

Machine Learning

# Linear Regression with multiple variables

---

## Multiple features

## Multiple features (variables).

Size (feet <sup>2</sup> )	Price (\$1000)
$x$	$y$
2104	460
1416	232
1534	315
852	178
...	...

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

## Multiple features (variables).

Size (feet <sup>2</sup> )	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
...	...	...	...	...

## Multiple features (variables).

Size (feet <sup>2</sup> )	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
...	...	...	...	...

Notation:

$n$  = number of features

$x^{(i)}$  = input (features) of  $i^{th}$  training example.

$x_j^{(i)}$  = value of feature  $j$  in  $i^{th}$  training example.

Hypothesis:

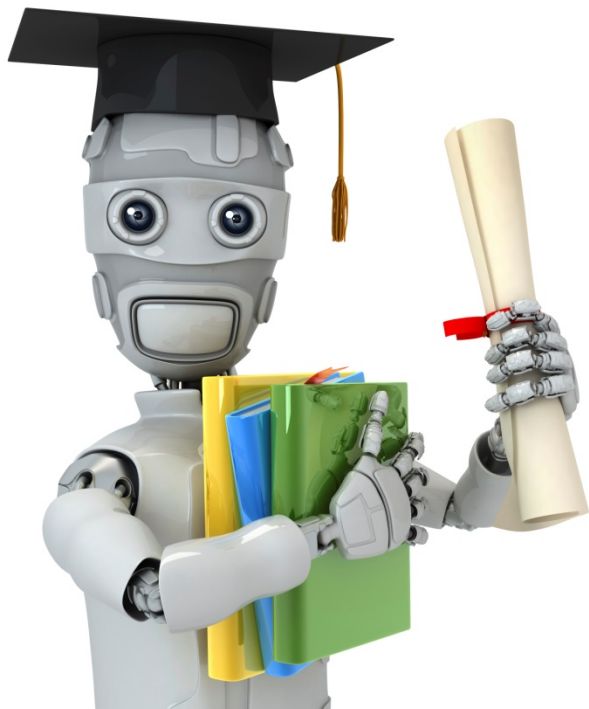
Previously: 
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

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

For convenience of notation, define  $x_0 = 1$ .

Multivariate linear regression.





Machine Learning

# Linear Regression with multiple variables

---

## Gradient descent for multiple variables



Hypothesis:  $h_{\theta}(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

Parameters:  $\theta_0, \theta_1, \dots, \theta_n$

Cost function:

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

Gradient descent:

Repeat {

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

}

(simultaneously update for every  $j = 0, \dots, n$ )

# Gradient Descent

Previously ( $n=1$ ):

Repeat {

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

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

(simultaneously update  $\theta_0, \theta_1$ )

}

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  $\theta_j$  for  
 $j = 0, \dots, n$ )

}

---

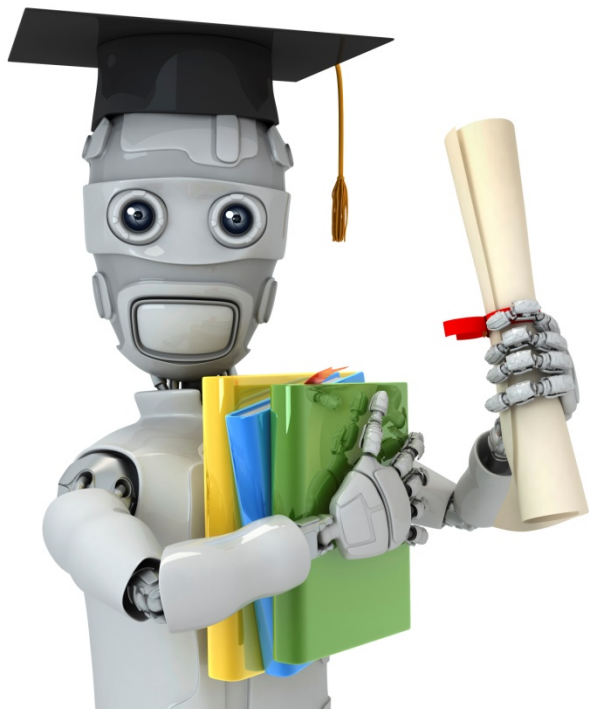
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})x_0^{(i)}$$

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

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

...





Machine Learning

# Linear Regression with multiple variables

---

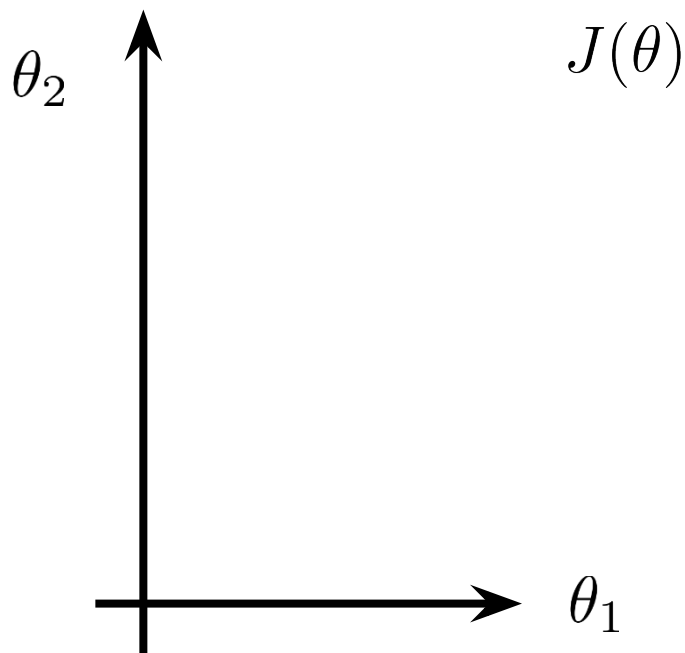
Gradient descent in practice I: Feature Scaling

# Feature Scaling

Idea: Make sure features are on a similar scale.

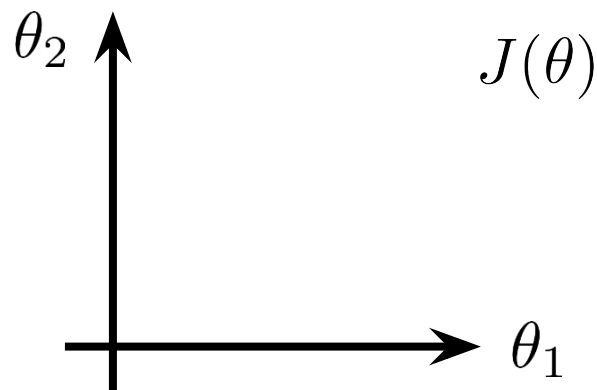
E.g.  $x_1 = \text{size (0-2000 feet}^2\text{)}$

$x_2 = \text{number of bedrooms (1-5)}$



$$x_1 = \frac{\text{size (feet}^2\text{)}}{2000}$$

$$x_2 = \frac{\text{number of bedrooms}}{5}$$



## Feature Scaling

Get every feature into approximately a  $-1 \leq x_i \leq 1$  range.

# Mean normalization

Replace  $x_i$  with  $x_i - \mu_i$  to make features have approximately zero mean  
(Do not apply to  $x_0 = 1$ ).

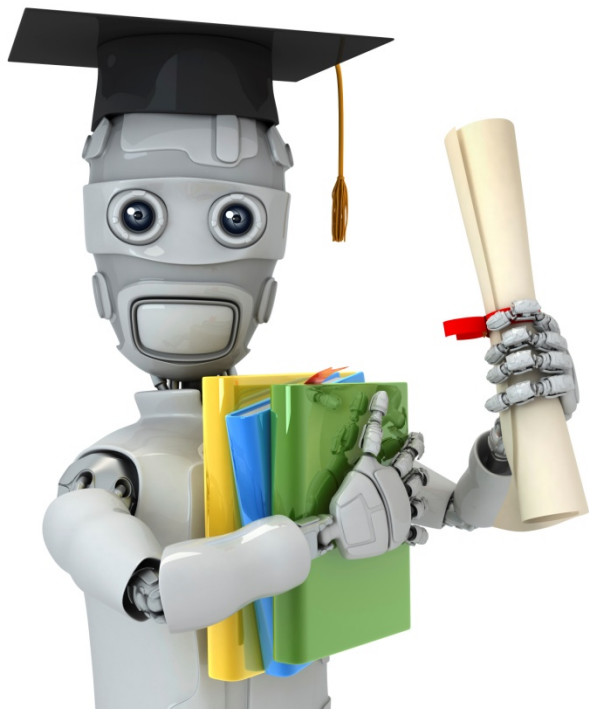
E.g.  $x_1 = \frac{size - 1000}{2000}$

$$x_2 = \frac{\#bedrooms - 2}{5}$$

$$-0.5 \leq x_1 \leq 0.5, -0.5 \leq x_2 \leq 0.5$$







Machine Learning

# Linear Regression with multiple variables

---

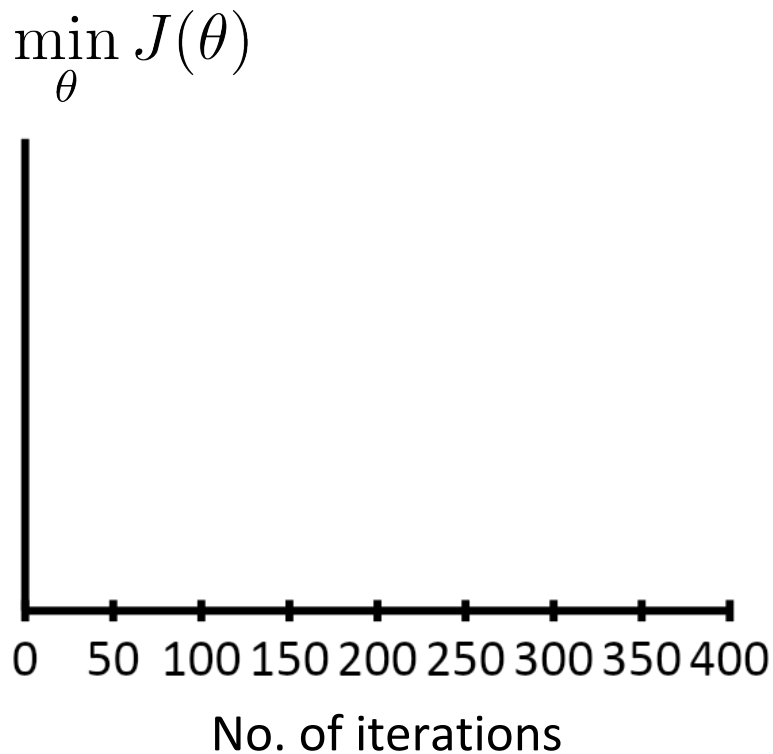
Gradient descent in practice II: Learning rate

# Gradient descent

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

- “Debugging”: How to make sure gradient descent is working correctly.
- How to choose learning rate  $\alpha$ .

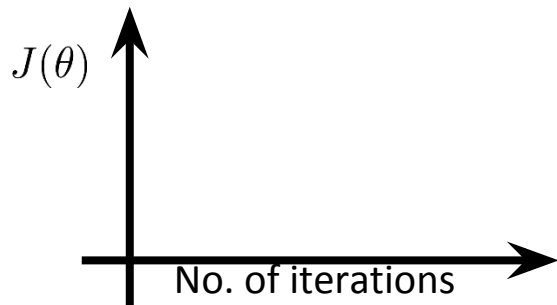
## Making sure gradient descent is working correctly.



Example automatic  
convergence test:

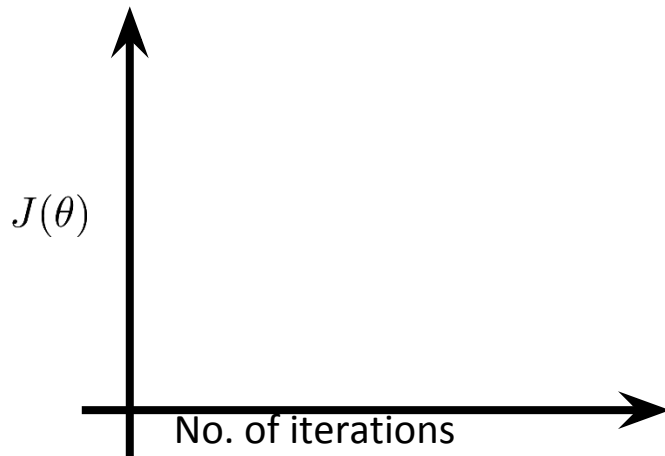
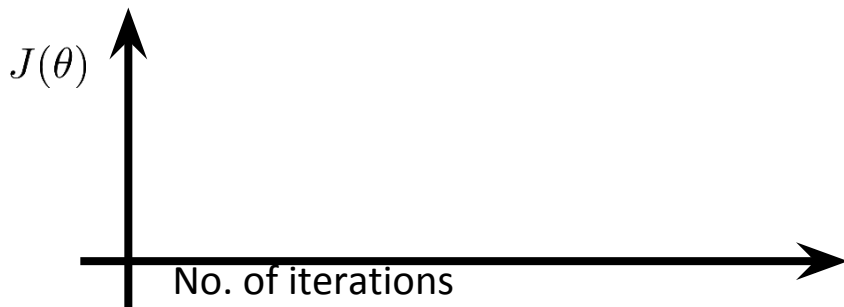
Declare convergence if  $J(\theta)$   
decreases by less than  $10^{-3}$   
in one iteration.

## Making sure gradient descent is working correctly.



Gradient descent not working.

Use smaller  $\alpha$ .



- For sufficiently small  $\alpha$ ,  $J(\theta)$  should decrease on every iteration.
- But if  $\alpha$  is too small, gradient descent can be slow to converge.

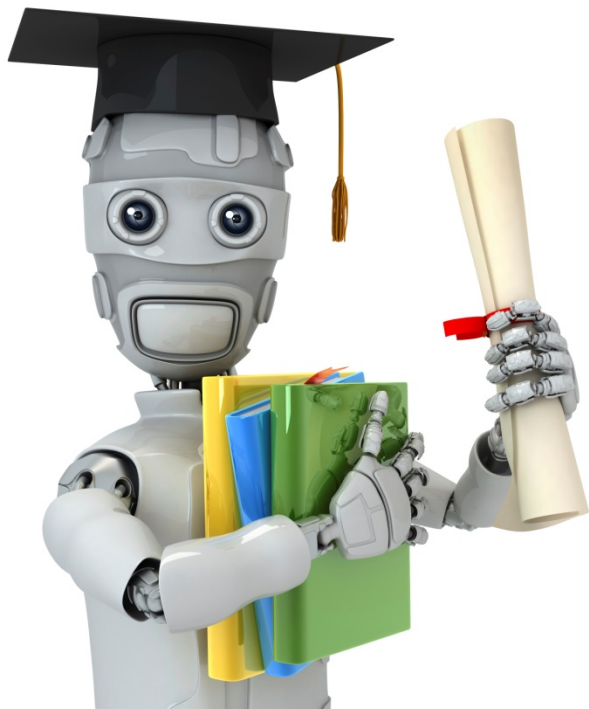
## Summary:

- If  $\alpha$  is too small: slow convergence.
- If  $\alpha$  is too large:  $J(\theta)$  may not decrease on every iteration; may not converge.

To choose  $\alpha$ , try

$\dots, 0.001, \quad , 0.01, \quad , 0.1, \quad , 1, \dots$





Machine Learning

# Linear Regression with multiple variables

---

Features and  
polynomial regression

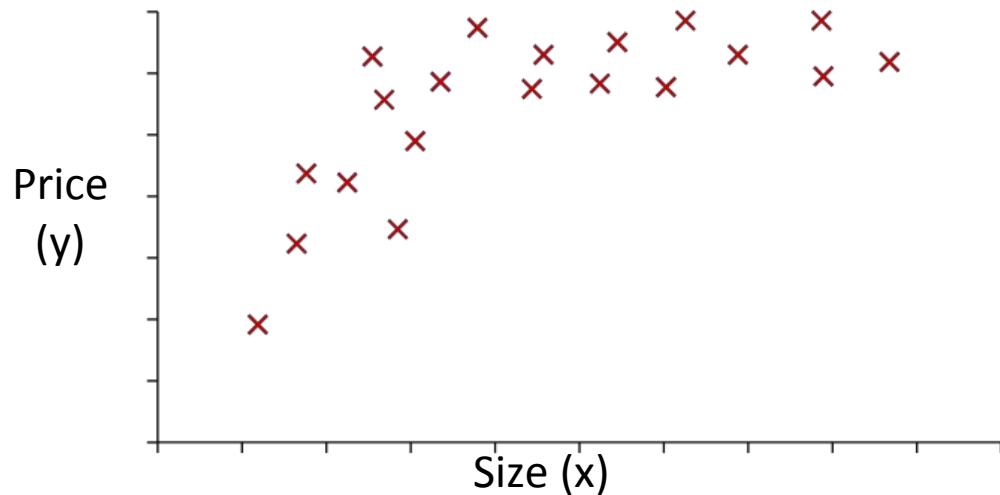
# Housing prices prediction

$$h_{\theta}(x) = \theta_0 + \theta_1 \times \textit{frontage} + \theta_2 \times \textit{depth}$$





# Polynomial regression



$$\theta_0 + \theta_1 x + \theta_2 x^2$$

$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

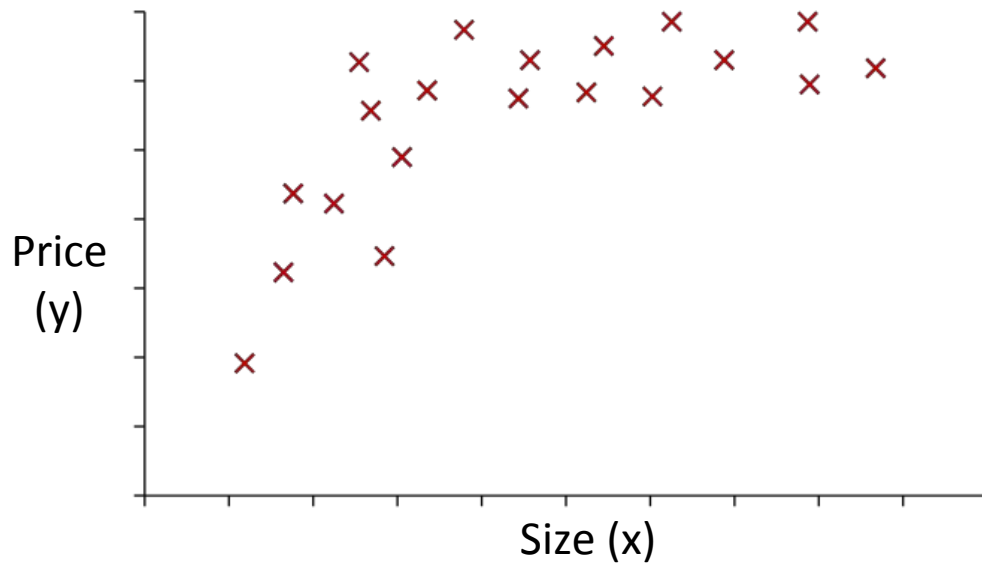
$$\begin{aligned} h_{\theta}(x) &= \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \\ &= \theta_0 + \theta_1(\text{size}) + \theta_2(\text{size})^2 + \theta_3(\text{size})^3 \end{aligned}$$

$$x_1 = (\text{size})$$

$$x_2 = (\text{size})^2$$

$$x_3 = (\text{size})^3$$

# Choice of features



$$h_{\theta}(x) = \theta_0 + \theta_1(\text{size}) + \theta_2(\text{size})^2$$

$$h_{\theta}(x) = \theta_0 + \theta_1(\text{size}) + \theta_2\sqrt{(\text{size})}$$