Linear regression

Lesson #1: Linear regression with one variable

Lesson #2: Linear regression with multiple variables

Duration: 5 hrs

Source: Machine Learning, Andrew Ng, coursera.org

Lesson #2: Linear regression with multiple variables

- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descend for multiple variables
    - Feature Scaling
    - Learning rate
    - Features and polynomial regression
    - Normal equation

Multiple features (variables).

Size (feet^2)	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^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
…	…	…	…	…

Multiple features (variables).

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

$(\mathrm{m} = 47)$

Notation:

n = number of features $(n = 4)$

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

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

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

$x_3^{(2)} = 2$

Hypothesis:

Previously: $h_\theta(x) = \theta_0 + \theta_1 x$ (one variable)

Multiple variables:

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

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

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

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

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1} \qquad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \in \mathbb{R}^{n+1}$$

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_n x_n$$
$$= \theta^T x$$

$$[\theta_0 \theta_1 \theta_2 \ldots \theta_n] \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

θ^T x

$1 \times (n+1)$ matrix

Multivariate linear regression.

Lesson #2: Linear regression with multiple variables

- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descent for multiple variables

$x_0 = 1$

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: $\theta_0, \theta_1, \ldots, \theta_n$ θ $(n+1)$ *dimensional vector*

Cost function:

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

$J(\theta)$

Gradient descent:

Repeat $\{$

$J(\theta)$

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

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

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

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

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

$\}$

$x_0^{(i)}=1$

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

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

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

$\ldots$

Lesson #2: Linear regression with multiple variables

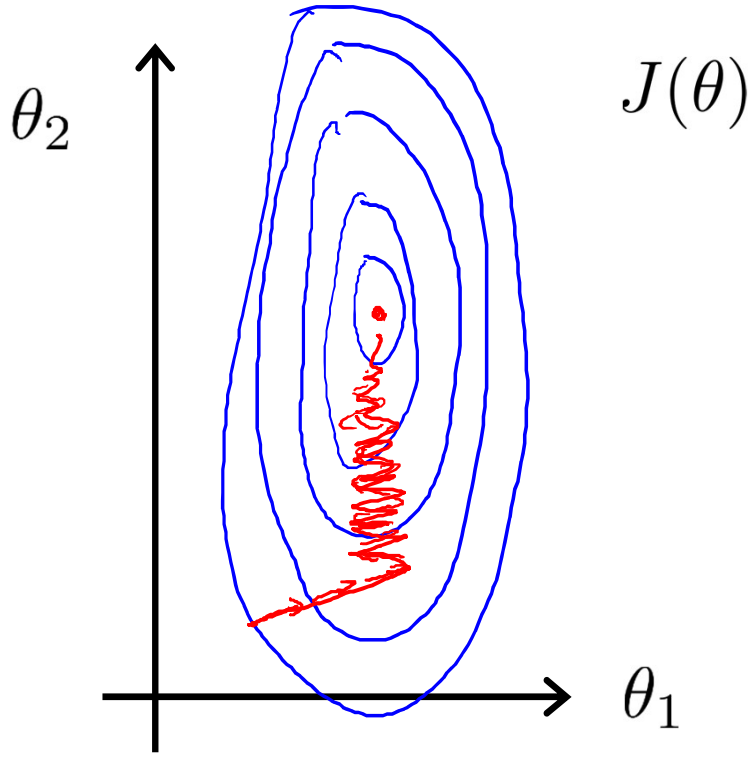
- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descent for multiple variables
    - Feature Scaling
    - Learning rate
    - Features and polynomial regression
    - Normal equation

Feature Scaling

Idea: Make sure features are on a similar scale.

E.g. x_1 = size (0-2000 feet^2)

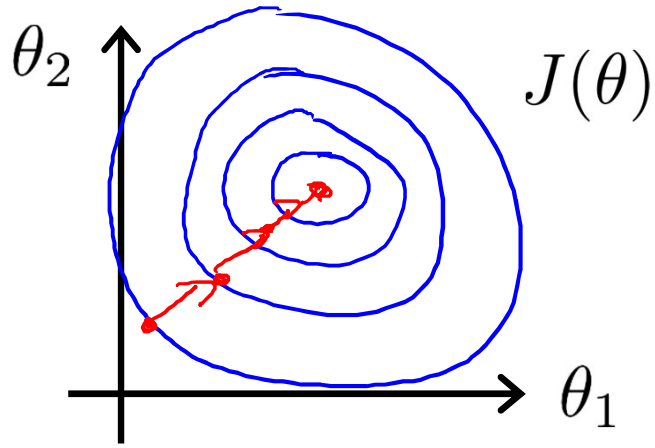
x_2 = number of bedrooms (1-5)



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

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

$-1 \le x_1 \le 1$ $\quad$ $-1 \le x_1 \le 1$



Feature Scaling

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

$$x_0 = 1$$
$$0 \leq x_1 \leq 3$$
$$-2 \leq x_2 \leq 0.5$$
$$-100 \leq x_3 \leq 100 \quad \text{Too big}$$
$$-0.0001 \leq x_4 \leq 0.0001 \quad \text{Too small}$$

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

$x_2 = \frac{\#bedrooms - 2}{4}$ 1-5 bedrooms

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

$$x_i \leftarrow \frac{x_i - \mu_i}{s_i}$$

μ_i: *average value of* x_i *in training set*

s_i: *range* (max − min) *or standard deviation*

Lesson #2: Linear regression with multiple variables

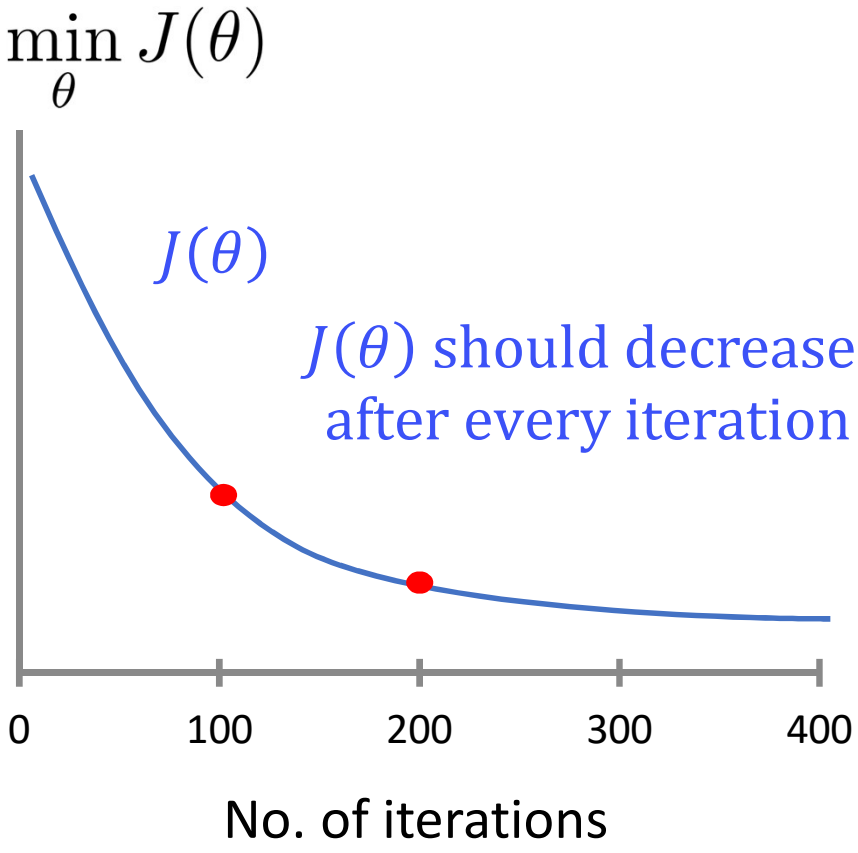
- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descent for multiple variables
    - Feature Scaling
    - Learning rate
    - Features and polynomial regression
    - Normal equation

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 α.

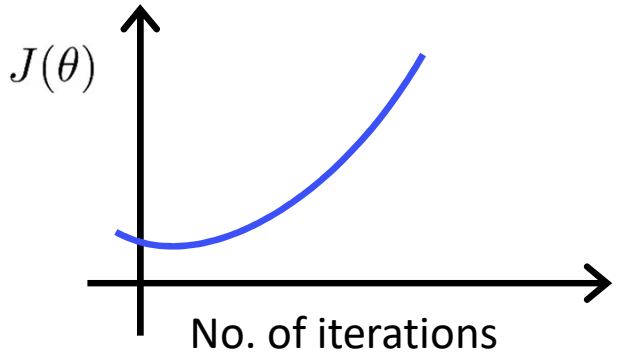
Making sure gradient descent is working correctly.

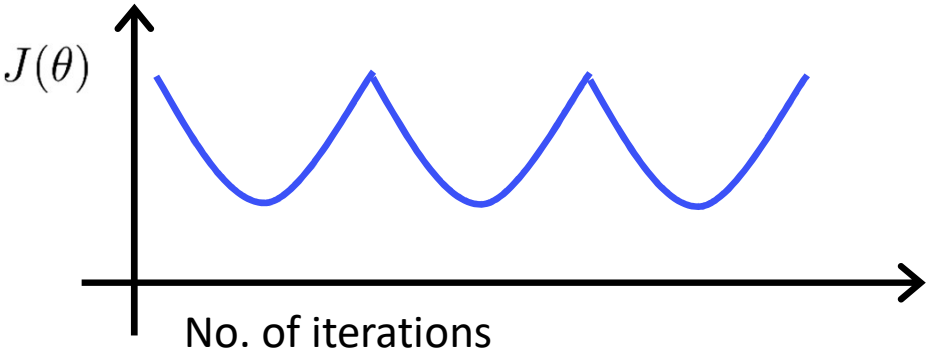


Example automatic convergence test:

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

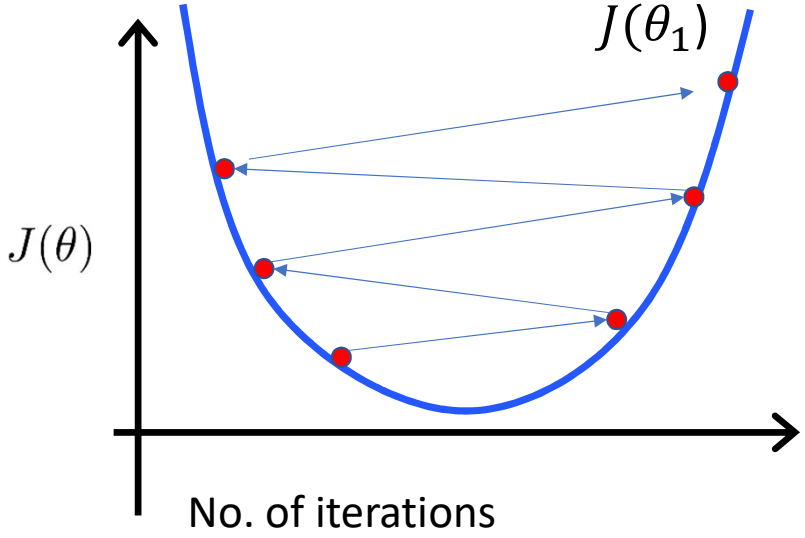
Making sure gradient descent is working correctly.





Gradient descent not working.

Use smaller α.



- 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.

To choose α, try

$$\ldots, 0.001, 0.003, 0.01, 0.03, 0.1, 0.3, 1, \ldots$$

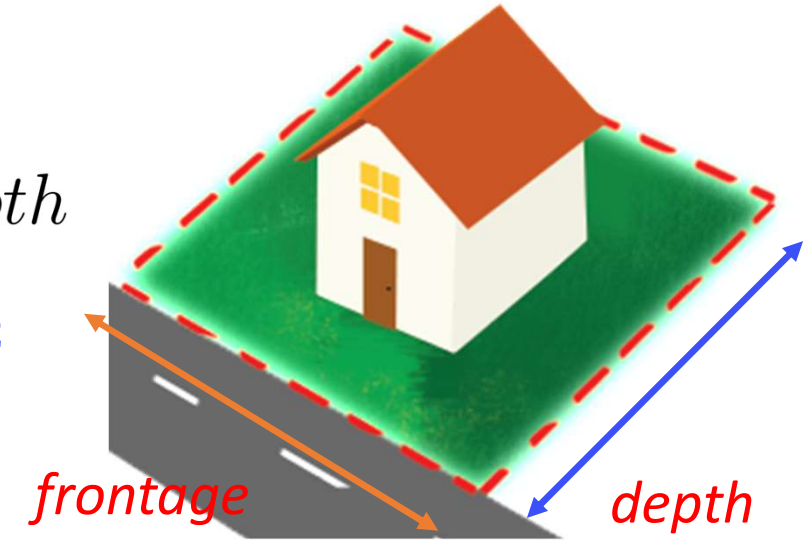
Lesson #2: Linear regression with multiple variables

- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descent for multiple variables
    - Feature Scaling
    - Learning rate
    - Features and polynomial regression
    - Normal equation

Housing prices prediction

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

x_1 x_2

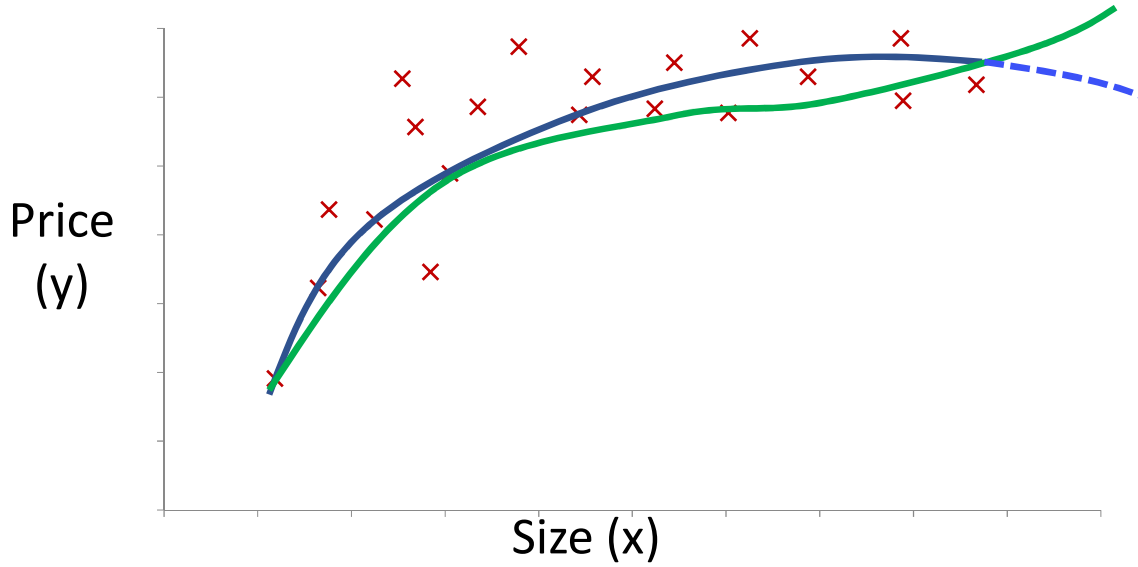


$\underline{Area}$:

$$x = frontage \times depth$$

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

Polynomial regression



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

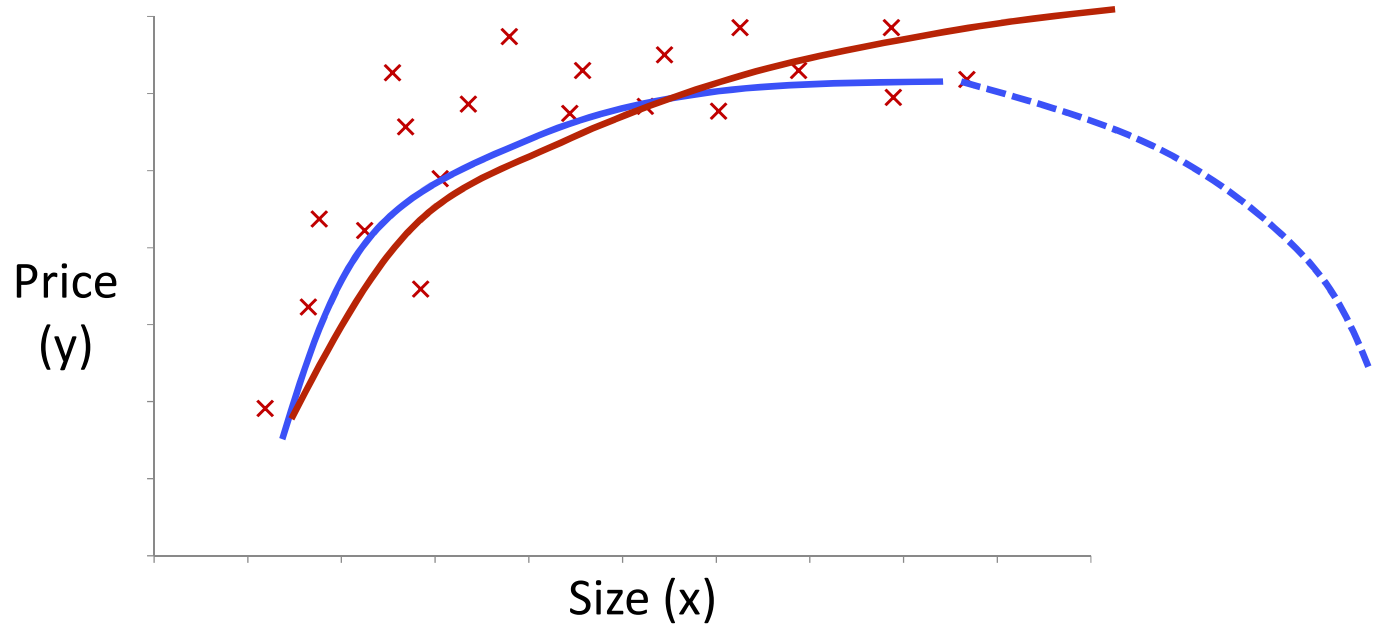
$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

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

$$x_1 = (size)$$
$$x_2 = (size)^2$$
$$x_3 = (size)^3$$

$size : 1 - 1000$
$size^2 : 1 - 1000{,}0000$
$size^3 : 1 - 10^9$

Choice of features



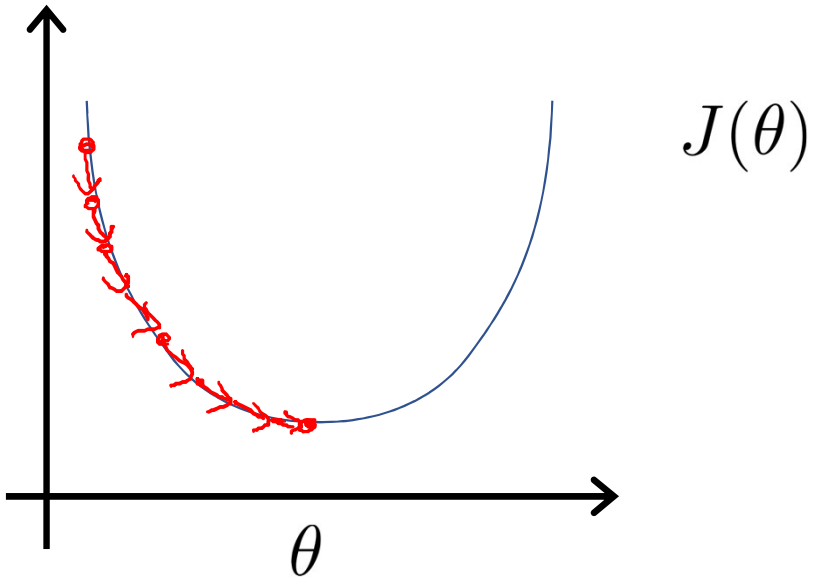
$$h_\theta(x) = \theta_0 + \theta_1(size) + \theta_2(size)^2$$

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

Lesson #2: Linear regression with multiple variables

- **Duration**: 2 hrs
- **Outline**:
  - Model representation
  - Gradient descent for multiple variables
    - Feature Scaling
    - Learning rate
    - Features and polynomial regression
    - Normal equation

Gradient Descent



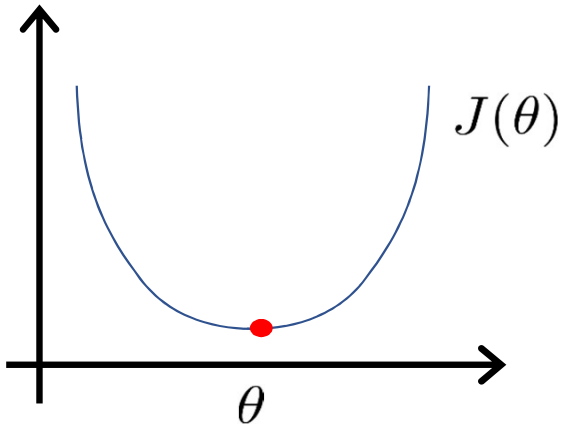
Normal equation: Method to solve for θ analytically.

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

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

$$\frac{d}{d\theta} J(\theta) = \ \ldots = 0$$

Solve for θ



$$\theta \in \mathbb{R}^{n+1} \qquad J(\theta_0, \theta_1, \ldots, \theta_m) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

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

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

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

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

m x $(n + 1)$

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

m – dimensional vector

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

Examples: $m = 5.$

x_0	Size (feet^2) x_1	Number of bedrooms x_2	Number of floors x_3	Age of home (years) x_4	Price ($1000) 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
1	3000	4	1	38	540

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \\ 1 & 3000 & 4 & 1 & 38 \end{bmatrix} \qquad y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \\ 540 \end{bmatrix}$$

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

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

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

$$X = \begin{bmatrix} \ldots & (x^{(1)})^T & \ldots \\ \ldots & (x^{(2)})^T & \ldots \\ & \vdots & \\ \ldots & (x^{(m)})^T & \ldots \end{bmatrix}$$

mx$(n+1)$

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

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

$$X = \begin{bmatrix} 1 & x_1^{(1)} \\ 1 & x_2^{(1)} \\ & \vdots \\ 1 & x_m^{(1)} \end{bmatrix} \quad y = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

mx2

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

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

Set $A = X^T X$

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

Matlab `pinv(X'*X)*X'*y`

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

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

NO Feature Scaling

$0 \le x_1 \le 1$

$0 \le x_2 \le 1000$

$0 \le x_3 \le 10^{-5}$

m training examples, n features.

Gradient Descent	Normal Equation
• Need to choose α. • Needs many iterations. • Works well even when n is large.	• No need to choose α. • Don't need to iterate. • Need to compute $(X^T X)^{-1}$ • Slow if n is very large.

Normal equation

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

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

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

- Redundant features (linearly dependent).
  E.g. x_1 = size in feet^2
  x_2 = size in m^2

  $x_{1} = (3.28)^2 x_2$

- Too many features (e.g. $m \leq n$).
  - Delete some features, or use regularization.