Machine Learning

Linear Regression

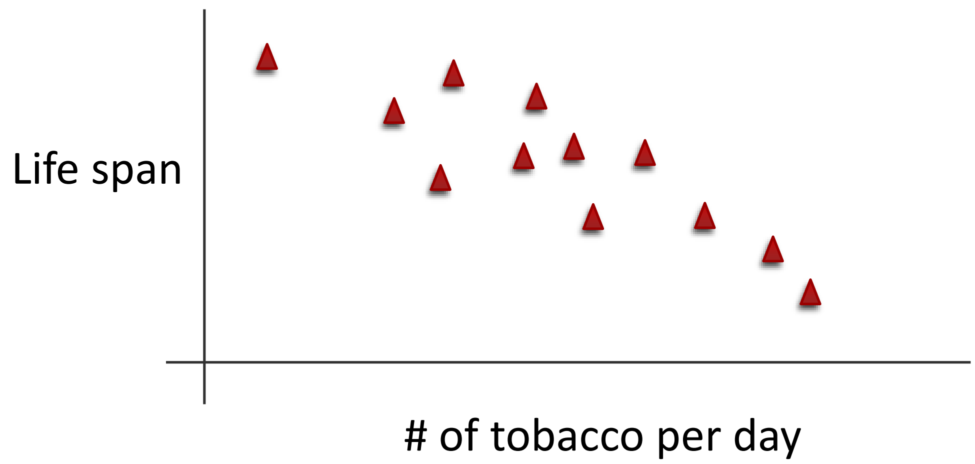
Professor: Cheolsoo Park

Acknowledgement

- Andrew Ng's ML class
  - https://www.coursera.org/learn/machine-learning

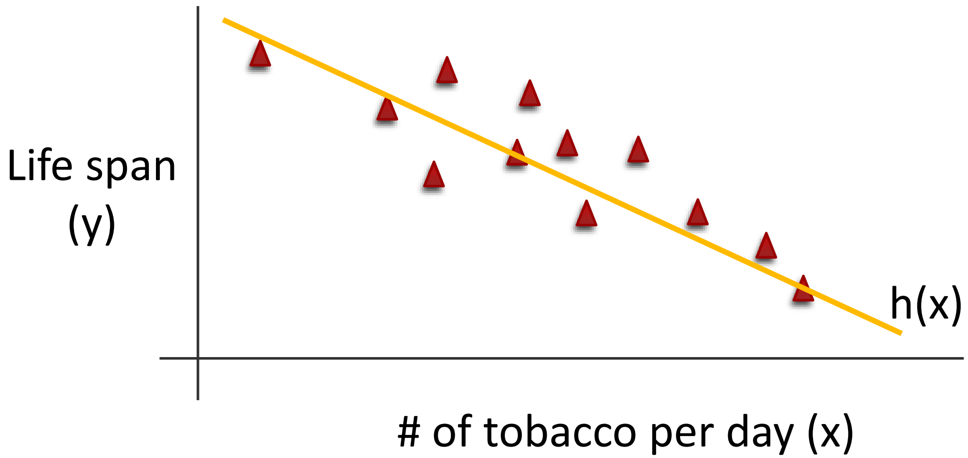
One Variable Linear Regression

- Smoking vs lifespan



- Supervised Learning
  - Classification : discrete output
  - Regression : continuous value output

One Variable Linear Regression



- Linear regression with one variable

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

Cost function

- Parameter estimation in linear regression problem
- Hypothesis : $h_\theta(x) = \theta_0 + \theta_1 x$

  , where θ_i $(i = 0 \ and \ 1)$ are parameters
- Let's find the parameters, θ_i
- Cost function

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

- To estimate $\theta_0 \ and \theta_1$, *minimize J(*θ_0*,* θ_1*)*

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

Cost Function

- Linear regression hypothesis

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

- Cost function

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

- To estimate θ_0 and θ_1, minimize J(θ_0, θ_1)

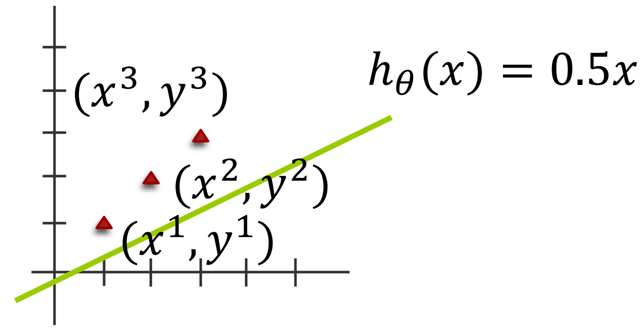
$$\underset{\theta_0, \theta_1}{minimize}\, J(\theta_0, \theta_1)$$

Cost Function

- Example

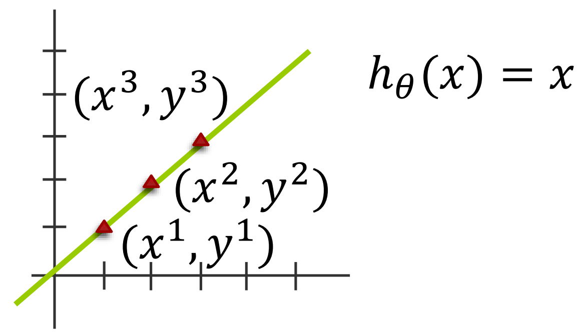
When $\theta_0 = 0, and\ \theta_1 = 0.5$

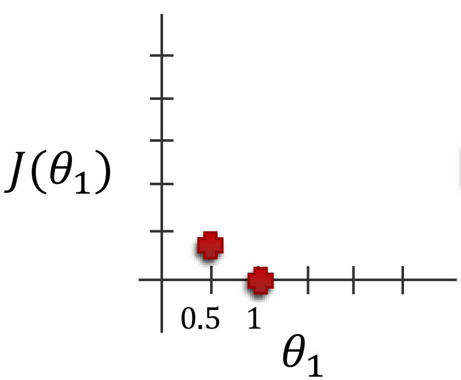
$$J(\theta_0, \theta_1) = \frac{1}{2n} \sum_{k=1}^{n} \left(0.5x^{(k)} - y^{(k)}\right)^2 =$$
$$\frac{1}{2\cdot 3}\left((0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right) = 0.58$$

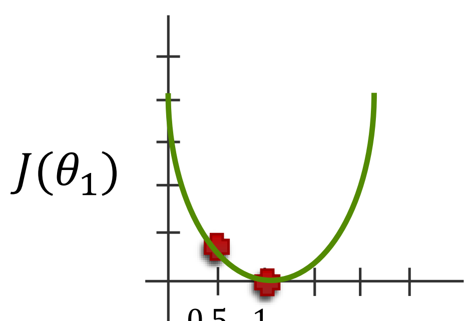


When $\theta_0 = 0, and\ \theta_1 = 1$

$$J(\theta_0, \theta_1) = \frac{1}{2n} \sum_{k=1}^{n} \left(h_\theta\left(x^{(k)}\right) - y^{(k)}\right)^2$$
$$\frac{1}{2n} \sum_{k=1}^{n} \left(x^{(k)} - y^{(k)}\right)^2 = \frac{1}{2\cdot 3}\left(0^2 + 0^2 + 0^2\right) = 0$$



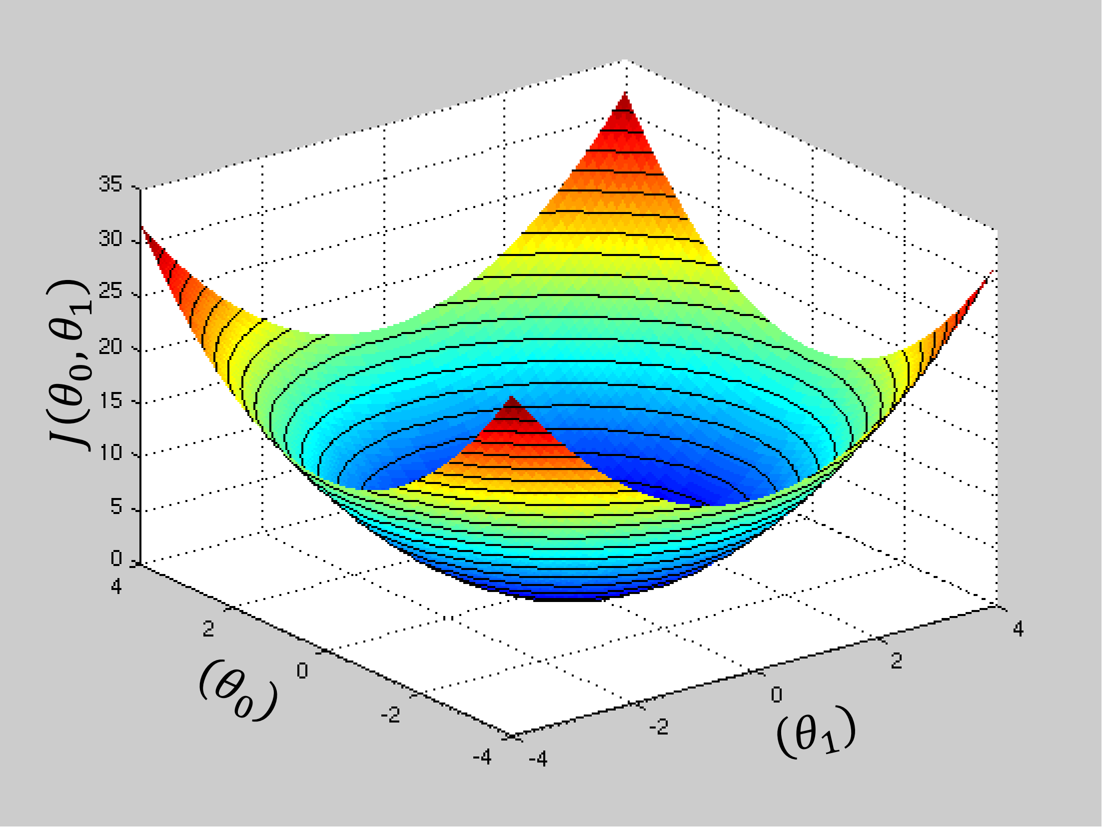




θ_1

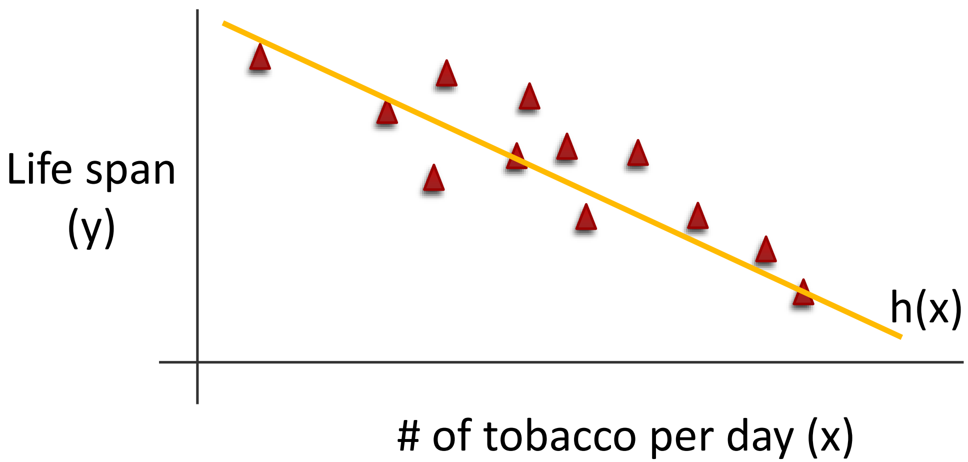
Cost Function

- $J(\theta_0, \theta_1)$ with two parameters

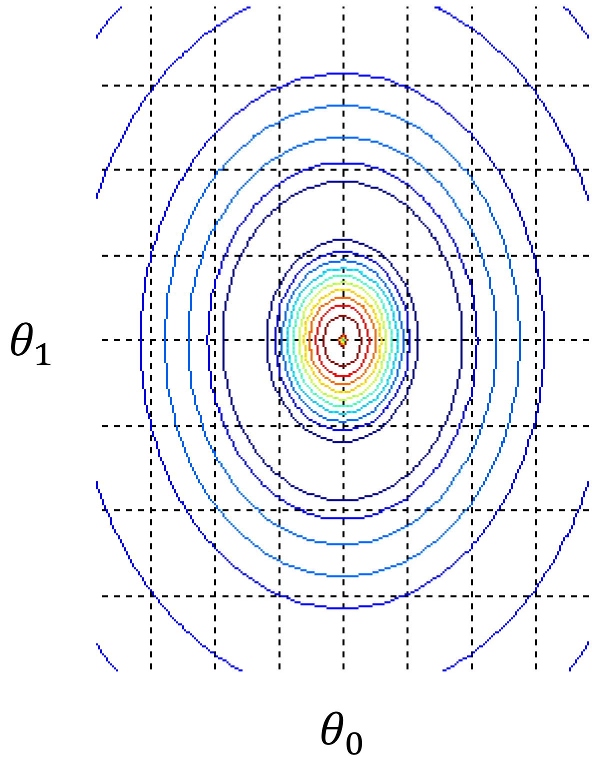


Cost Function

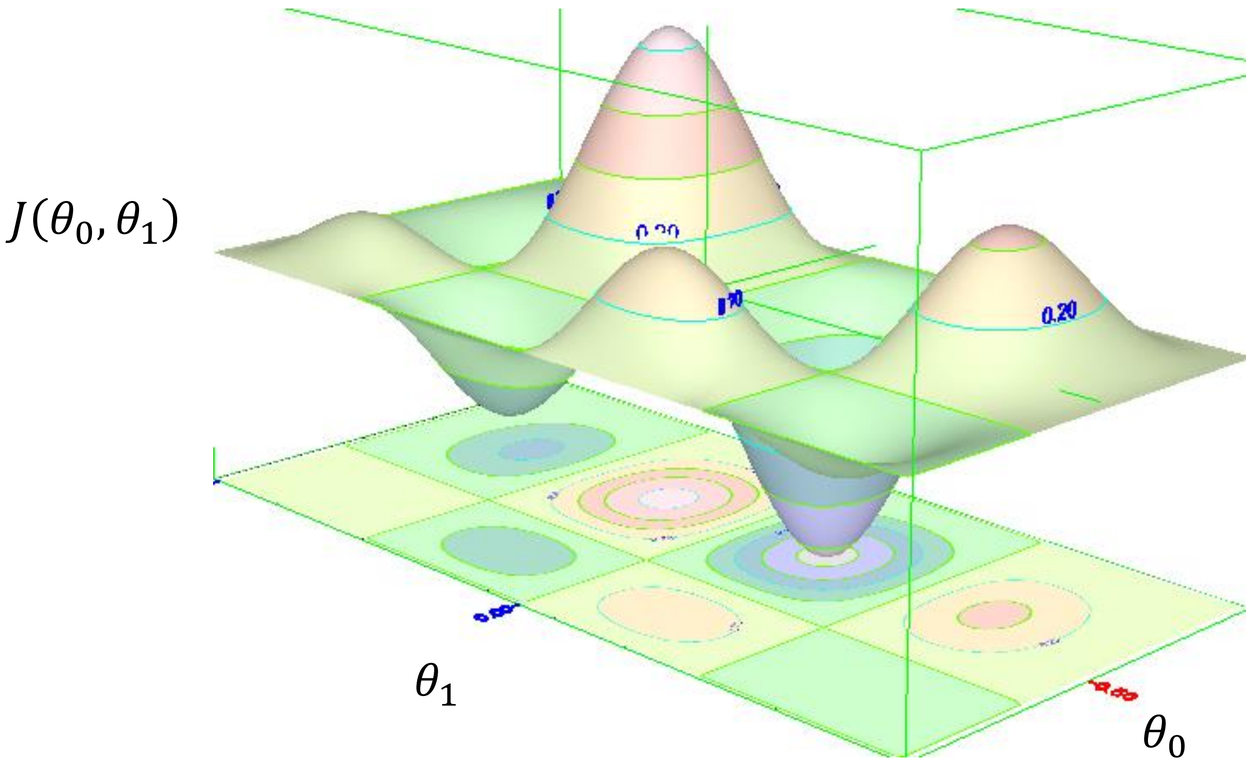
$h_\theta(x) = \theta_0 + \theta_1 x$



$J(\theta_0, \theta_1)$



Gradient Descent Algorithm

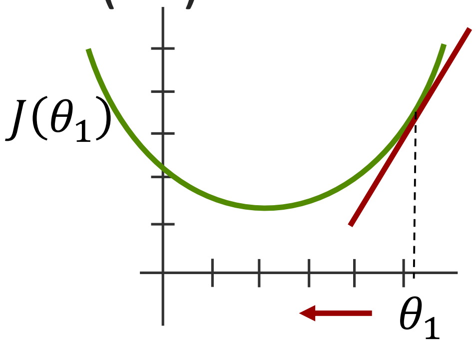


Gradient Descent Algorithm

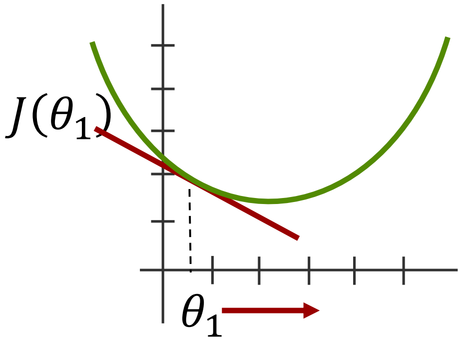
- Repeat the function below until it converges

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

where α (>0) is a learning rate



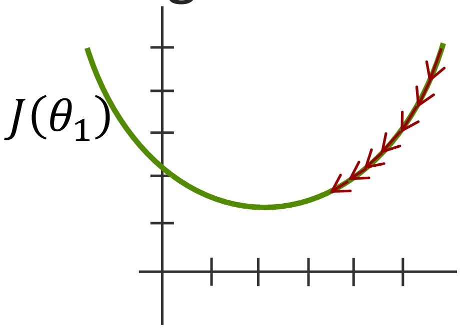
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$



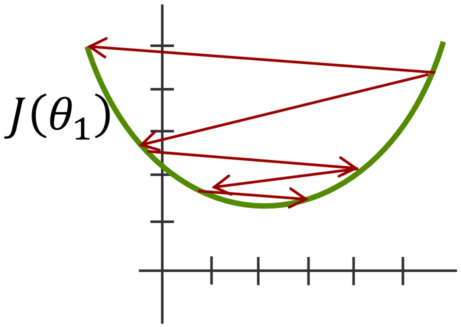
Gradient Descent Algorithm

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

- Too small α makes gradient descent slow



- Too large α makes gradient descent could overshoot the minimum.

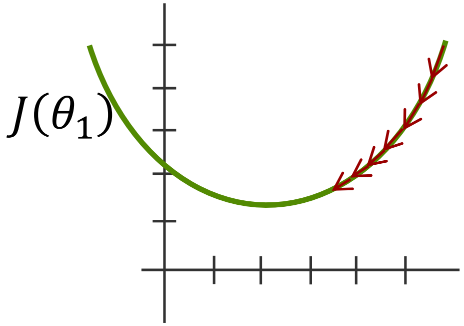


- Gradient descent can converge to a local minimum

Gradient Descent Algorithm

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

- Since gradient descent takes smaller steps by itself, we don't have to adjust or decrease α over time.



Gradient Descent for Linear Regression

- Linear regression hypothesis

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

- Cost function

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

- Gradient descent algorithm

Repeat the function below until it converges

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

Gradient Descent for Linear Regression

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

$$= \frac{d}{d\theta_j}\frac{1}{2n}\sum_{k=1}^{n}\left(\theta_0 + \theta_1 x^{(k)} - y^{(k)}\right)^2$$

$$j = 0 \Rightarrow \frac{d}{d\theta_0}J(\theta_0,\theta_1) = \frac{1}{n}\sum_{k=1}^{n}\left(h_\theta\left(x^{(k)}\right) - y^{(k)}\right)$$

$$j = 1 \Rightarrow \frac{d}{d\theta_1}J(\theta_0,\theta_1) = \frac{1}{n}\sum_{k=1}^{n}\left(h_\theta\left(x^{(k)}\right) - y^{(k)}\right) x^{(k)}$$

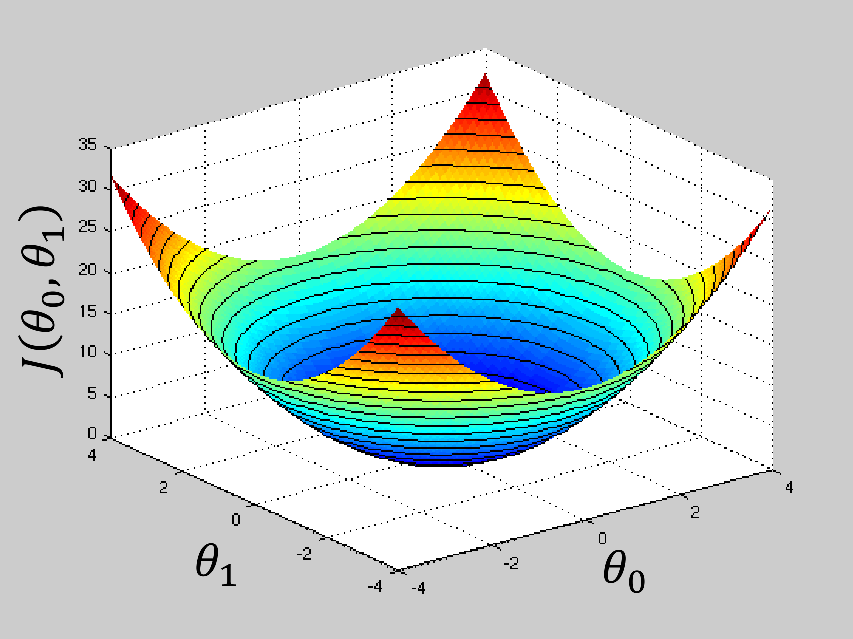
Gradient Descent for Linear Regression

Repeat the function below until it converges

$$\theta_0 \coloneqq \theta_0 - \alpha \frac{1}{n} \sum_{k=1}^{n} \left(h_\theta\left(x^{(k)}\right) - y^{(k)} \right)$$

$$\theta_1 \coloneqq \theta_1 - \alpha \frac{1}{n} \sum_{k=1}^{n} \left(h_\theta\left(x^{(k)}\right) - y^{(k)} \right) x^{(k)}$$

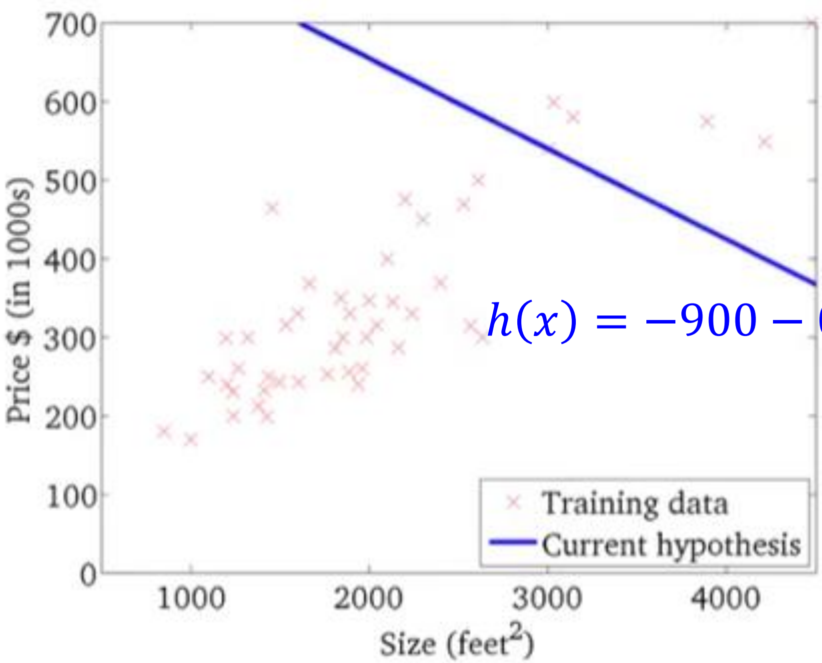
Update θ_0 and θ_1 simultaneously



Gradient Descent for Linear Regression

$h_\theta(x)$

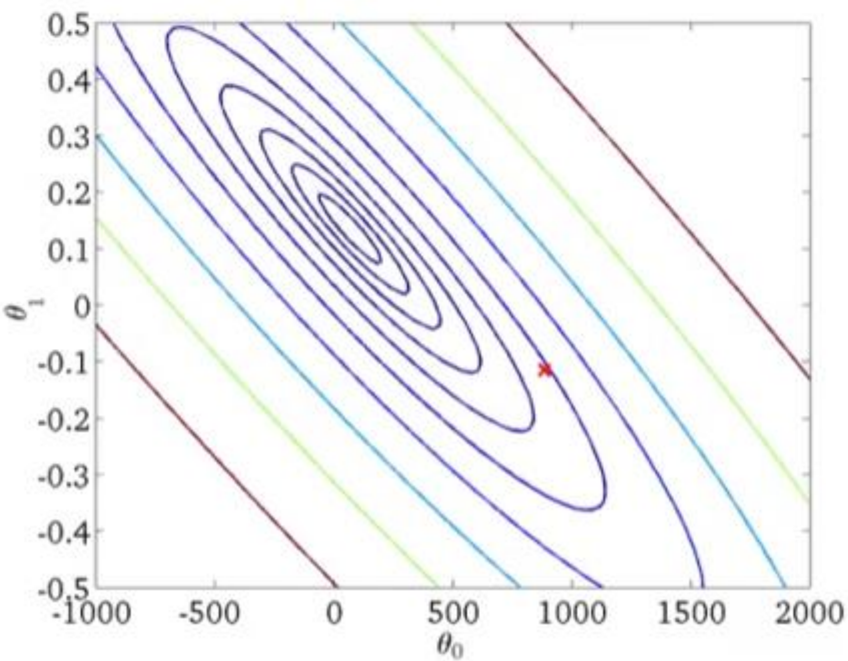
(for fixed θ_0, θ_1, this is a function of x)



$h(x) = -900 - 0.1x$

$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

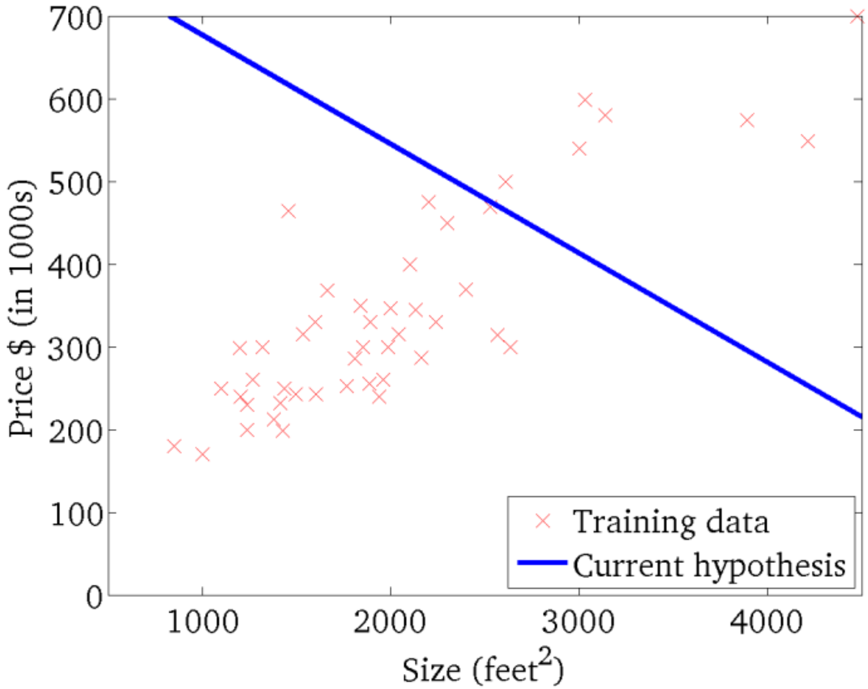


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

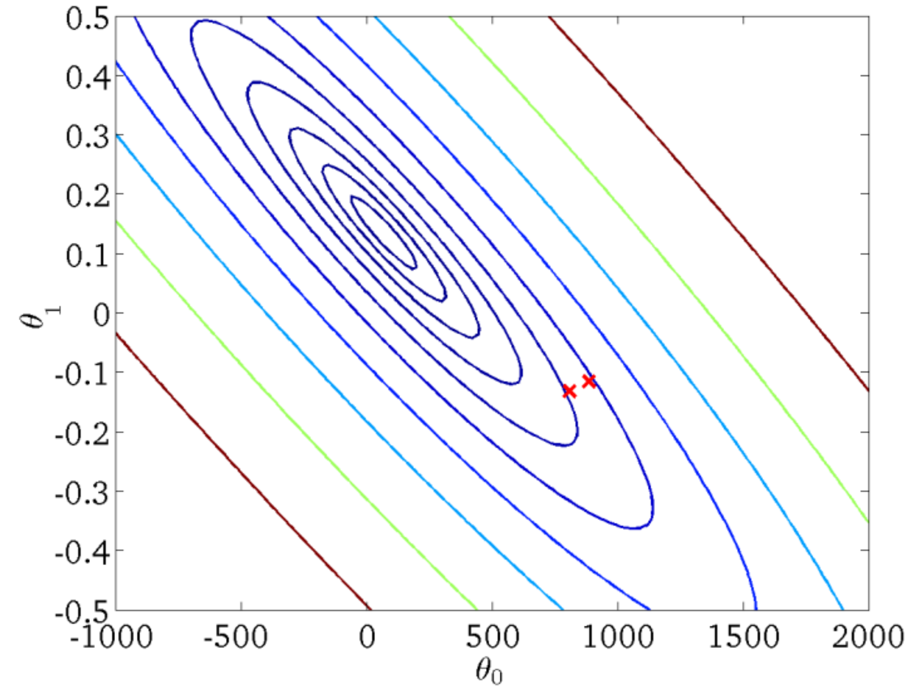
$h_\theta(x)$

(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)

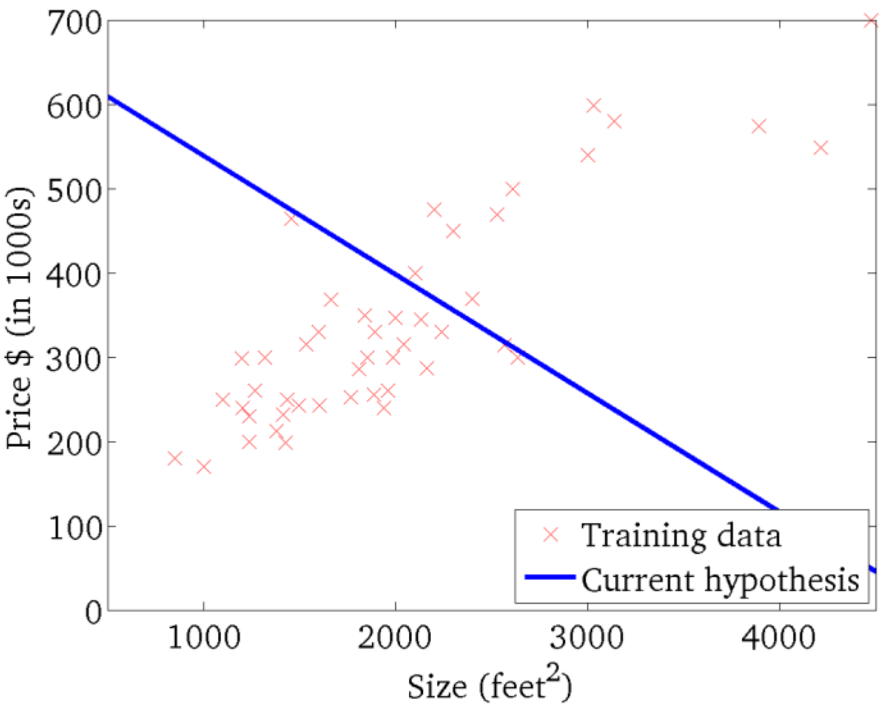


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

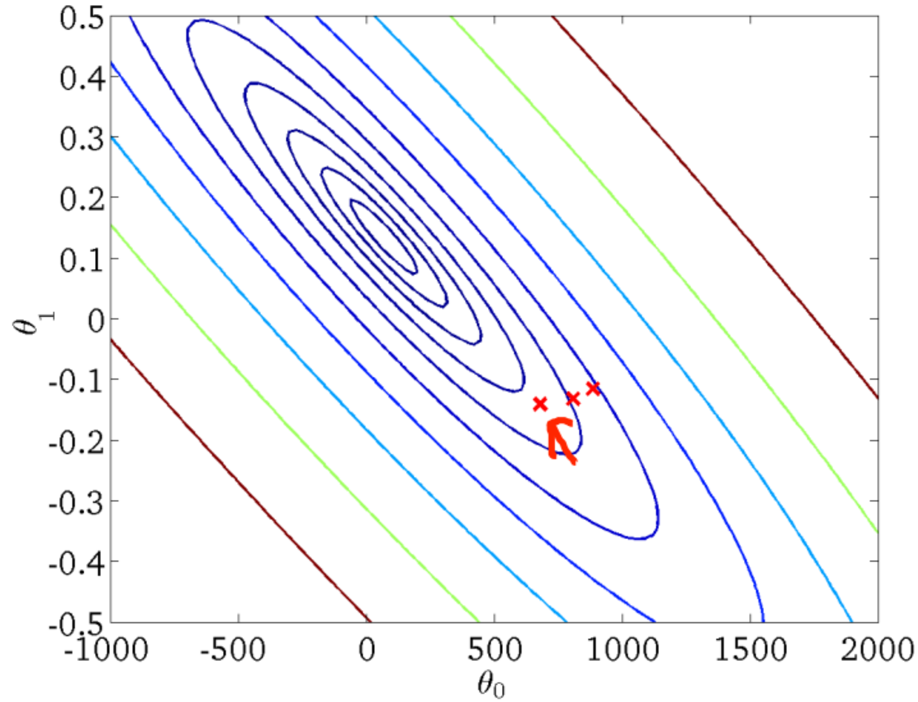
$$h_\theta(x)$$

(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)

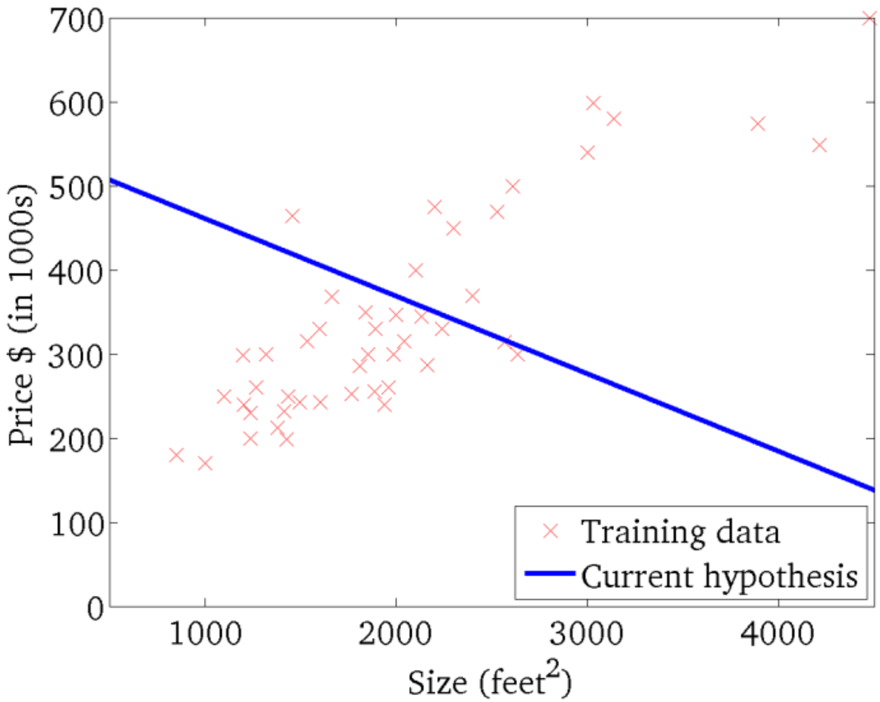


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

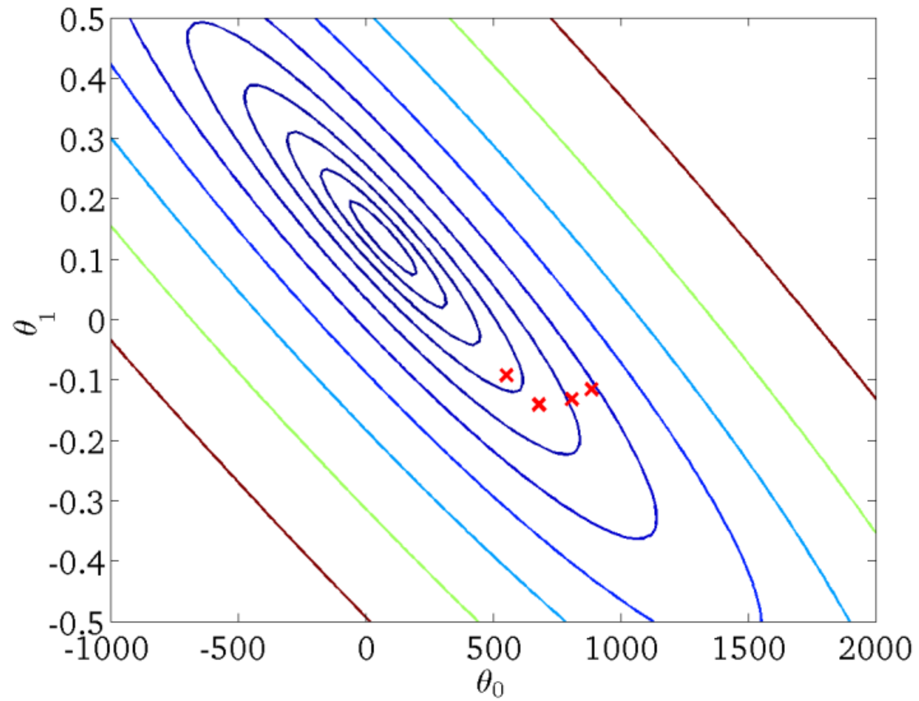
$h_\theta(x)$

(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)

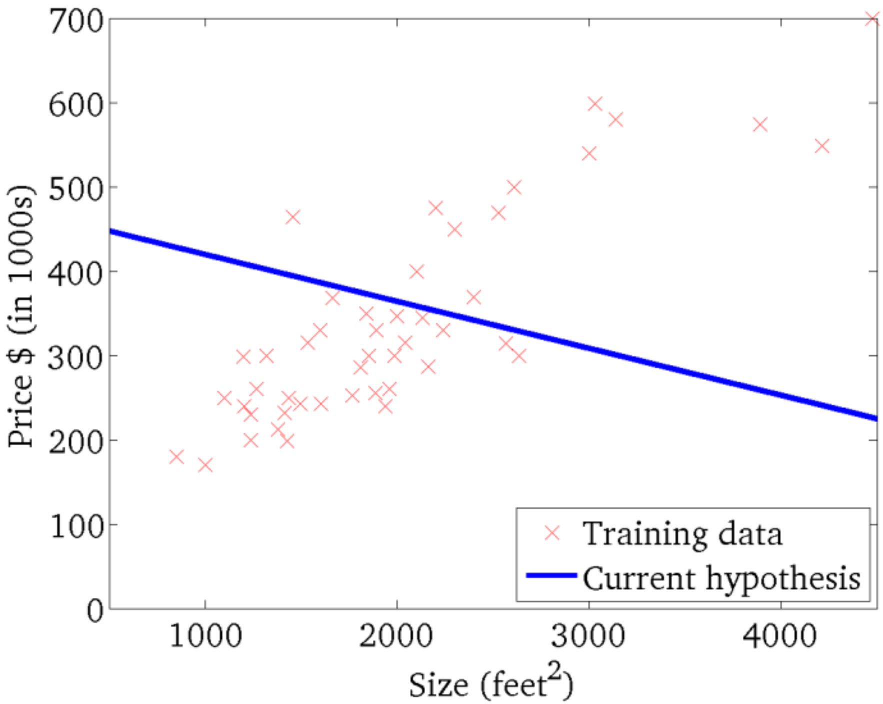


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

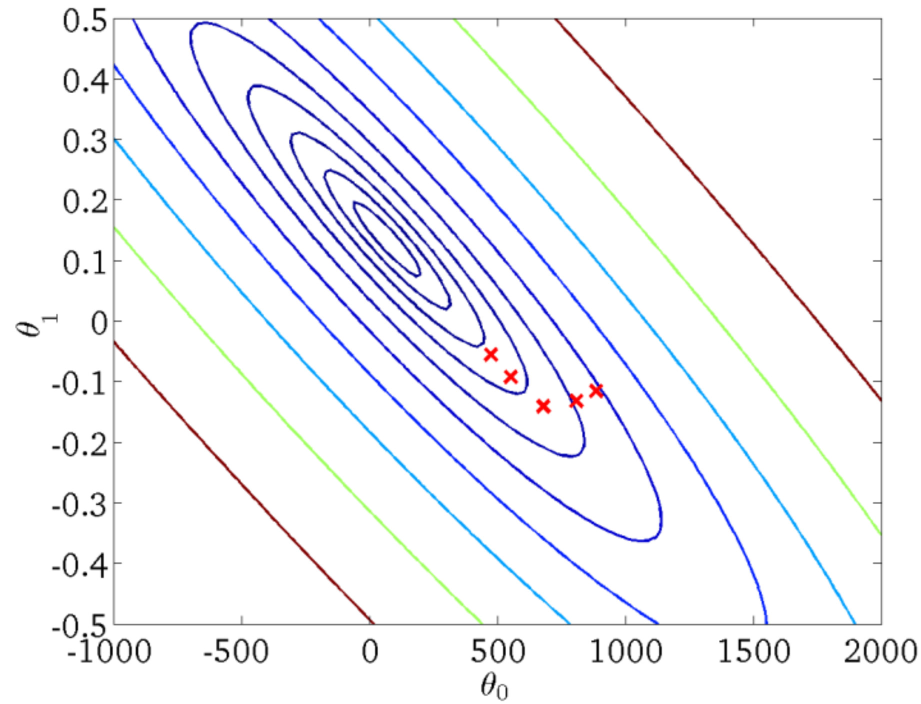
$$h_\theta(x)$$

(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)

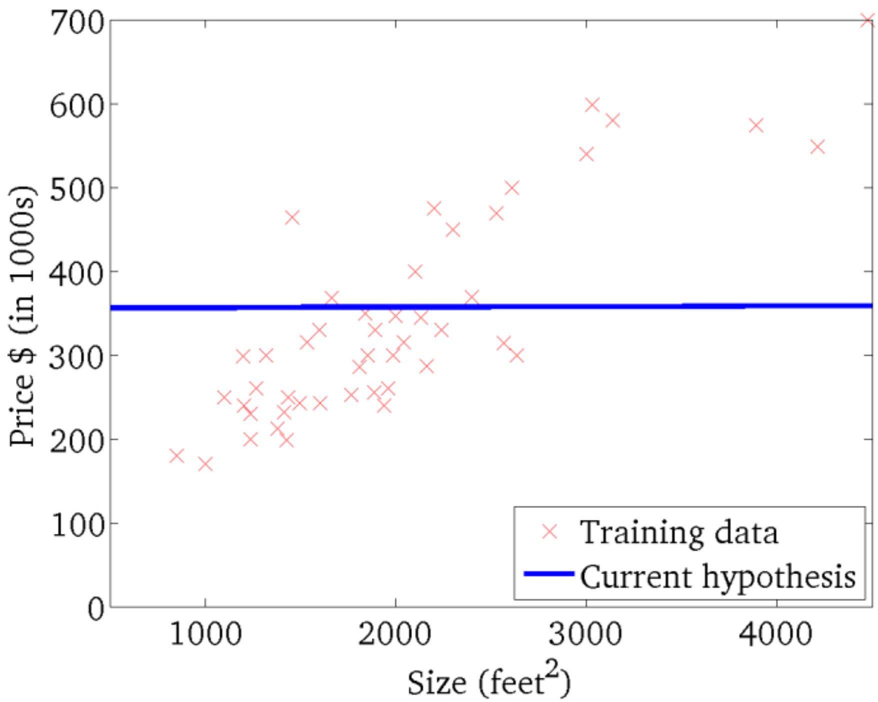


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

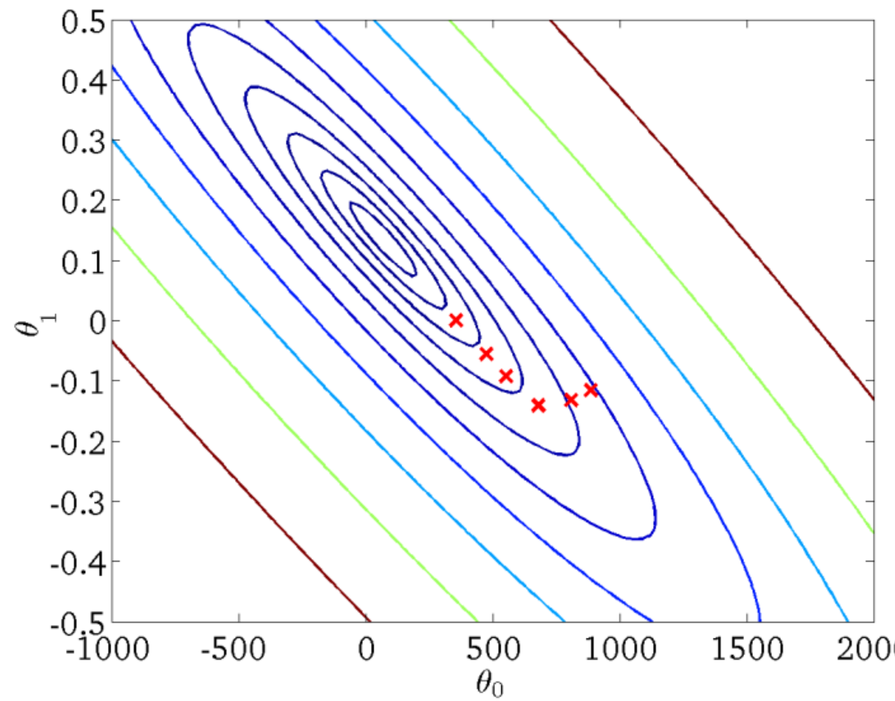
$h_\theta(x)$

(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)

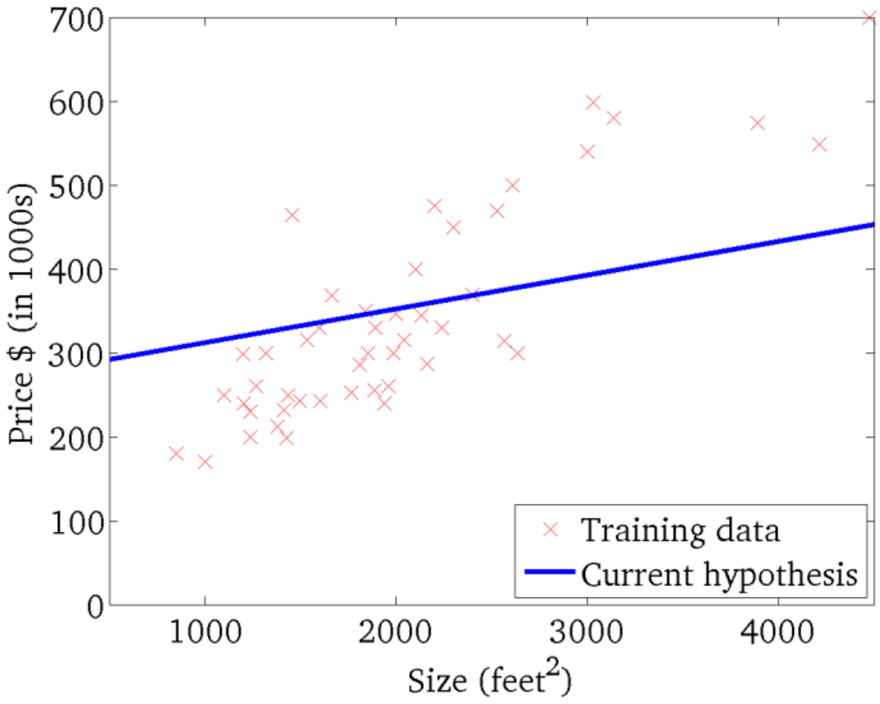


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

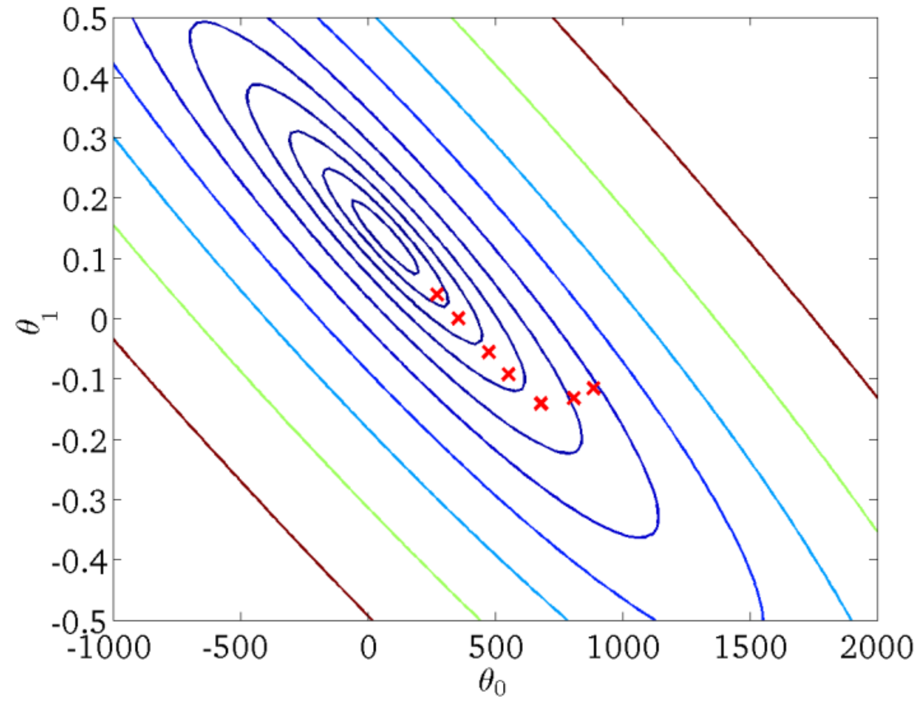
$h_\theta(x)$

(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)

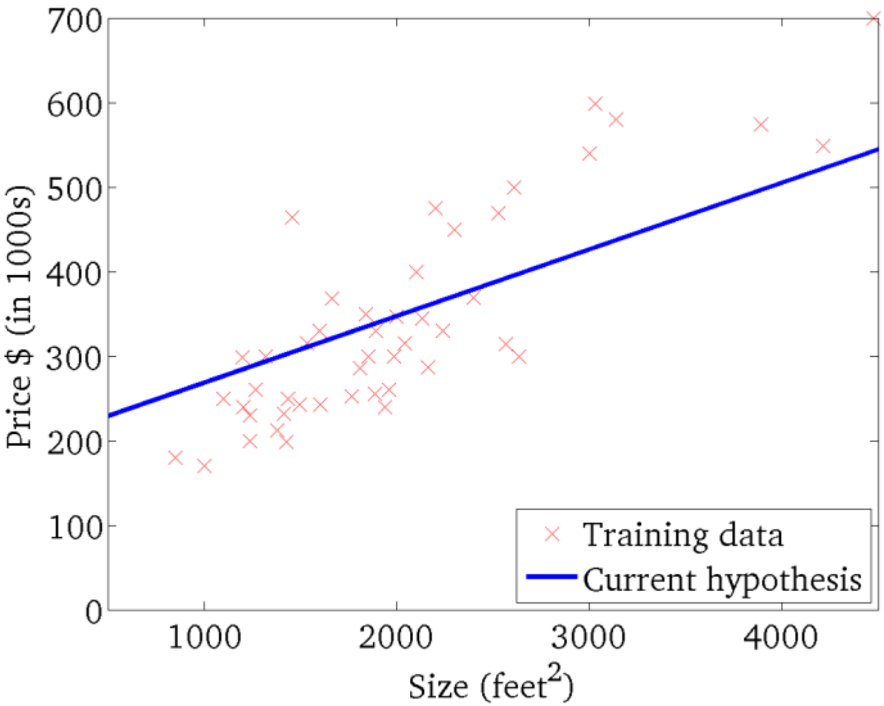


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

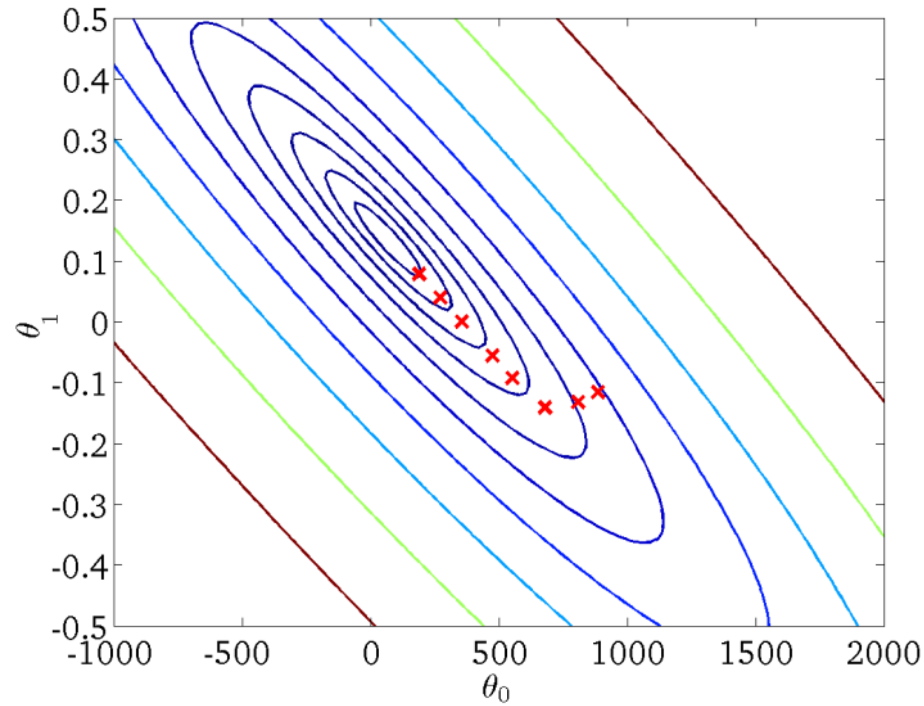
$h_\theta(x)$

(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)

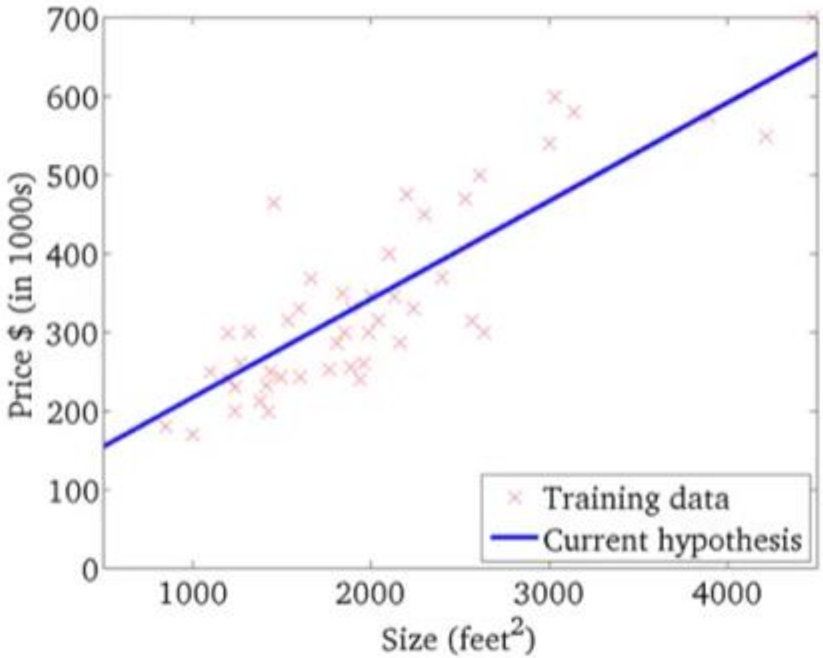


Machine Learning by Andrew Ng.

Gradient Descent for Linear Regression

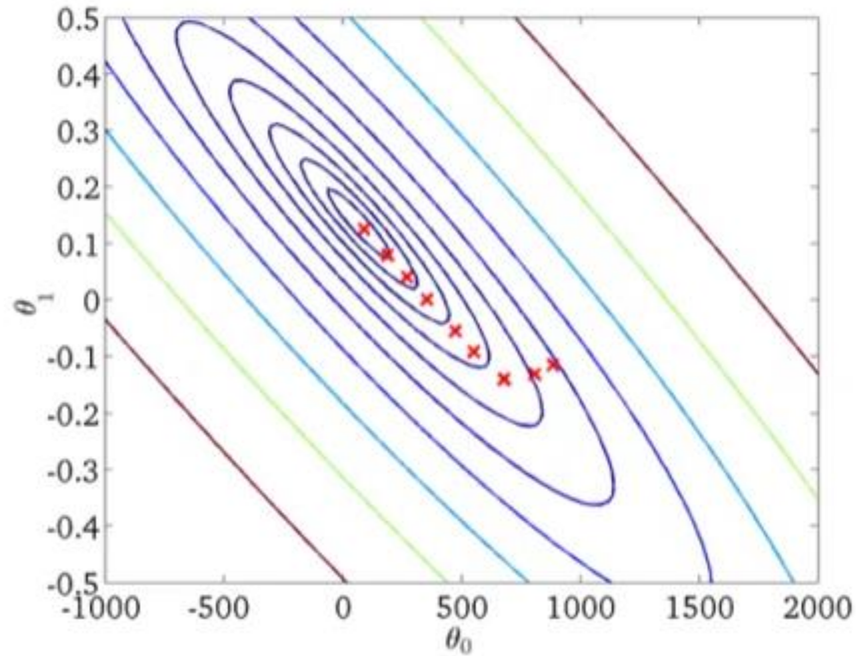
$h_\theta(x)$

(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)



Machine Learning by Andrew Ng.

Linear Regression with multiple variables

- When we have multiple variables(features) to interpret a phenomenon
  - for example, lifespan vs smoking/exercise/fine dust/diet and so on….

- Hypothesis for the multiple variables

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4 + \cdots$$

➔ $h_\theta(x)=\boldsymbol{\theta}^T\boldsymbol{X}$

, where $\boldsymbol{\theta} = \begin{bmatrix} \theta_{\mathbf{0}} \\ \theta_{\mathbf{1}} \\ \theta_{\mathbf{2}} \\ \dots \\ \theta_{\boldsymbol{n}} \end{bmatrix}$ and $\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_{\mathbf{0}} \\ \boldsymbol{x}_{\mathbf{1}} \\ \boldsymbol{x}_{\mathbf{2}} \\ \dots \\ \boldsymbol{x}_{\boldsymbol{n}} \end{bmatrix}$ $(x_0 = 1)$

Linear Regression with multiple variables

- Hypothesis of linear regression

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

- Cost function

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

- Gradient descent algorithm

Repeat the function below until it converges

$$\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta_0, \theta_1, \dots \theta_n) \text{ for j=0,…,n}$$

Linear Regression with multiple variables

Repeat the function below until it converges

$$\theta_j := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta_0, \theta_1, \dots \theta_n) \text{ for j=0,…,n}$$

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

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

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

.

.

.

Feature Scaling

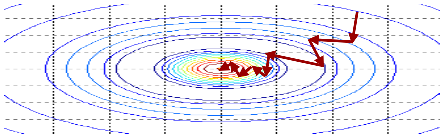
- Features should be in similar range
  - Set the range into $-1 \leq x_i \leq 1$
  - Feature normalization : zero mean and unit variance

$$\frac{x_i - \mu}{\sigma}$$

For example, $x_1 = \#\ of\ cigaret$ (0~40 pieces)

x_2 =seconds of exercise (0~7200 secs)

$\boldsymbol{\theta_2}$

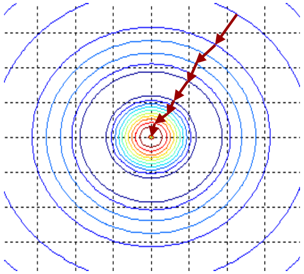


$\boldsymbol{\theta_1}$

$$x_1 = \frac{\#\ of\ cigaret}{40}$$

$$x_2 = \frac{\text{seconds of exercise}}{7200}$$

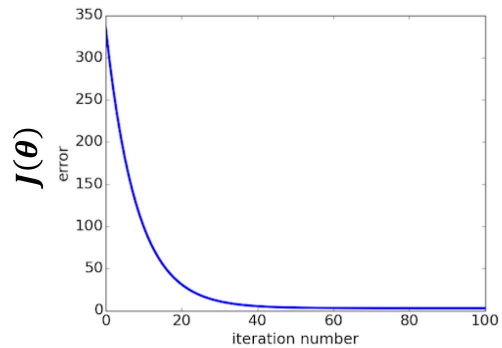
$\boldsymbol{\theta_2}$



$\boldsymbol{\theta_1}$

Convergence of Gradient Descent

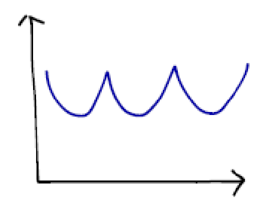
- Checking the convergence of gradient descent



- Sometimes, the stopping criterion of gradient descent is set as

$$J(\boldsymbol{\theta}) < 10^{-3}$$

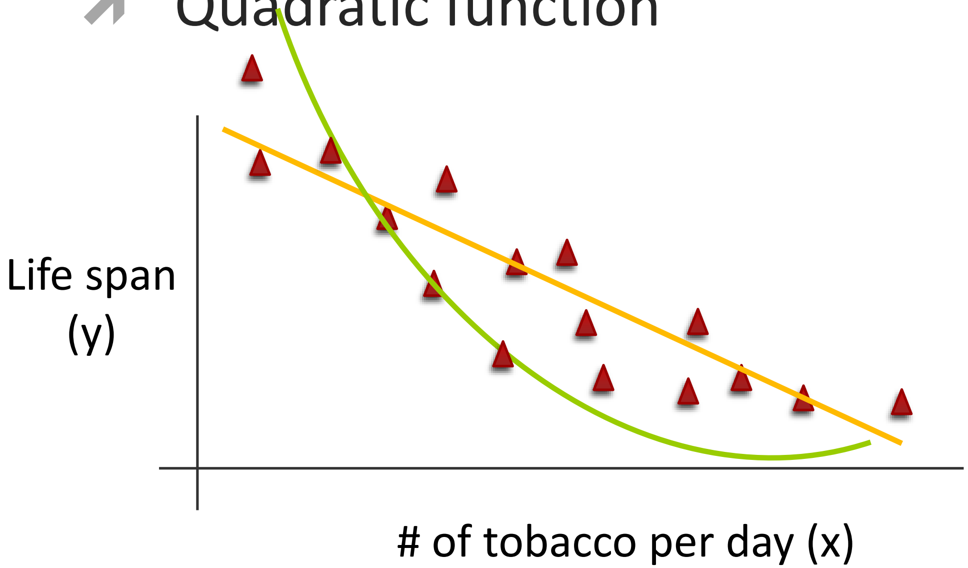
- Using too large α, $J(\boldsymbol{\theta})$ may not decrease on every iteration



- Small α makes $J(\boldsymbol{\theta})$ decrease on every iteration
- However, too small α causes too slow convergence

Polynomial Regression

- Quadratic function

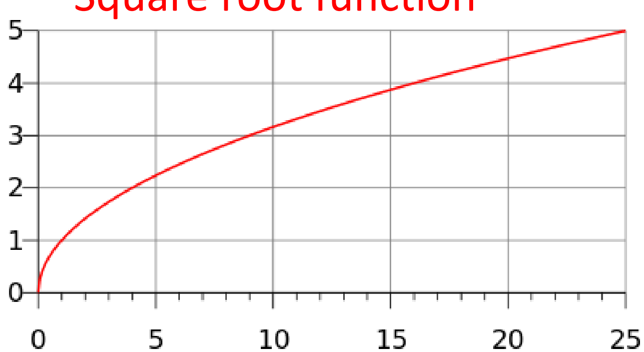


$$h_\theta(x) = \theta_0 + \theta_1 x_1$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2$$

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3$$

* Square root function



$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 \sqrt{x_2}$$

Normal Equation

- Normal equation : analytic way to solve for θ

	Smoking (cigarette #)	Exercise (Seconds/day)	Fine dust ($\mu g/m^3$)	Diet (kcal)	Lifespan (years)
x_0	x_1	x_2	x_3	x_4	y
1	5	3600	20	2000	90
1	0	1000	60	2500	75
1	10	7200	150	3500	57
1	40	500	100	1000	45

$$\mathbf{x} = \begin{bmatrix} 1 & 5 & 3600 & 20 & 2000 \\ 1 & 0 & 1000 & 60 & 2500 \\ 1 & 10 & 7200 & 150 & 3500 \\ 1 & 40 & 500 & 100 & 1000 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 90 \\ 75 \\ 57 \\ 45 \end{bmatrix}$$

$$\boldsymbol{y} = \boldsymbol{\theta_0 x_0} + \boldsymbol{\theta_1 x_1} + \boldsymbol{\theta_2 x_2} + \boldsymbol{\theta_3 x_3} + \boldsymbol{\theta_4 x_4} \Rightarrow \boldsymbol{\theta} = \left(\boldsymbol{X^T X}\right)^{-1} \boldsymbol{X^T y}$$

Gradient Descent vs Normal Equation

- When m is the sample numbers and n is the number of features
- Gradient Descent
  - Needs α
  - Takes many iterations
  - is OK with large n
- Normal Equation
  - Needs no α
  - Needs no iteration
  - Need to compute $(X^T X)^{-1}$
  - Slow with large n