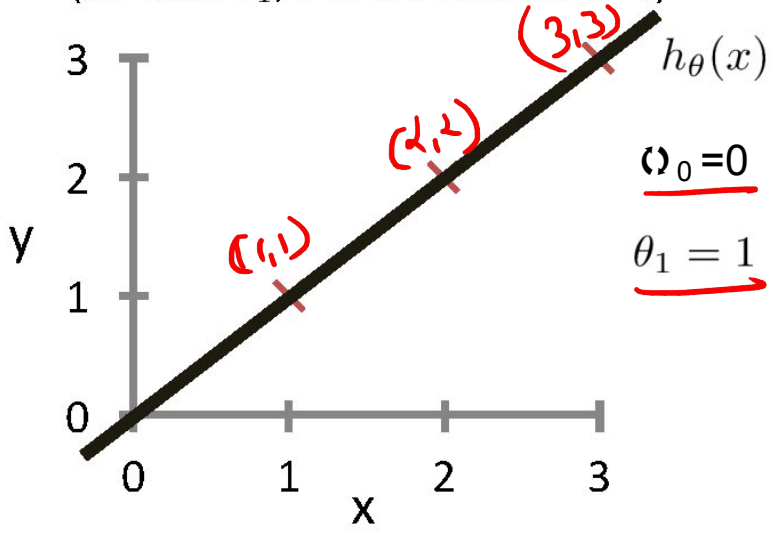
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

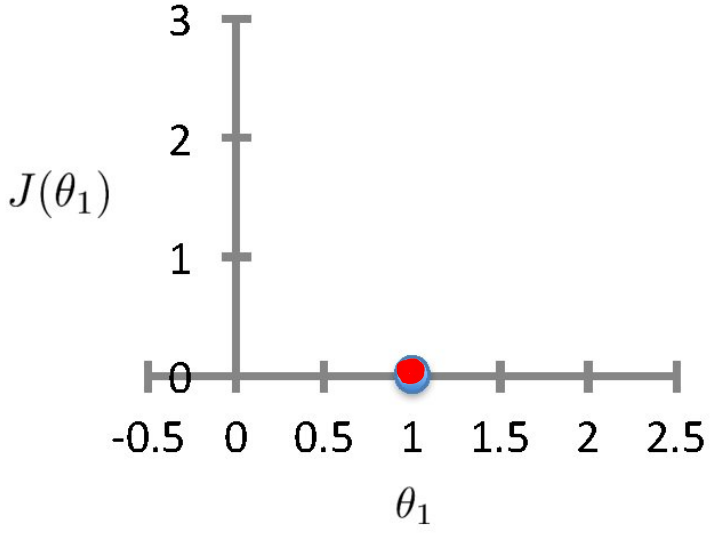
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



Based on example by Andrew Ng

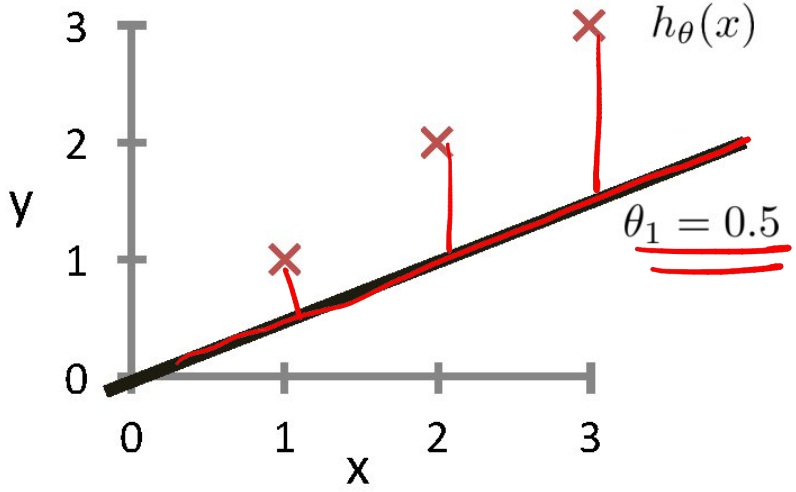
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

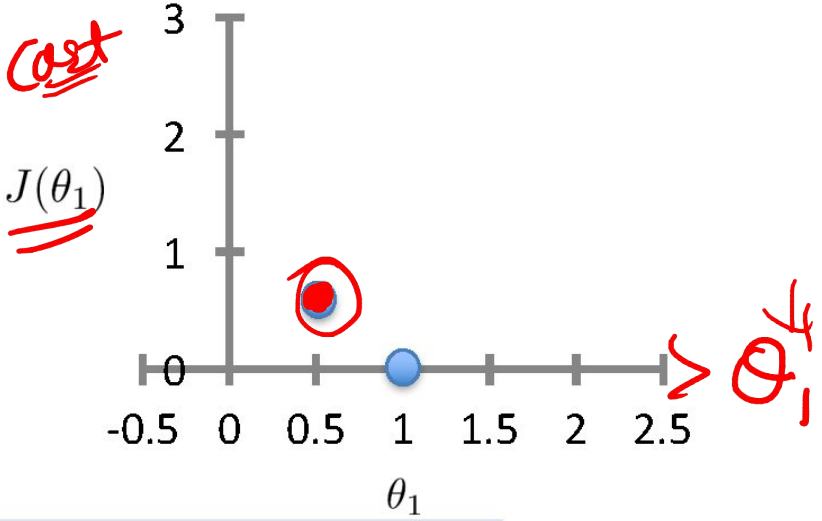
$h_\theta(x)$

(for fixed θ_1, this is a function of x)



$J(\theta_1)$

(function of the parameter θ_1)



$$J([0, 0.5]) = \frac{1}{2 \times 3}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right] \approx 0.58$$

Based on example by Andrew Ng

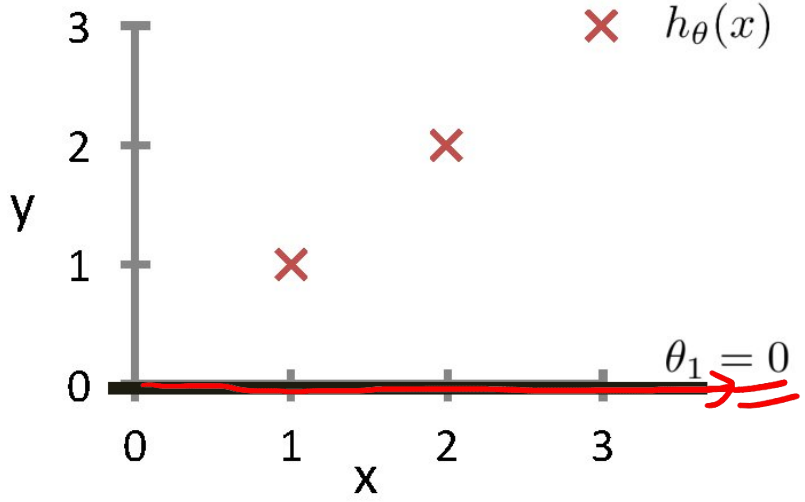
Intuition Behind Cost Function

$$J(\boldsymbol{\theta}) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_{\boldsymbol{\theta}}\left(\boldsymbol{x}^{(i)}\right) - y^{(i)}\right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

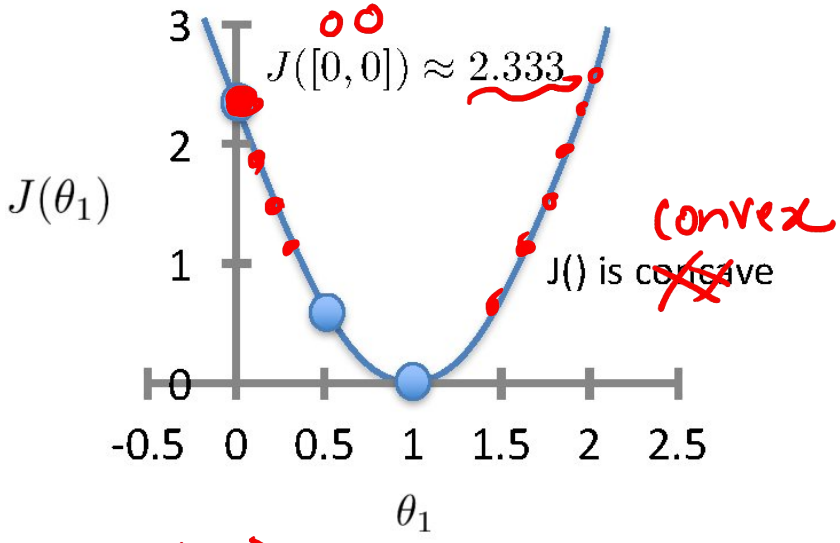
$h_\theta(x)$

(for fixed θ_1, this is a function of x)

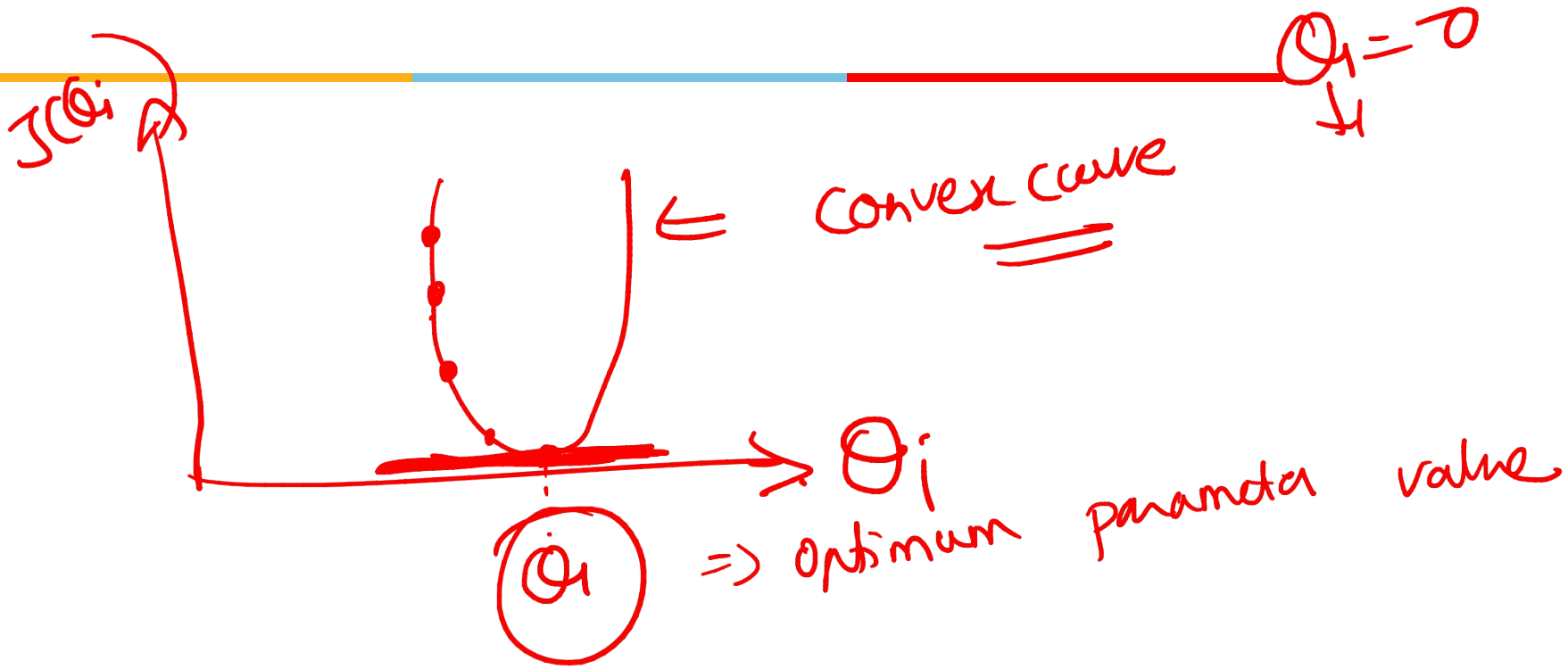


$J(\theta_1)$

(function of the parameter θ_1)



Based on example by Andrew Ng


$\theta_1 = 0$
$J(\theta_j)$
convex curve
θ_j
θ_1
=> optimum parameter value

Linear Regression Summary

- Hypothesis function $h_\theta(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$
- Parameters to learn θ_i
- Cost function $J(\theta) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_\theta\left(x^{(i)}\right) - y^{(i)} \right)^2$
- Goal $\min_\theta J(\theta)$

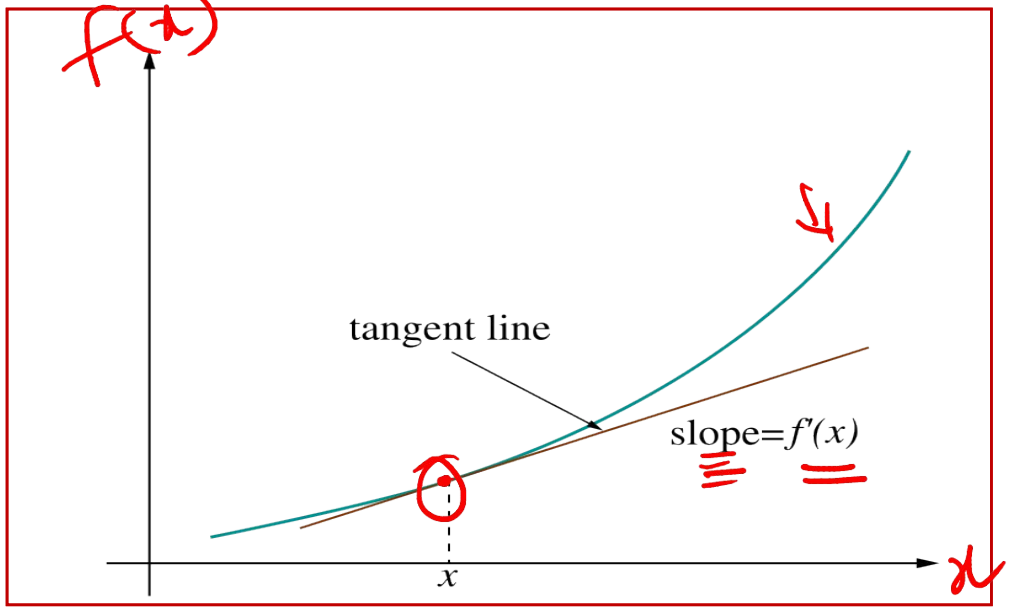
Gradient Descent

Gradient

$$f(x,y) = x^2y + \sin(y)$$

$$\frac{\partial f}{\partial x} = 2xy$$

$$\frac{\partial f}{\partial y} = x^2 + \cos(y)$$

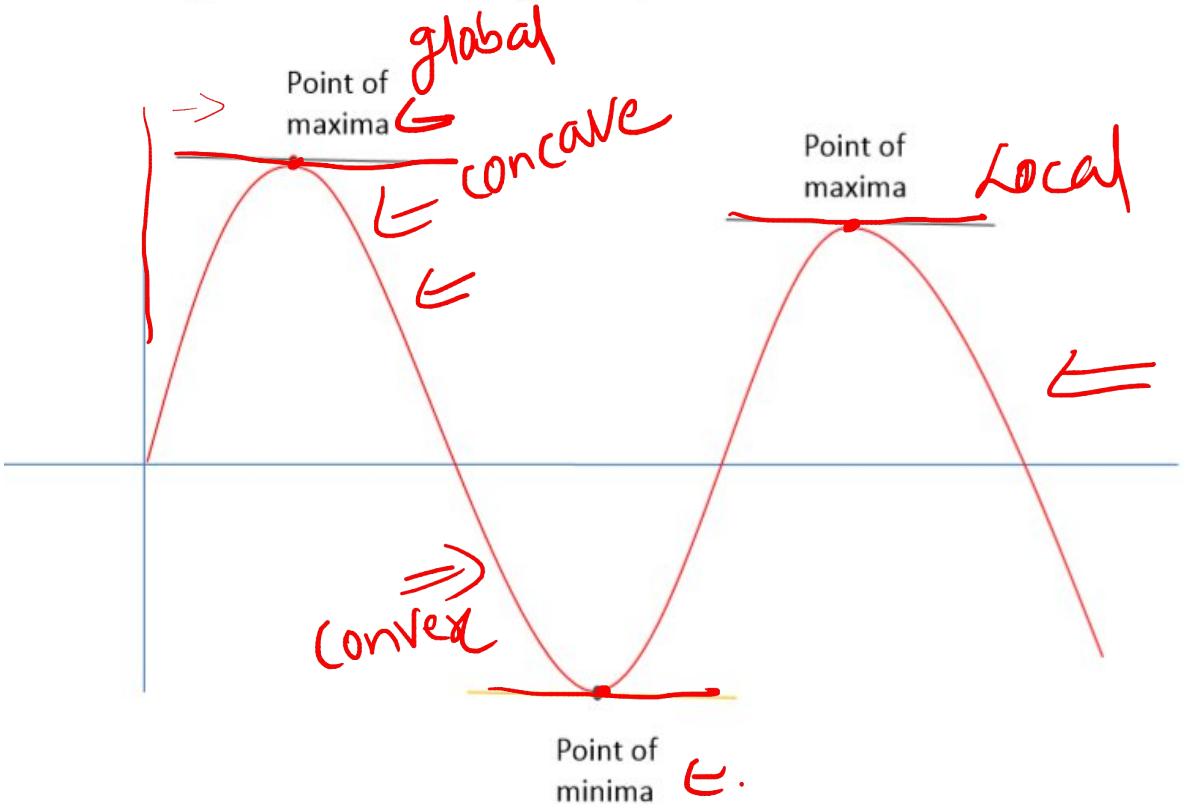


– Gradient puts these two partial derivatives together in a vector as follows:

$$\nabla f(x,y) = \nabla\left(x^2y + \sin(y)\right) = \begin{bmatrix} 2xy \\ x^2 + \cos(y) \end{bmatrix}$$

Maxima and Minima

- For maxima and minima $m = dy/dx = \tan 0^0 = 0$
- $dy/dx = 0$ means tangent is parralel to X –axis.



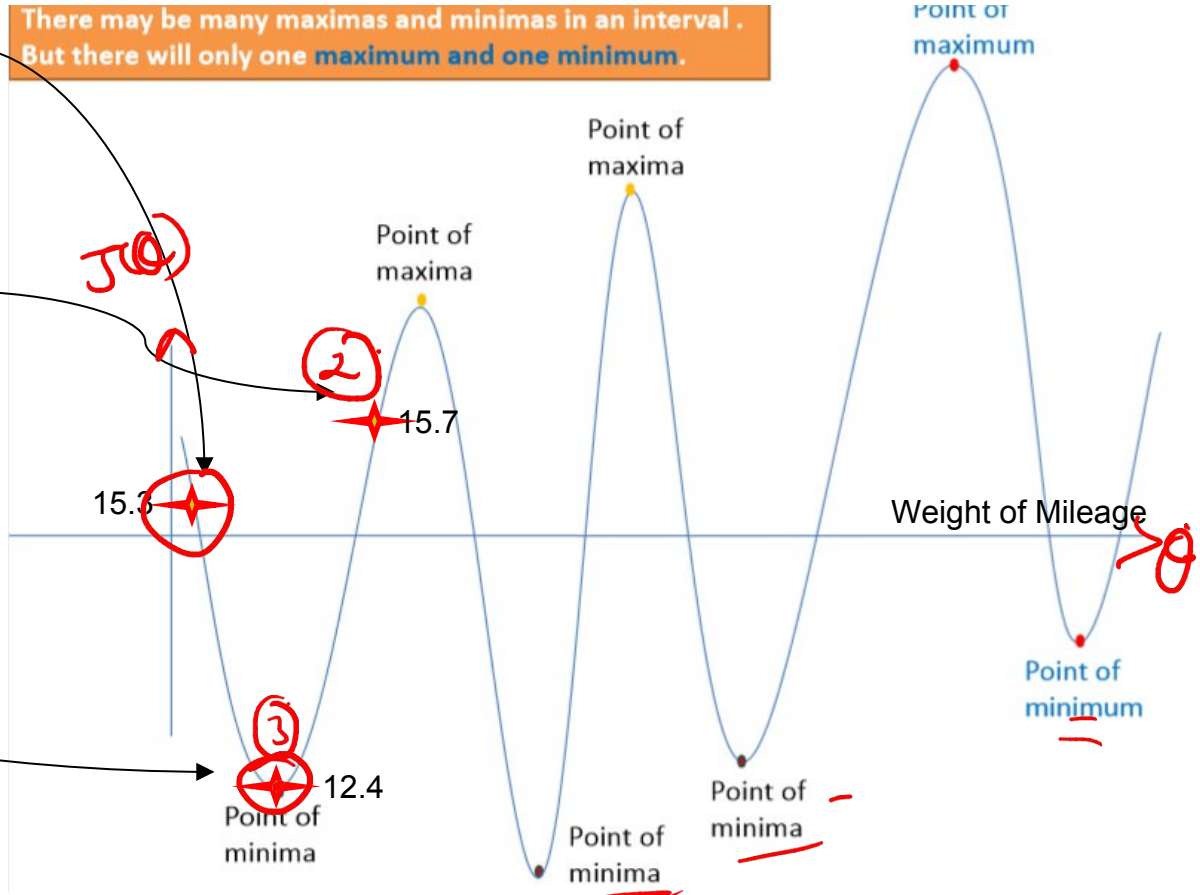
Local minima
global minima

$f'(x) = 0$

Notion of Maxima and Minima of a function in Machine Learning

1. CarPrice = 1 + 0.05Mileage
2. CarPrice = 1 + 0.25Mileage
3. CarPrice = 1 + 0.75Mileage

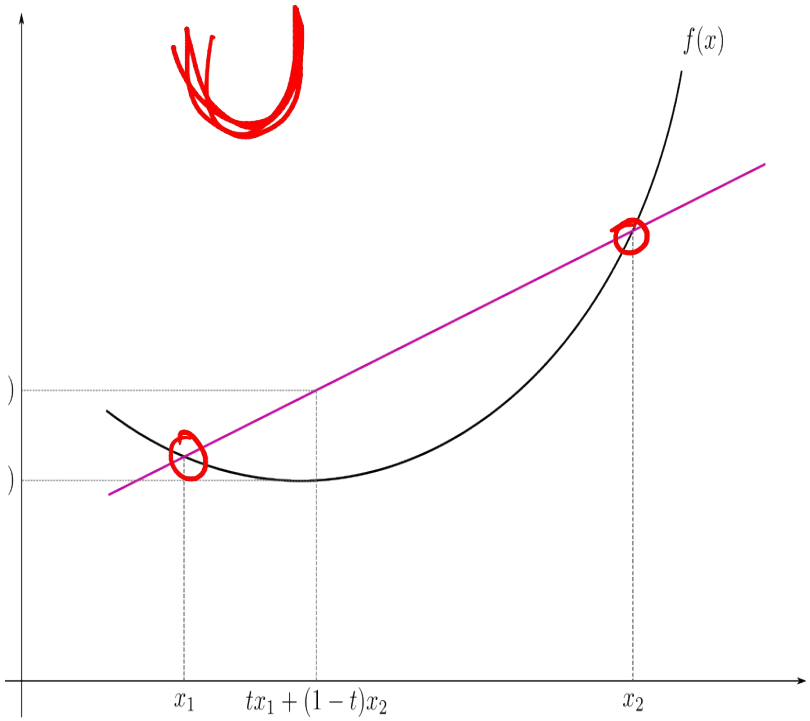
Mileage (in kmpl)	*Car Price (in cr)*
9.8	10.48
9.12	1.75
9.5	6.95
10	2.51



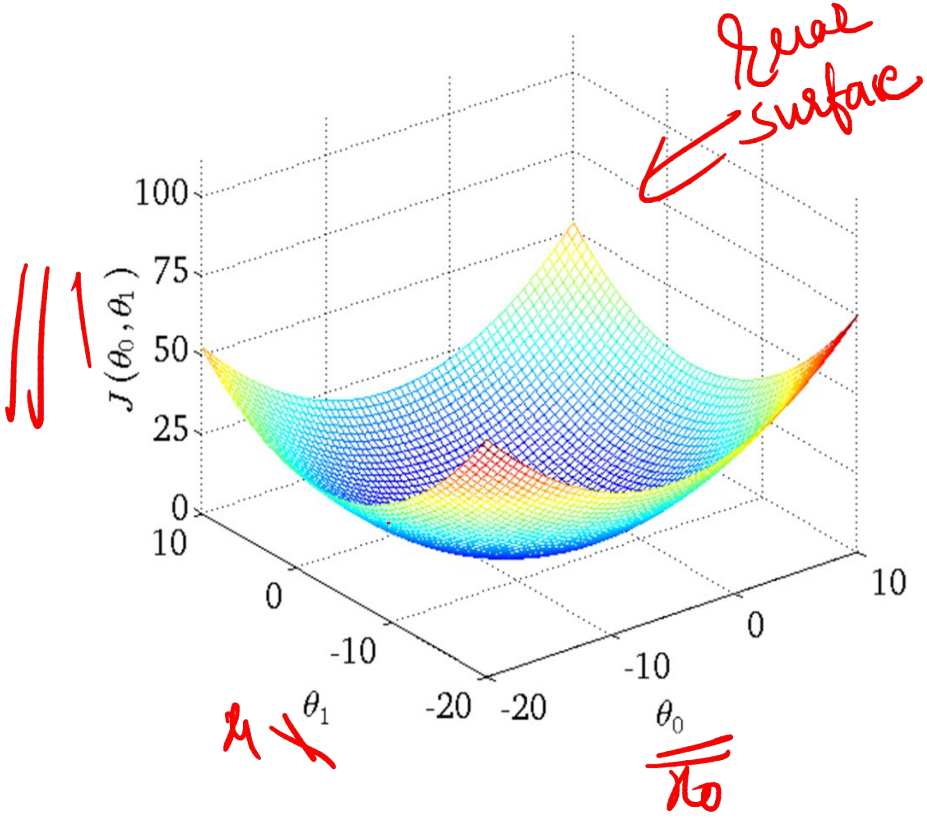
Sum of absolute Error / Training Error

Convex Function

Univariate



Multivariate



Real-valued function defined on an n-dimensional interval is called **convex** if the line segment between any two points on the graph of the function lies above or on the graph

BITS Pilani, Pilani Campus

Basic Search Procedure

- Choose initial value for θ
- Until we reach a minimum:
  - Choose a new value for θ to reduce $J(\theta)$

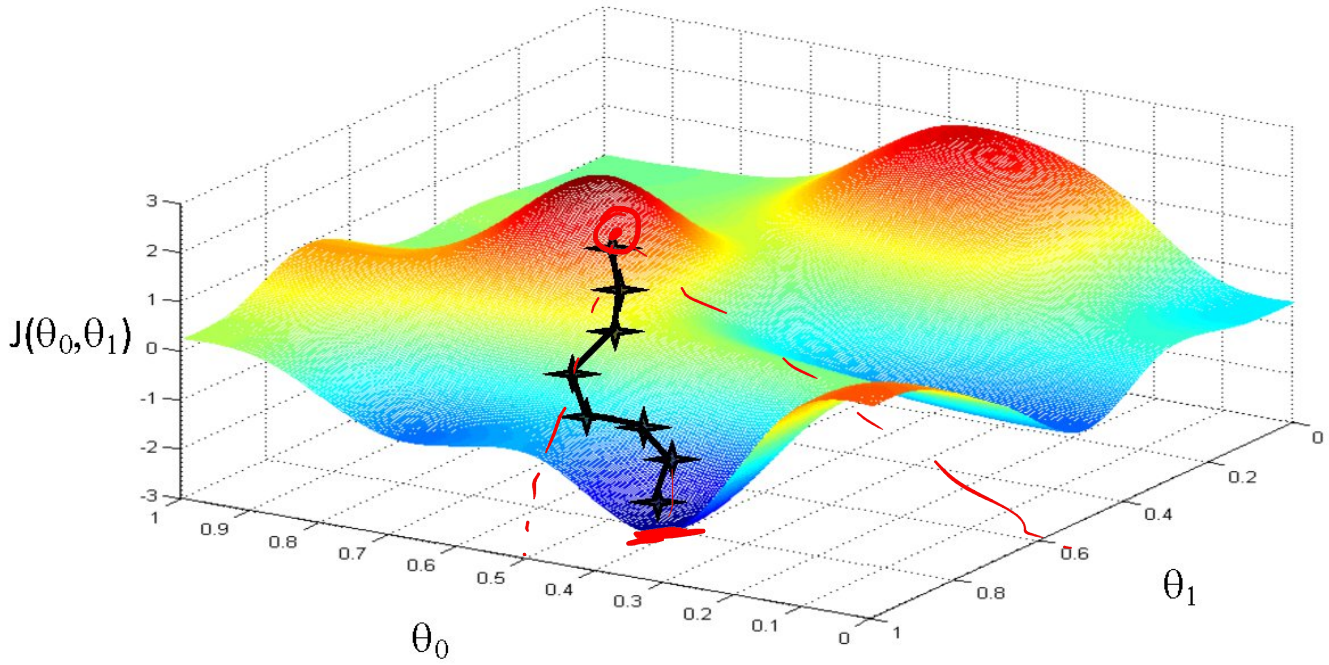


Figure by Andrew Ng

Basic Search Procedure

- Choose initial value for θ
- Until we reach a minimum:
  - Choose a new value for θ to reduce $J(\theta)$

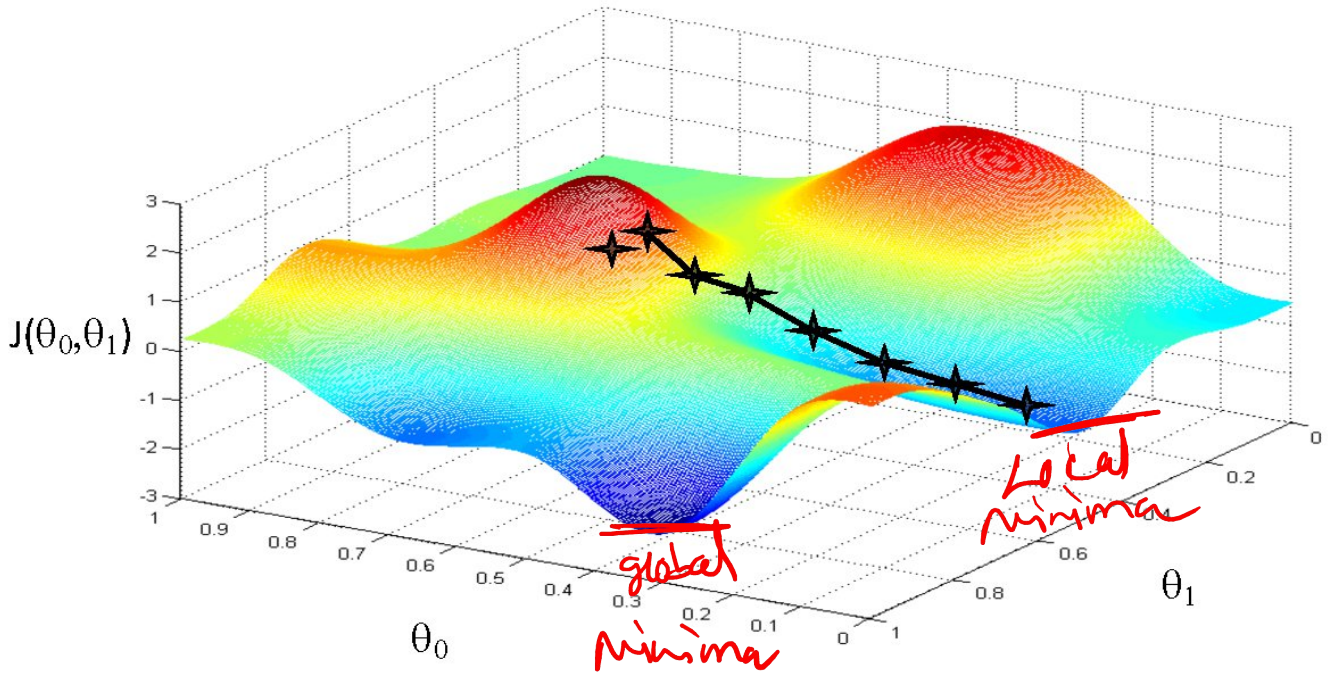
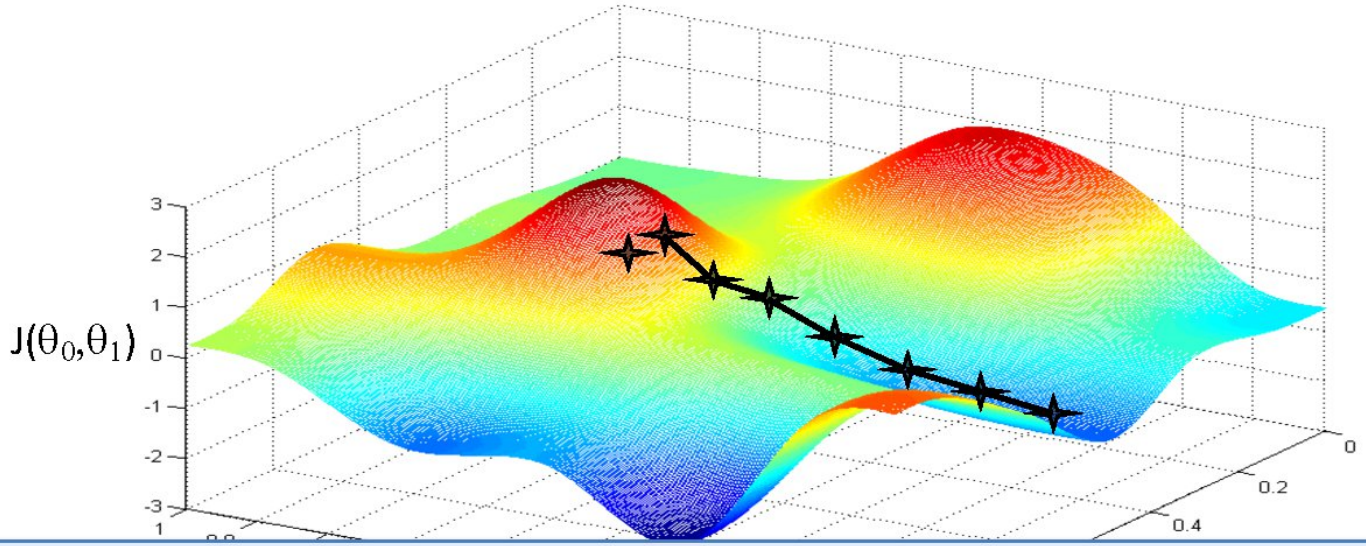


Figure by Andrew Ng

Basic Search Procedure

- Choose initial value for θ
- Until we reach a minimum:
  - Choose a new value for θ to reduce $J(\theta)$



Since the least squares objective function is convex (concave), we don't need to worry about local minima

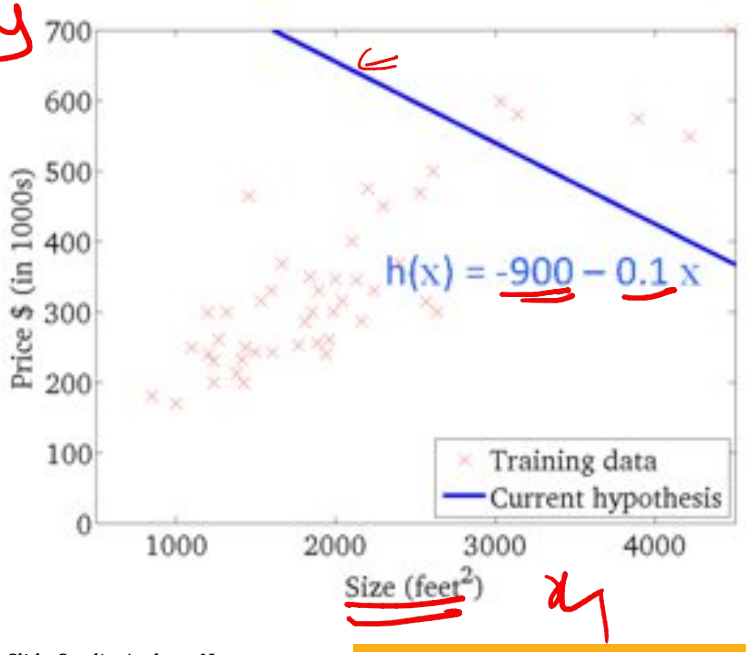
Figure by Andrew Ng

Gradient Descent

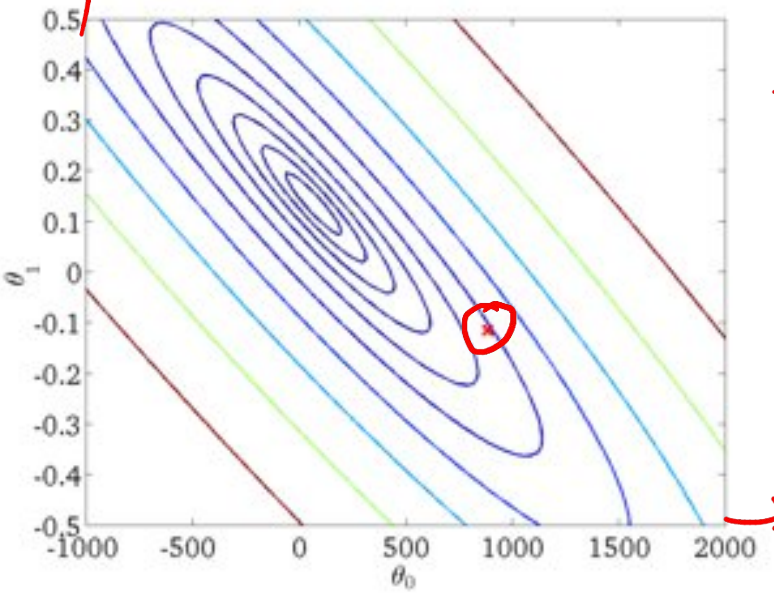
Size in feet^2 (x)	Price ($) in 1000's (y)
2104	460
1416	232
1534	315
852	178
…	…

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

$h_\theta(x)$ (for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$ (function of the parameters θ_0, θ_1)



Contour plots

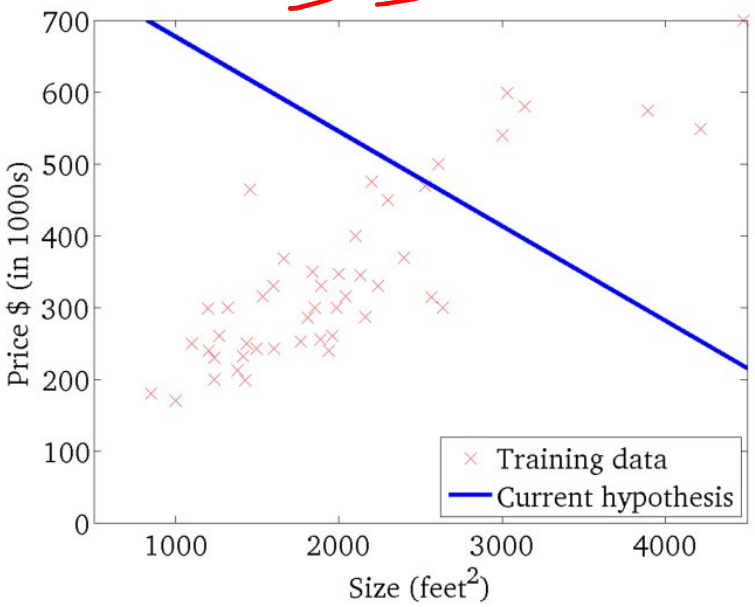
Khan academy

Slide Credit: Andrew Ng

Gradient Descent

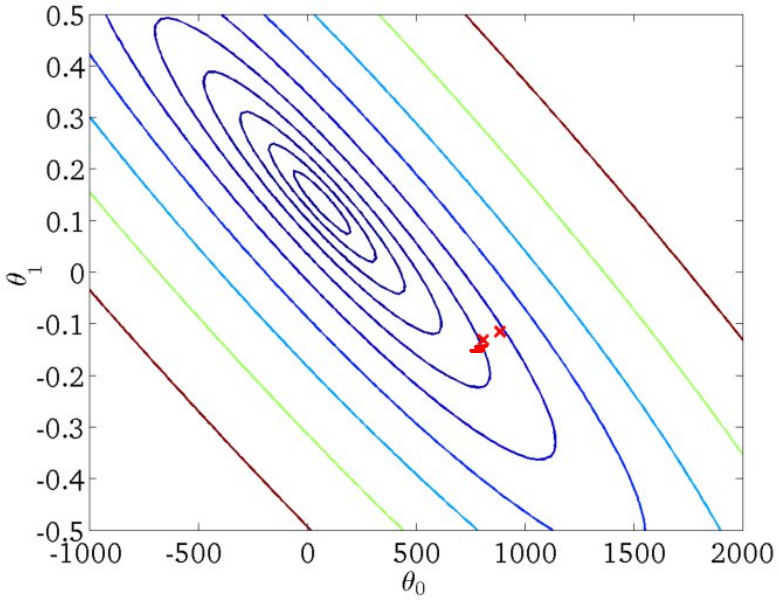
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

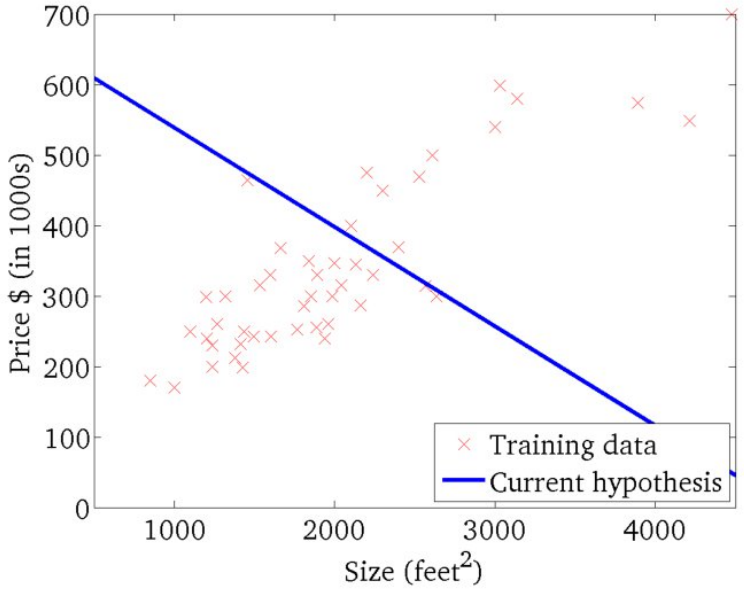


Slide by Andrew Ng

Gradient Descent

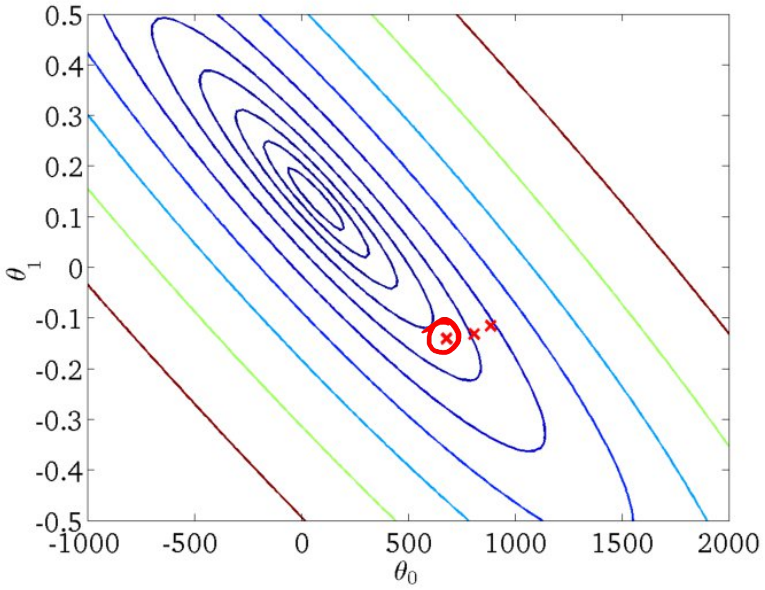
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

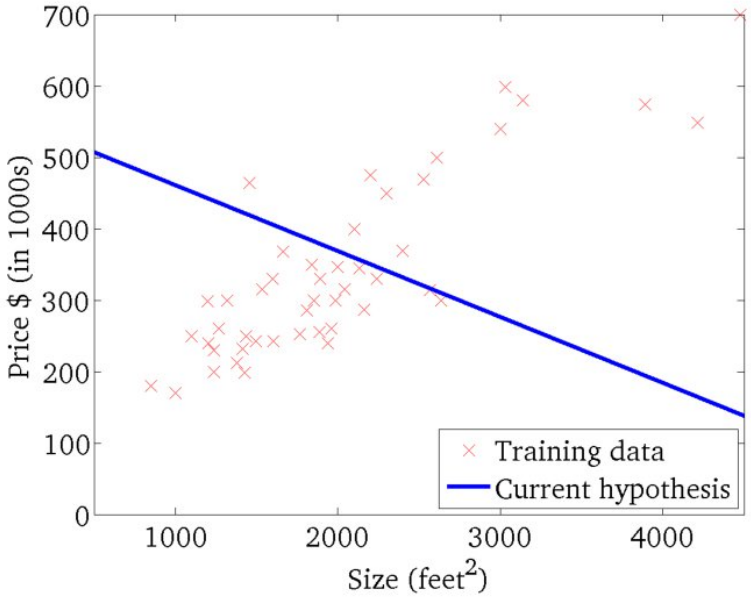


Slide by Andrew Ng

Gradient Descent

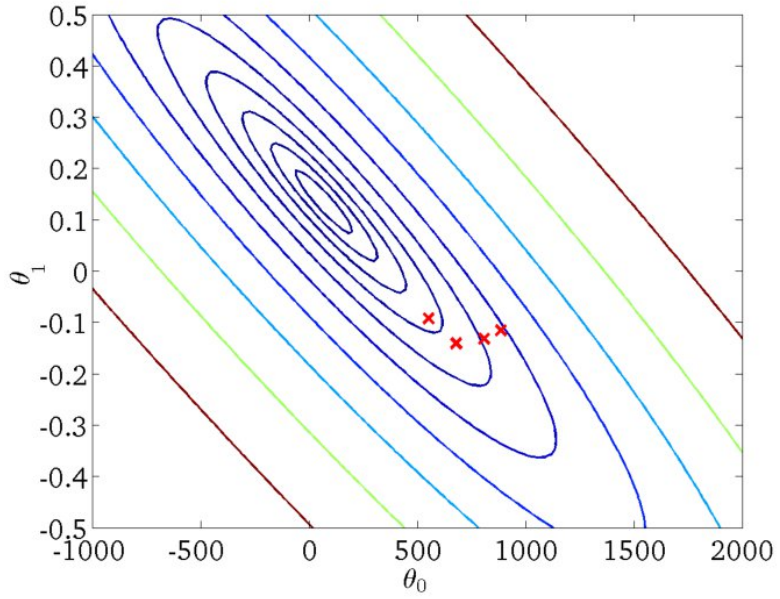
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

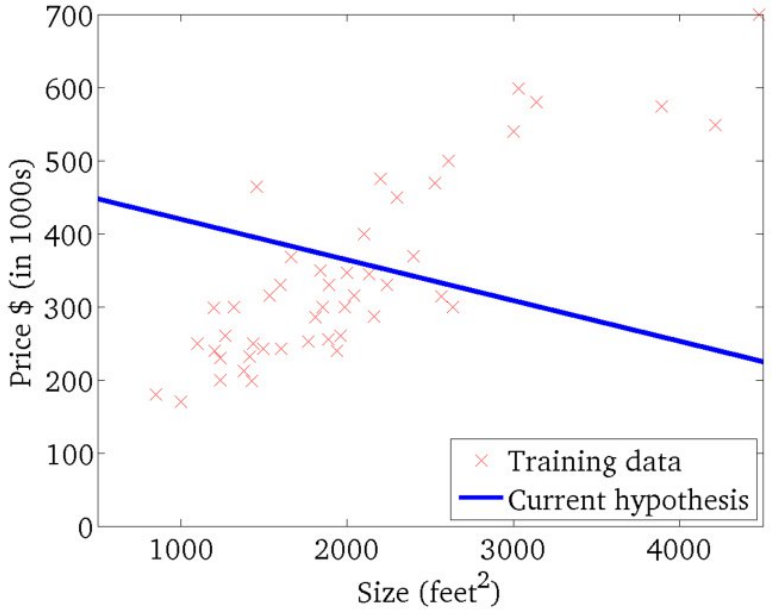


Slide by Andrew Ng

Gradient Descent

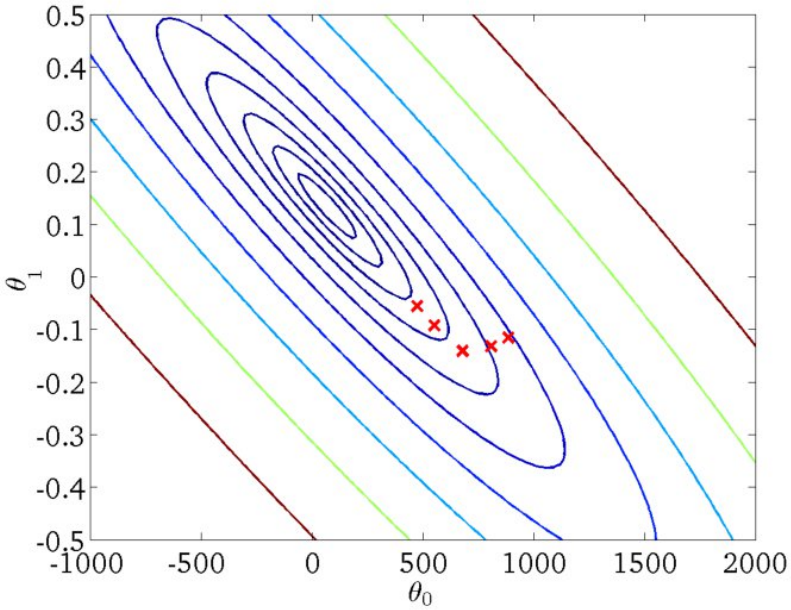
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

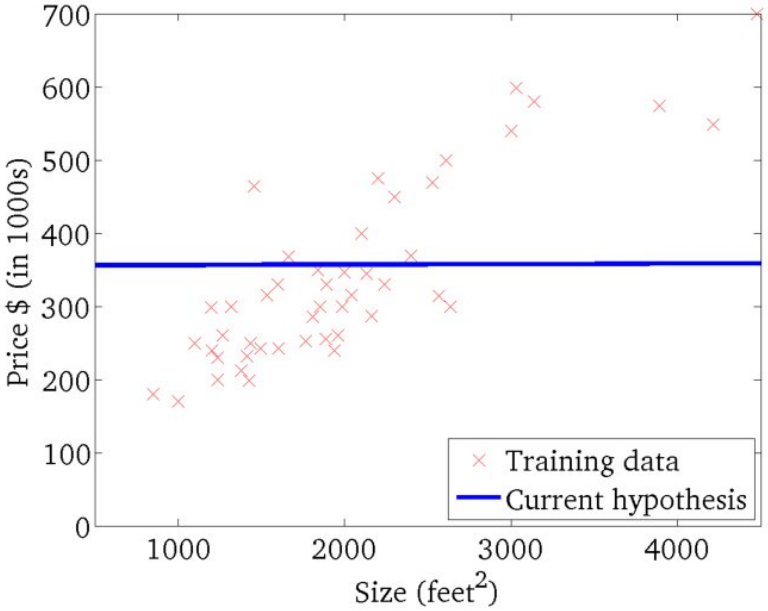


Slide by Andrew Ng

Gradient Descent

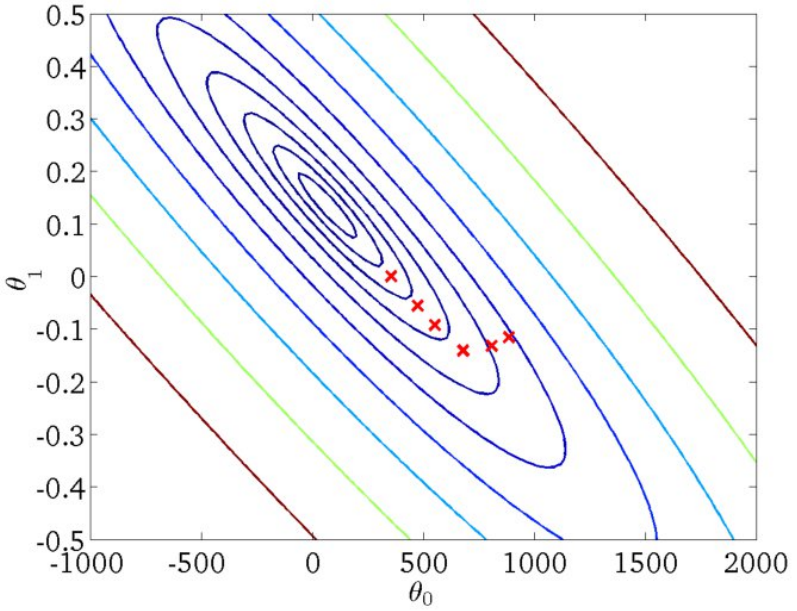
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

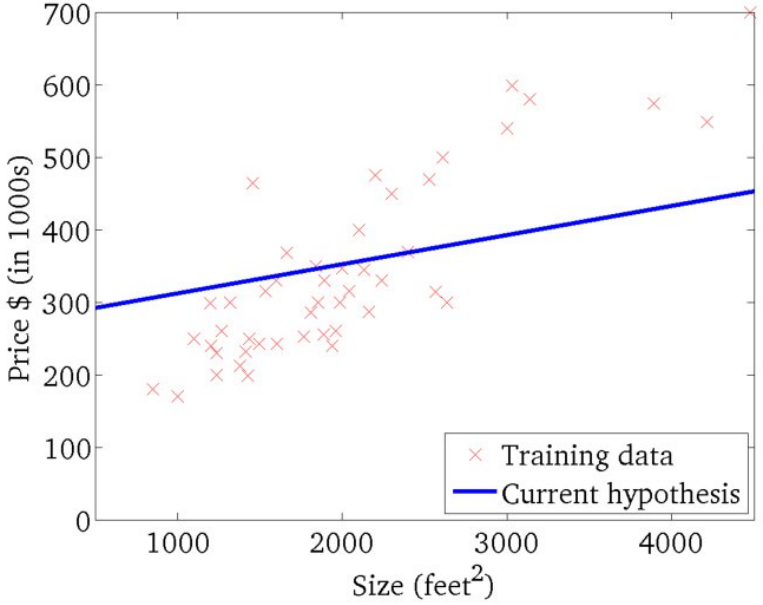


Slide by Andrew Ng

Gradient Descent

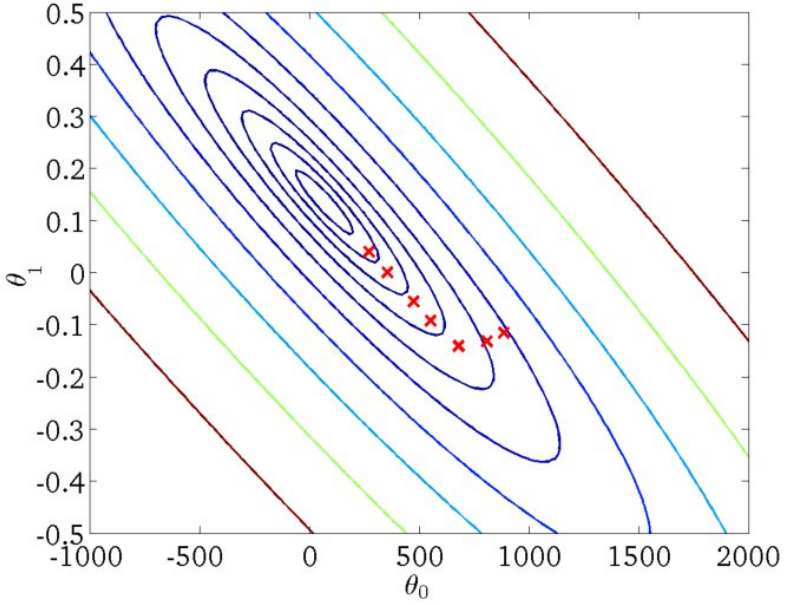
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

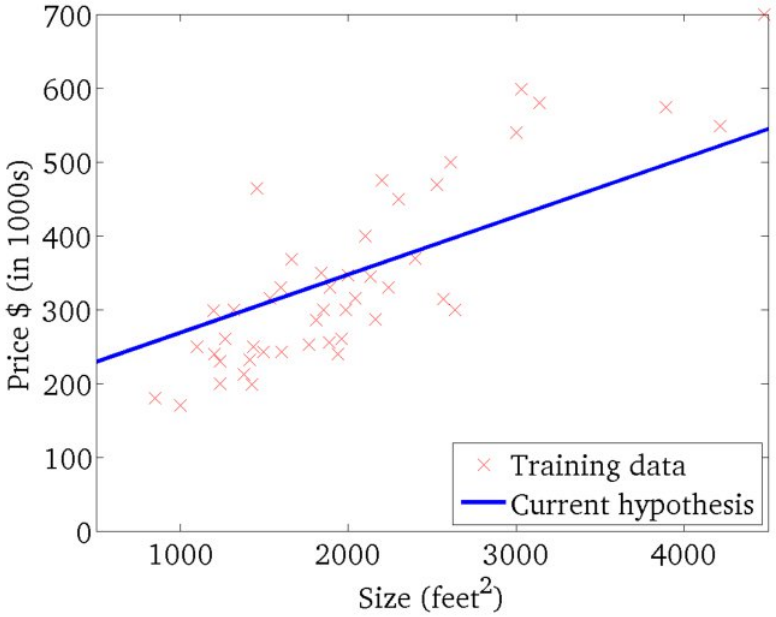


Slide by Andrew Ng

Gradient Descent

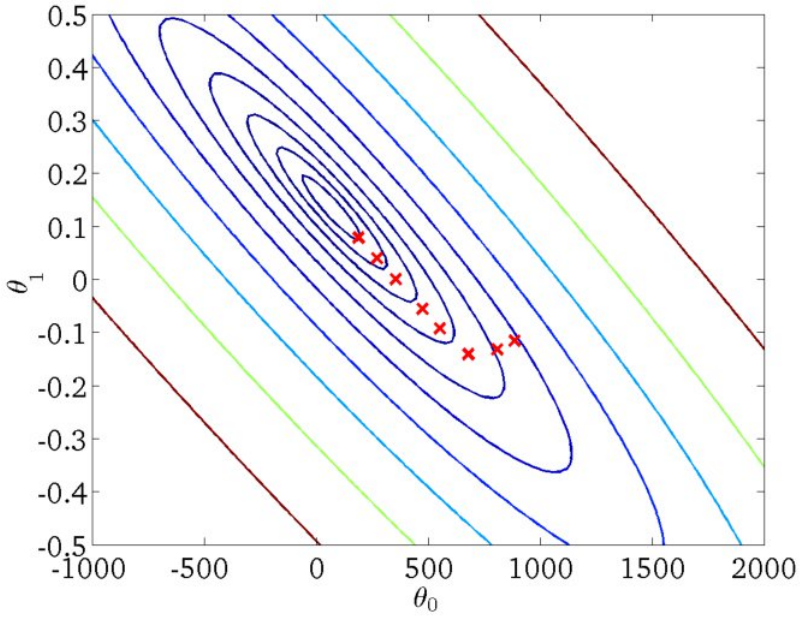
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

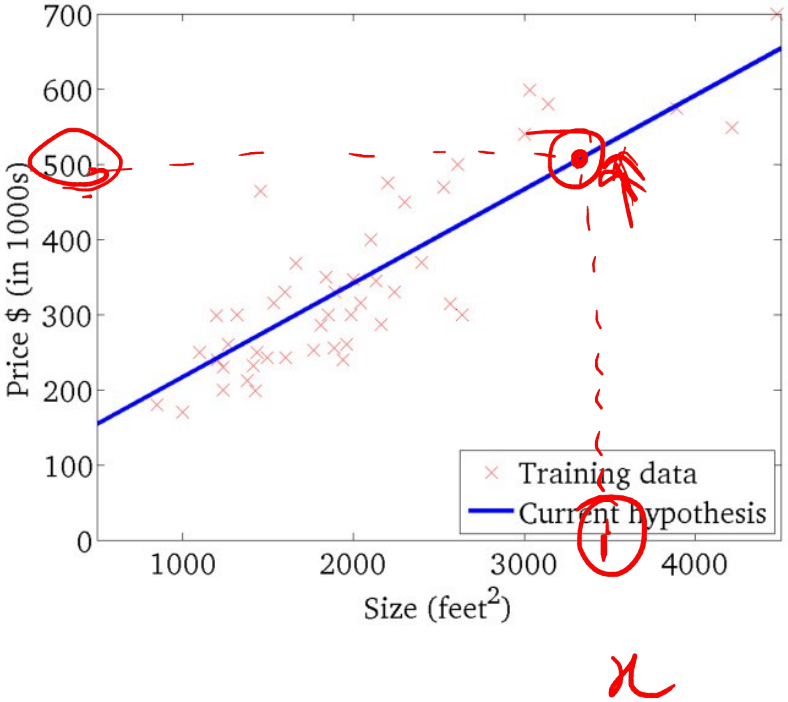


Slide by Andrew Ng

Gradient Descent

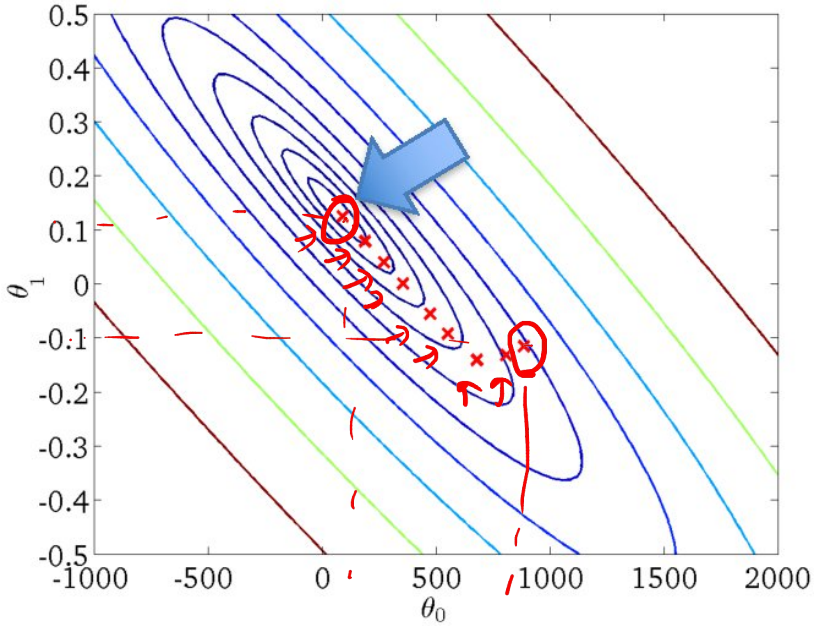
$h_\theta(x)$

(for fixed θ_0, θ_1, this is a function of x)



$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)



Slide by Andrew Ng

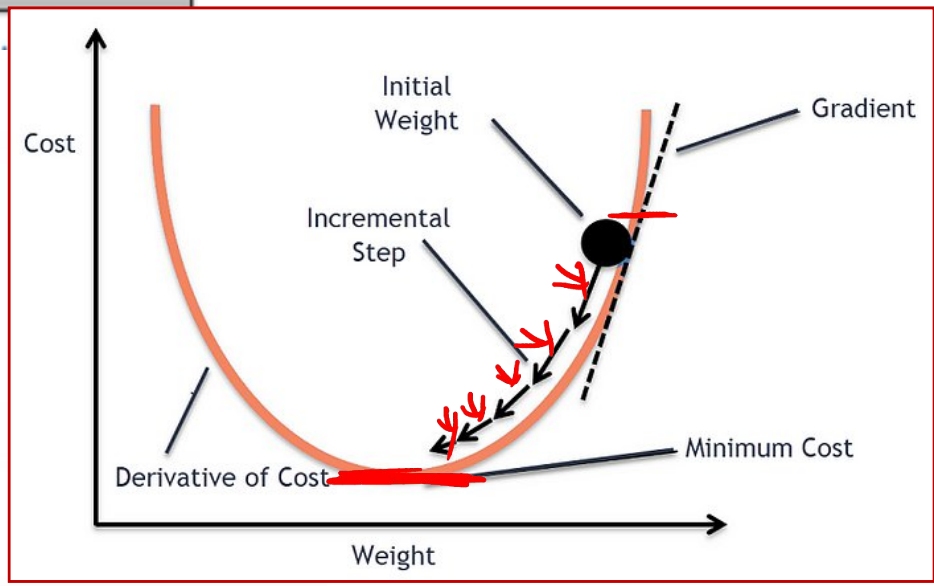
Gradient Descent Algorithm

- Initialize θ
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

simultaneous update for j = 0 ... d

learning rate (small)
e.g., $\alpha = 0.05$



Have some function $J(\theta_0, \theta_1)$

Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

Outline:

- Start with some θ_0, θ_1
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

$$\sum_{j=0}^{d} x_j \theta_j \quad j = 0 \ldots d$$

$$\Rightarrow x_0\theta_0 + x_1\theta_1 + x_2\theta_2 + x_3\theta_3$$

$$\theta_0^{new} = \theta_0^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_0}$$

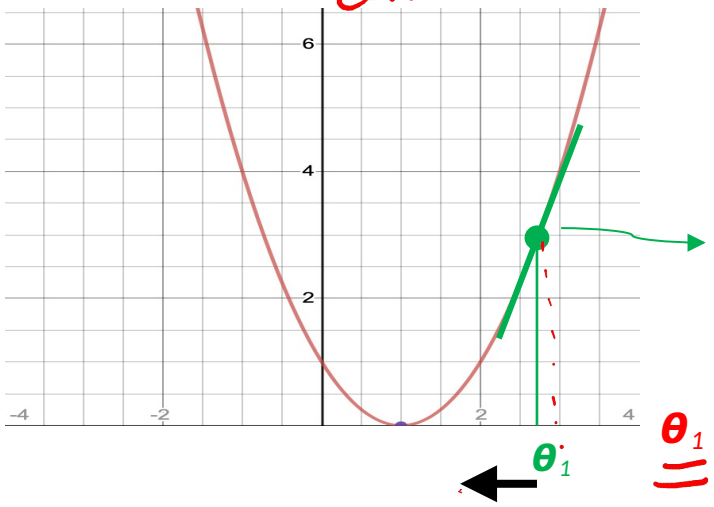
$$\theta_1^{new} = \theta_1^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_1}$$

$$\theta_d^{new} = \theta_d^{new} - \alpha \frac{\partial J(\theta)}{\partial \theta_d}$$

$$\theta_j^{new} = \theta_j^{old} - \alpha \frac{\partial J(\theta)}{\partial \theta_j} \quad j = 0 \ldots d$$

Gradient Descent Intuition

- optimization objective is to minimize $J(\theta_1)$ θ_0, θ_1



$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

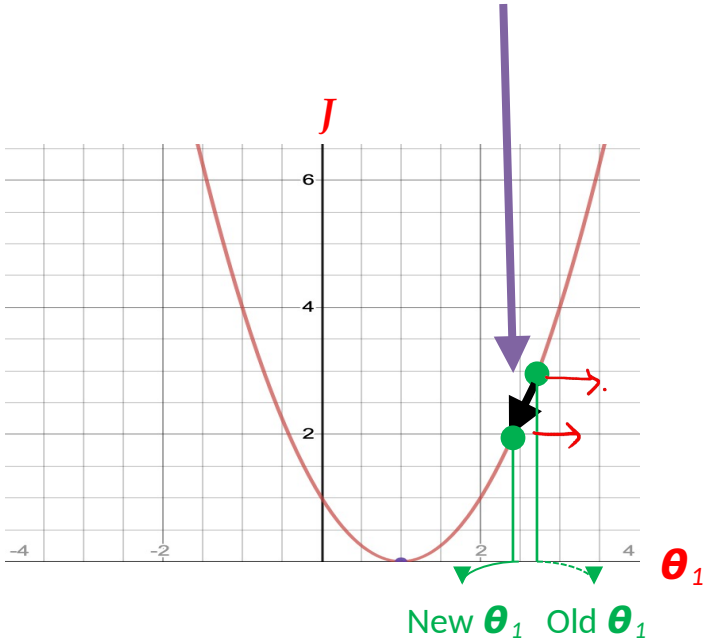
$$= \theta_1 - \alpha\,(Positive\ Number)$$

Decrease θ_1 by a certain value

The Impact of Partial Derviative

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1

Step towards negative direction of a gradient



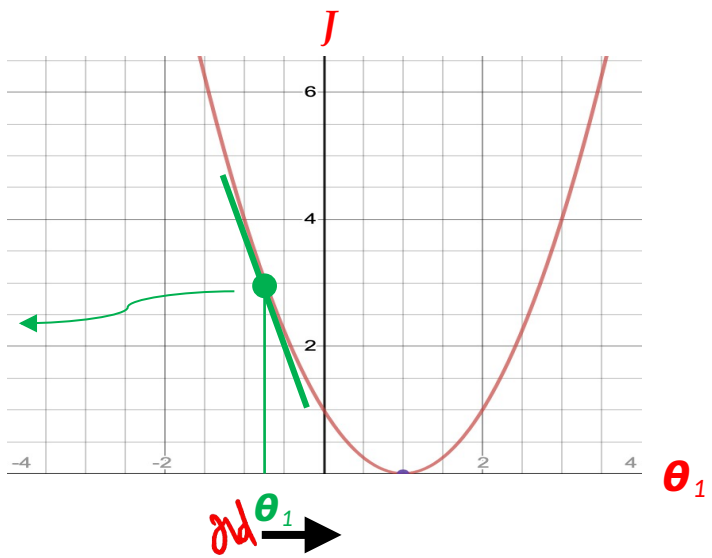
$$\theta_1 = \theta_1 - \alpha \frac{d J(\theta_1)}{d \theta_j}$$

$$= \theta_1 - \alpha \, (Positive\ Number)$$

Decrease θ_1 by a certain value

The Impact of Partial Derviative

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1



Negative Derivative

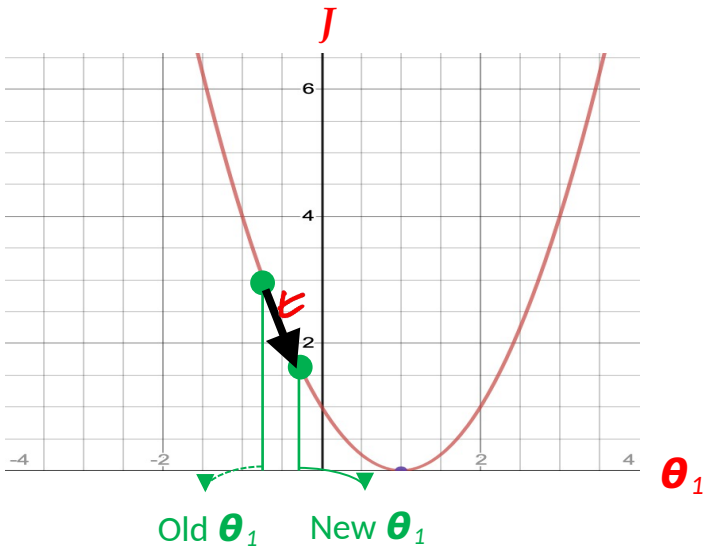
$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

$$= \theta_1 - \alpha\,(Negative\ Number)$$

Increase θ_1 by a certain value

The Impact of Partial Derviative

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1



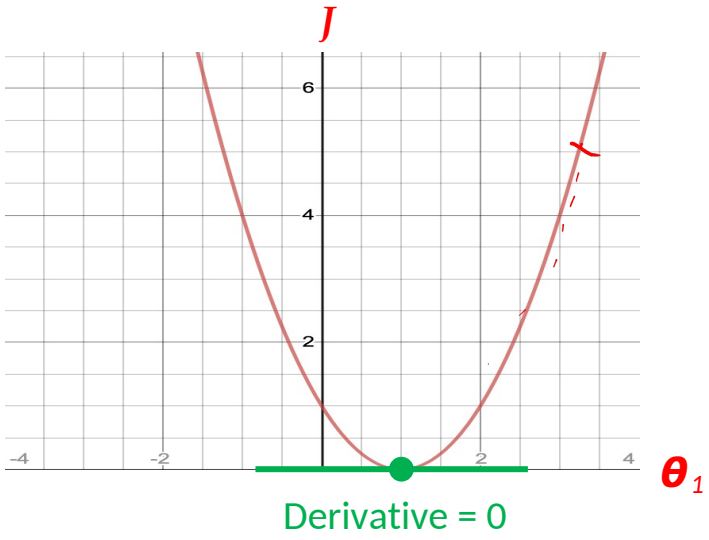
$$\theta_1 = \theta_1 - \alpha \frac{d J(\theta_1)}{d \theta_j}$$

$$= \theta_1 - \alpha \, (Negative\ Number)$$

Increase θ_1 by a certain value

The Impact of Partial Derviative

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1



$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

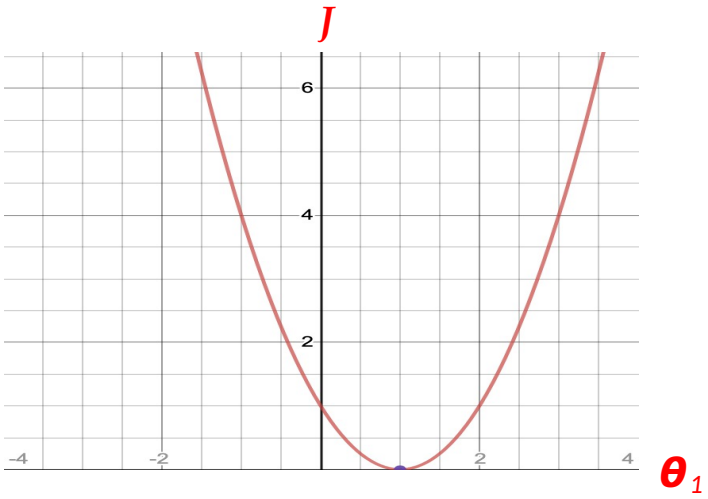
$$= \theta_1 - \alpha\,(Zero)$$

θ_1 remains the same, hence, gradient descent *converges*

The Impact of Learning Rate

α or learning rate
hyper parameters

- optimization objective is to minimize $J(\theta_1)$ θ_0, θ_1



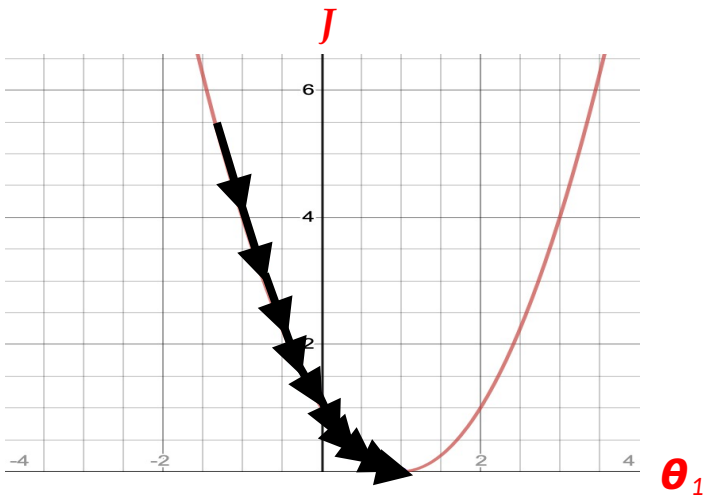
$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

Learing Rate

What happens if α is too small?

The Impact of Learning Rate

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1



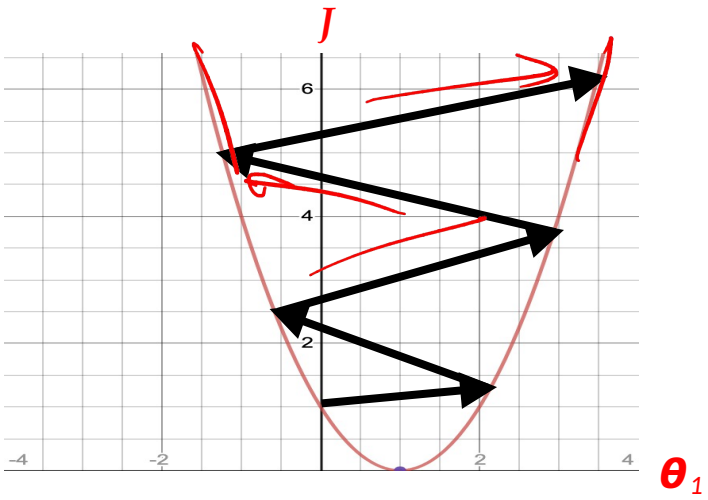
$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

$$= \theta_1 - (Too\ Small\ Number) \frac{d\,J(\theta_1)}{d\,\theta_j}$$

θ_1 changes only a tiny bit on each step, hence, gradient descent will render slow (will take more time to converge)

The Impact of Learning Rate

- optimization objective is to minimize $J(\theta_1)$ θ_0,θ_1



$$\theta_1 = \theta_1 - \alpha \frac{d\,J(\theta_1)}{d\,\theta_j}$$

$$= \theta_1 - (Too\ Large\ Number)\ \frac{d\,J(\theta_1)}{d\,\theta_j}$$

θ_1 changes a lot (and probably faster) on each step, hence, gradient descent *will potentially overshoot the minimum and, accordingly, fail to converge (or even diverge)*

Gradient Descent algorithm :

Effect of Learning Rate

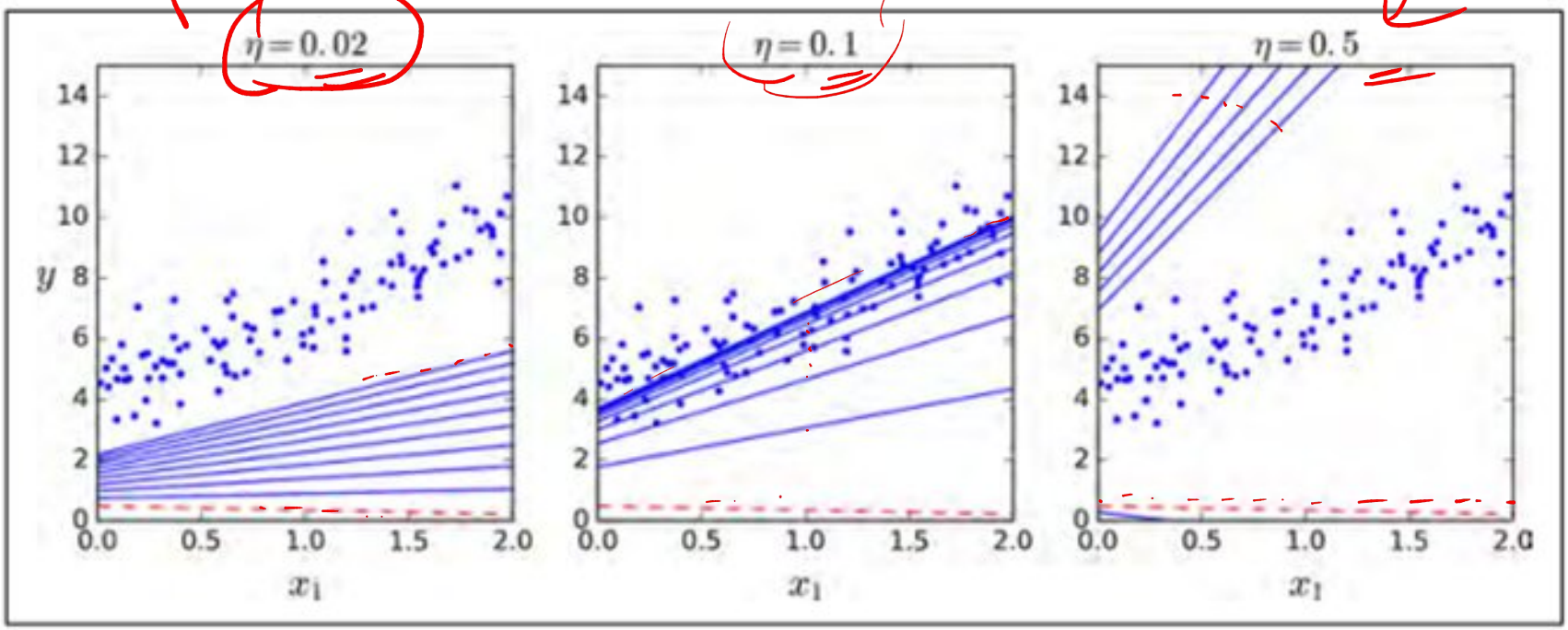


Figure 4-8. Gradient Descent with various learning rates

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Gradient Descent algorithm :

Effect of Feature Scaling

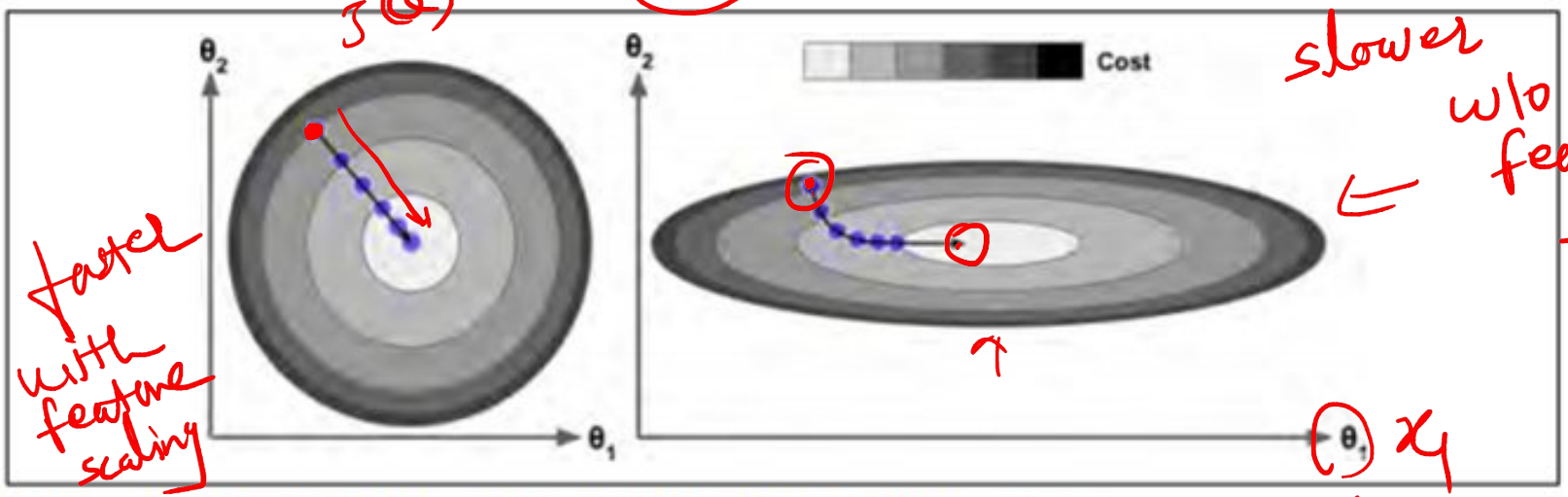


Figure 4-7. Gradient Descent with and without feature scaling

Cost Function Linear Regression

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

$$J(\boldsymbol{\theta}) = \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2$$

For insight on J(), let's assume $x \in \mathbb{R}$ so $\boldsymbol{\theta} = [\theta_0, \theta_1]$

$$\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right)^2$$

Derivative of cost function for one training example

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x,$$

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2} (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) \\
&= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
\frac{\partial J(\theta)}{\partial \theta_j} &= (h_\theta(x) - y)\, x_j \qquad (\hat{y} - y)
\end{aligned}$$

$\frac{\partial}{\partial \theta_0}\ \theta_0 x_0 + \theta_1 x_1 \cdots \theta_n x_n - y$

$x_0 \frac{\partial \theta_0}{\partial \theta_0} \Rightarrow x_0 \qquad \theta_j \Rightarrow x_j$

Derivative of cost function for n training example

- Initialize θ
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update
for j = 0 ... d

For Linear Regression:

$$\frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta}) = \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(x^{(i)} \right) - y^{(i)} \right)^2$$

$$= \frac{\partial}{\partial \theta_j} \frac{1}{2n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right)^2$$

$$= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) \times \frac{\partial}{\partial \theta_j} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right)$$

$$= \frac{1}{n} \sum_{i=1}^{n} \left(\sum_{k=0}^{d} \theta_k x_k^{(i)} - y^{(i)} \right) x_j^{(i)}$$

Gradient Descent for Linear Regression

Incorrect Implementation

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

$\theta_0 := temp0$

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

$\theta_1 := temp1$

repeat until convergence {

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

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

}

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

Update θ_0 and θ_1 simultaneously

$$temp0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$

$$temp1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

$\theta_0 := temp0$

$\theta_1 := temp1$

Gradient Descent for Linear Regression

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

simultaneous update for j = 0 ... d

- To achieve simultaneous update
  - At the start of each GD iteration, compute $h_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{(i)} \right)$
  - Use this stored value in the update step loop
- Assume convergence when $\| \boldsymbol{\theta}_{new} - \boldsymbol{\theta}_{old} \|_2 < \epsilon$

L$_2$ norm: $\| \boldsymbol{v} \|_2 = \sqrt{\sum_i v_i^2} = \sqrt{v_1^2 + v_2^2 + \ldots + v_{|v|}^2}$

Gradient Descent: Variants

- **Batch** gradient descent refers to calculating the derivative from all training data before calculating an update.

Initialize the Parameters $(\theta_0^1, \theta_1^1, \ldots..)$

K=1

Repeat until Convergence {

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

$$\theta_1^{k+1} = \theta_1^k - \frac{\alpha}{m} \sum_{i=1}^{m} ((h_\theta(x^{(i)}) - y^{(i)}) * x_1^{(i)})$$

$$\theta_2^{k+1} = \theta_2^k - \frac{\alpha}{m} \sum_{i=1}^{m} ((h_\theta(x^{(i)}) - y^{(i)}) * x_2^{(i)})$$

……

K=k+1

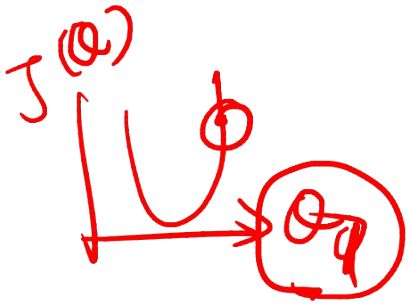
}

return $(\theta_0^1, \theta_1^1, \ldots..)$

BITS Pilani, Deemed to be University under Section 3 of UGC Act, 1956

Gradient Descent: Variants

- **Minibatch** refers to calculating derivative of mini groups of training data before calculating an update.



Divide the training instances into "N" batches each of size "m"
Initialize the Parameters $(\theta_0^1, \theta_1^1, \ldots..)$
K=1
Repeat until Convergence {
Repeat for every batch in 1 : N , each with 'm' instances {

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

$$\theta_1^{k+1} = \theta_1^k - \frac{\alpha}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)}) - y^{(i)}) * x_1^{(i)})$$

$$\theta_2^{k+1} = \theta_2^k - \frac{\alpha}{m}\sum_{i=1}^{m}((h_\theta(x^{(i)}) - y^{(i)}) * x_2^{(i)})$$

……
K=k+1
}
}
return $(\theta_0^1, \theta_1^1, \ldots..)$

Gradient Descent: Variants

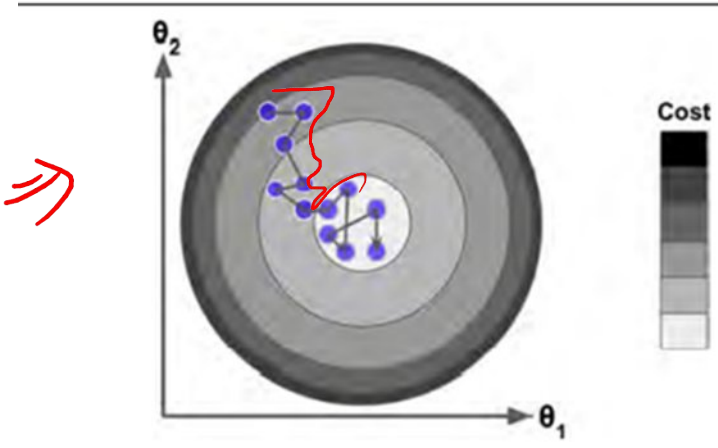
- **Stochastic gradient** descent refers to calculating the derivative from each training data instance and calculating the update immediately

Randomly shuffle training instances
Initialize the Parameters $(\theta_0^1, \theta_1^1, \ldots..)$
K=1
Repeat until Convergence {
Sample with replacement, only one random training instance "i" at a time

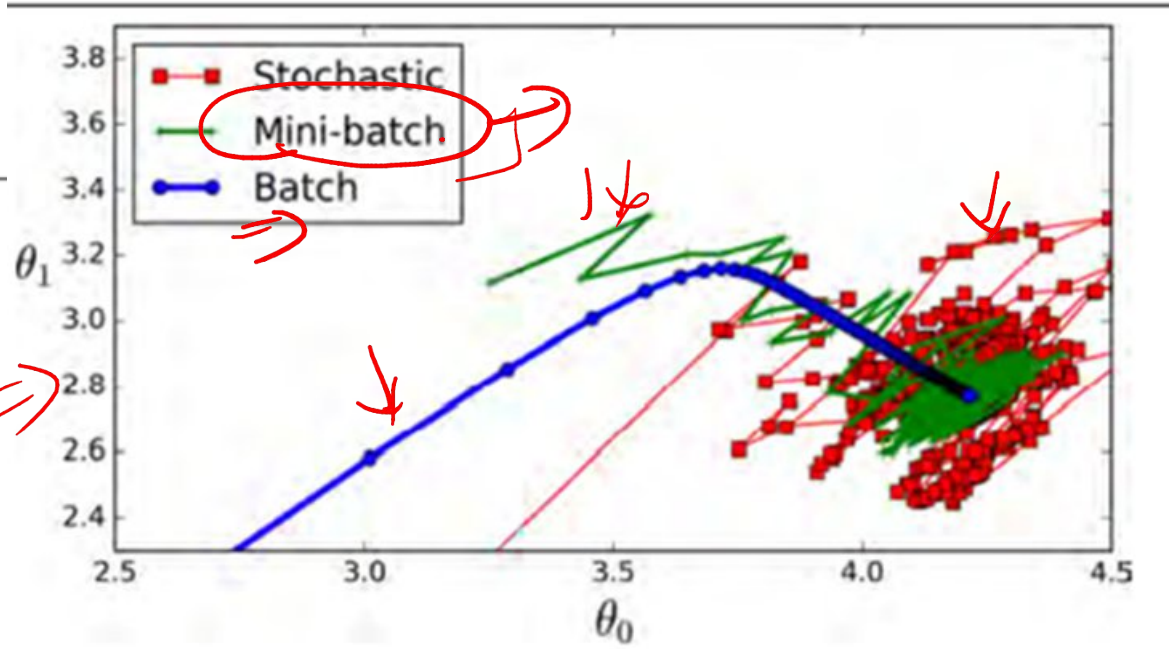
$$\theta_0^{k+1} = \theta_0^k - \alpha\left(h_\theta(x^{(i)}) - y^{(i)}\right)$$
$$\theta_1^{k+1} = \theta_1^k - \alpha\left(h_\theta(x^{(i)}) - y^{(i)}\right) * x_1^{(i)}$$
$$\theta_2^{k+1} = \theta_2^k - \alpha\left(h_\theta(x^{(i)}) - y^{(i)}\right) * x_2^{(i)}$$

……
K=k+1
}
}
return $(\theta_0^1, \theta_1^1, \ldots..)$

Gradient Descent: Variants - Effect on Convergence



Stochastic Gradient Descent



1. Gradient Descent paths in parameter space

BITS Pilani, Pilani Campus

Fit a linear Regression Line : Gradient Descent

Steps :

(Assuming : 'n' no.of.instances and two predictors {x1, x2} and linear regression)

1. Identification of the equations $y = w_0+w_1X_1+W_2X_2$
2. Cost function & derivative
    1. $W_0` = w_0 – 1/n * \text{learning rate} * (\text{sum} (w_0+w_1X_1+W_2X_2 - y))$
    2. $W_1` = w_1 – 1/n * \text{learning rate} * (\text{sum} (w_0+w_1X_1+W_2X_2 - y) * x_1)$
    3. $W_2` = w_2 – 1/n * \text{learning rate} * (\text{sum} (w_0+w_1X_1+W_2X_2 - y)*x_2)$
3. Apply the equations

Fit a linear Regression Line : Gradient Descent

Fit a linear regression. Show only the first iteration of Gradient descent algorithm using learning rate of **0.02** for the following data , if the Relative Risk of Coronary Heart Disease is believed to be only linearly dependent on BMI as well as Diastolic Pressure. Assume the intercept of the regression model as **5** and the slope of independent variables as **-0.03 (negative)**.

Patient	Systolic Pressure mm Hg	Diastolic Pressure mm Hg	BMI	Waist Circumference Threshold cm	RR-CHD (Relative Risk of Coronary Heart Disease)
1	140	80	35	100	1.81
2	120	80	25	80	1.22
3	130	100	30	60	1.71

Fit a linear Regression Line : Gradient Descent

- Initialize θ
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_\theta \left(x^{(i)} \right) - y^{(i)} \right) x_j^{(i)}$$

simultaneous update for j = 0 ... d

Patient	Systolic Pressure mm Hg	Diastolic Pressure mm Hg	BMI	Waist Circumference Threshold cm	RR-CHD (Relative Risk of Coronary Heart Disease)
1	140	80	35	100	1.81
2	120	80	25	80	1.22
3	130	100	30	60	1.71

Steps :

1. Identification of the equations: **RR-CHD = 5 – 0.03 * BMI – 0.03 * DiastolicPressure**
2. Cost function & derivative
   1. W_0` = w_0 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD))
      = 5 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD))
   2. W_1` = w_1 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * BMI)
      = -0.03 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * BMI)
   3. W_2 = w_2 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * DiastolicPresure)
      = -0.03 – 1/3 * 0.02 * (sum (5-0.03BMI-0.03DiastolicPresure - RRCHD) * DiastolicPresure)

3. Apply the equations : Answer at **the end of first iteration**:
   W0 = 5.0016 , W1 = 0.0476, w2= 0.179

References

- https://cs229.stanford.edu/lectures-spring2022/main_notes.pdf
- Pattern recognition and machine learning, Christopher bishop - CH3