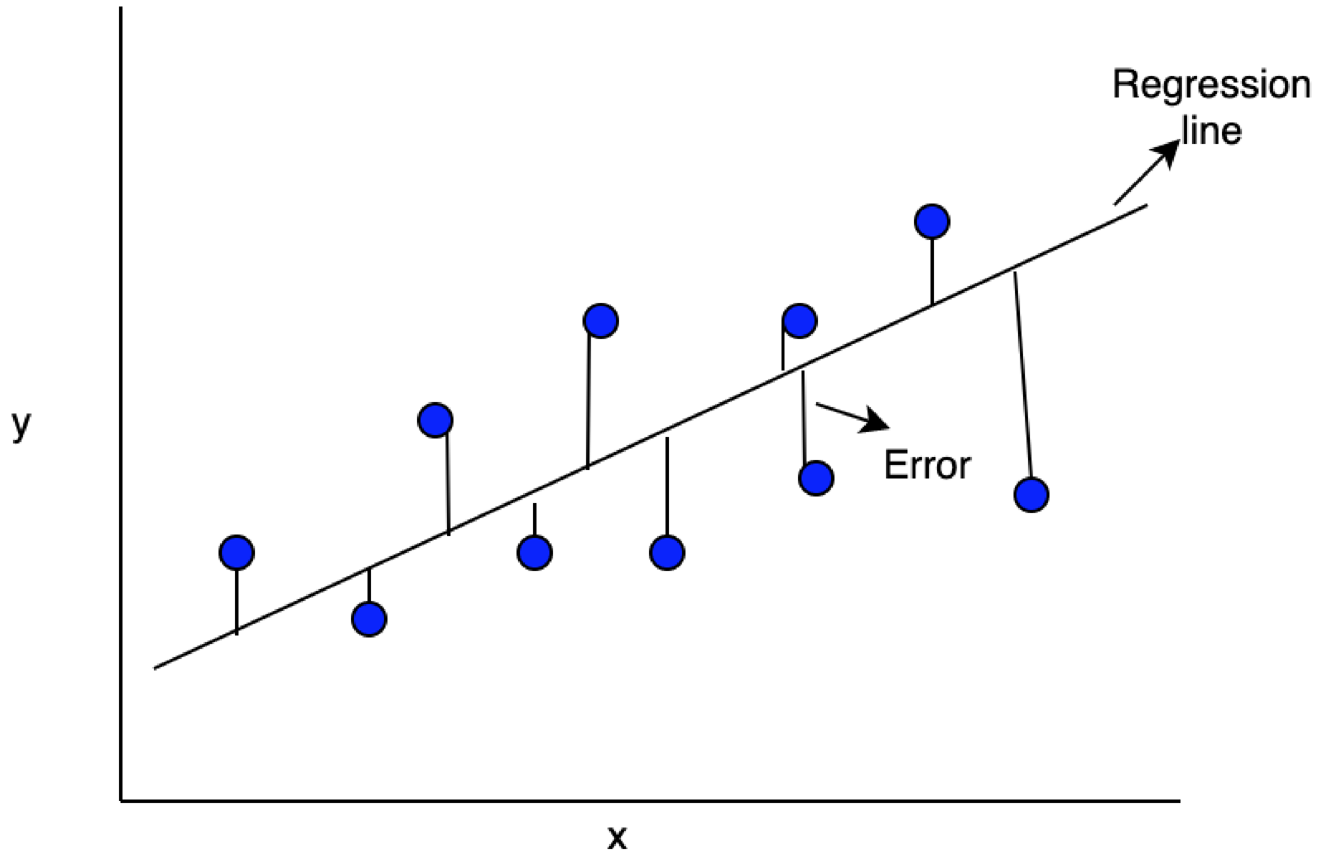
CSDS503 / COMP552 – Advanced Machine Learning

Faizad Ullah

Linear Regression

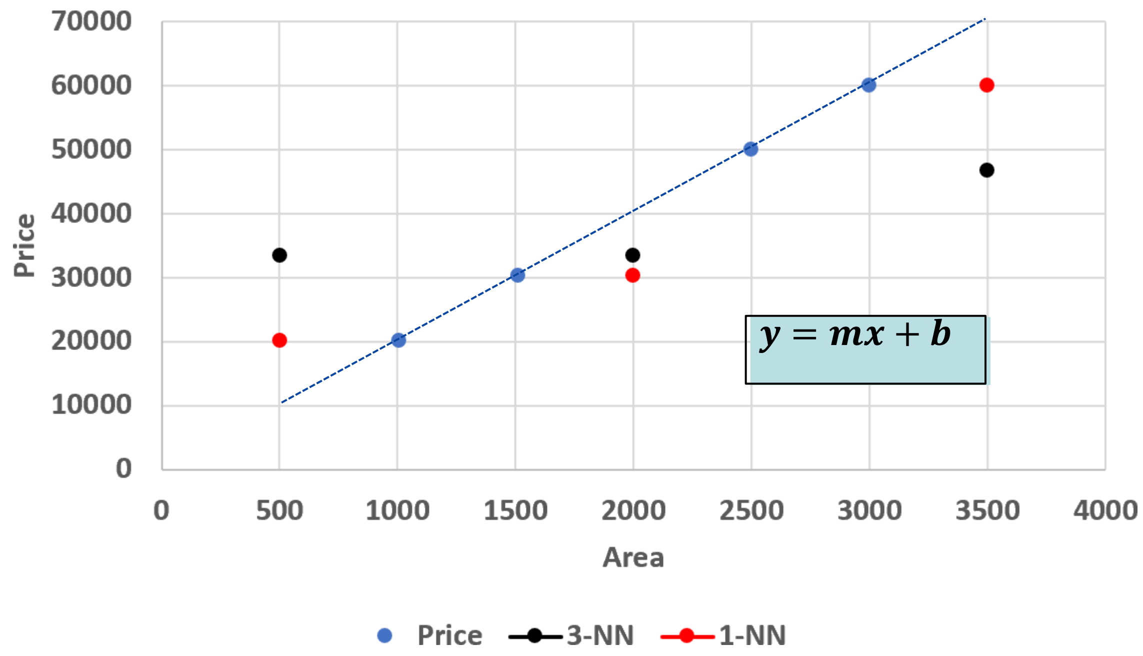

Regression line
y
Error
x

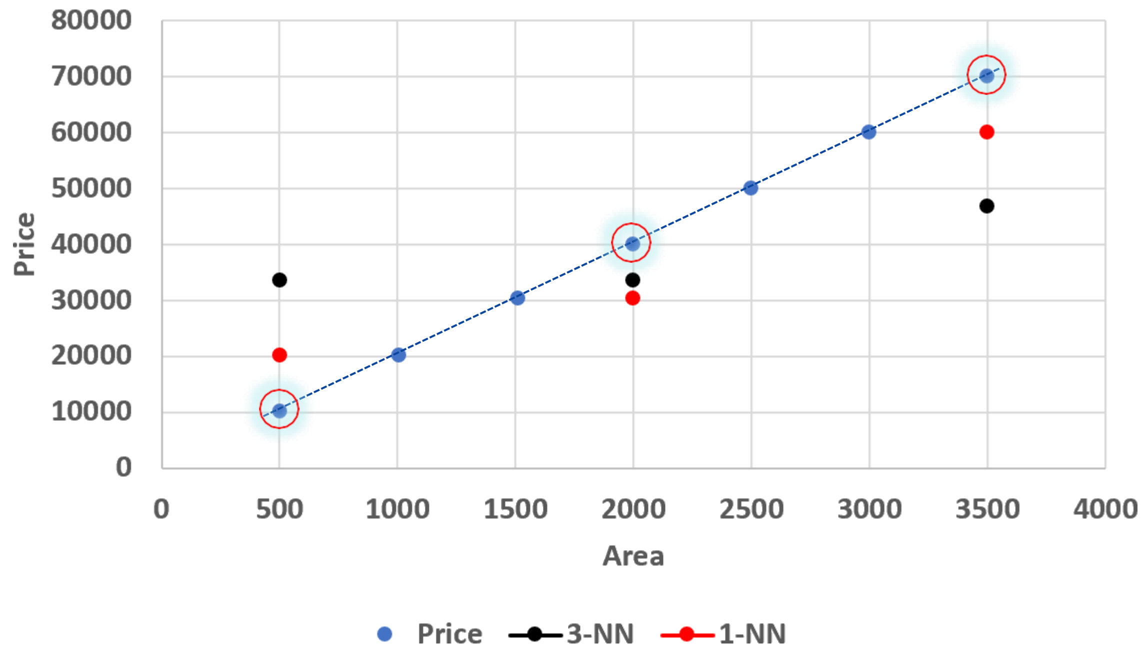
Find the price!

Area	Price
1,510	30,250
1,005	20,150
2,500	50,050
3,000	60,050

Price for Area: 500? 2000?, 3500?

Area	1-NN	3-NN
500	20,150	30,150
2,000	30,250	33,483.33
3,500	60,050	50,050





The Dot Product is Commutative

For two vectors, $\boldsymbol{A}$ and $\boldsymbol{B}$

$$\boldsymbol{A}.\boldsymbol{B} = \boldsymbol{B}.\boldsymbol{A}$$

As

$$\boldsymbol{A}.\boldsymbol{B} = A^T B$$

And

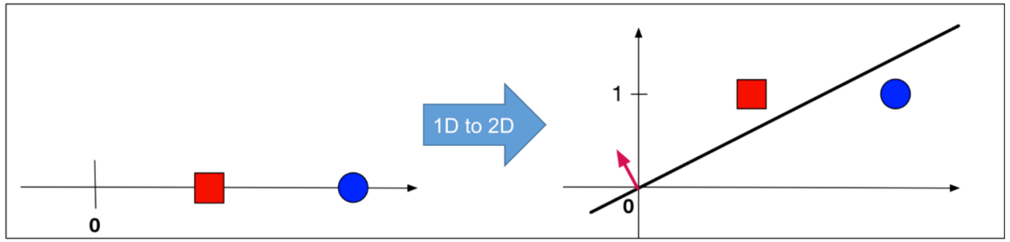
$$\boldsymbol{B}.\boldsymbol{A} = B^T A$$

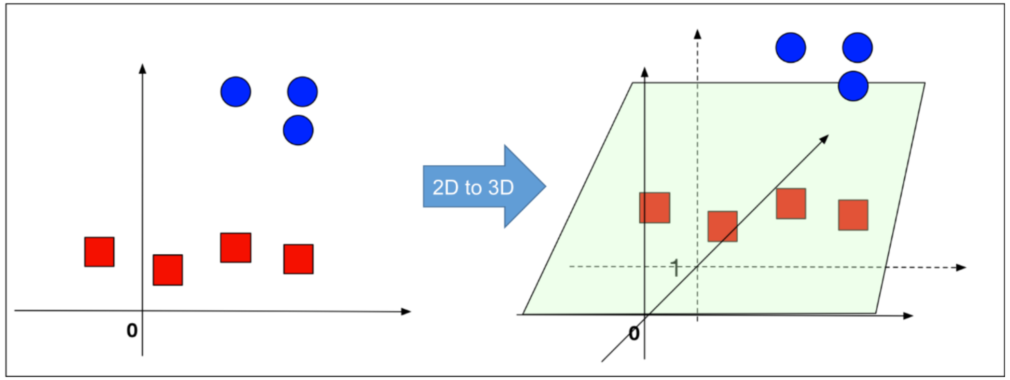
Therefore,

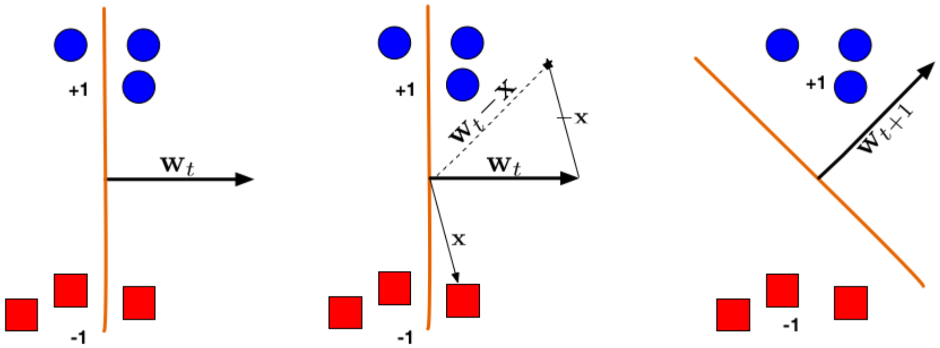
$$A^T B = B^T A$$

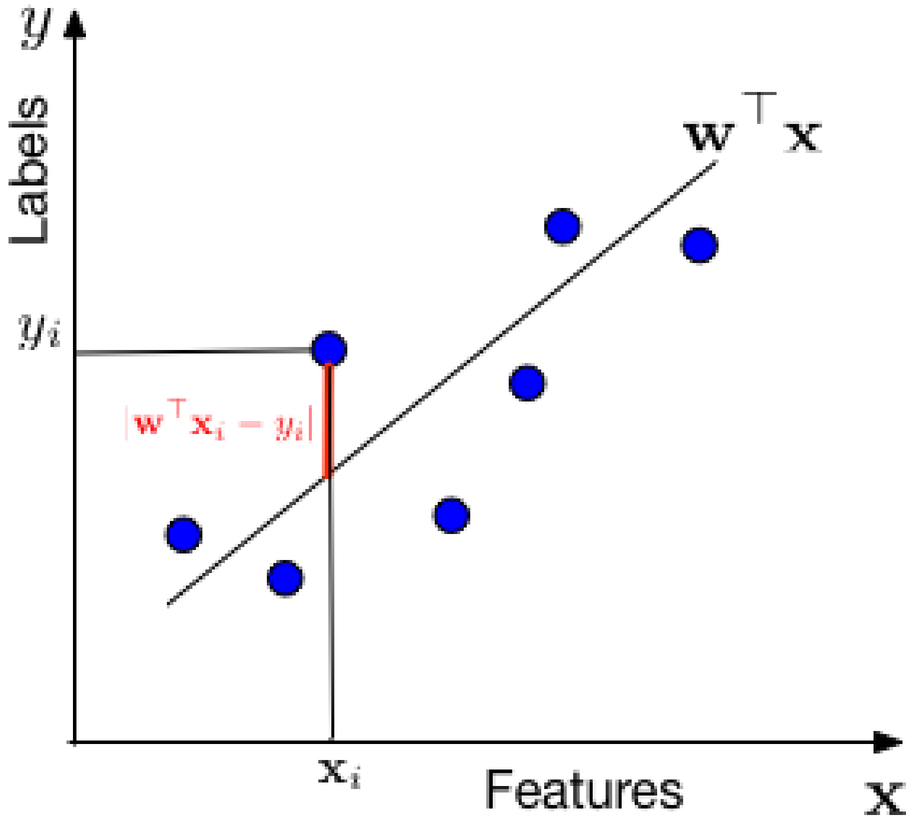
Geometric Interpretation of absorbing the Bias

(Kilian Weinberger, Lecture 3: The Perceptron, https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html)









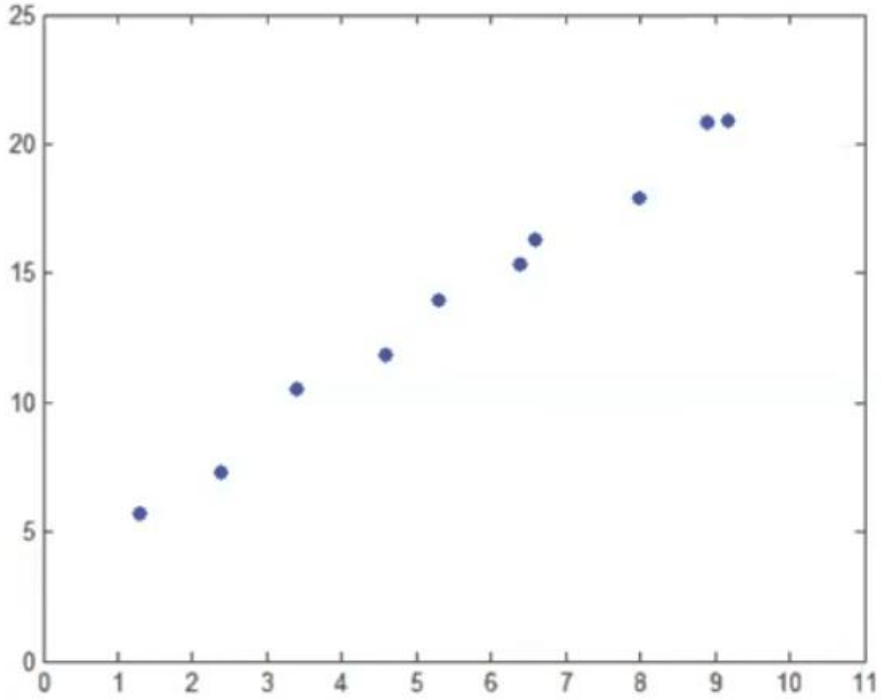
y
Labels
$\mathbf{w}^\top \mathbf{x}$
y_i
$|\mathbf{w}^\top \mathbf{x}_i - y_i|$
$\mathbf{x}_i$
Features
$\mathbf{x}$

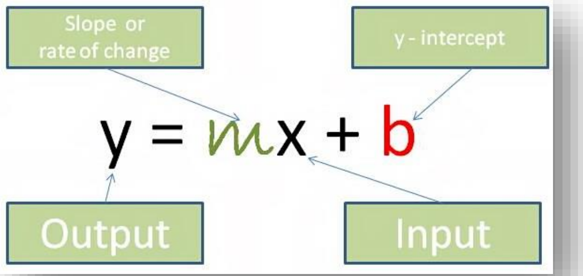
A quick overview of it all

- Fitting a line to data
- How to predict using the equation of line
- How to fit the line to data?
    - Cost function: Mean Square Error
    - Plot of the cost function
    - Gradients of the cost function
    - Stepping down the slopes: Gradient descent
    - Convex and non-convex cost functions
    - Local and global minima

Linear Regression

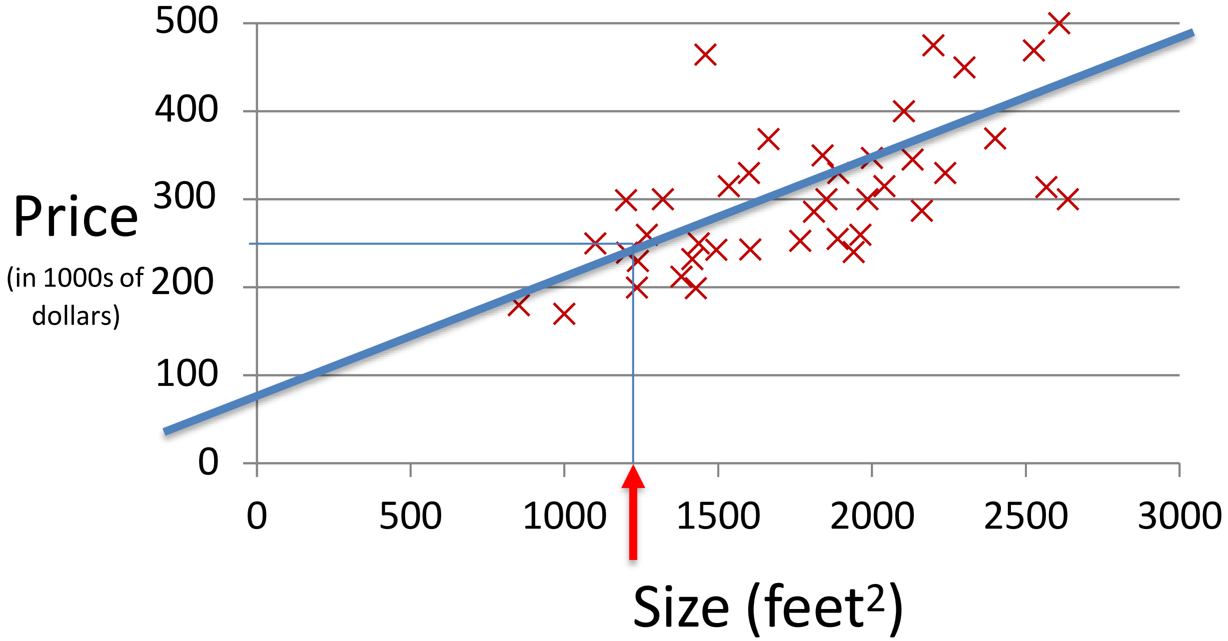
x	t
1.3	5.7
2.4	7.3
3.4	10.5
4.6	11.8
5.3	13.9
6.6	16.3
6.4	15.3
8.0	17.9
8.9	20.8
9.2	20.9





Linear Regression with one variable

Size ($feet^2$)	Price $(\times 1000)$
1500	190
2250	285
2740	420
2318	300
2500	350
1250	180
…	…



Notation:

$\boldsymbol{m}$ = Number of training samples

$\boldsymbol{n}$ = Number of features

$\boldsymbol{x_j}$ = jth feature

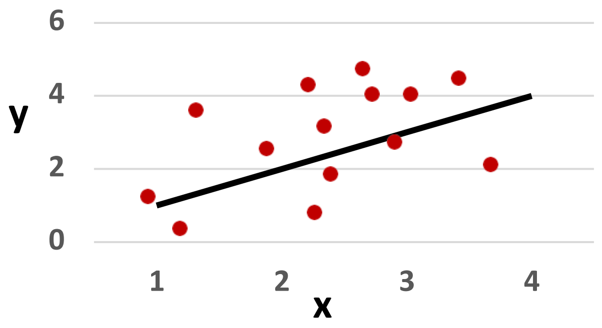
$\boldsymbol{y}$ = Label

$(\boldsymbol{x^i}, \boldsymbol{y^i})$: the ith sample in the dataset

Linear Regression with one variable

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

Linear regression with one variable, univariate linear regression, simple linear regression



Choose θ_0, θ_1 such that $h_\theta(x) \approx y$ for our training examples (x, y).

Parameters

$$\theta_0, \theta_1$$

Cost function

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

Goal:

$$Minimize_{\theta_0,\theta_1} J(\theta_0, \theta_1)$$

A simplified case

Hypothesis

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

Parameters

$$\theta_0, \theta_1$$

Cost function

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

Goal:

$$Minimize_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

Assume $\boldsymbol{\theta_0 = 0}$

Hypothesis

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

Parameters

$$\theta_1$$

Cost function

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

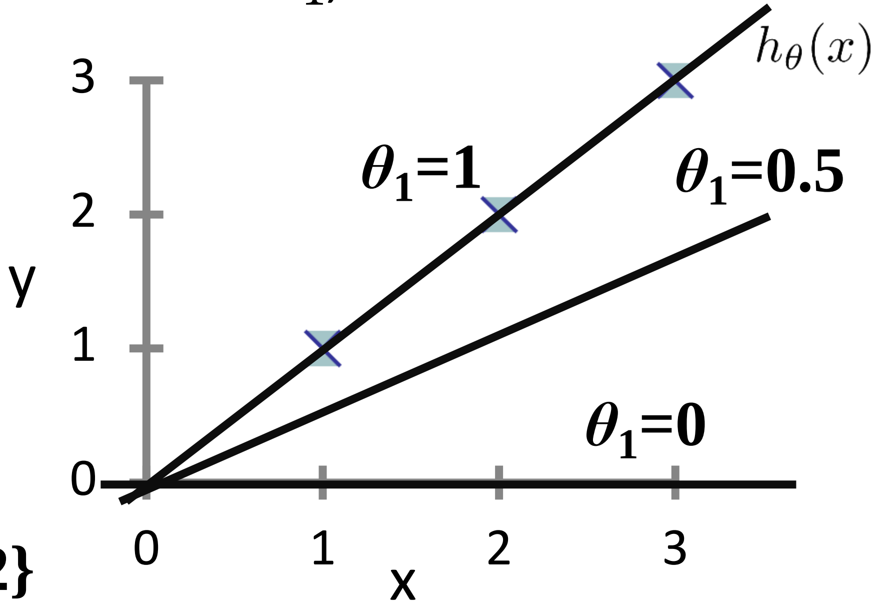
Goal:

$$Minimize_{\theta_1} J(\theta_1)$$

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

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

x	y
1	1
2	2
3	3



$\theta_1 = \{0,1,2\}$

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

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

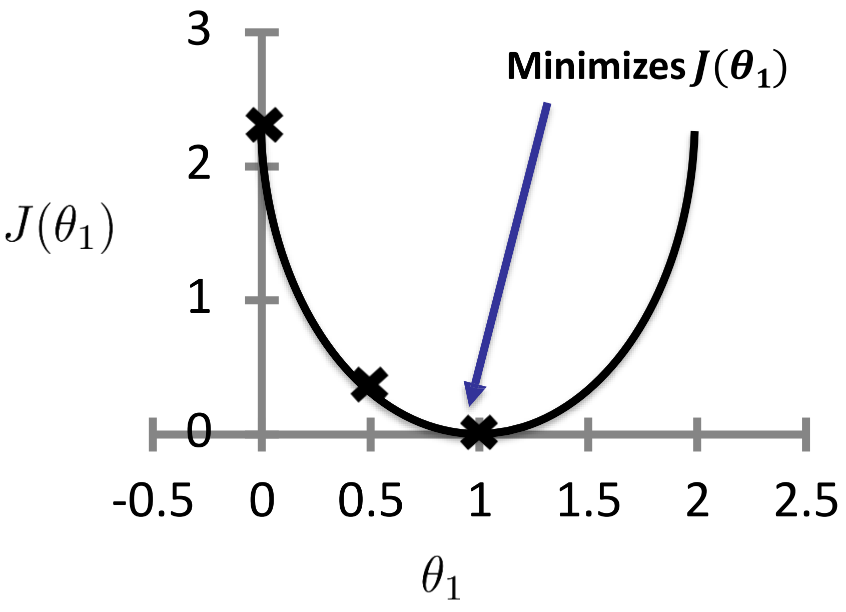
$J(\theta_1)$

(function of the parameter θ_1)

$$J(1) = \frac{1}{2(3)}\left((1-1)^2 + (2-2)^2 + (3-3)^2\right) = 0$$

$$J(0.5) = \frac{1}{6}\left((0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right) = 0.58$$

$$J(0) = \frac{1}{6}\left((1)^2 + (2)^2 + (3)^2\right) = 2.3$$



Using both of the “knobs”

Hypothesis

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

Parameters

$$\theta_0, \theta_1$$

Cost function

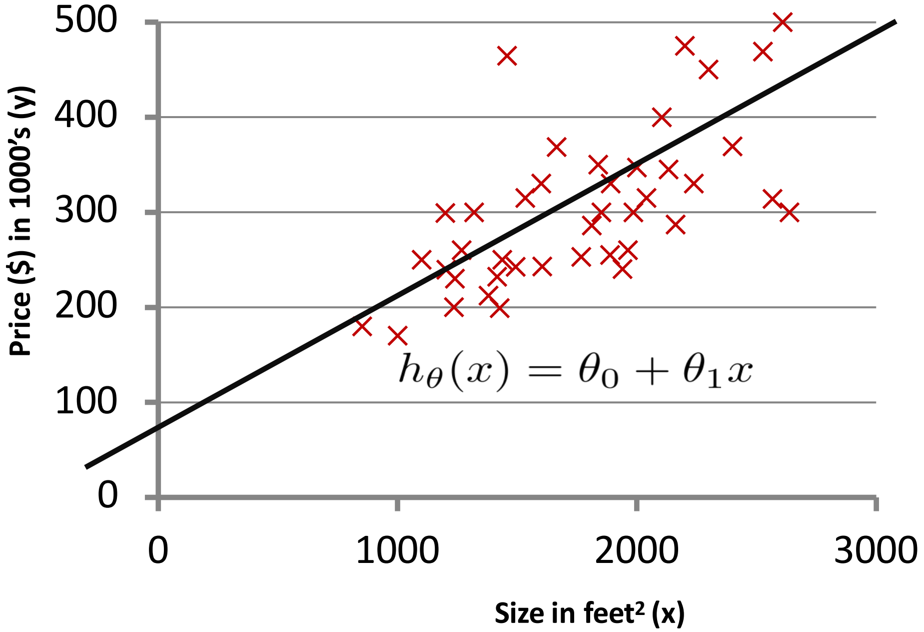
$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

Goal:

$$Minimize_{\theta_0, \theta_1} J(\theta_0, \theta_1)$$

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

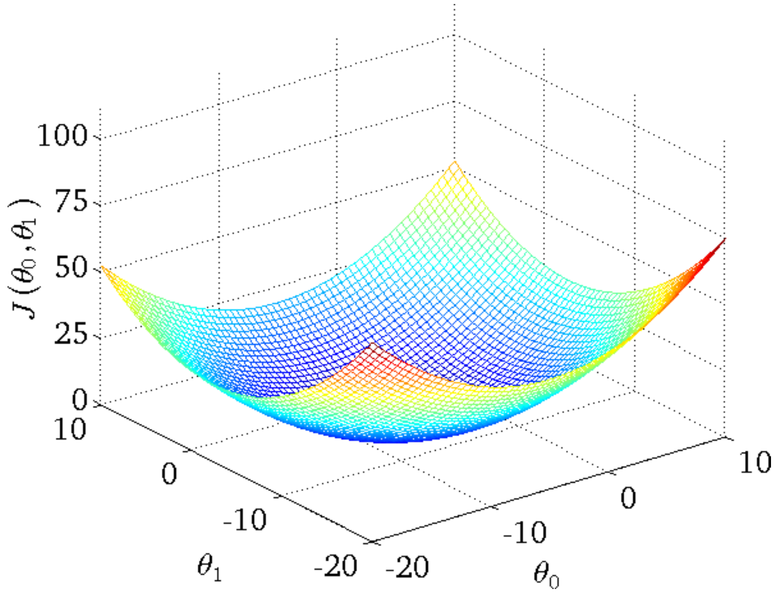
(for fixed θ_0, θ_1, this is a function of x)



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

(function of the parameters θ_0, θ_1)

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

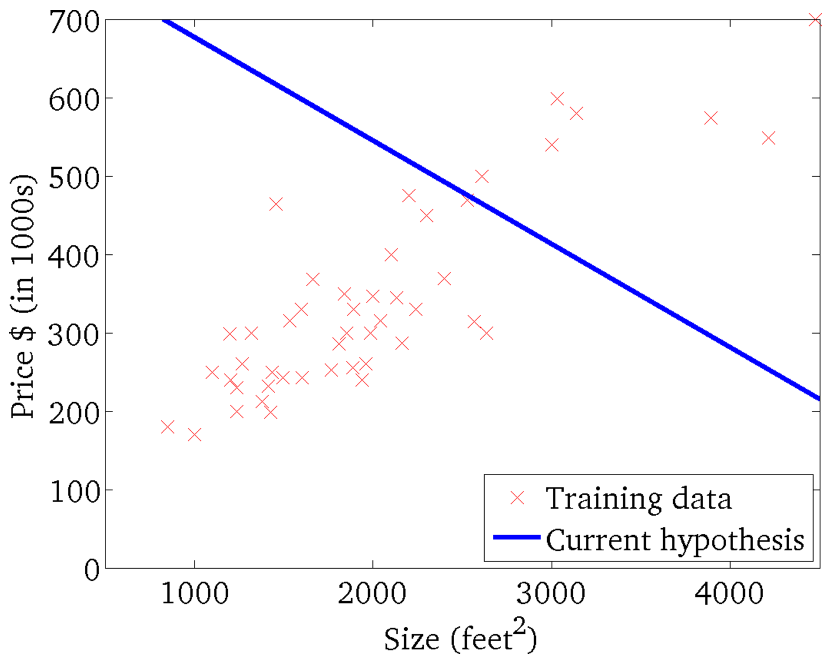


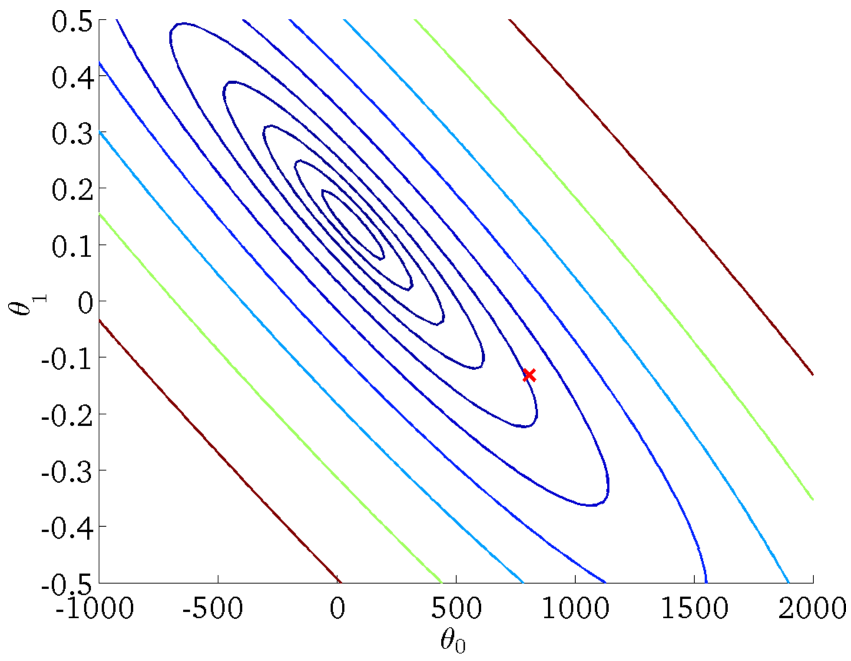
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)





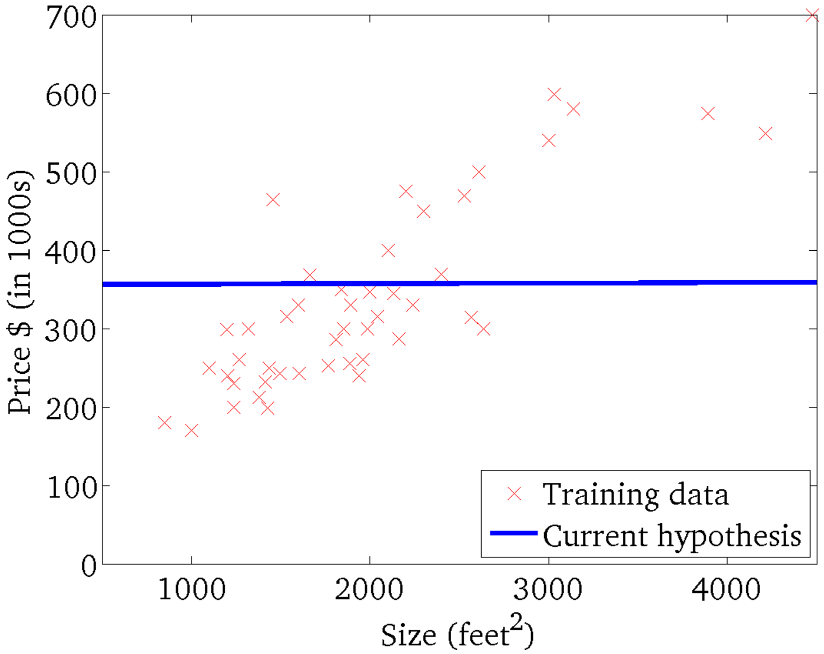
Andrew Ng

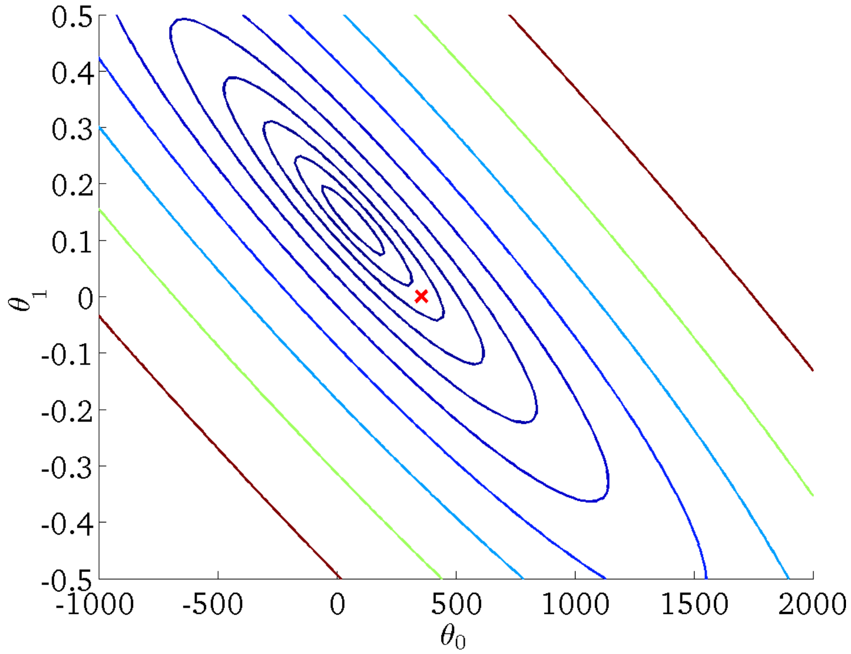
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)





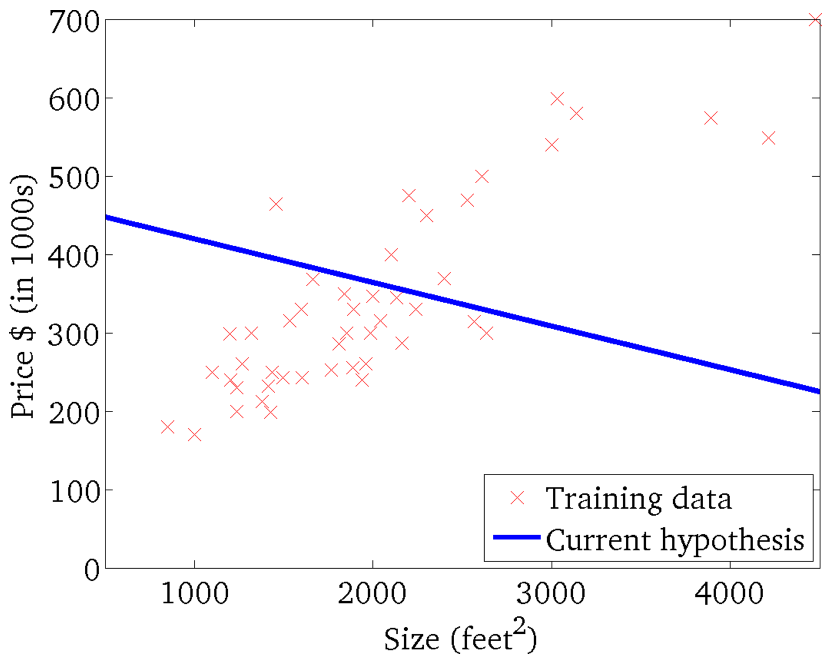
Andrew Ng

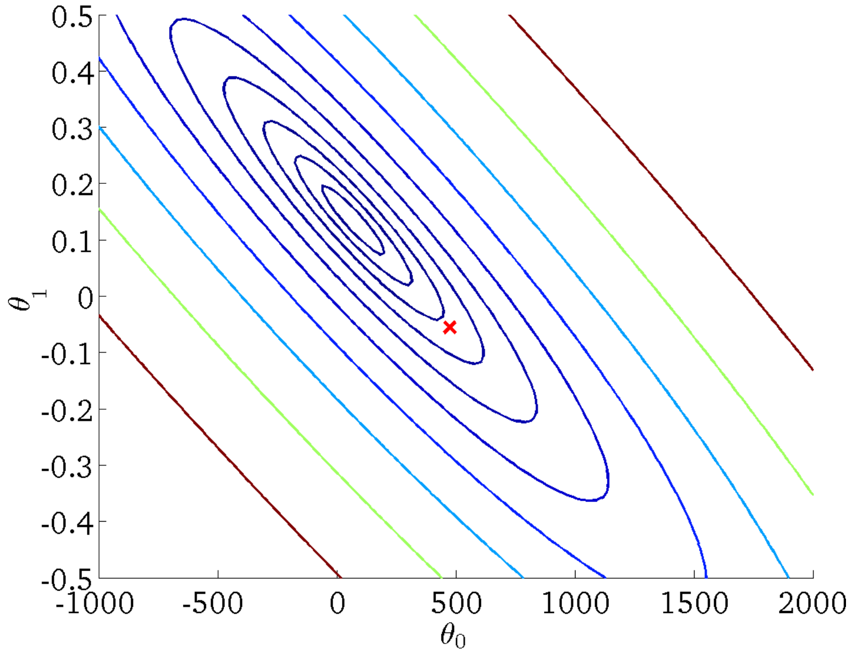
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)





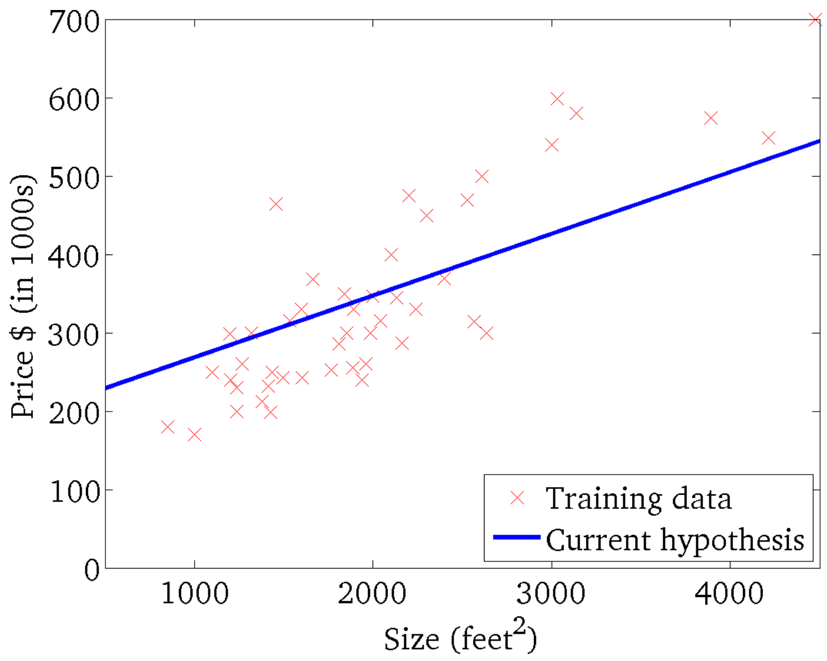
Andrew Ng

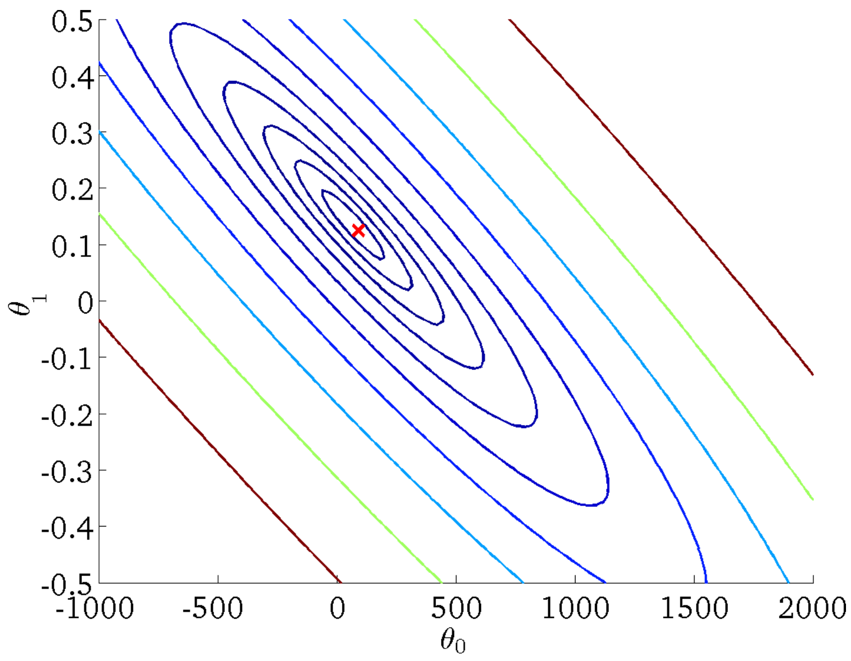
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)





Andrew Ng

Gradient Descent Algorithm

The Gradient Descent Algorithm

Goal: Minimize $J(\theta_0, \theta_1)$

Outline:

- Start with some (θ_0, θ_1) - For example (0,0)
- Keep updating (θ_0, θ_1) to reduce $J(\theta_0, \theta_1)$
  - Until we **hopefully** reach **a** minimum

A simplified version of gradient descent

Assume again that we set $\theta_0 = 0$ and our hypothesis and cost function practically have only one coefficient,θ_1.

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

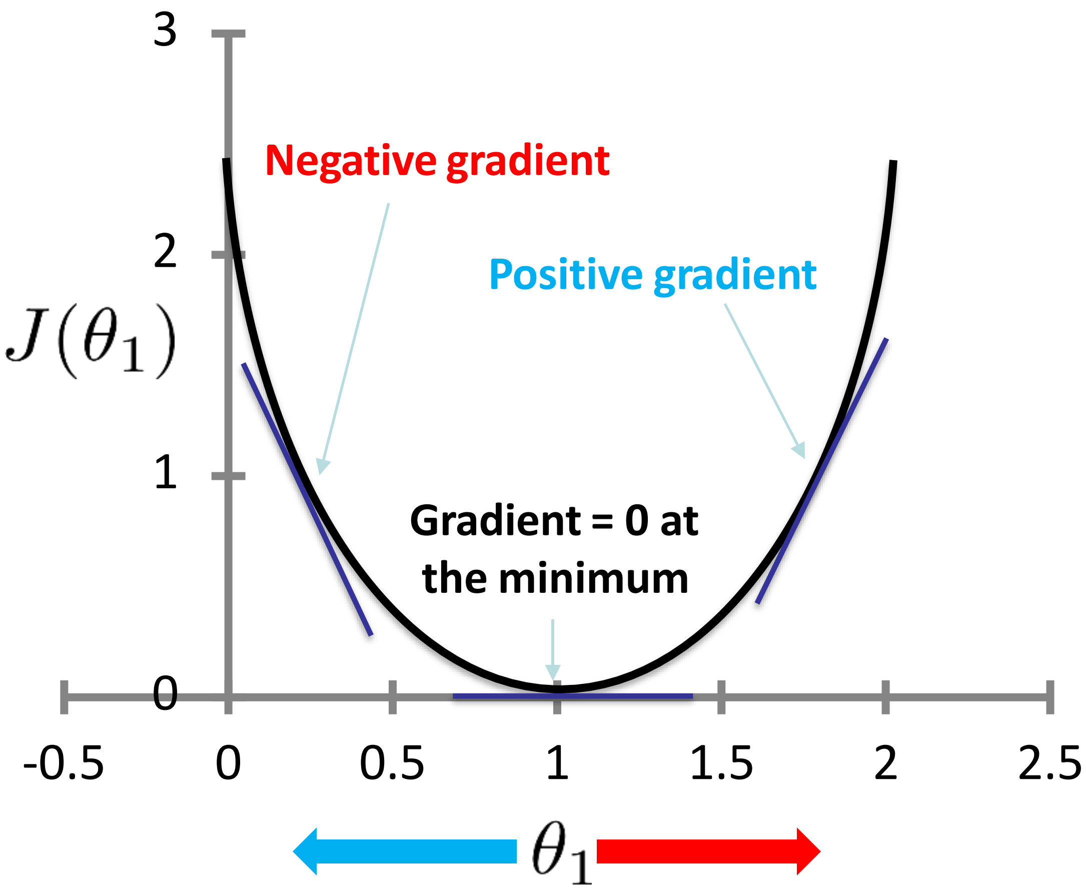
$repeat\ until\ convergence\{$

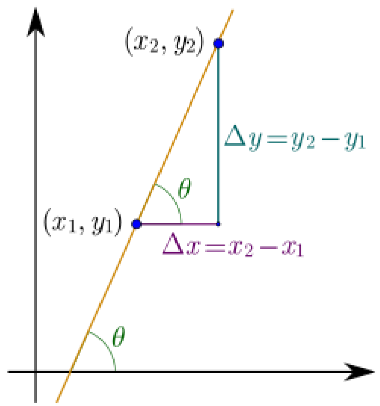
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1}(J(\theta_1))$$

$\}$

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

Direction of step





Gradient = $\frac{y_2 - y_1}{x_2 - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$

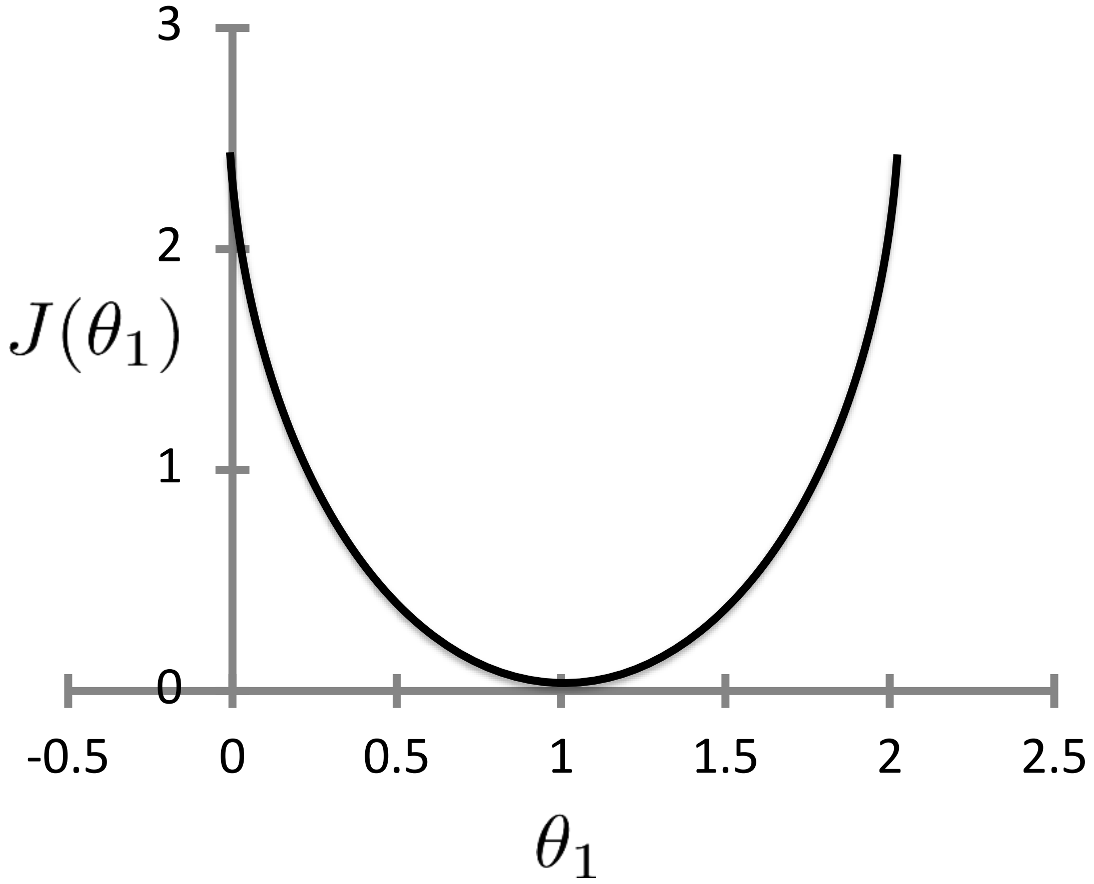
The limiting case is when x_2 approaches x_1 $= \frac{\Delta y}{\Delta x} as \Delta x$→0 = $\frac{dy}{dx}$

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

$\theta_1 := \theta_1 - \alpha(negative)$: Increases θ_1

$\theta_1 := \theta_1 - \alpha(positive)$: Decreases θ_1

Step size



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

Gradient = $\frac{y_2 - y_1}{x_2 - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$

The limiting case is when x_2 approaches x_1

$= \frac{\Delta y}{\Delta x} as \Delta x \rightarrow 0 = \frac{dy}{dx}$

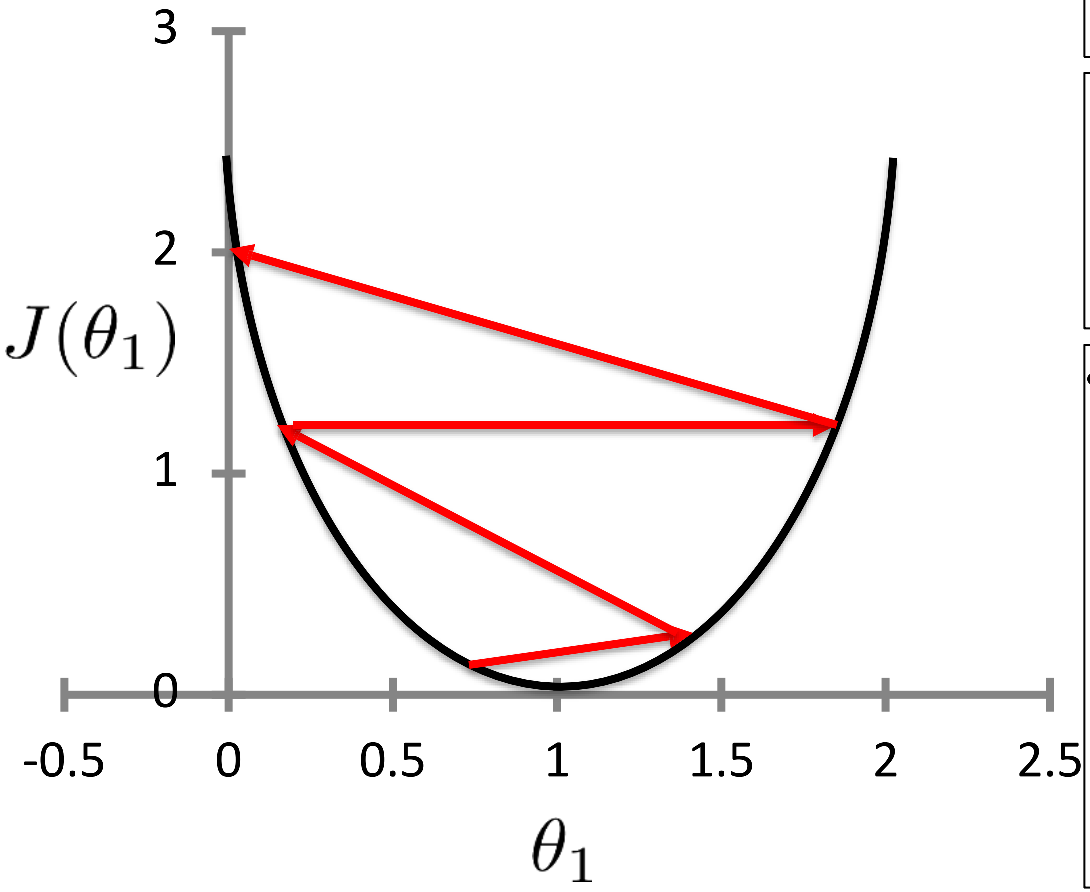
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1}\left(J(\theta_1)\right)$$

$\theta_1 := \theta_1 - \alpha\,(negative)$: Increases θ_1

$\theta_1 := \theta_1 - \alpha\,(positive)$: Decreases θ_1

- In addition to α, the steepness of the gradient also control the step size.
- The step sizes become smaller as we get closer to the minimum, even with a fixed α.
- With α too small, takes a long time to reach the minimum

Step size



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

Gradient = $\frac{y_2 - y_1}{x_2 - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$

The limiting case is when x_2 approaches x_1 $= \frac{\Delta y}{\Delta x} as \Delta x$ → 0 = $\frac{dy}{dx}$

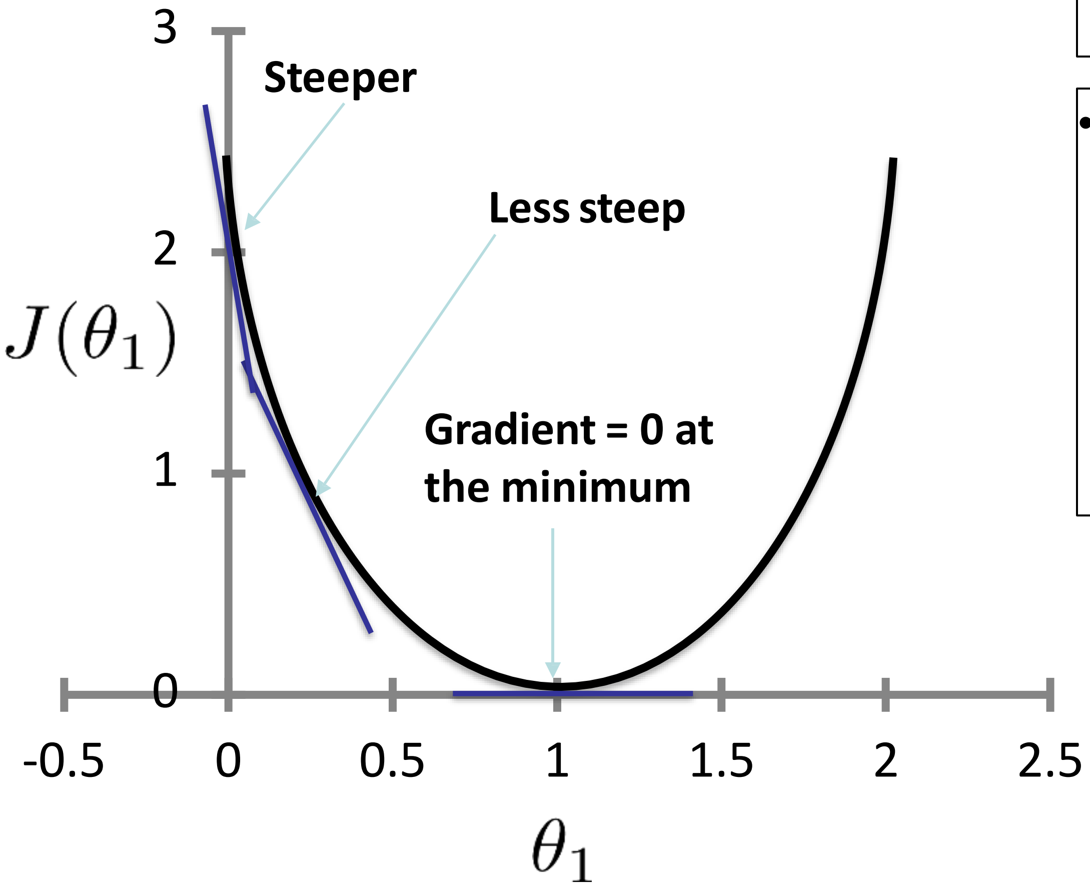
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1}(J(\theta_1))$$

$\theta_1 := \theta_1 - \alpha(negative)$: Increases θ_1

$\theta_1 := \theta_1 - \alpha(positive)$: Decreases θ_1

- In addition to α, the steepness of the gradient also control the step size.
- The step sizes become smaller as we get closer to the minimum, even with a fixed α.
- With α too small, takes a long time to reach the minimum
- With α too big, we can miss the minimum, and may fail to converge

Step size



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

Gradient = $\frac{y_2 - y_1}{x_2 - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1}$

The limiting case is when x_2 approaches x_1

$= \frac{\Delta y}{\Delta x}$ as $\Delta x \to 0 = \frac{dy}{dx}$

- In addition to α, the steepness of the gradient also control the step size.
- The step sizes become smaller as we get closer to the minimum, even with a fixed α.
- No need to decrease α with time or number of steps

Gradient Descent by Hand

$$\boldsymbol{h_\theta(x) = \theta_1 x}$$

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

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

$$\theta_1 = 0.1$$

$$\alpha = 0.1$$

$$repeat\ until\ convergence\{$$

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

$$\}$$

x	Y	predictions1	predictions2
1	2	0.1	0.986
2	4	0.2	1.972
3	6	0.3	2.958

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

Cost = 8.423, theta_1 = 0.1
Theta_1 = 0.986 (after first iteration)

Cost = 2.39, theta_1 = 0.986
Theta_1 = 1.466 (after second iteration)

Gradient Descent by Hand

$$\boldsymbol{h_\theta(x) = \theta_0 + \theta_1 x}$$

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

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

x	Y
1	2
2	4
3	6

$$\theta_1 = 0.1$$

$$\alpha = 0.5$$

$repeat\ until\ convergence\{$

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

$\}$

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

Gradient Descent by Hand

$$\boldsymbol{h_\theta}\,(\boldsymbol{x}) = \boldsymbol{\theta_0} + \boldsymbol{\theta_1 x}$$

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

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

x	Y
1	2
2	4
3	6

$$\theta_1 = 0.1$$

$$\alpha = 0.9$$

$repeat\ until\ convergence\{$

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

$\}$

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

Gradient Descent by Hand

$$\boldsymbol{h_\theta(x) = \theta_0 + \theta_1 x}$$

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

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

x	Y
2	4
4	8
6	12

$$\theta_1 = 0.5$$

$$\alpha = 0.1$$

$repeat\ until\ convergence\{$

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

$\}$

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

Gradient Descent Algorithm

$$\text{repeat until convergence } \{$$
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad (\text{for } j = 0 \text{ and } j = 1)$$
$$\}$$

(Where α is the learning rate. E.g., 1, 0.1, 0.01, 0.001,… etc.)

Correct: **Simultaneous update**

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\theta_1 := \text{temp1}$$

Incorrect:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$
$$\theta_0 := \text{temp0}$$
$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$
$$\theta_1 := \text{temp1}$$

Gradient Descent Algorithm

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \qquad \text{(simultaneously update } j = 0 \text{ and } j = 1) \\
&\}
\end{aligned}
$$

Gradient Descent Algorithm

$$\text{repeat until convergence } \{$$
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$
$$(\text{for } j = 1 \text{ and } j = 0)$$
$$\}$$

Linear Regression Model

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

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

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

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

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

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

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

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

Gradient Descent Algorithm

repeat until convergence $\{$

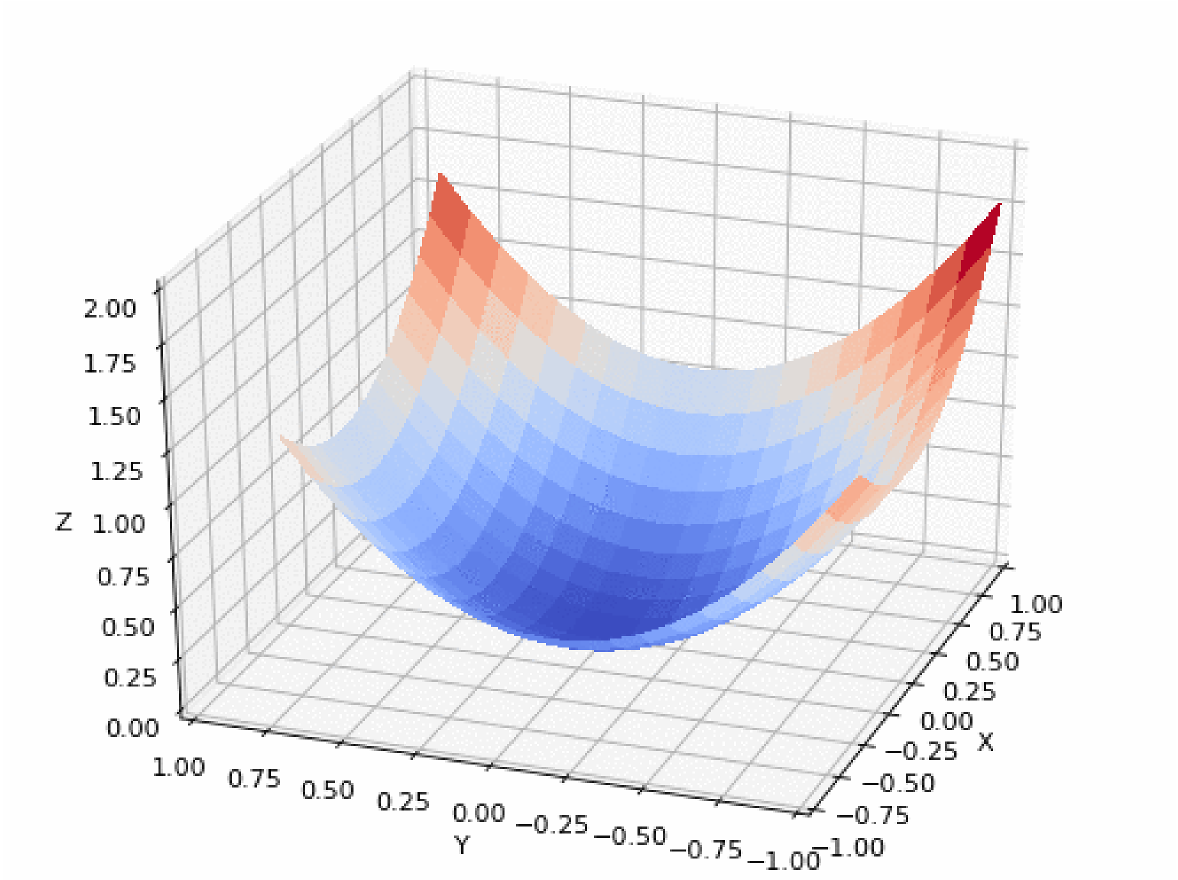
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right)$$

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

$\}$

update θ_0 and θ_1 simultaneously

Convex function



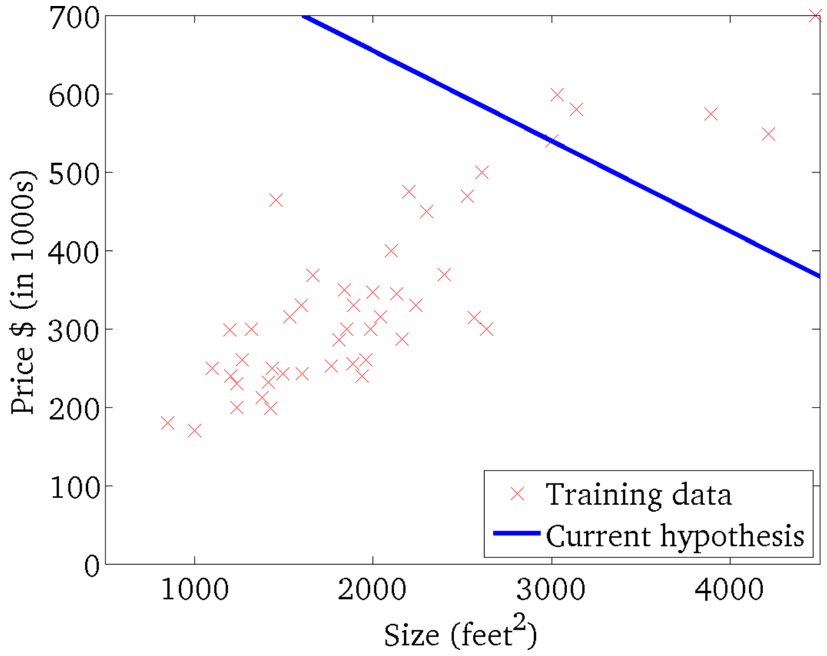
https://github.com/pvigier/gradient-descent

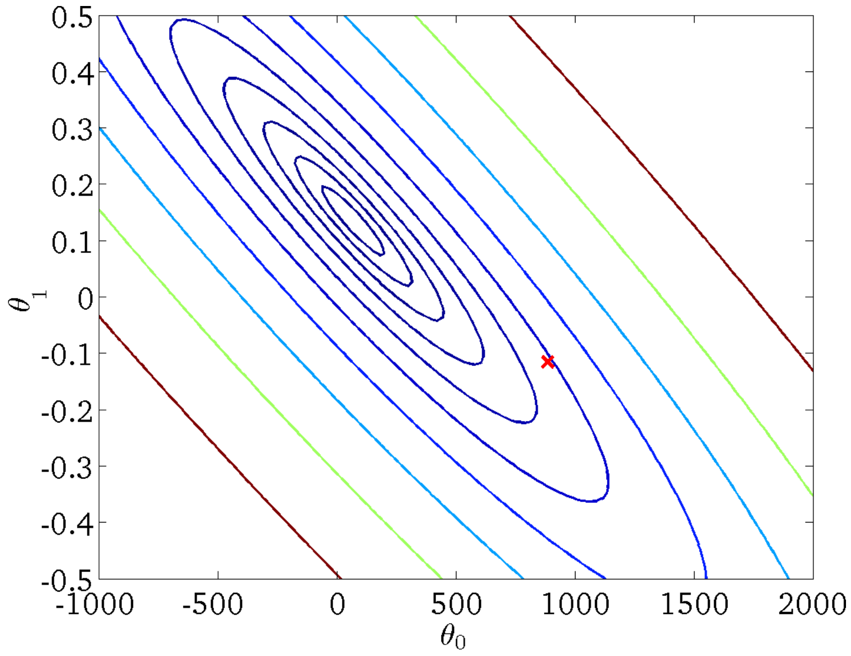
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



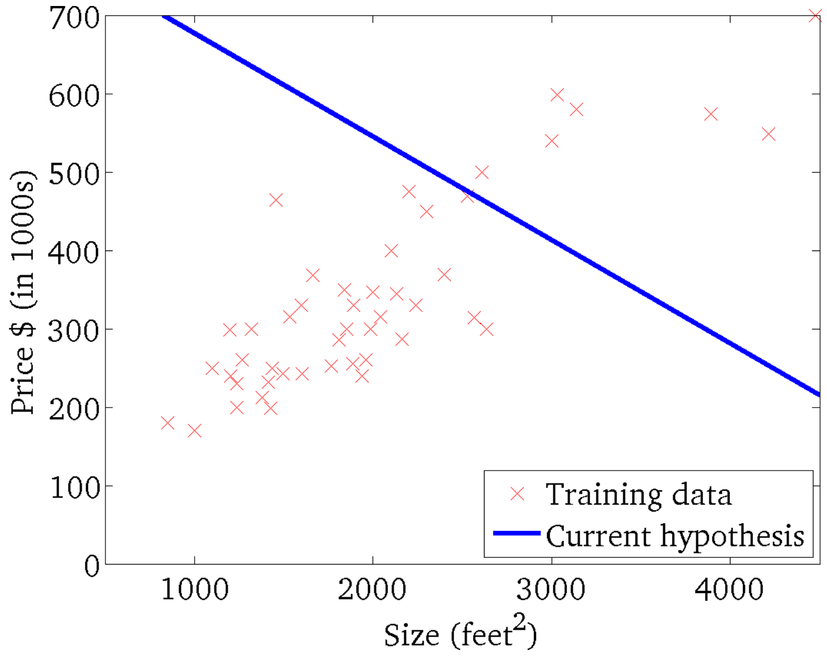


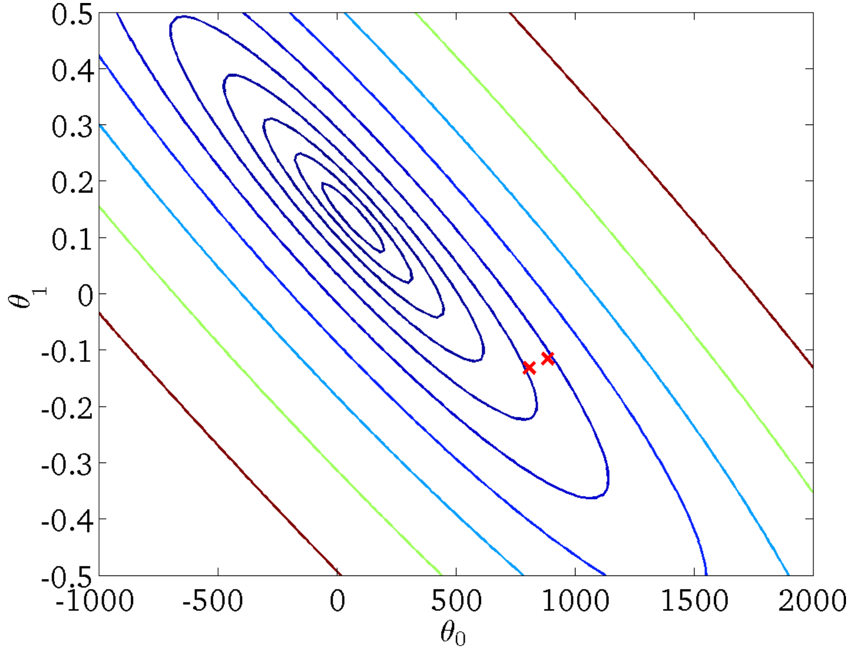
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



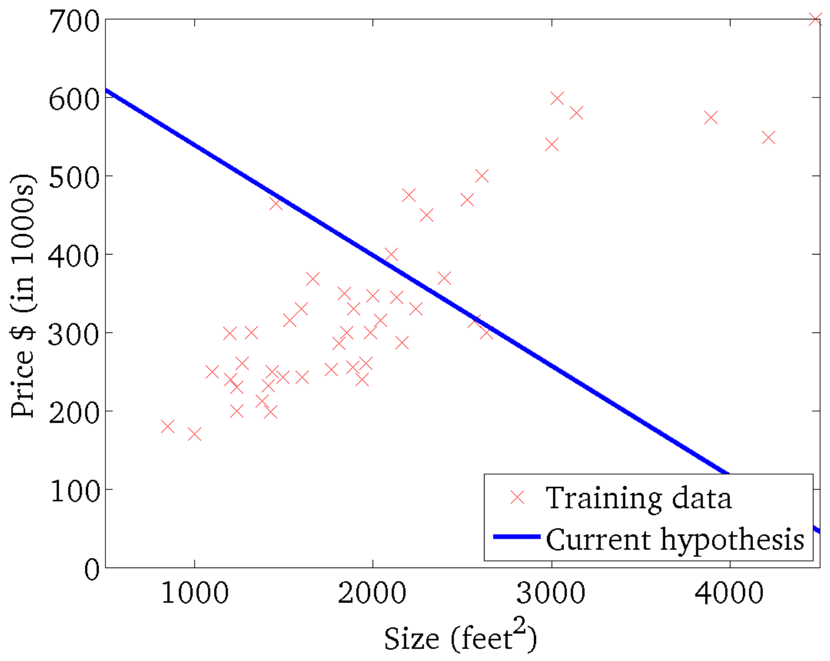


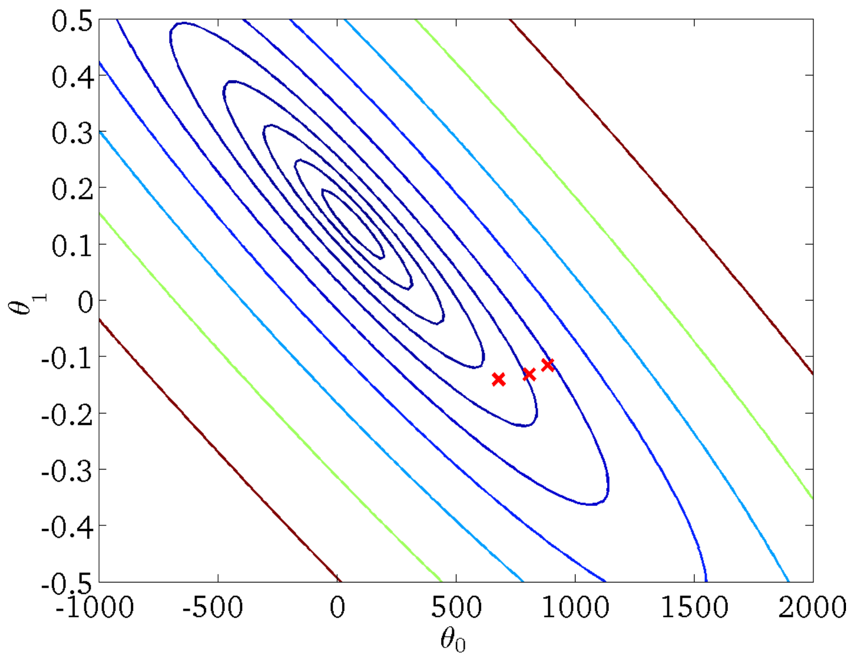
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



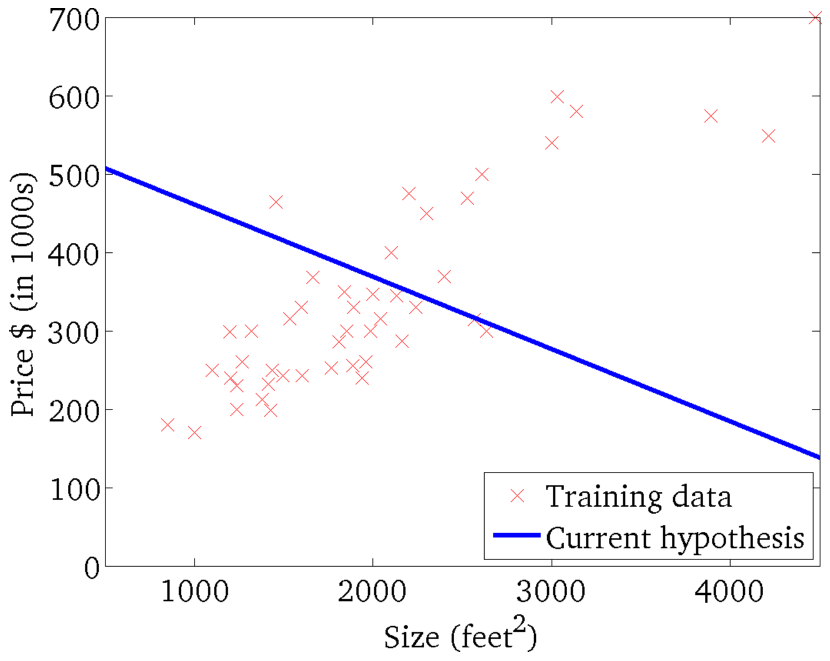


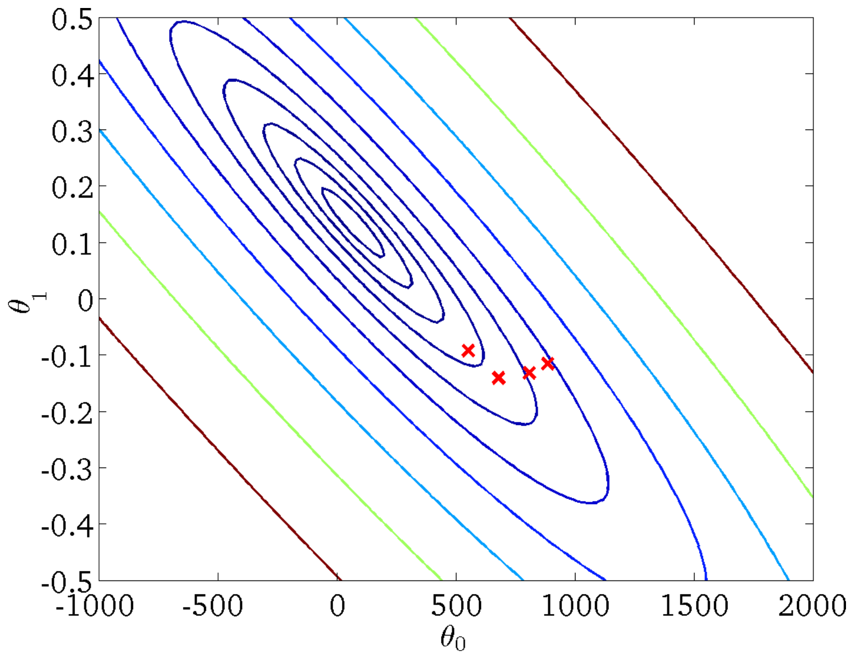
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



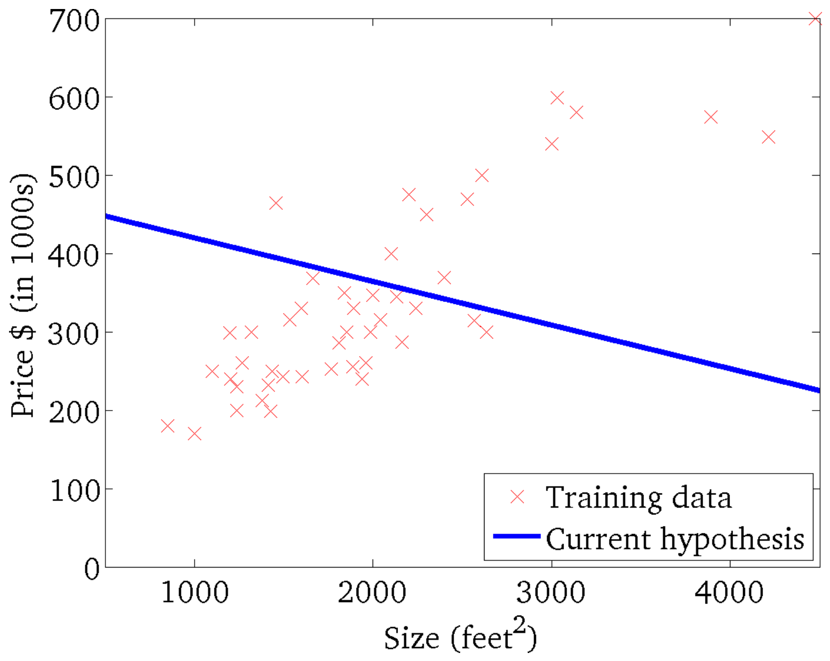


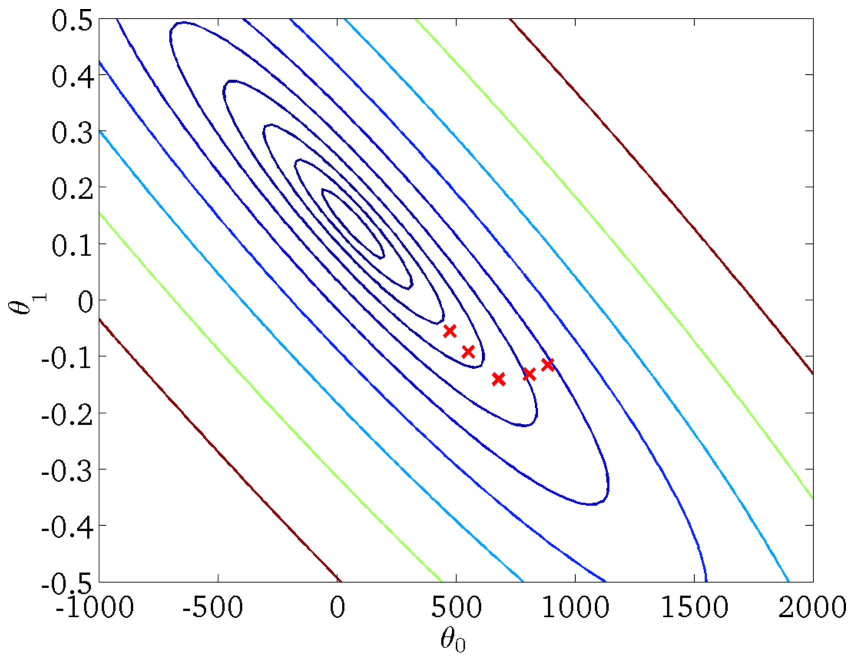
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



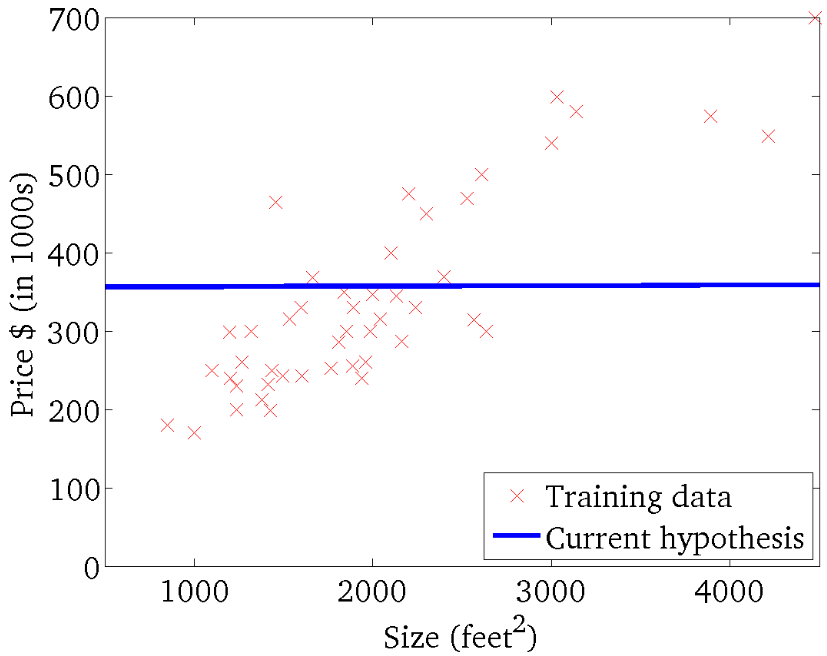


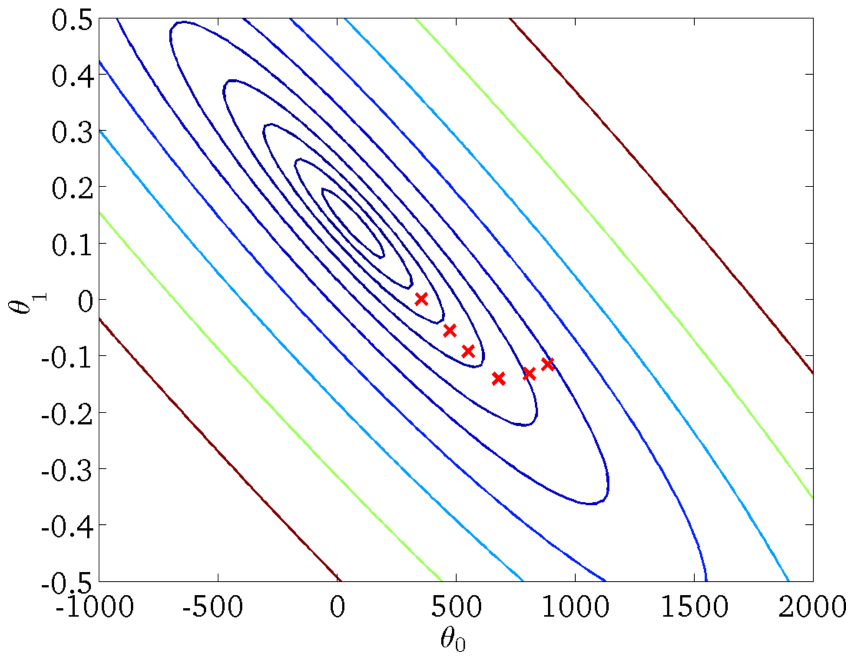
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



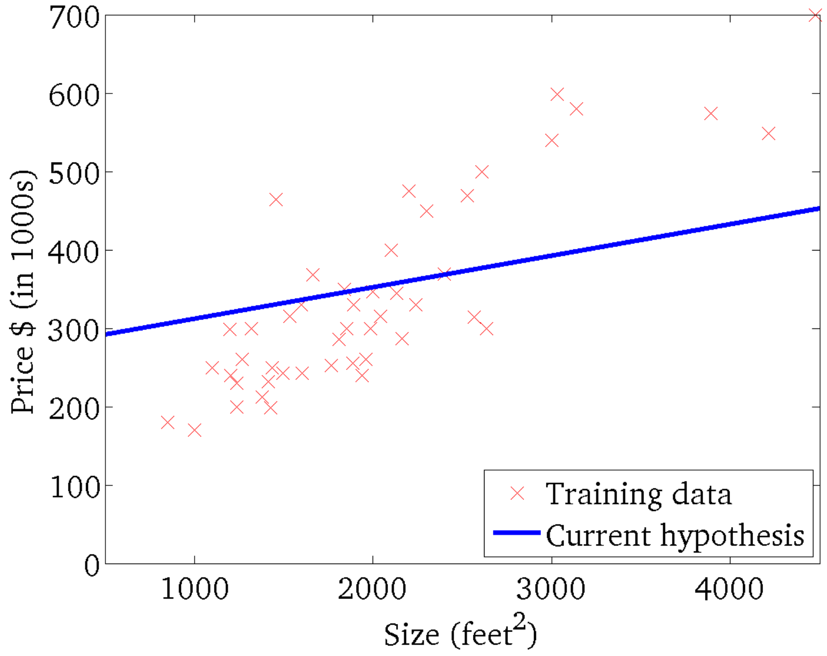


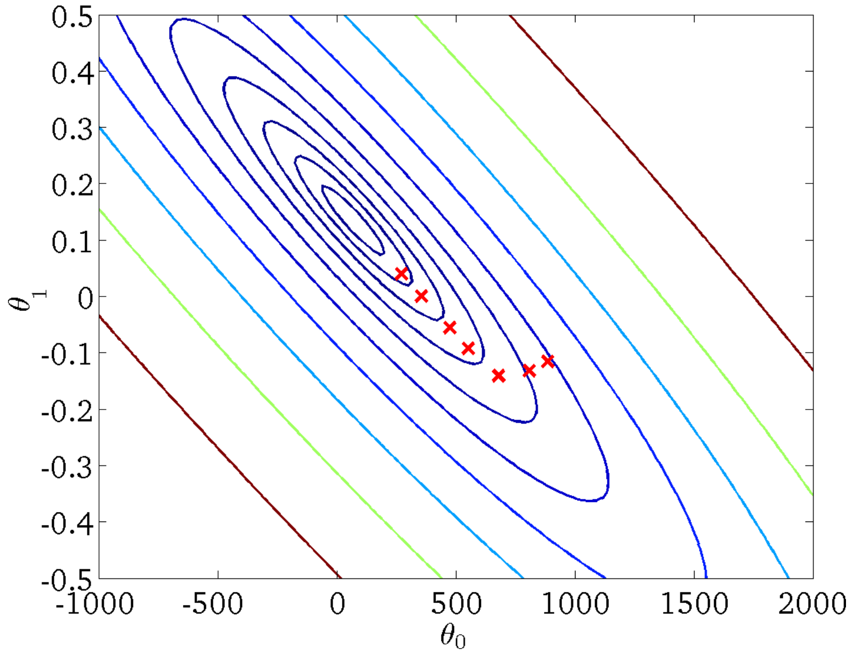
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



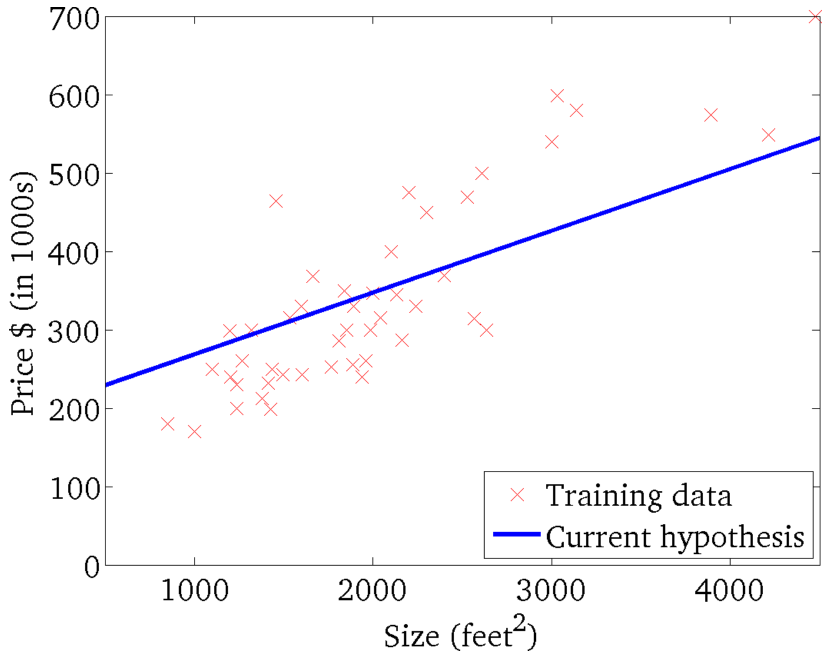


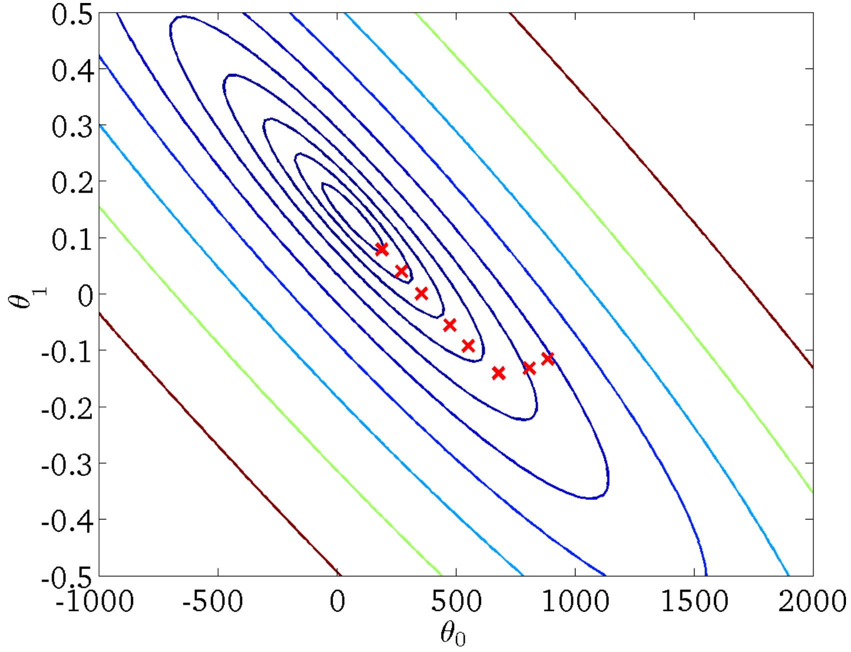
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)



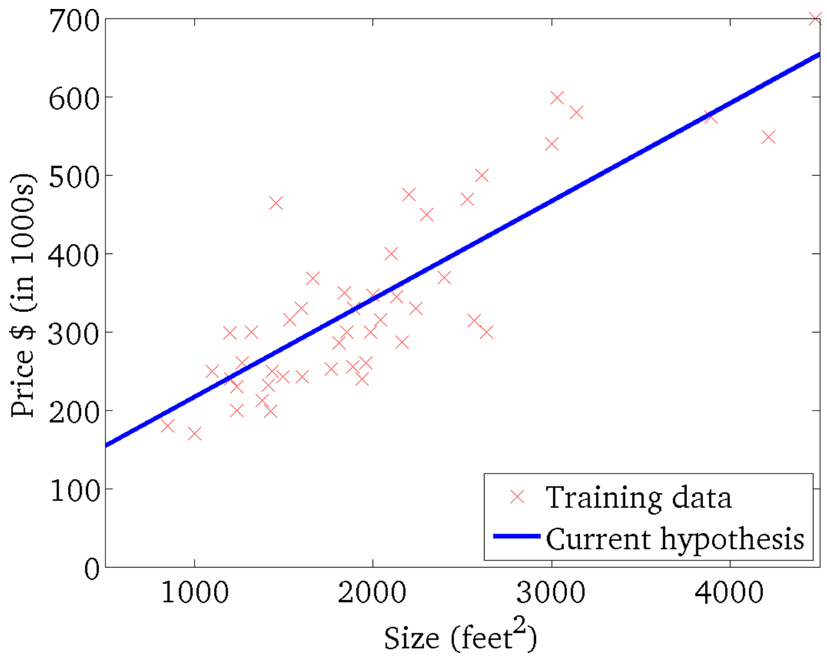


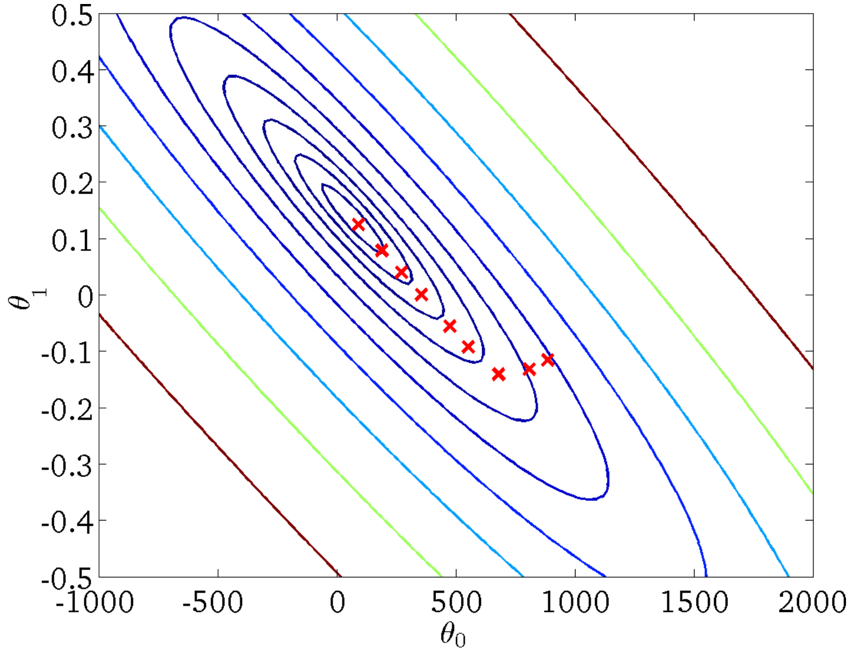
$$h_\theta(x) = \theta_0 + \theta_1 x$$

(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)





$$w_j \leftarrow w_j - \eta \frac{\partial E}{\partial w_j}$$

$$w_j \leftarrow w_j + \Delta w_j$$

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j}$$

$$\Delta w_j = \eta (t - y) x_j^{(i)}$$

Batch Gradient Descent

"Batch": Each step of gradient descent uses all the training examples.

$$
\begin{aligned}
&\text{repeat until convergence } \{ \\
&\quad \theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \\
&\quad \theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)} \right) \cdot x^{(i)} \\
&\}
\end{aligned}
$$

Types of Gradient Descent

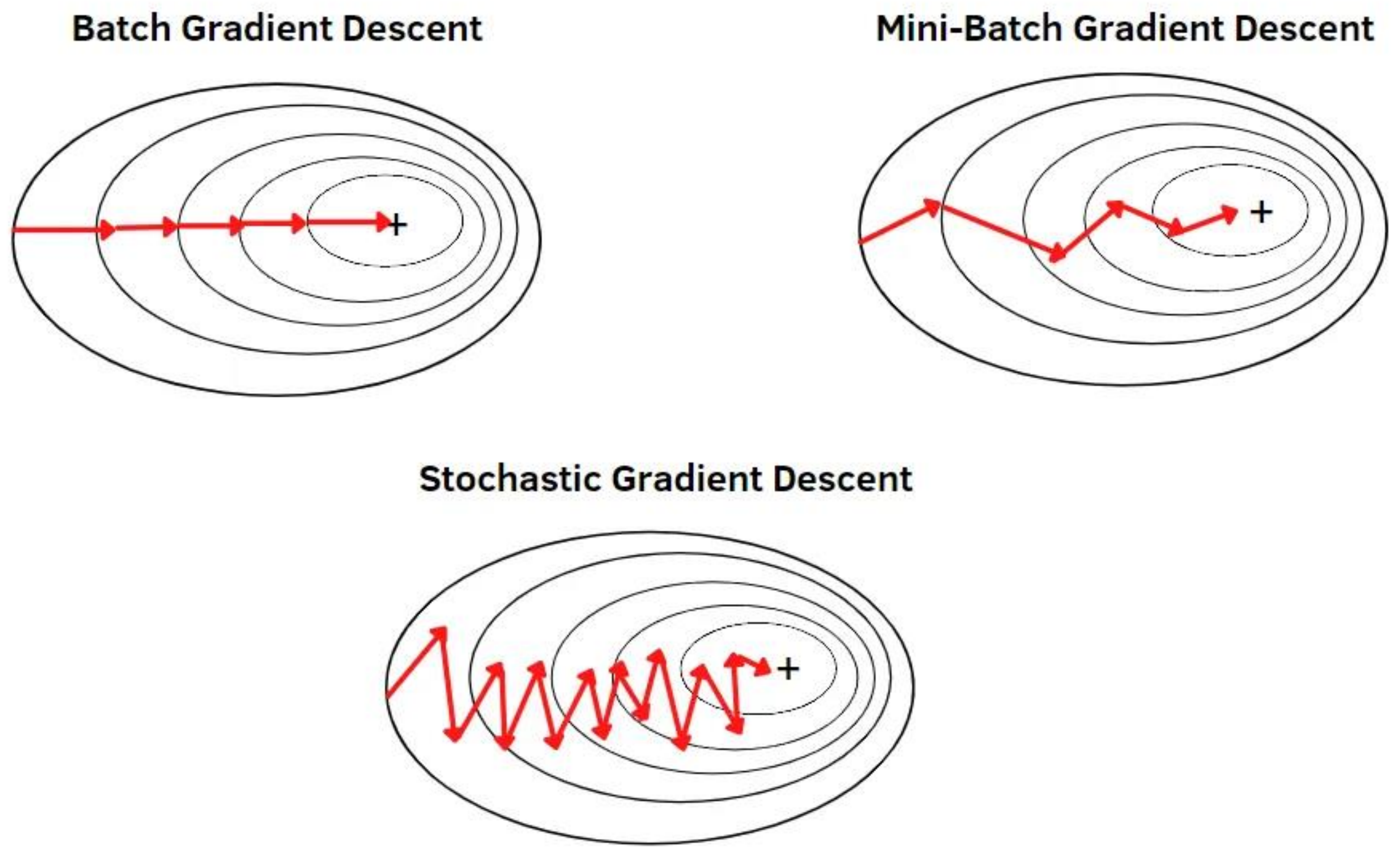
Batch gradient descent: Computes the gradient of the cost function with respect to to the parameters θ for the entire training dataset (m instances).

Stochastic gradient descent: SGD performs a parameter update for *each* training example $x^{(i)}$ and label $y^{(i)}$

Mini-batch gradient descent: Mini-batch gradient descent takes the best of both worlds and performs an update for random mini-batches of k training examples, where $k < m$.

Read: https://ruder.io/optimizing-gradient-descent/

Types of Gradient Descent



https://medium.com/@pehansanchitha01/gradient-descent-optimization-variants-algorithms-and-challenges-ffbfcd707c89

Multivariate

Single feature (variable)

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

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

Multiple features (variables)

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
…	…	…	…	…

$$h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

Generally speaking, weights are indicating the importance of each feature.

Notation:

m = Number of training samples

n = Number of features

x = Feature

y = Label

($x_j^{(i)}$, $y_j^{(i)}$): the jth feature of the ith sample in the dataset

Andrew Ng

Hypothesis

Previously:

$$\boldsymbol{h_\theta(x) = \theta_0 + \theta_1 x}$$

Now:

$$\boldsymbol{h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

where, $X = [x_1\ x_2\ \ldots\ x_n]$ is the n-dimensional feature vector, and $\Theta = [\theta_0\ \theta_1\ \ldots\ \theta_n]$ is an (n+1)-dimensional vector of weights.

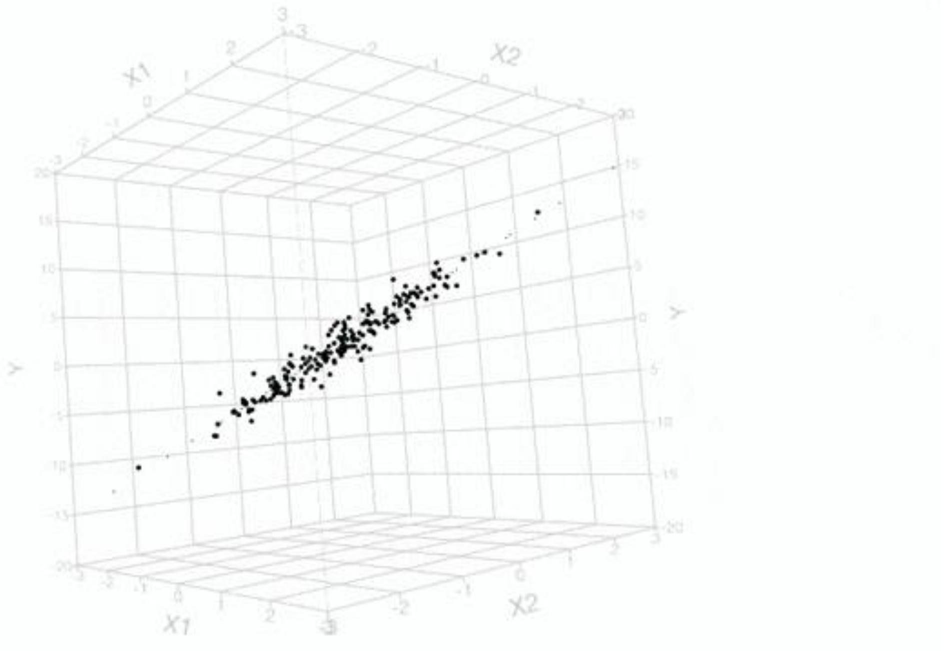
$$\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n}}$$
$$\boldsymbol{\Theta} \in \mathbb{R}^{\boldsymbol{n+1}}$$

Geometric Interpretation

$$\boldsymbol{h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

where, $X = [x_1\ x_2\ \ldots\ x_n]$ is the n-dimensional feature vector, and $\Theta = [\theta_0\ \theta_1\ \ldots\ \theta_n]$ is an (n+1)-dimensional vector of weights.

$\boldsymbol{h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2}$ (a 2-D hyperplane in 3-D)



https://www.jmp.com/en_us/statistics-knowledge-portal/what-is-multiple-regression/fitting-multiple-regression-model.html

Hypothesis

$$\boldsymbol{h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

where, $X = [x_1\ x_2\ \ldots\ x_n]$ is the n-dimensional feature vector, and $\Theta = [\theta_0\ \theta_1\ \ldots\ \theta_n]$ is an (n+1)-dimensional vector of weights.

$$\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n}}, \boldsymbol{\Theta} \in \mathbb{R}^{\boldsymbol{n+1}}$$

To make this more uniform, assume $x_0 = 1$ to get:

$$\boldsymbol{h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

where, $X = [x_0\ x_1\ x_2\ \ldots\ x_n]$ is the (n+1)-dimensional feature vector, and

$\Theta = [\theta_0\ \theta_1\ \ldots\ \theta_n]$ is an (n+1)-dimensional vector of weights.

$$\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n+1}}, \boldsymbol{\Theta} \in \mathbb{R}^{\boldsymbol{n+1}}$$

Vectorizing the notation

$$\boldsymbol{h_{\theta}(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

Now we can redefine our hypothesis as:

$\Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}, X = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}$ and where $\Theta \in \mathbb{R}^{n+1}$ and $X \in \mathbb{R}^{n+1}$

and,

$$\boldsymbol{h_{\Theta}(X) = \Theta^T X}$$

Multivariate linear regression

For m-training instances

$$\boldsymbol{h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n}$$

$$\Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix}, X = \begin{bmatrix} x_0^{(1)} & x_0^{(2)} & x_0^{(3)} & & x_0^{(m)} \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \dots & x_1^{(m)} \\ \dots & \dots & \dots & & \dots \\ x_n^{(1)} & x_n^{(2)} & x_n^{(3)} & & x_n^{(m)} \end{bmatrix}$$

$n \times 1$ $\quad$ $n \times m$

$$h_\Theta(X) = \Theta^T X = [\theta_0\ \theta_1\ \dots \theta_n] \begin{bmatrix} x_0^{(1)} & x_0^{(2)} & x_0^{(3)} & & x_0^{(m)} \\ x_1^{(1)} & x_1^{(2)} & x_1^{(3)} & \dots & x_1^{(m)} \\ \dots & \dots & \dots & & \dots \\ x_n^{(1)} & x_n^{(2)} & x_n^{(3)} & & x_n^{(m)} \end{bmatrix} = [h_\theta(x^{(0)})\ \ h_\theta(x^{(1)})\ \dots\ h_\theta(x^{(m)})]$$

$1 \times n$ $\quad$ $n \times m$ $\quad = \quad$ $1 \times m$

Multivariate Gradient Descent

Multivariate Gradient Descent

Hypothesis: $h_\theta\ (x) = h_\Theta(X) = \Theta^T X = \theta_0 x_0 + \theta_1 x_1 + \cdots + \theta_n x_n$

Parameter vector: $\Theta = \theta_0, \theta_1 \ldots \theta_n$

Feature vector: $\boldsymbol{X} = x_0 x_1 \cdots x_n$

Cost Function: $J(\theta_1) = \dfrac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Gradient descent:

repeat{

$\theta_j \coloneqq \theta_j - \alpha \frac{\partial}{\partial \theta_j} \left(J(\Theta)\right)$ //simultaneously update for all j = 0 … n

}

Gradient Descent

Previously (n=1):

Repeat $\{$

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

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

(simultaneously update θ_0, θ_1)

$\}$

New algorithm $(n \geq 1)$:

Repeat $\{$

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

(simultaneously update θ_j for $j = 0, \ldots, n$)

$\}$

$$\theta_0 := \theta_0 - \alpha \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_0^{(i)}$$

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

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

$\ldots$

Andrew Ng

Sources

- **Worldwide Calculus: Lines, Planes, and Hyperplanes,** https://www.youtube.com/watch?v=-sNDkhE2Vsk
- **Akshay L Chandra – medium.com**
  - **McCulloch-Pitts Neuron—Mankind's First Mathematical Model Of A Biological Neuron:** https://towardsdatascience.com/mcculloch-pitts-model-5fdf65ac5dd1
  - **Perceptron: The Artificial Neuron (An Essential Upgrade To The McCulloch-Pitts Neuron):** https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d
  - **Perceptron Learning Algorithm: A Graphical Explanation Of Why It Works:** https://towardsdatascience.com/perceptron-learning-algorithm-d5db0deab975
- **Prof. Mitesh M. Khapra** (https://www.cse.iitm.ac.in/~miteshk/) on NPTEL's (http://nptel.ac.in/) Deep Learning course (https://onlinecourses.nptel.ac.in/noc18_cs41/preview)
- **Machine Learning for Intelligent Systems**, Kilian Weinberger, Cornell, Lectures 3-6, https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/lecturenote03.html
  - **Dr. Agha Ali Lectures**
- **Perceptrons. An Introduction to Computational Geometry. Marvin Minsky and Seymour Papert. M.I.T. Press, Cambridge, Mass., 1969.** https://science.sciencemag.org/content/165/3895/780