

Machine Learning

Linear Regression with multiple variables

Multiple features

Multiple features (variables).

Size (feet ²)	Price (\$1000)
$\rightarrow x$	$y \leftarrow$
2104	460
1416	232
1534	315
852	178
...	...

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



Multiple features (variables).

<u>Size (feet²)</u>	<u>Number of bedrooms</u>	<u>Number of floors</u>	<u>Age of home (years)</u>	Price (\$1000)
x_1	x_2	x_3	x_4	y
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...

Notation:

- $n = 4$ = number of features
- $x^{(i)}$ = input (features) of i^{th} training example.
- $x_j^{(i)}$ = value of feature j in $\underline{i^{th}}$ training example.

$x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$

$x_3^{(2)} = 2$

Hypothesis:

Previously:

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

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

E.g. $\underline{h_{\theta}(x)} = \underline{80} + \underline{0.1x_1} + \underline{0.01x_2} + \underline{3x_3} - \underline{2x_4}$

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

For convenience of notation, define $x_0 = 1.$ ($x_0^{(i)} = 1$)

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$\Theta = \begin{bmatrix} \Theta_0 \\ \Theta_1 \\ \Theta_2 \\ \vdots \\ \Theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

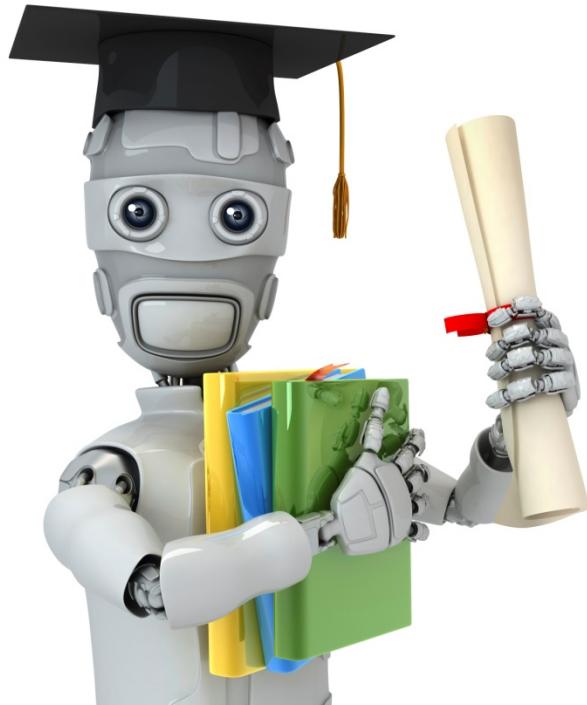
$$h_{\theta}(x) = \underline{\Theta_0x_0 + \Theta_1x_1 + \cdots + \Theta_nx_n}$$

$$= \boxed{\Theta^T x}$$

$$\begin{bmatrix} \Theta_0 & \Theta_1 & \cdots & \Theta_n \end{bmatrix} \Theta^T$$

$(n+1) \times 1$ matrix
 $\Theta^T x$

Multivariate linear regression. 



Machine Learning

Linear Regression with multiple variables

Gradient descent for multiple variables

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

Parameters: $\underline{\theta_0, \theta_1, \dots, \theta_n}$ Θ n+1 - dimensional vector

Cost function:

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

Gradient descent:

Repeat {
 $\rightarrow \theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \dots, \theta_n)$ $\mathcal{J}(\Theta)$
 }
 ↑ simultaneously update for every $j = 0, \dots, n$

Gradient Descent

Previously (n=1):

Repeat {

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

$\frac{\partial}{\partial \theta_0} J(\theta)$

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

(simultaneously update θ_0, θ_1)

}

New algorithm ($n \geq 1$):

Repeat {

$$\frac{\partial}{\partial \theta_j} J(\theta)$$

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

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

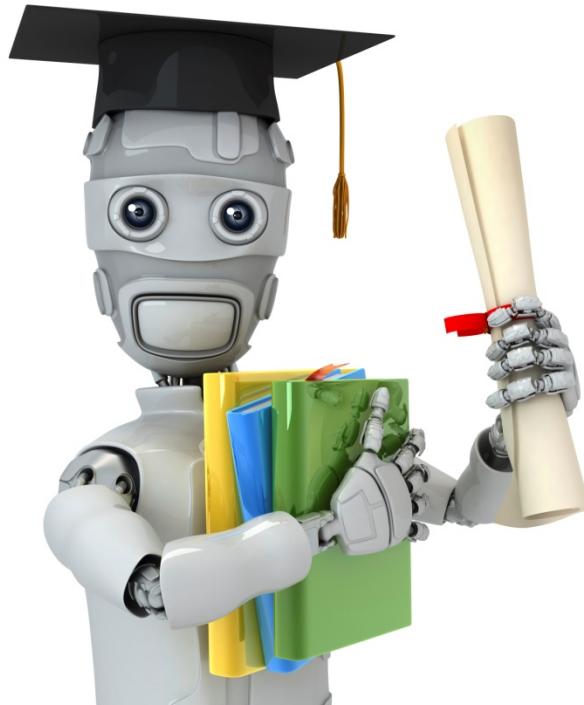
}

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

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

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

...



Machine Learning

Linear Regression with multiple variables

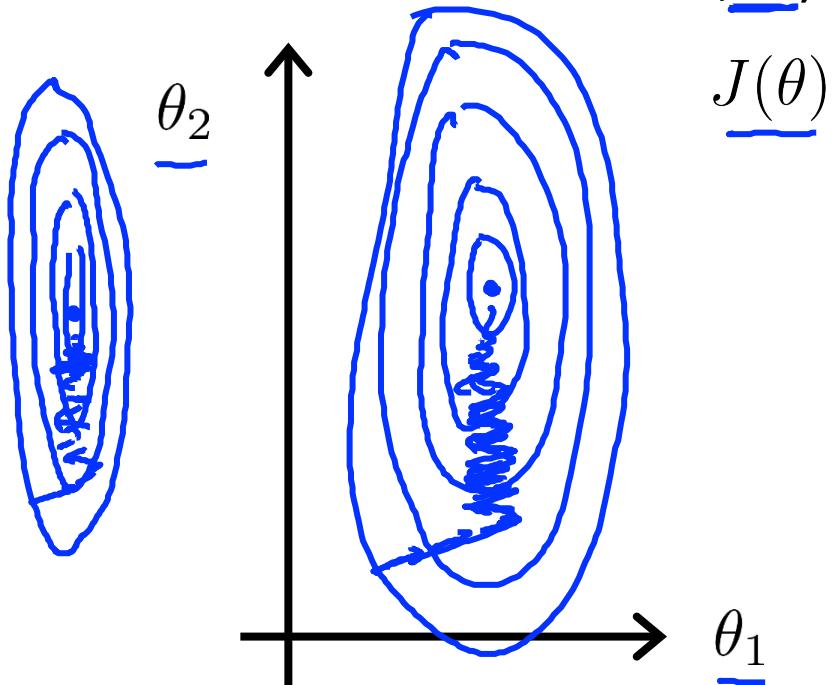
Gradient descent in practice I: Feature Scaling

Feature Scaling

Idea: Make sure features are on a similar scale.

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

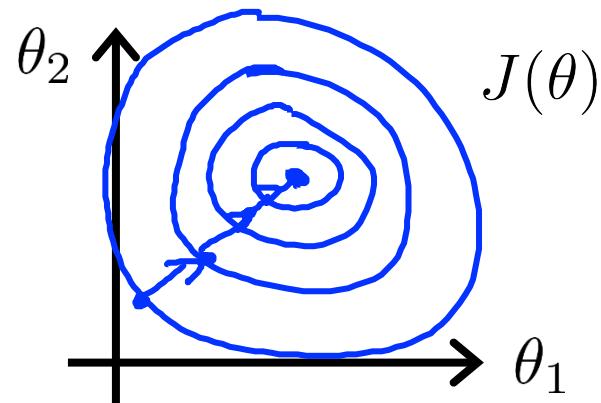
$x_2 = \text{number of bedrooms } (1\text{-}5)$



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

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

$$0 \leq x_1 \leq 1 \quad 0 \leq x_2 \leq 1$$



Feature Scaling

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

$$x_0 = 1$$

$$6 \leq x_1 \leq 3 \quad \checkmark$$

$$-2 \leq x_2 \leq 0.5 \quad \checkmark$$

$$-100 \leq x_3 \leq 100 \quad \times$$

$$-0.0001 \leq x_4 \leq 0.0001 \quad \times$$

$$\boxed{-1 \leq x_i \leq 1}$$

$$-3 \text{ to } 3 \quad \checkmark$$

$$-\frac{1}{2} \text{ to } \frac{1}{2} \quad \checkmark$$

Mean normalization

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

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

Average size ≈ 100

$$x_2 = \frac{\#\text{bedrooms} - 2}{5 - 4}$$

1-5 bedrooms

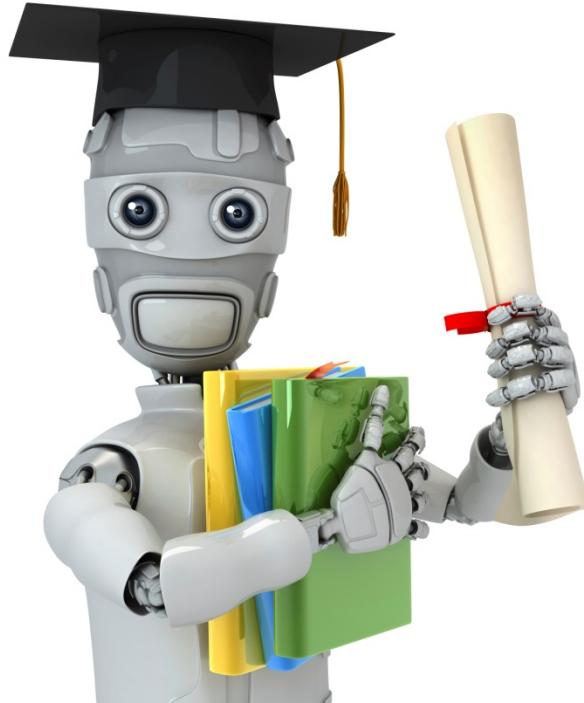
$$\rightarrow [-0.5 \leq x_1 \leq 0.5, -0.5 \leq x_2 \leq 0.5]$$

$$x_1 \leftarrow \frac{x_1 - \mu_1}{\sigma_1}$$

avg value of x_1 in training set

range ($\max - \min$)
(or standard deviation)

$$x_2 \leftarrow \frac{x_2 - \mu_2}{\sigma_2}$$



Machine Learning

Linear Regression with multiple variables

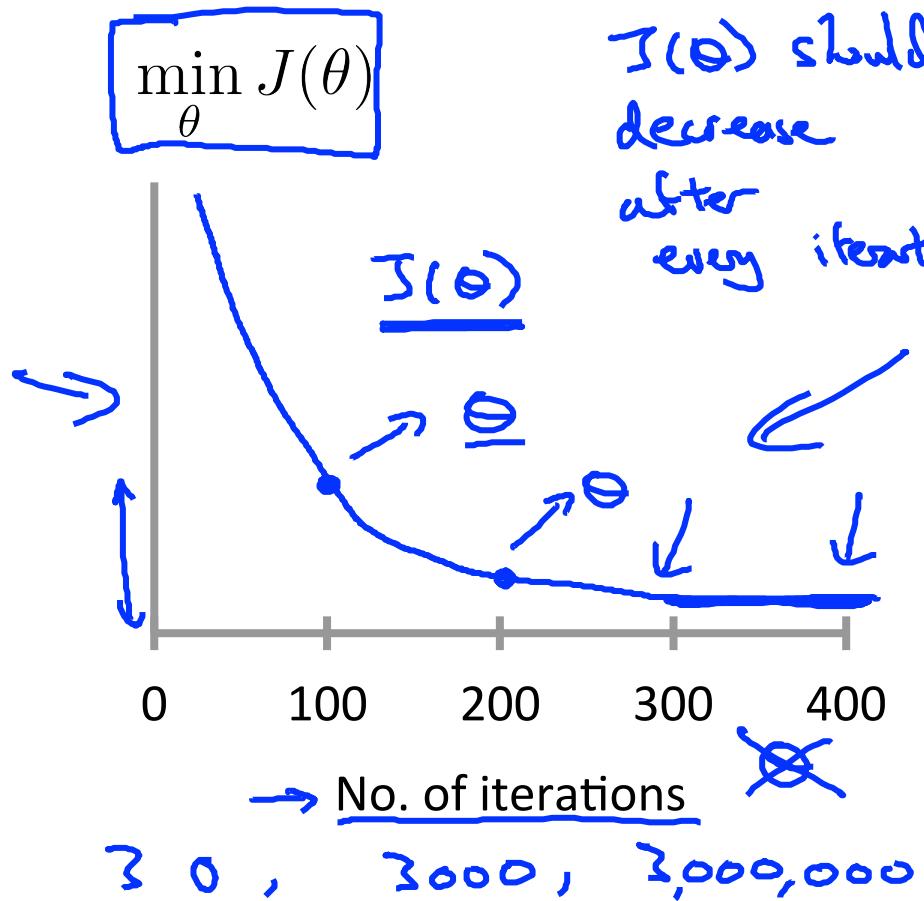
Gradient descent in practice II: Learning rate

Gradient descent

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

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

Making sure gradient descent is working correctly.



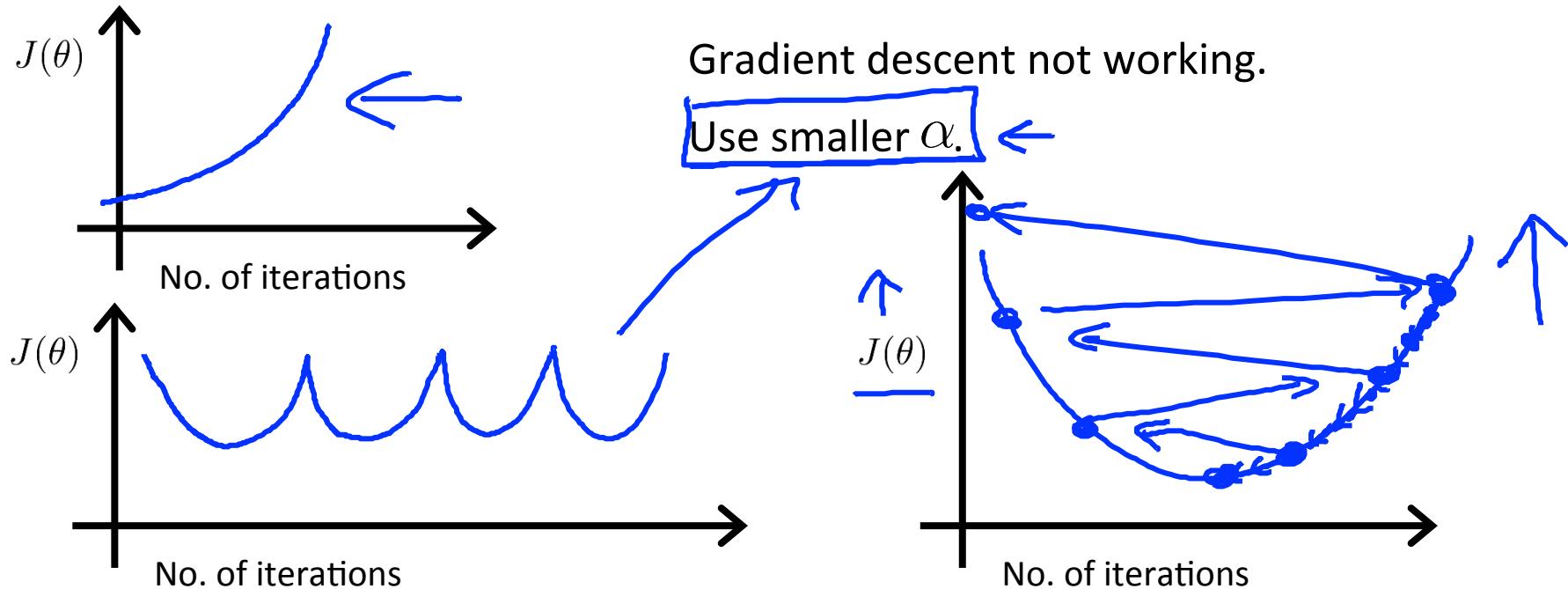
$J(\theta)$ should decrease after every iteration.

→ Example automatic convergence test:

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

$$10^{-3}$$

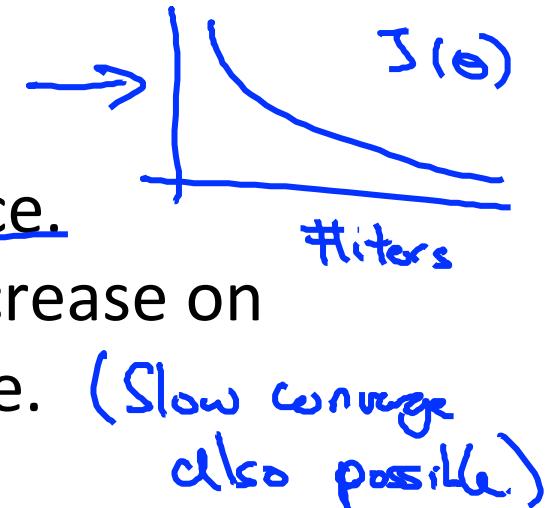
Making sure gradient descent is working correctly.



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

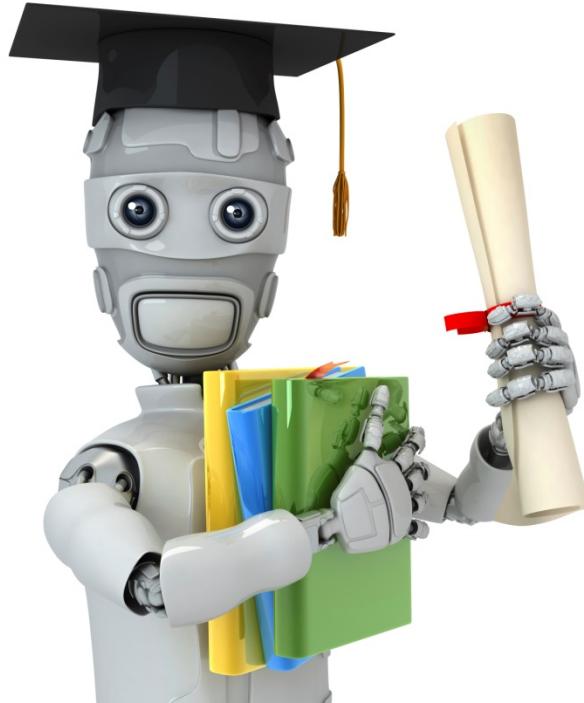
Summary:

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



To choose α , try

$$\dots, \underbrace{0.001}_{\uparrow}, \underbrace{0.003}_{\approx 3x}, \underbrace{0.01}_{\approx 3x}, \underbrace{0.03}_{3x}, \underbrace{0.1}_{\approx 3x}, \underbrace{0.3}_{3x}, \underbrace{1}_{\approx 3x}, \dots$$



Machine Learning

Linear Regression with multiple variables

Features and
polynomial regression

Housing prices prediction

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

x_1
-



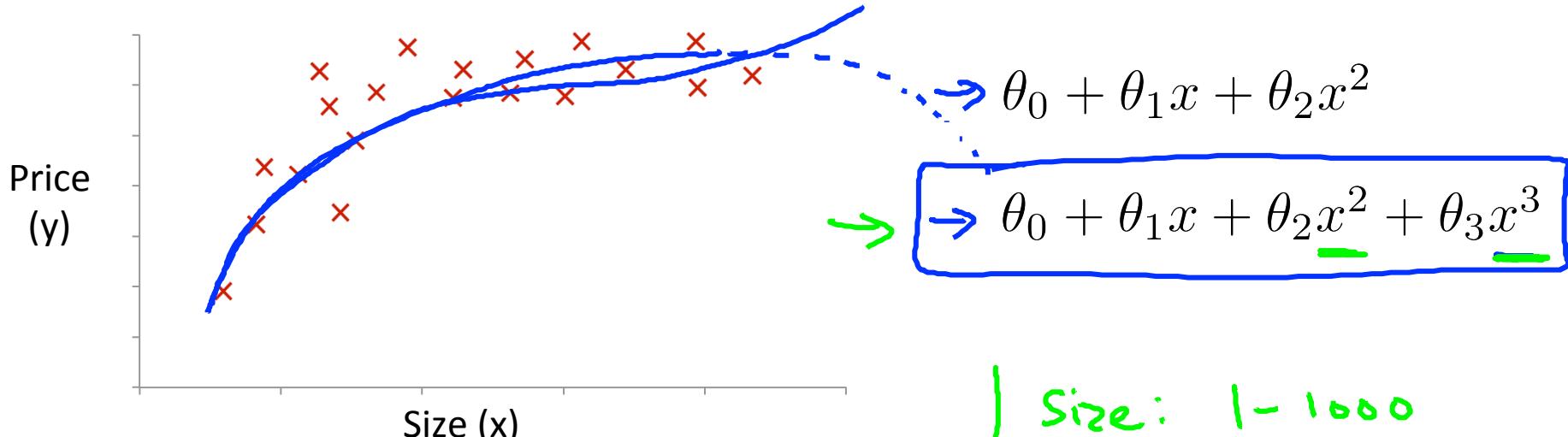
Area

$$\times = \underline{\text{frontage} \times \text{depth}}$$

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

~ land area

Polynomial regression



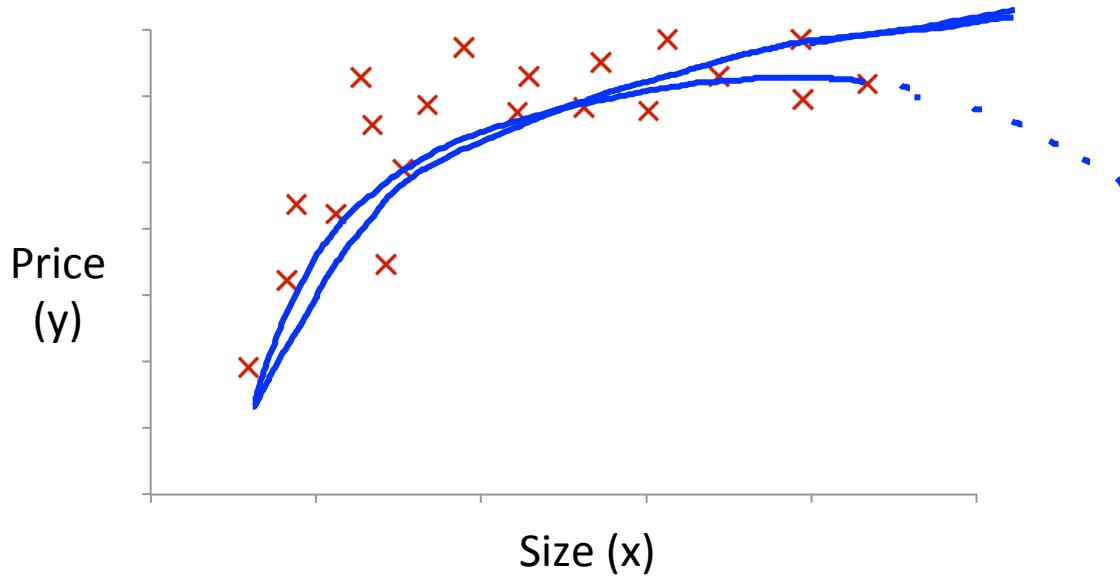
$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$= \theta_0 + \theta_1(\text{size}) + \theta_2(\text{size})^2 + \theta_3(\text{size})^3$$

$$\begin{aligned} \rightarrow x_1 &= (\text{size}) \\ \rightarrow x_2 &= (\text{size})^2 \\ \rightarrow x_3 &= (\text{size})^3 \end{aligned}$$

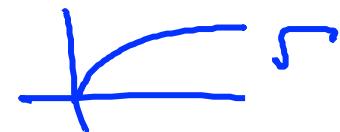
Size: 1 - 1000
Size²: 1 - 1000, 000
Size³: 1 - 10⁹

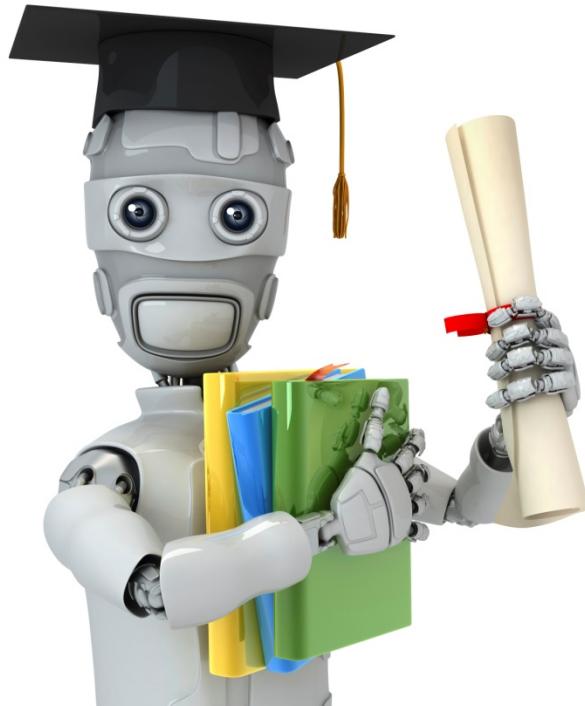
Choice of features



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

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



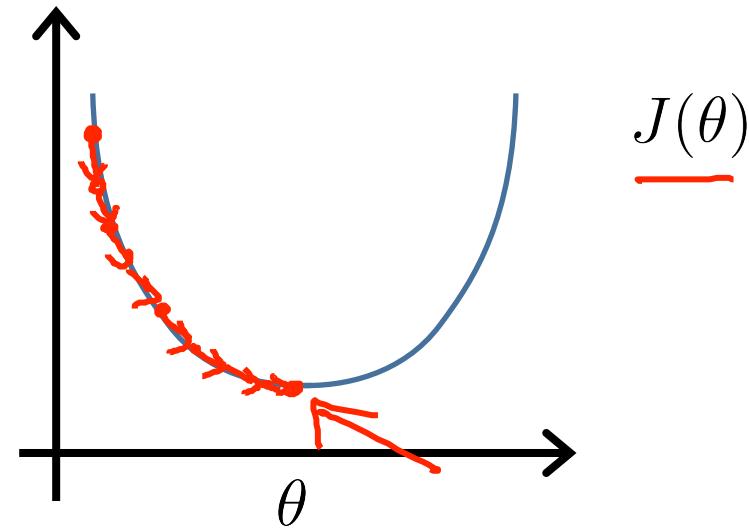


Machine Learning

Linear Regression with multiple variables

Normal equation

Gradient Descent



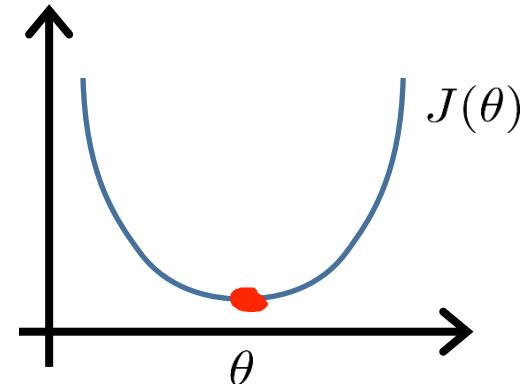
Normal equation: Method to solve for $\underline{\theta}$ analytically.

Intuition: If 1D ($\theta \in \mathbb{R}$)

$$\rightarrow J(\theta) = a\theta^2 + b\theta + c$$

$$\frac{\partial}{\partial \theta} J(\theta) = \dots \stackrel{\text{set}}{=} 0$$

Solve for θ



$$\theta \in \mathbb{R}^{n+1}$$

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

$$\frac{\partial}{\partial \theta_j} J(\theta) = \dots \stackrel{\text{set}}{=} 0 \quad (\text{for every } j)$$

Solve for $\theta_0, \theta_1, \dots, \theta_n$

Examples: $m = 4$.

	Size (feet ²)	Number of bedrooms	Number of floors	Age of home (years)	Price (\$1000)
x_0	x_1	x_2	x_3	x_4	y
1	2104	5	1	45	460
1	1416	3	2	40	232
1	1534	3	2	30	315
1	852	2	1	36	178

Diagram illustrating the data matrix X and the price vector y :

$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$

$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$

$\theta = (X^T X)^{-1} X^T y$

$m \times (n+1)$

m -dimensional vector

m examples $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$; n features.

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ x_1^{(i)} \\ x_2^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix} \in \mathbb{R}^{n+1} \quad X = \begin{bmatrix} \cdots & (x^{(1)})^\top & \cdots \\ \cdots & (x^{(1)})^\top & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & (x^{(m)})^\top & \cdots \end{bmatrix}$$

(design matrix)

E.g. If $x^{(i)} = \begin{bmatrix} 1 \\ x_1^{(i)} \end{bmatrix}$

$$X = \begin{bmatrix} 1 & x_1^{(1)} \\ 1 & x_1^{(2)} \\ \vdots & \vdots \\ 1 & x_1^{(m)} \end{bmatrix}_{m \times 2}$$

$$y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

$$\Theta = (X^T X)^{-1} X^T y$$

$$\theta = \boxed{(X^T X)^{-1} X^T y}$$

$(X^T X)^{-1}$ is inverse of matrix $X^T X$.

Set $A := X^T X$

$$(X^T X)^{-1} = A^{-1}$$

Octave: $\text{pinv}(X' * X) * X' * y$

$$\text{pinv}(X^T * X) * X^T * y$$

$$\theta = \boxed{(X^T X)^{-1} X^T y}$$

$$\min_{\theta} J(\theta)$$

$$\left| \begin{array}{l} X' \\ X^T \\ \hline \cancel{\text{Feature Scaling}} \\ 0 \leq x_1 \leq 1 \\ 0 \leq x_2 \leq 1000 \\ 0 \leq x_3 \leq 10^{-5} \end{array} \right| \checkmark$$

m training examples, n features.

Gradient Descent

- • Need to choose α .
- • Needs many iterations.
- Works well even when n is large.

$$\underline{n = 10^6}$$

Normal Equation

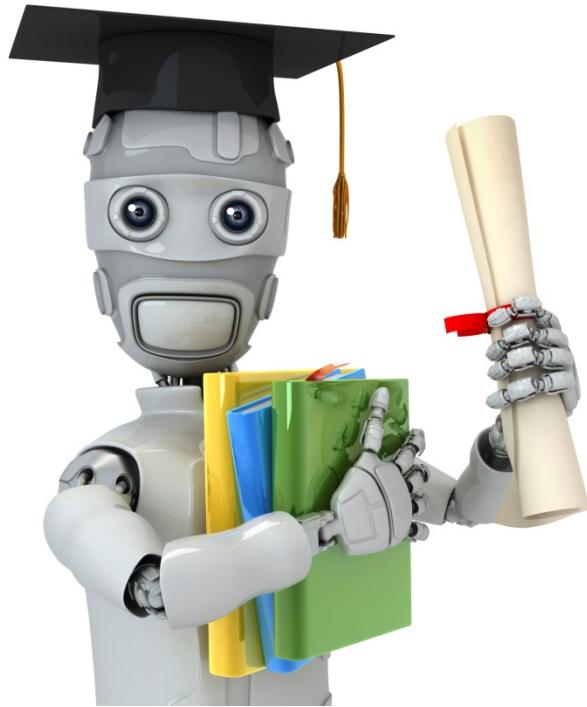
- • No need to choose α .
- • Don't need to iterate.
- Need to compute
$$(X^T X)^{-1}$$
 $n \times n$ $O(n^3)$
- Slow if n is very large.

$$n = 100$$

$$n = 1000$$

$$\dots - n = \underline{10000}$$





Machine Learning

Linear Regression with multiple variables

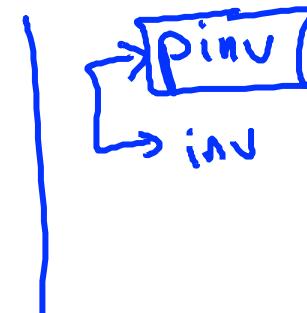
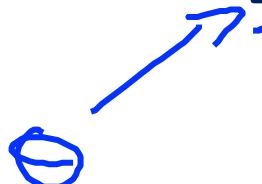
Normal equation
and non-invertibility
(optional)

Normal equation

$$\theta = \underline{(X^T X)^{-1} X^T y}$$

$X^T X$

- What if $X^T X$ is non-invertible? (singular/degenerate)
- Octave: `pinv(X' * X) * X' * y`



What if $X^T X$ is non-invertible?



- Redundant features (linearly dependent).

E.g.

$$\begin{aligned}x_1 &= \text{size in feet}^2 \\x_2 &= \text{size in m}^2 \\x_1 &= (3.28)^2 x_2\end{aligned}$$

$$1_m = 3.28 \text{ feet}$$

$$\rightarrow \underline{n = 10} \leftarrow$$

$$\rightarrow \underline{n = 100} \leftarrow$$

$$\Theta \in \mathbb{R}^{101}$$

- Too many features (e.g. $m \leq n$).

- Delete some features, or use regularization.

↓ later