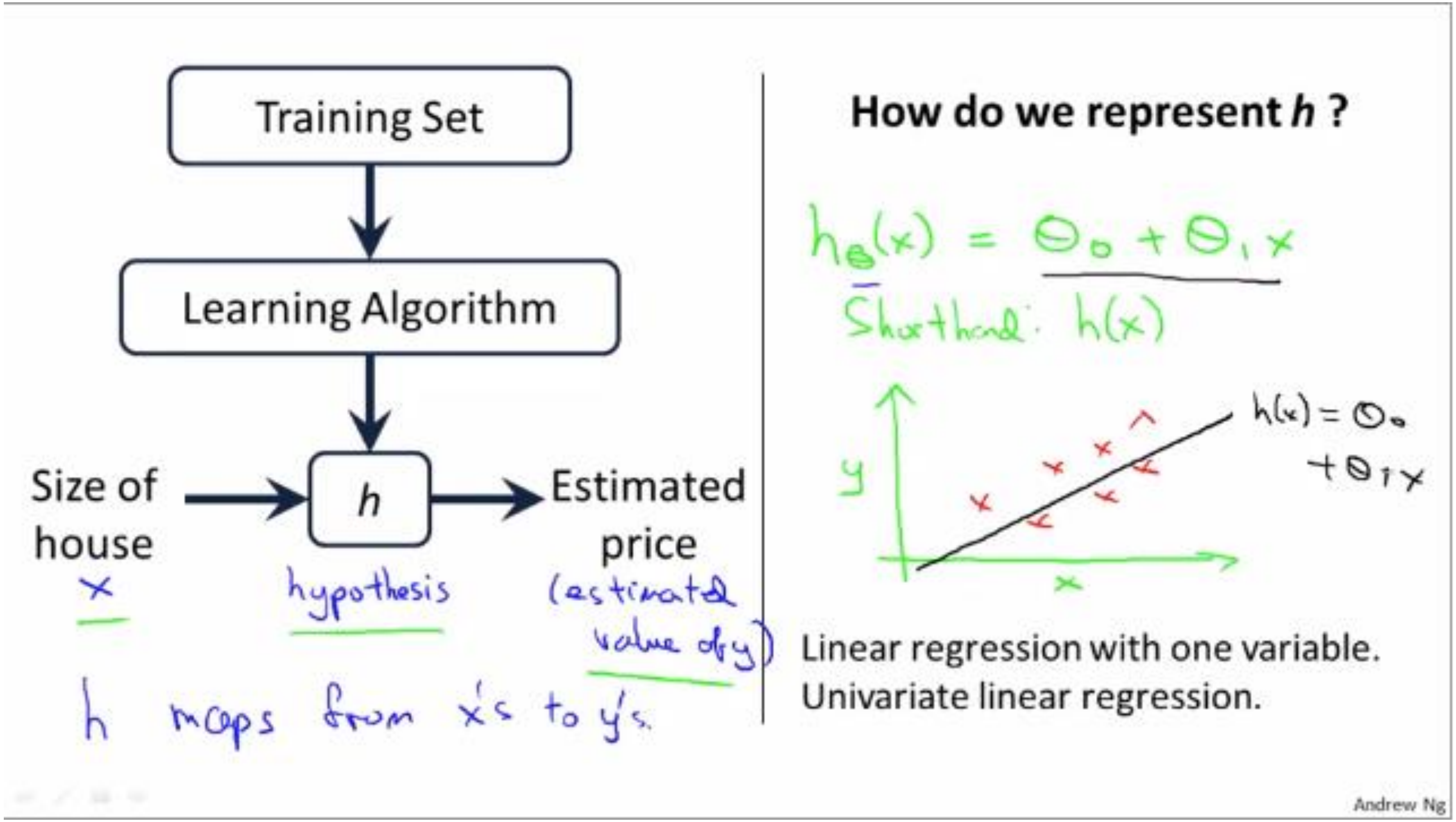
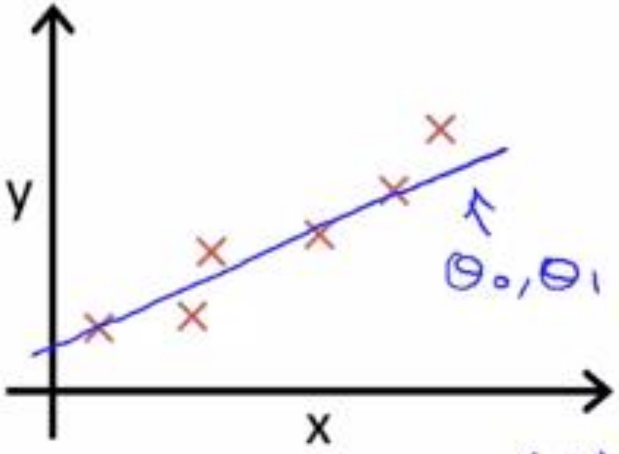
Linear Regression

Simple Linear Regression



Cost Function



$(x^{(i)}, y^{(i)})$

Idea: Choose θ_0, θ_1 so that $h_\theta(x)$ is close to y for our training examples (x, y)

x, y

training examples

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

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

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

$$\underset{\theta_0, \theta_1}{\text{minimize}} \ \underbrace{J(\theta_0, \theta_1)}_{\text{Cost function}}$$

Squared error function

Andrew Ng

Used for linear regression

Cost Function - Intuition I

Hypothesis:

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

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

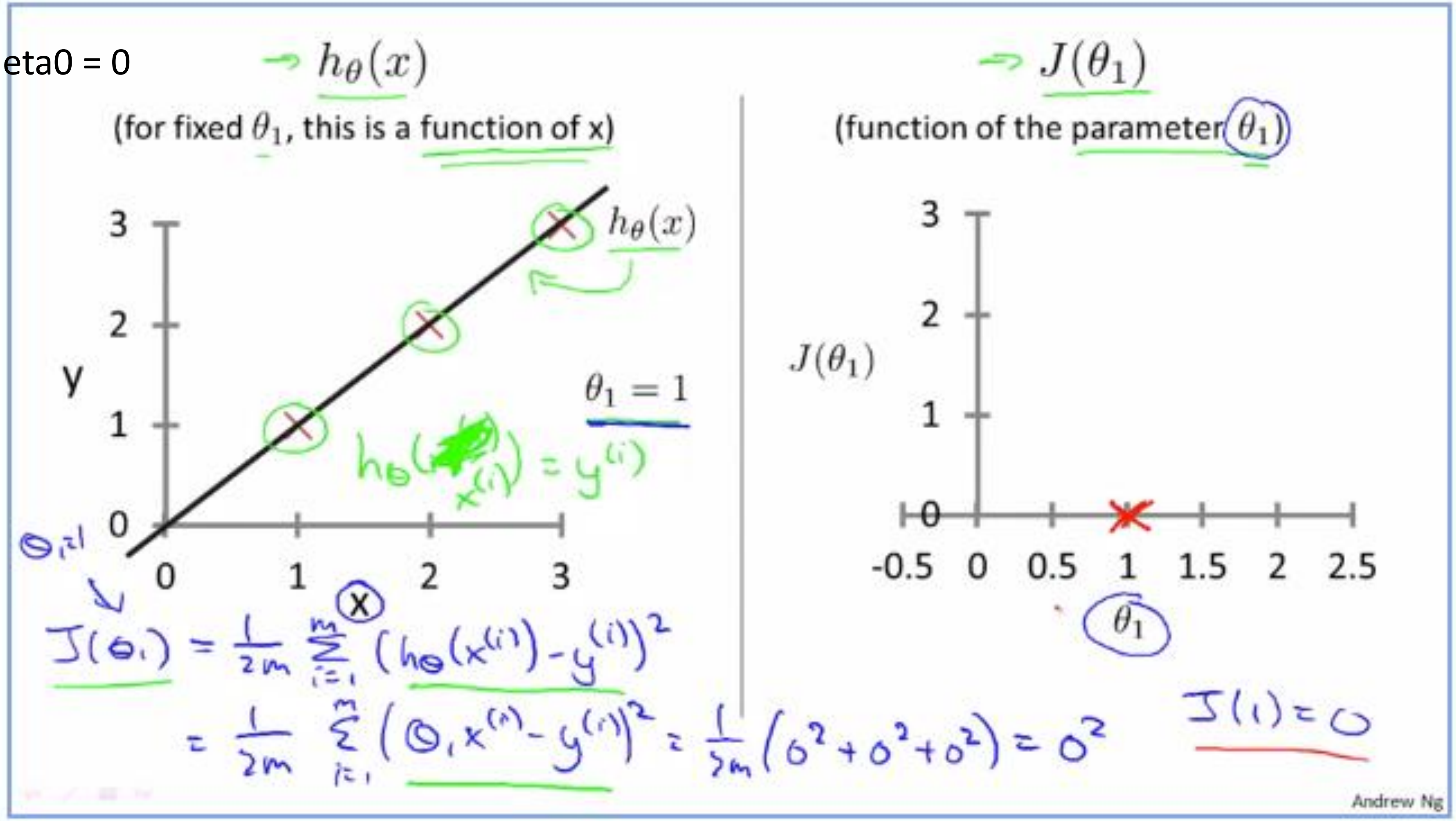
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

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

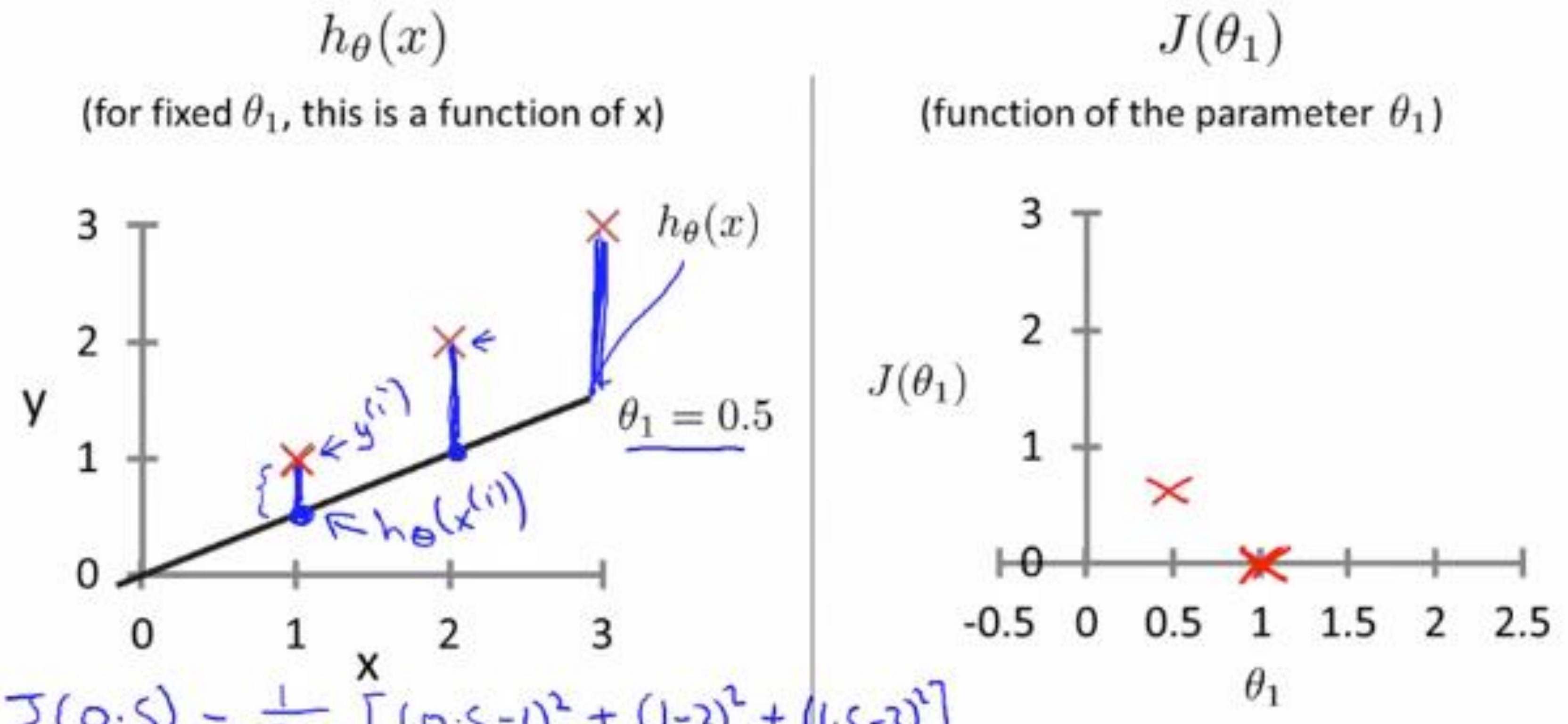
Andrew Ng

Cost Function - Intuition I

Consider Theta0 = 0



Cost Function - Intuition I

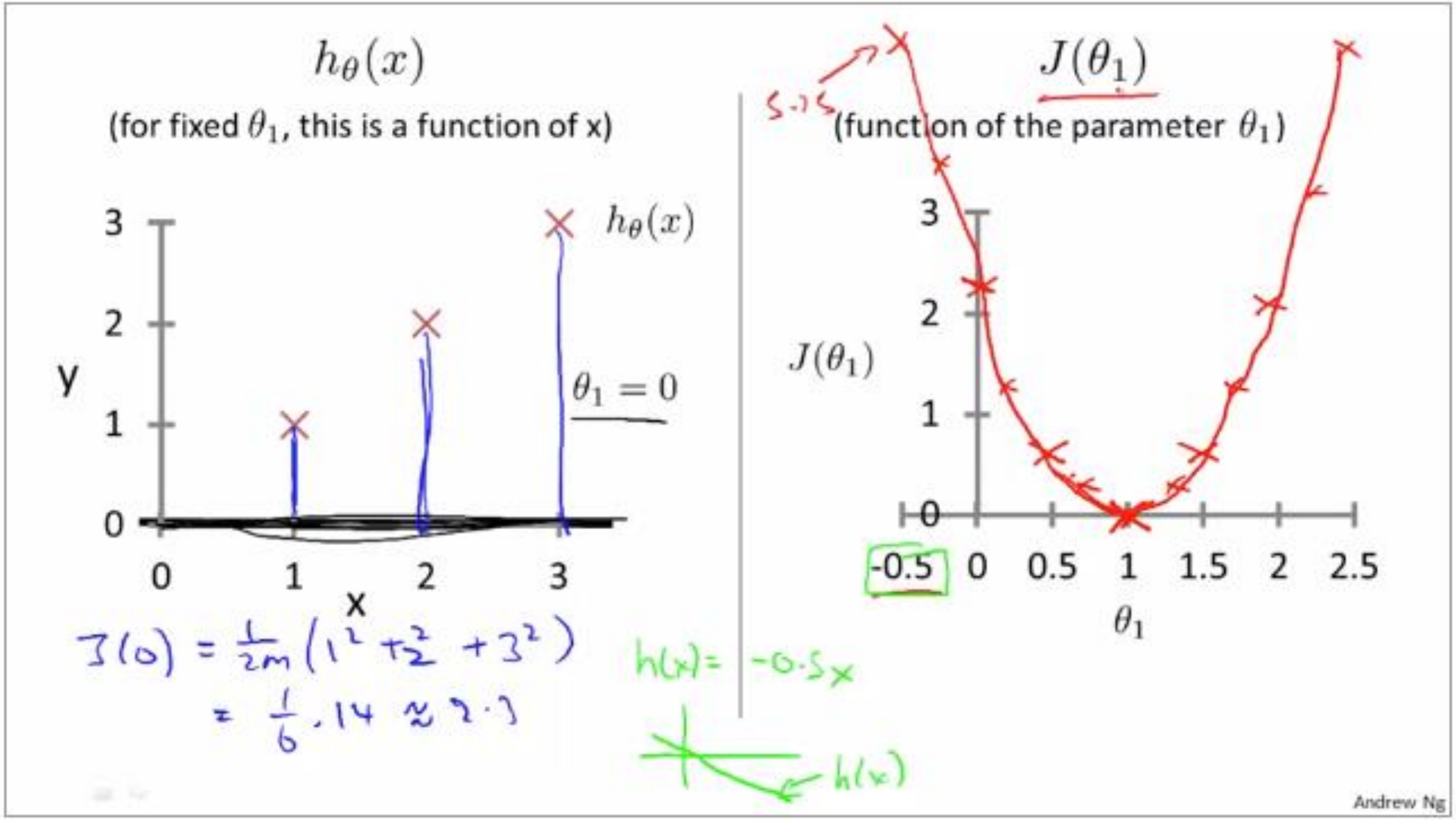


$$J(0.5) = \frac{1}{2m}\left[(0.5-1)^2 + (1-2)^2 + (1.5-3)^2\right]$$

$$= \frac{1}{2\times 3}(3.5) = \frac{3.5}{6} \approx 0.58$$

Andrew Ng

Cost Function - Intuition I



Cost Function - Intuition II

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

Parameters: θ_0, θ_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: $\underset{\theta_0, \theta_1}{\text{minimize}} \; J(\theta_0, \theta_1)$

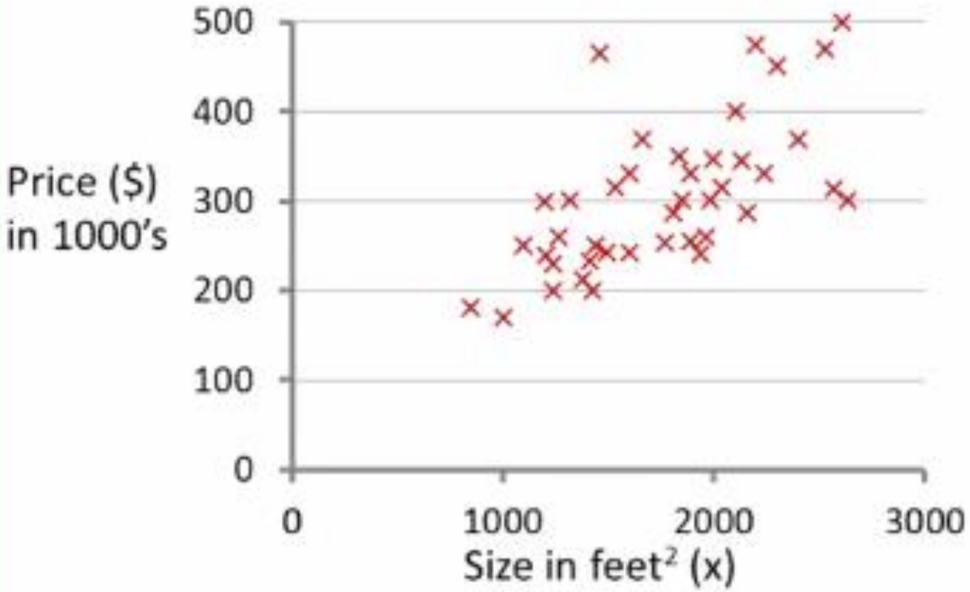
Andrew Ng

Considering both parameters

Cost Function - Intuition II

$h_\theta(x)$

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

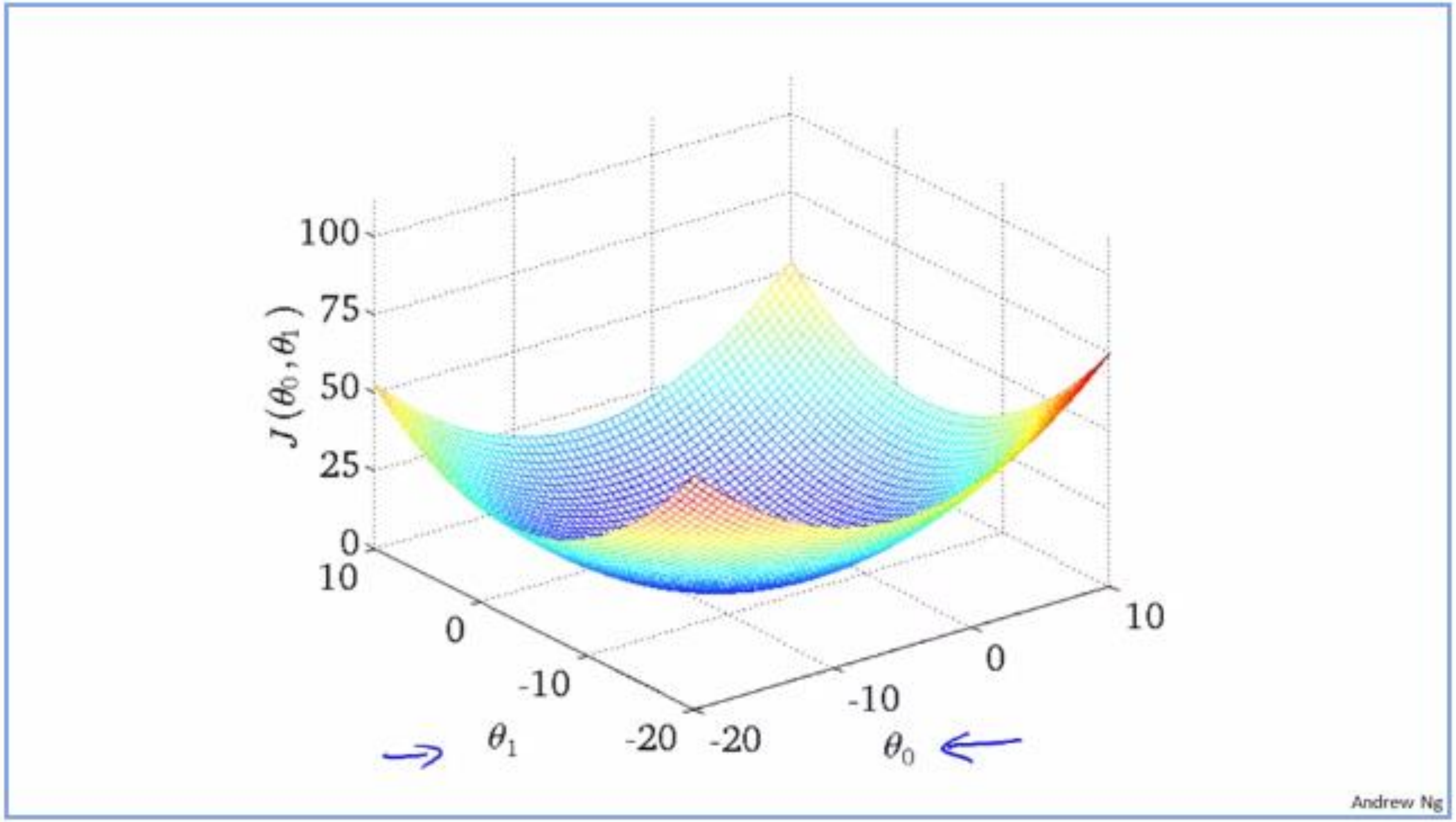


$J(\theta_0, \theta_1)$

(function of the parameters θ_0, θ_1)

Andrew Ng

Cost Function - Intuition II

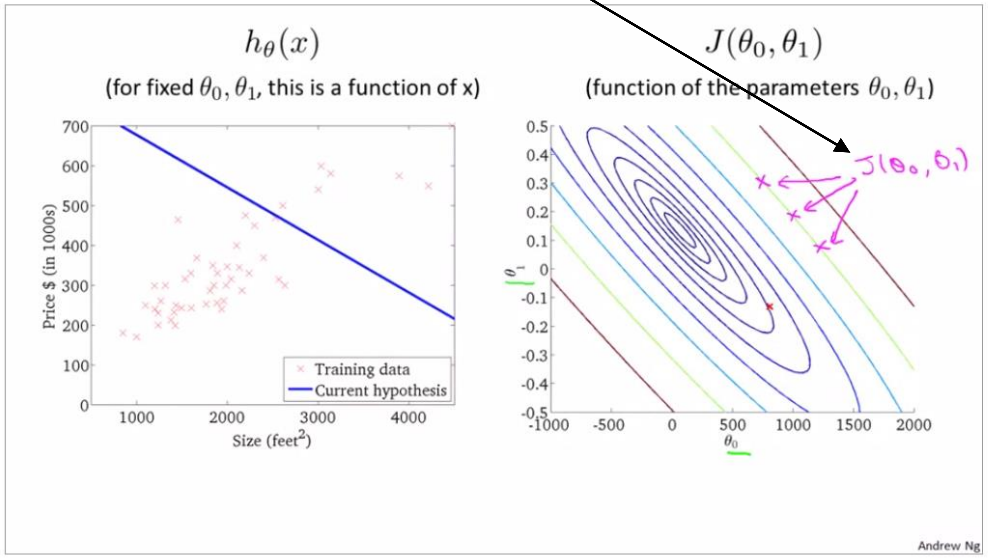


https://towardsdatascience.com/linear-regression-cost-function-gradient-descent-normal-equations-1d2a6c878e2c

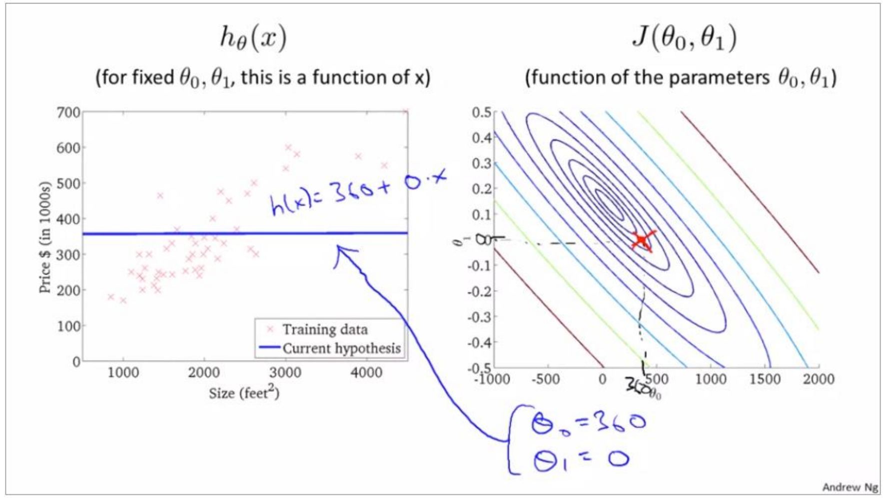
Any point on the same eclipses will give us the same value of error function J.

contour plot

Cost Function - Intuition II

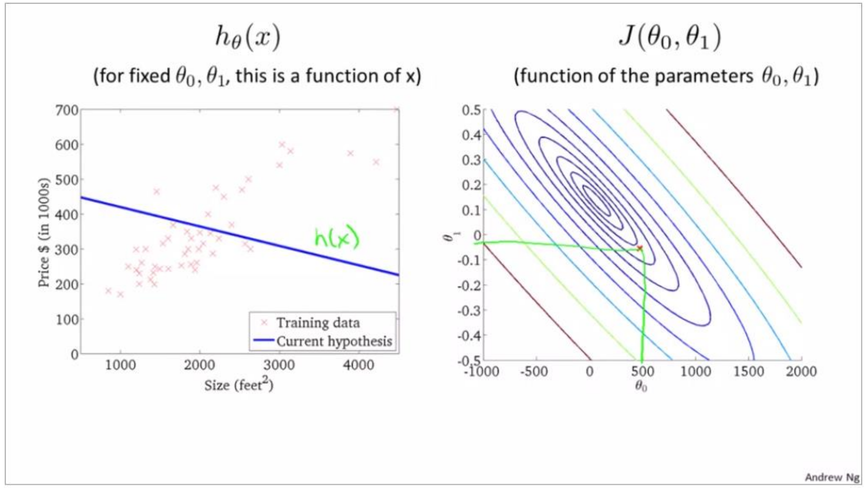


Cost Function - Intuition II

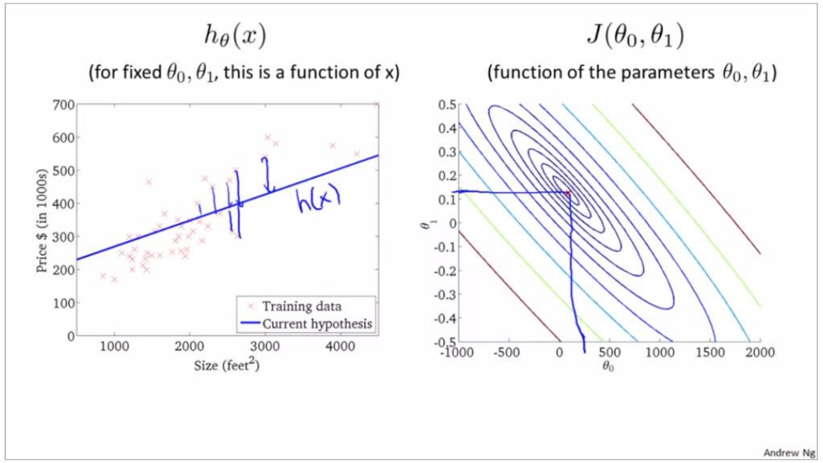


We want algorithm which Automatically finding the value of Theta0 and Theta1 and minimize J (Cost function)

Cost Function - Intuition II



Cost Function - Intuition II



Gradient Descent

- Automatically finding the value of Theta0 and Theta1 and minimize J (Cost function) for Linear Regression.
- Used for other algorithm as well.

Have some function $J(\theta_0, \theta_1)$ $J(\theta_0, \theta_1, \theta_2, \dots, \theta_n)$

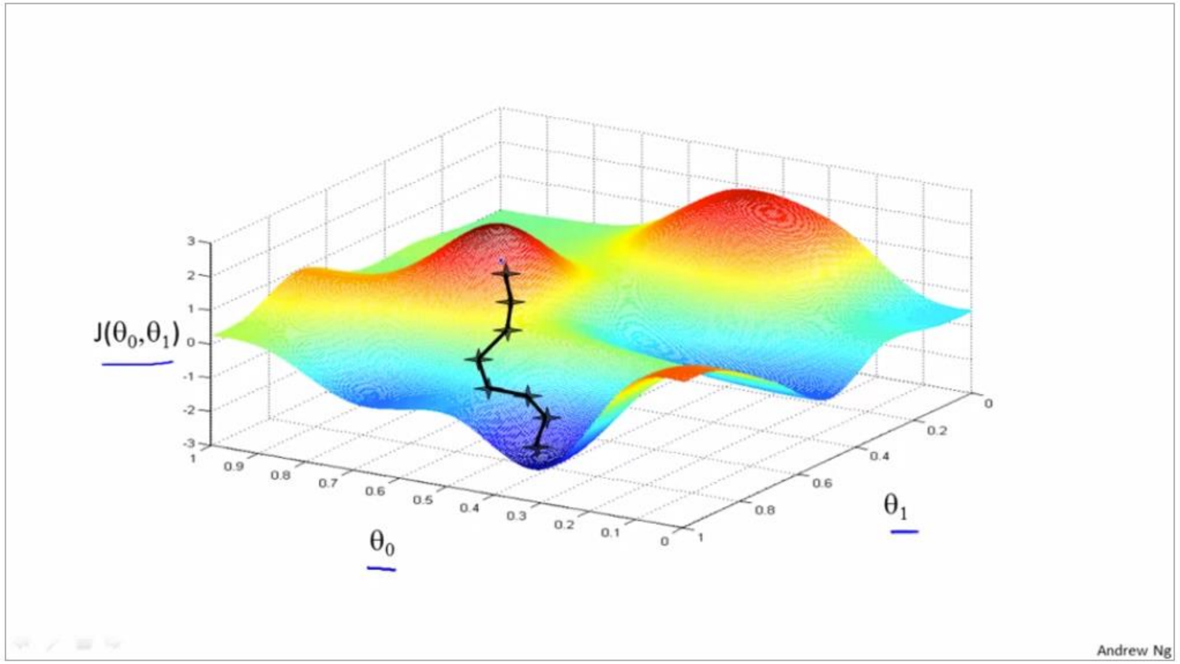
Want $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$ $\min_{\theta_0 \dots \theta_n} J(\theta_0, \dots, \theta_n)$

Outline:

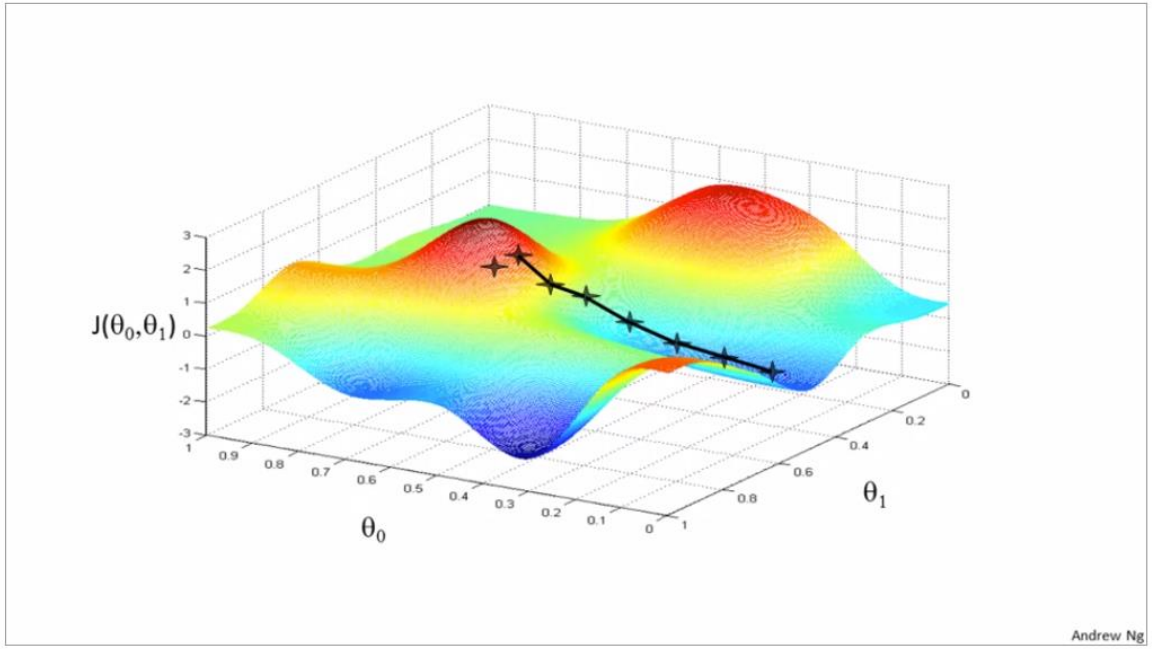
- Start with some θ_0, θ_1 (say $\theta_0 = 0, \theta_1 = 0$)
- Keep changing θ_0, θ_1 to reduce $J(\theta_0, \theta_1)$ until we hopefully end up at a minimum

Andrew Ng

Gradient Descent



Gradient Descent



Gradient Descent

Gradient descent algorithm

Assignment

$a := b$

$a := a+1$

Truth assertion

$a = b$

$a = a+1$ ✗

θ_0, θ_1

repeat until convergence {

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

}

learning rate

Simultaneously update θ_0 and θ_1

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:

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

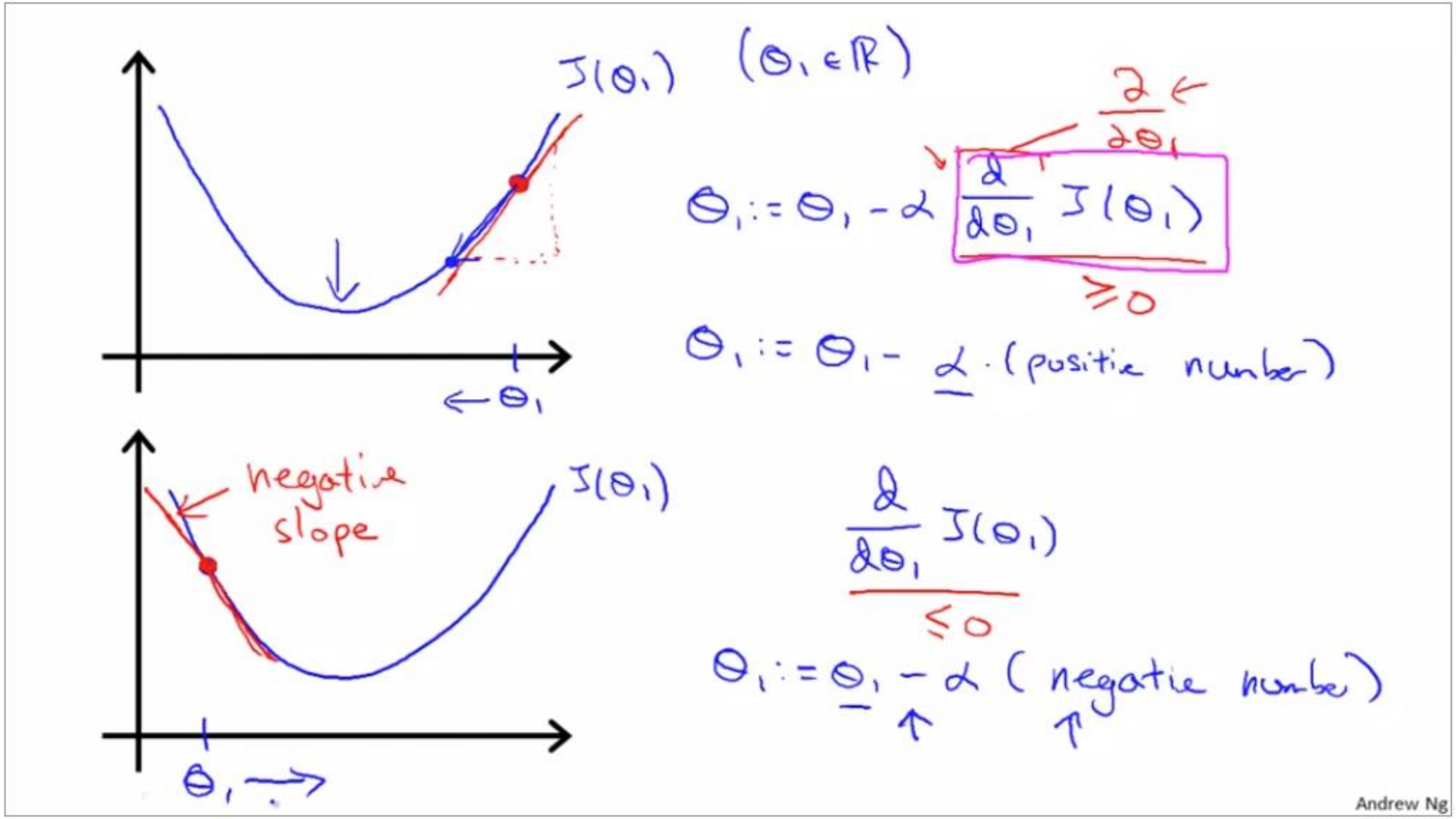
$\theta_0 := \text{temp0}$

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

$\theta_1 := \text{temp1}$

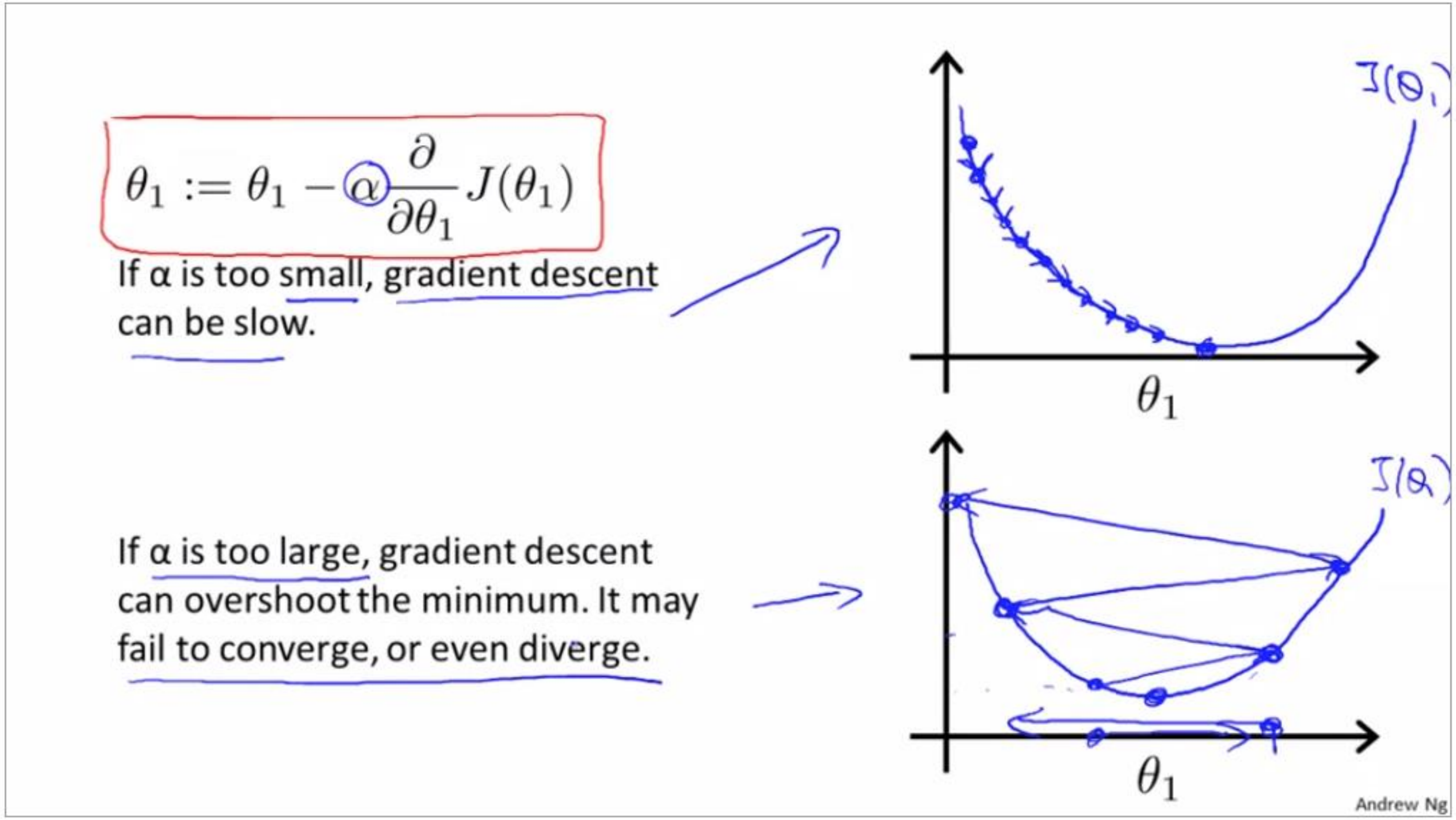
Andrew Ng

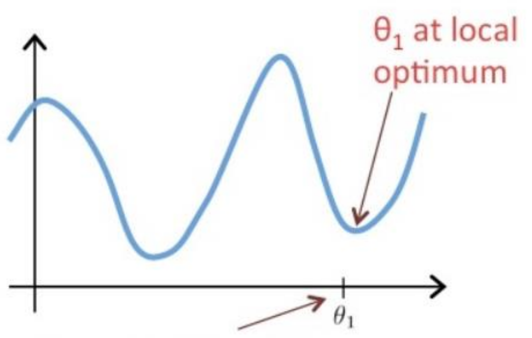
Gradient Descent Intuition



All about slope: https://www.occc.edu/wp-content/legacy/sem/mathhandouts/All%20About%20Slopes.pdf

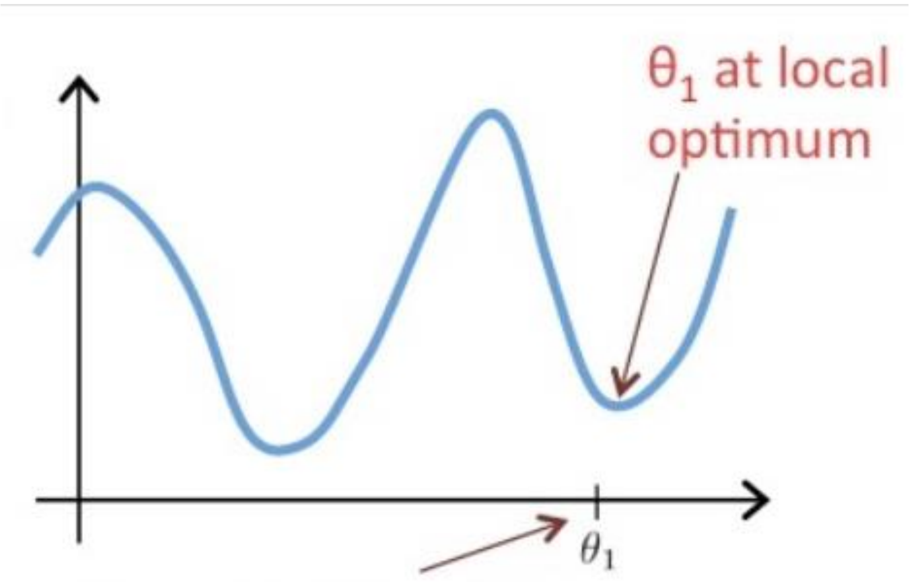
Gradient Descent Intuition





Question: Suppose theta_1 is at a local optimum of J(\theta_1), such as shown in the figure. What will one step of gradient descent do?

Question: Suppose theta_1 is at a local optimum of J(\theta_1), such as shown in the figure. What will one step of gradient descent do?

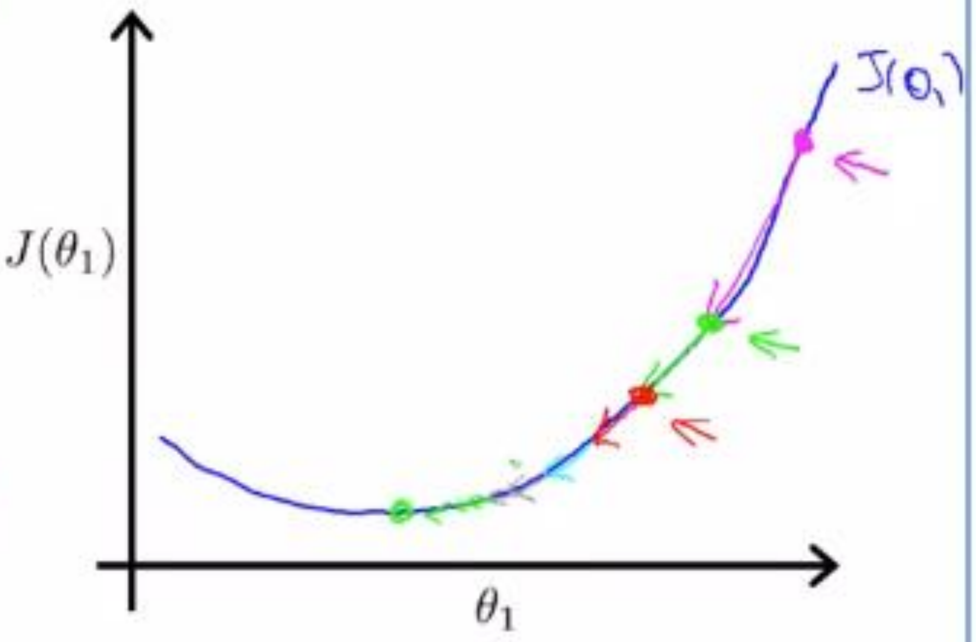


Gradient Descent Intuition

Gradient descent can converge to a local minimum, even with the learning rate α fixed.

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

As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease α over time.



Andrew Ng

Gradient Descent For Linear Regression

Machine Learning

Linear regression with one variable

Gradient descent for linear regression

Gradient Descent For Linear Regression

Gradient descent algorithm

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

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

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

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

Andrew Ng

Gradient Descent For Linear Regression

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

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

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

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

Andrew Ng

Gradient descent algorithm

$$\frac{\partial}{\partial\theta_0} J(\theta_0, \theta_1)$$

repeat until convergence {

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

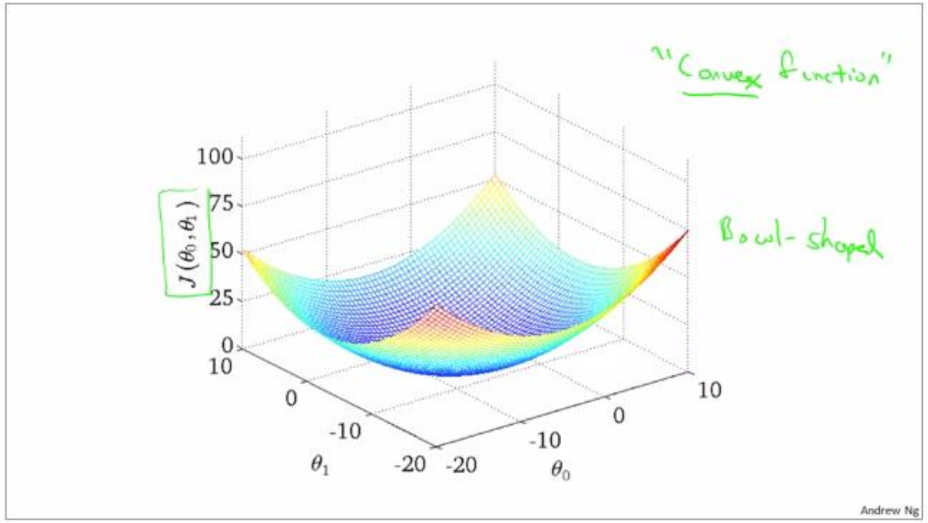
$$\theta_1 := \theta_1 - \alpha \boxed{\frac{1}{m}\sum_{i=1}^{m}\left(h_\theta(x^{(i)}) - y^{(i)}\right)\cdot x^{(i)}}$$

}

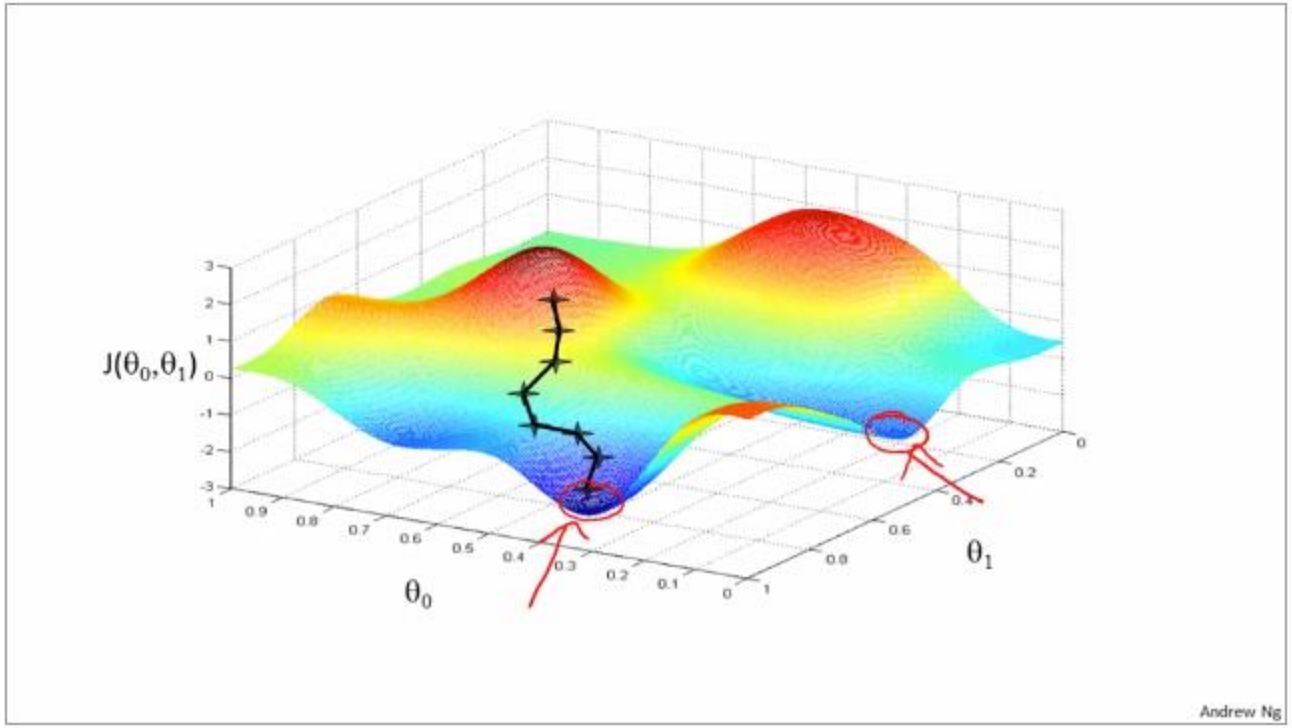
$$\frac{\partial}{\partial\theta_1} J(\theta_0, \theta_1)$$

update θ_0 and θ_1 simultaneously

Andrew Ng

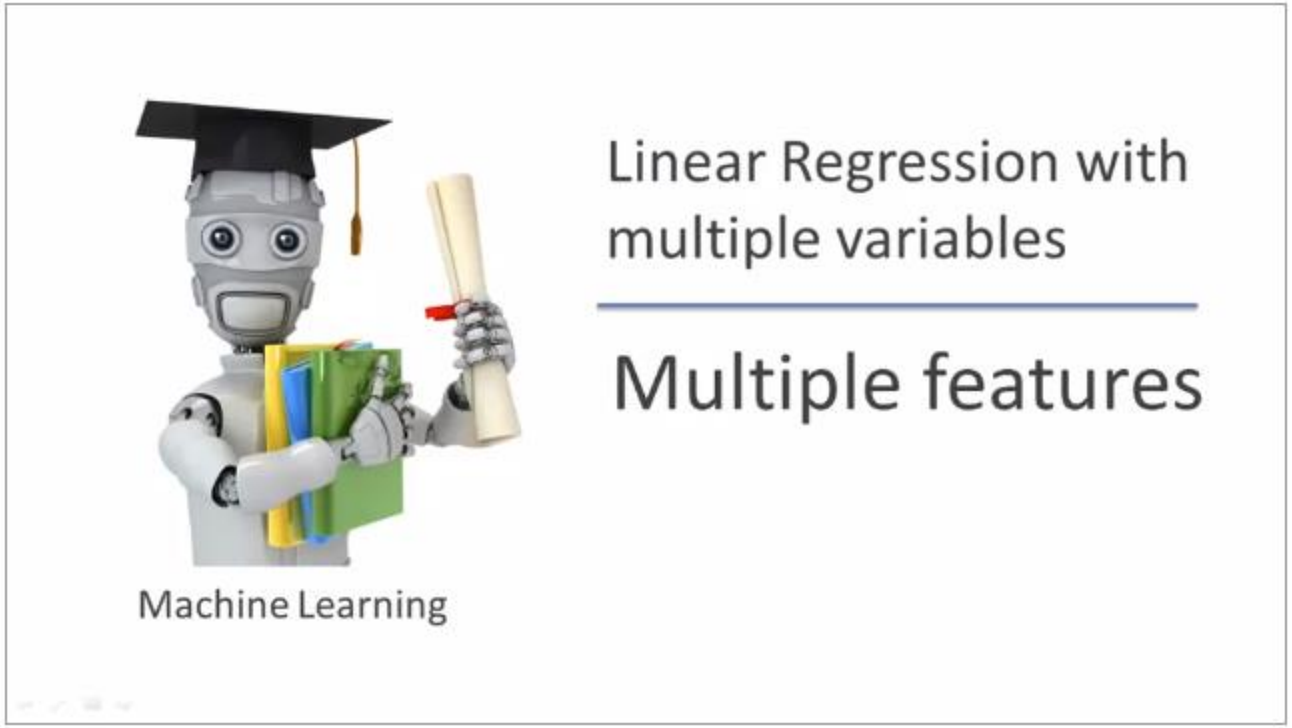


Same local and global minima



Multiple local minima

Multiple Features



Multiple Features

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
...	...	...	...	...

$m = 47$

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

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.

Andrew Ng

Multiple Features

Hypothesis:

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

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

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

Andrew Ng

Multiple Features

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

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

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

$$\underbrace{[\theta_0 \ \theta_1 \cdots \theta_n]}_{\substack{\theta^T \\ (n+1)\times 1 \\ \text{matrix}}} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}}_{x}$$

$$\theta^T x$$

$$h_\theta(x) = \theta_0 \underbrace{x_0}_{=1} + \theta_1 x_1 + \cdots + \theta_n x_n$$

$$= \boxed{\theta^T x.}$$

Multivariate linear regression.

Andrew Ng

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: $\cancel{\theta_0, \theta_1, \ldots, \theta_n}$ θ n+1 - dimensional vector

Cost function:

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

$J(\theta)$

Gradient descent:

Repeat {

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

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

Andrew Ng

Gradient Descent for Multiple Variables

Gradient Descent

Previously (n=1):

Repeat {

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

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

$x_1^{(i)}$

(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 $j = 0, \ldots, n$)

}

$x_0^{(i)} = 1$

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

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

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

$\ldots$

Andrew Ng

Gradient Descent in Practice I - Feature Scaling

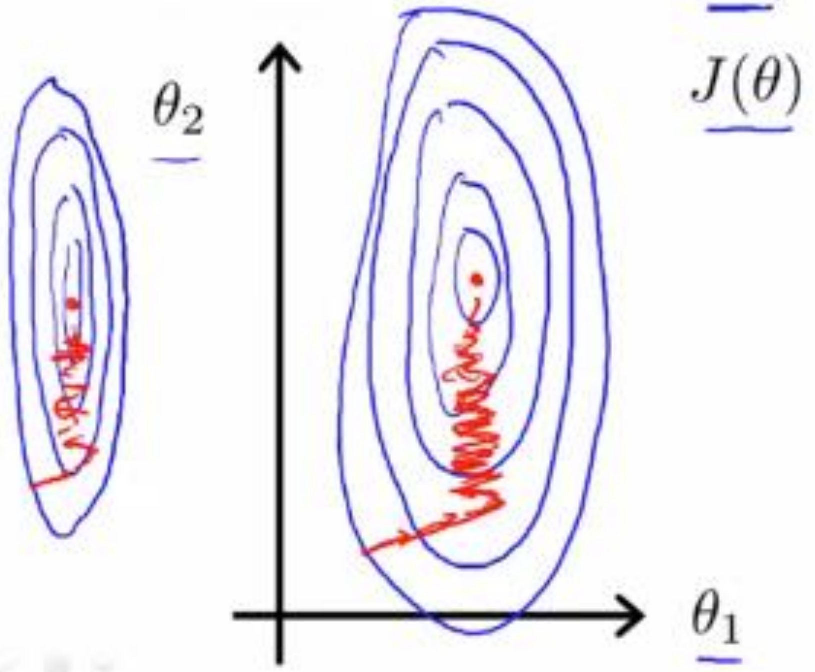
Make gradient Descent run much faster and take fewer steps

Feature Scaling

Idea: Make sure features are on a similar scale.

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

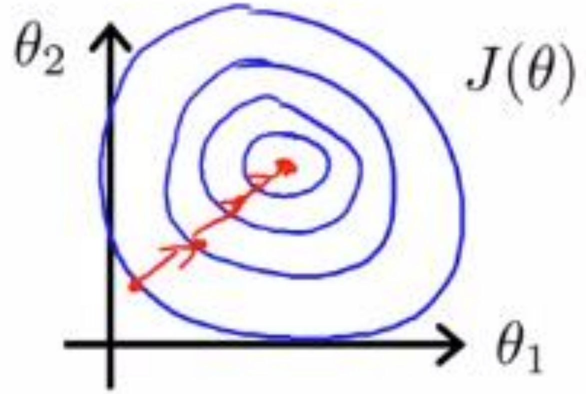
x_2 = number of bedrooms (1-5) $\leftarrow$



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

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

$0 \le x_1 \le 1 \qquad 0 \le x_2 \le 1$



Andrew Ng

Gradient Descent in Practice I - Feature Scaling

Feature Scaling

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

$x_0 = 1$

$0 \le x_1 \le 3$ ✓

$-2 \le x_2 \le 0.5$ ✓

$-100 \le x_3 \ \boxed{100}$ ✗

$-0.0001 \le x_4 \le \boxed{0.0001}$ ✗

Andrew Ng

Gradient Descent in Practice I - Feature Scaling

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. $\boxed{x_1 = \frac{size - 1000}{2000}}$ Average size = 100

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

$-0.5 \le x_1 \le 0.5, \; -0.5 \le x_2 \le 0.5$

$x_1 \leftarrow \frac{x_1 - \mu_1}{s_1}$ avg value of x_1 in training set

range (max - min) (or standard deviation)

$x_2 \leftarrow \frac{x_2 - \mu_1}{s_2}$

Andrew Ng

Gradient Descent in Practice II - Learning Rate

Gradient descent

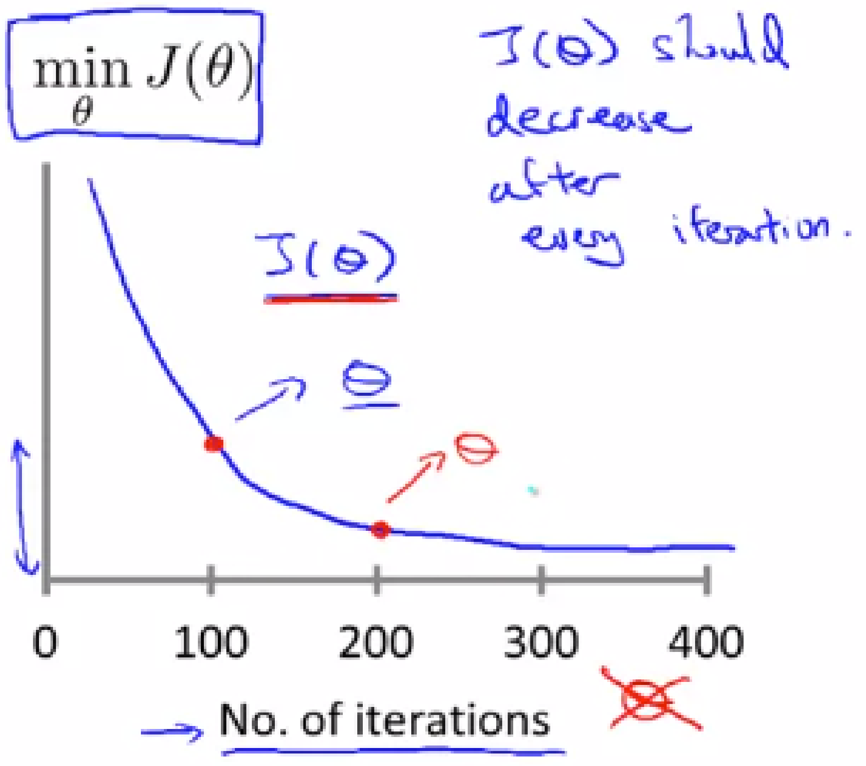
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

- "Debugging": How to make sure gradient descent is working correctly.
- How to choose learning rate $\boxed{\alpha}$.

Andrew Ng

Gradient Descent in Practice II - Learning Rate

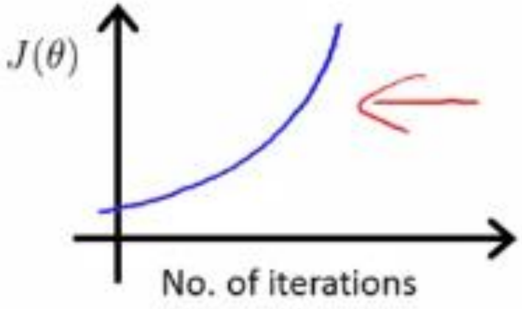
Making sure gradient descent is working correctly.



Andrew Ng

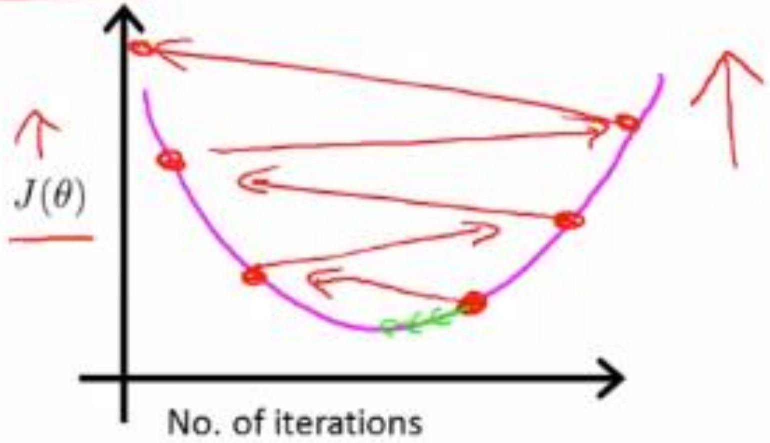
Gradient Descent in Practice II - Learning Rate

Making sure gradient descent is working correctly.



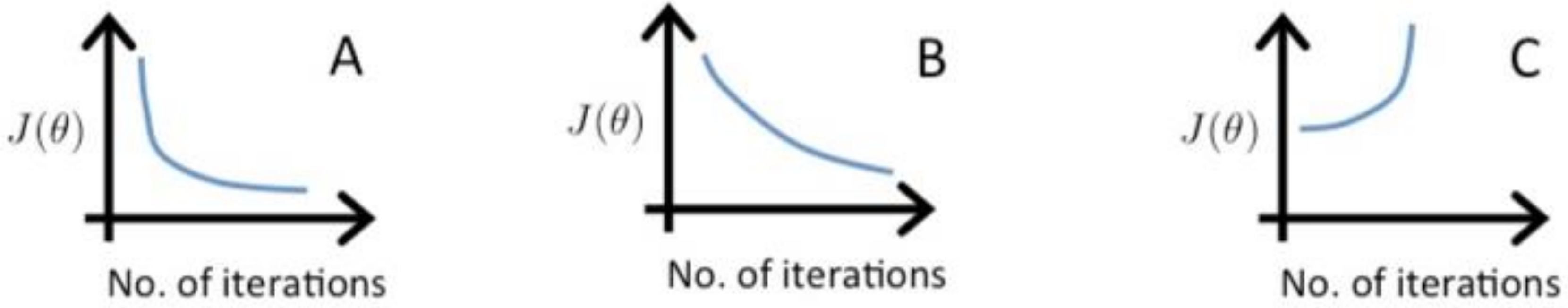
Gradient descent not working.

$\boxed{\text{Use smaller } \alpha.}$



Andrew Ng

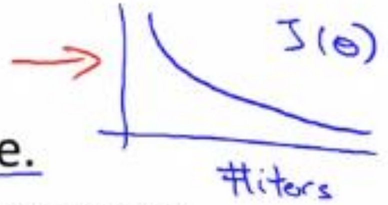
Suppose a friend ran gradient descent three times, with $\alpha = 0.01$, $\alpha = 0.1$, and $\alpha = 1$, and got the following three plots (labeled A, B, and C):



Which plots corresponds to which values of α?

◉ A is $\alpha = 0.1$, B is $\alpha = 0.01$, C is $\alpha = 1$.

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

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

Housing prices prediction

$$h_\theta(x) = \theta_0 + \theta_1 \times \underbrace{frontage}_{\underline{x_1}} + \theta_2 \times \underbrace{depth}_{\underline{x_2}}$$

<u>Area</u>

$x = \underline{frontage * depth}$

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

land area

Andrew Ng

m training examples, n features.

Gradient Descent	Normal Equation
• Need to choose α.	• No need to choose α.
• Needs many iterations.	• Don't need to iterate.
• Works well even when n is large.	• Need to compute $\boxed{(\boxed{X^T X})^{-1}}$ $n \times n$ $O(n^3)$
$n = 10^6$	• Slow if n is very large.
	$n = 100$
	$n = 1000$
	$n = 10000$

Andrew Ng

Logistic Regression

Logistic Regression

Classification

Classification

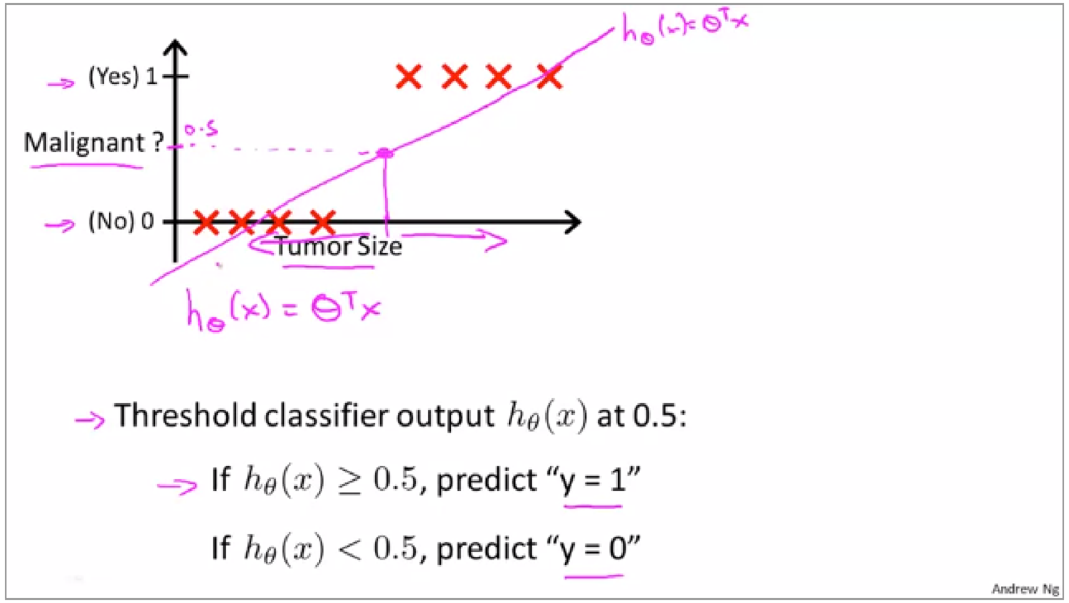
- Email: Spam / Not Spam?
- Online Transactions: Fraudulent (Yes / No)?
- Tumor: Malignant / Benign ?

$y \in \{0, 1\}$

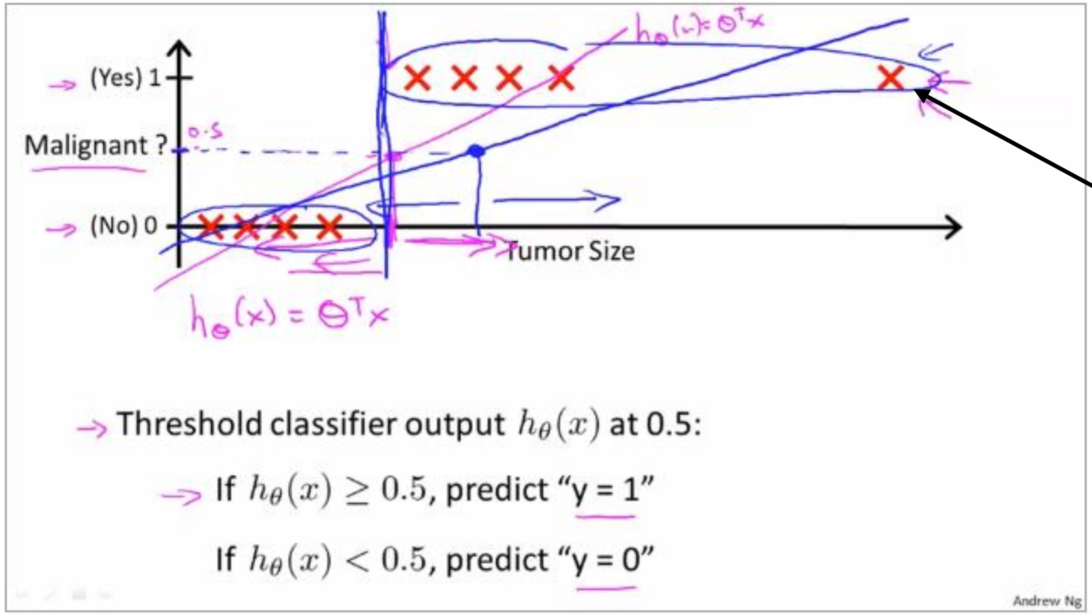
0: "Negative Class" (e.g., benign tumor)

1: "Positive Class" (e.g., malignant tumor)

Classification



Linear regression is doing good in this case



Linear regression is doing bad after adding one data in this case

Hypothesis Representation

Logistic Regression Model

Want $\boxed{0 \le h_\theta(x) \le 1}$

$$h_\theta(x) = g(\theta^T x)$$

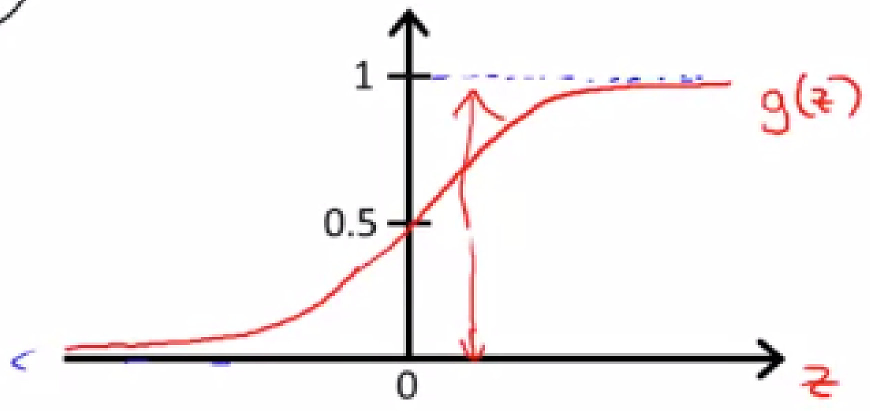
$$g(z) = \frac{1}{1+e^{-z}}$$

$\theta^T x$

$$\boxed{h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}}$$

Sigmoid function

Logistic function



Parameters θ.

Andrew Ng

Hypothesis Representation

Interpretation of Hypothesis Output

$h_\theta(x)$

$h_\theta(x)$ = estimated probability that $\boxed{y = 1}$ on input x

Example: If $x = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumorSize} \end{bmatrix}$

$h_\theta(x) = 0.7$ $y=1$

Tell patient that 70% chance of tumor being malignant

$h_\theta(x) = P(y=1|x;\theta)$

$y = 0$ or 1

"probability that y = 1, given x, parameterized by θ"

$\rightarrow P(y=0|x;\theta) + P(y=1|x;\theta) = 1$

$P(y=0|x;\theta) = 1 - P(y=1|x;\theta)$

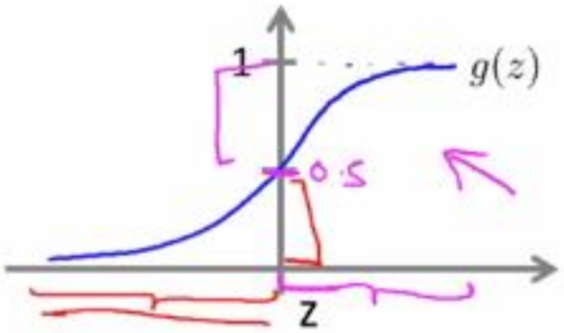
Andrew Ng

Decision Boundary

Logistic regression

$\rightarrow h_\theta(x) = g(\theta^T x) = P(y=1|x;\theta)$

$\rightarrow g(z) = \frac{1}{1+e^{-z}}$



Suppose predict "$y = 1$" if $h_\theta(x) \geq 0.5$

$\theta^T x \geq 0$

$g(z) \geq 0.5$

when $z \geq 0$

$h_\theta(x) = g(\theta^T x) \geq 0.5$

whenever $\theta^T x \geq 0$

z

predict "$y = 0$" if $h_\theta(x) < 0.5$

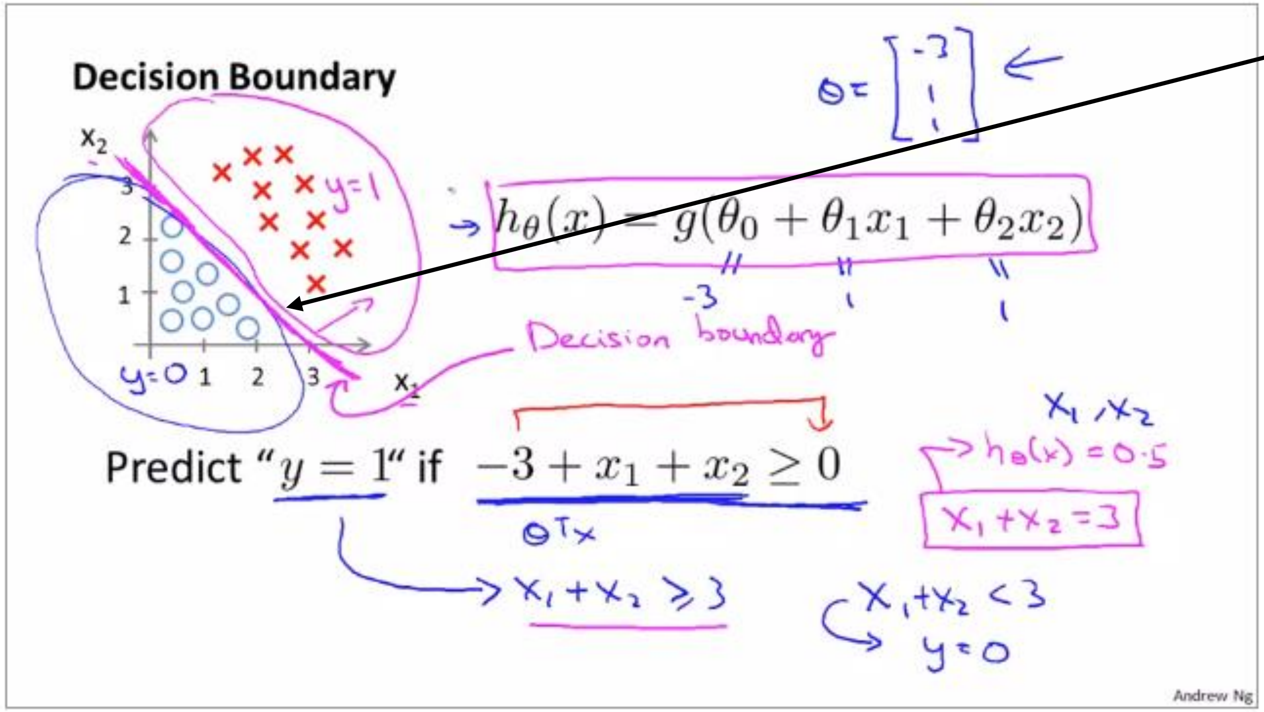
$h_\theta(x) = g(\theta^T x)$

$\theta^T x < 0$

$g(z) < 0.5$

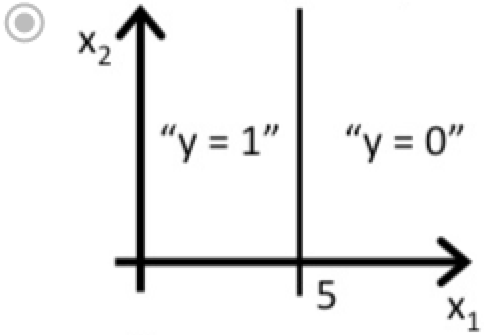
Andrew Ng

Decision Boundary



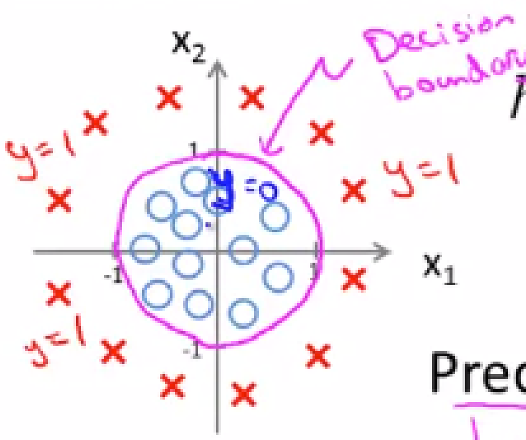
Decision boundary is property of hypothesis and Parameters but not of Data.

Consider logistic regression with two features x_1 and x_2. Suppose $\theta_0 = 5, \theta_1 = -1, \theta_2 = 0$, so that $h_\theta(x) = g(5 - x_1)$. Which of these shows the decision boundary of $h_\theta(x)$?



Decision Boundary

Non-linear decision boundaries



$$h_\theta(x) = g(\overset{=-1}{\theta_0} + \overset{=0}{\theta_1} x_1 + \overset{=0}{\theta_2} x_2 + \underset{=1}{\theta_3} x_1^2 + \underset{=1}{\theta_4} x_2^2)$$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

$\boxed{x_1^2 + x_2^2 = 1}$

$x_1^2 + x_2^2 \geq 1$

Cost Function

Training set: $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \cdots, (x^{(m)}, y^{(m)})\}$

m examples $\quad x \in \begin{bmatrix} x_0 \\ x_1 \\ \cdots \\ x_n \end{bmatrix} \mathbb{R}^{n+1} \quad x_0 = 1, y \in \{0, 1\}$

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}$$

How to choose parameters θ ?

Cost Function

Cost function

→ Linear regression: $J(\theta) = \boxed{\frac{1}{m}\sum_{i=1}^{m} \cancel{\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2} \quad \text{cost}(h_\theta(x^{(i)}), y)}$

→ $\text{Cost}(h_\theta(x^{(i)}), y^{(i)}) = \frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

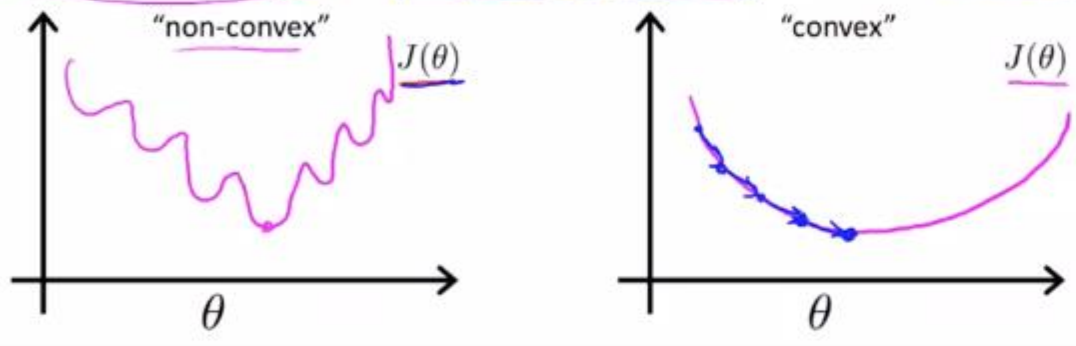
Cost Function

Cost function

→ ~~Linear~~ regression: $J(\theta) = \boxed{\frac{1}{m}\sum_{i=1}^{m} \cancel{\frac{1}{2}\left(h_\theta(x^{(i)}) - y^{(i)}\right)^2} \quad \text{cost}(h_\theta(x^{(i)}), y)}$

logistic

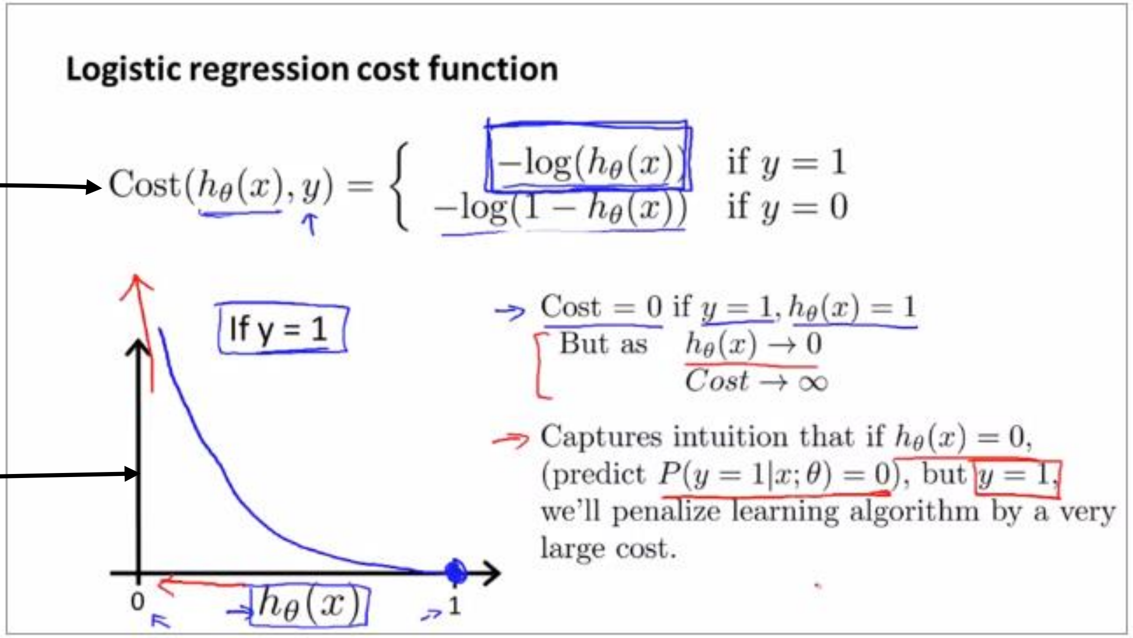
→ $\text{Cost}(h_\theta(x), y) = \boxed{\frac{1}{2}\left(h_\theta(x) - y\right)^2}$ ← $\boxed{\frac{1}{1+e^{-\theta^T x}}}$



Andrew Ng

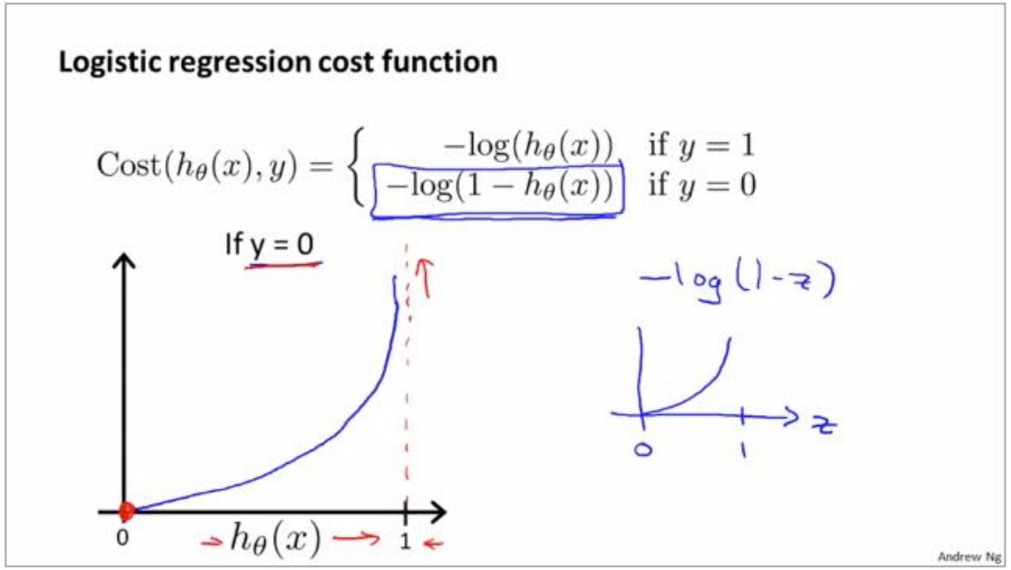
Due to sigmoid, this cost function will be non convex And that's why we do not use squared error loss in logistic regression

Cost Function

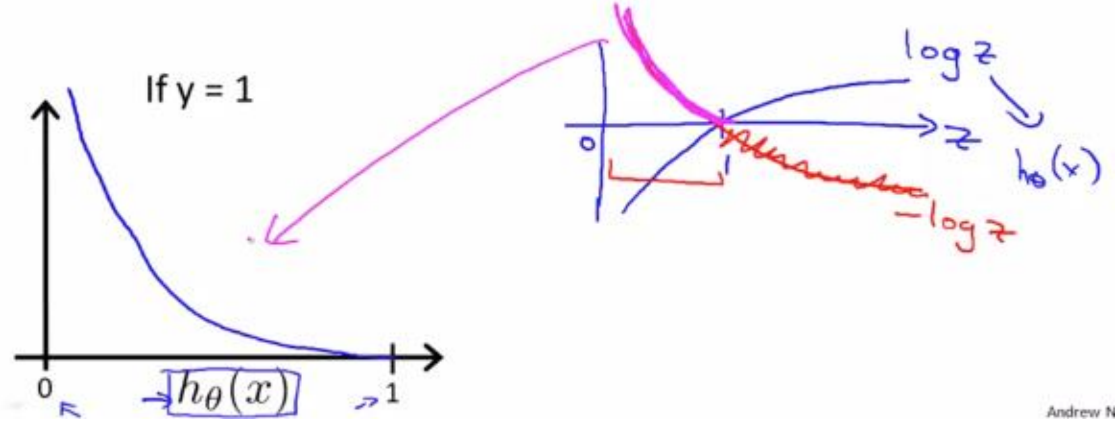


Cost

Cost Function



With proper choice of cost function we can have convex graph of it and it local optimum free



In logistic regression, the cost function for our hypothesis outputting (predicting) $h_\theta(x)$ on a training example that has label $y \in \{0, 1\}$ is:

$$\text{cost}(h_\theta(x), y) = \begin{cases} -\log h_\theta(x) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Which of the following are true? Check all that apply.

- [x] If $h_\theta(x) = y$, then $\text{cost}(h_\theta(x), y) = 0$ (for $y = 0$ and $y = 1$).
- [x] If $y = 0$, then $\text{cost}(h_\theta(x), y) \to \infty$ as $h_\theta(x) \to 1$.
- [ ] If $y = 0$, then $\text{cost}(h_\theta(x), y) \to \infty$ as $h_\theta(x) \to 0$.
- [x] Regardless of whether $y = 0$ or $y = 1$, if $h_\theta(x) = 0.5$, then $\text{cost}(h_\theta(x), y) > 0$.

Logistic regression cost function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\mathrm{Cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$$

Note: $y = 0$ or 1 always

$$\mathrm{Cost}(h_\theta(x), y) = -y\log(h_\theta(x)) - (1-y)\log(1-h_\theta(x))$$

Logistic regression cost function

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$= -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log\left(1-h_\theta(x^{(i)})\right)\right]$$

To fit parameters θ :

$$\boxed{\min_\theta J(\theta) \quad \text{Get } \theta}$$

To make a prediction given new x:

Output $h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

$p(y=1 \mid x;\theta)$

Gradient Descent

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

Want $\min_\theta J(\theta)$:

Repeat $\{$

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

(simultaneously update all θ_j)

$\}$

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

Gradient Descent

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

Want $\min_\theta J(\theta)$:

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$$

Repeat $\{$

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

(simultaneously update all θ_j)

$\}$

$h_\theta(x) = \theta^T x$

$h_\theta(x) = \frac{1}{1+e^{-\theta^T x}}$

Algorithm looks identical to linear regression!

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \boldsymbol{x}}}$$

$$\sigma(\mathrm{x}) = \frac{1}{1 + e^{-\mathrm{x}}}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{0 * (1 + e^{-\mathrm{x}}) - (1) * \big(e^{-\mathrm{x}} * (-1)\big)}{(1 + e^{-\mathrm{x}})^2}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{(e^{-\mathrm{x}})}{(1 + e^{-\mathrm{x}})^2} = \frac{1\text{-}1 + (e^{-\mathrm{x}})}{(1 + e^{-\mathrm{x}})^2} = \frac{1 + e^{-\mathrm{x}}}{(1 + e^{-\mathrm{x}})^2} - \frac{1}{(1 + e^{-\mathrm{x}})^2}$$

$$\frac{d\big(\sigma(\mathrm{x})\big)}{dx} = \frac{1}{1 + e^{-\mathrm{x}}} * \left(1 - \frac{1}{1 + e^{-\mathrm{x}}}\right) = \sigma(\mathrm{x})\big(1 - \sigma(\mathrm{x})\big)$$

Gradient Descent

$$J(\theta) = -\frac{1}{m}[\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1-y^{(i)}) \log (1-h_\theta(x^{(i)}))]$$

Step 1:

Applying Chain rule and writing in terms of partial derivatives.

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * \sum_{i=1}^{m} \left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \frac{\partial\left(h_\theta(x^{(i)})\right)}{\partial(\theta j)} \right]$$

$$+ \sum_{i=1}^{m} \left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * \frac{\partial\left(1 - h_\theta(x^{(i)})\right)}{\partial(\theta j)} \right]$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)} \right]$$

$$+ \sum_{i=1}^{m} \left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * (-\sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)} \right])$$

Step 2:

Evaluating the partial derivative using the pattern of the derivative of the sigmoid function.

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} * \frac{1}{h_\theta(x^{(i)})} * \sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)} \right]$$

$$+ \sum_{i=1}^{m} \left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * (-\sigma(z)\left(1 - \sigma(z)\right) * \frac{\partial(\theta^T x)}{\partial(\theta j)} \right])$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} \frac{1}{h_\theta(x^{(i)})} h_\theta(x^{(i)}) \left(1 - h_\theta(x^{(i)})\right) * x_j^i \right] +$$

$$\sum_{i=1}^{m} \left[\left(1 - y^{(i)}\right) * \frac{1}{\left(1 - h_\theta(x^{(i)})\right)} * \left(-h_\theta(x^{(i)})\left(1 - h_\theta(x^{(i)})\right) * x_j^i \right] \right)$$

Step 3:

Simplifying the terms by multiplication

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} \left[y^{(i)} * \left(1 - h_\theta\left(x^{(i)}\right)\right) * x_j^i - \left(1 - y^{(i)}\right) * h_\theta\left(x^{(i)}\right) * * x_j^i \right]$$

$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} [\ y^{(i)} \text{-} y^{(i)} * h_\theta\left(x^{(i)}\right) \text{-} h_\theta\left(x^{(i)}\right) + y^{(i)} * h_\theta\left(x^{(i)}\right)] * x_j^i\ )$$

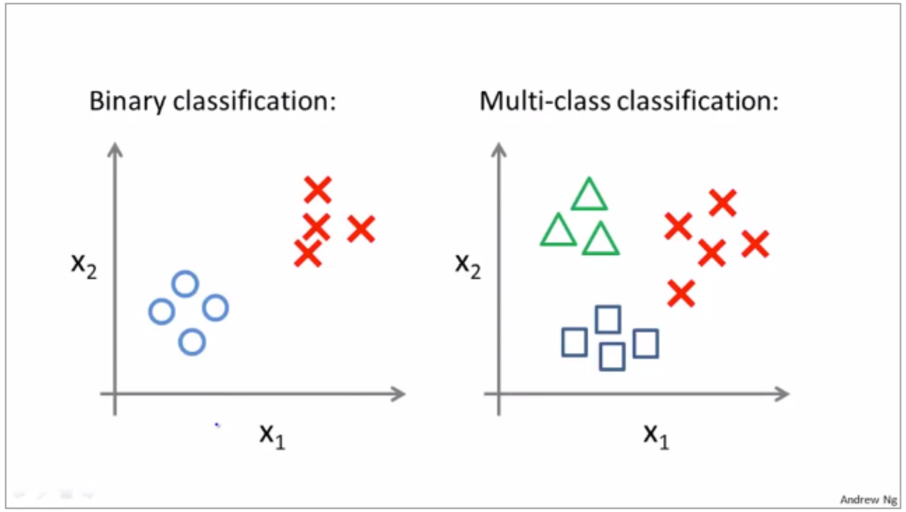
$$\frac{\partial(J(\theta))}{\partial(\theta j)} = -\frac{1}{m} * (\sum_{i=1}^{m} [\ y^{(i)} \text{-} h_\theta\left(x^{(i)}\right)] * x_j^i\ )$$

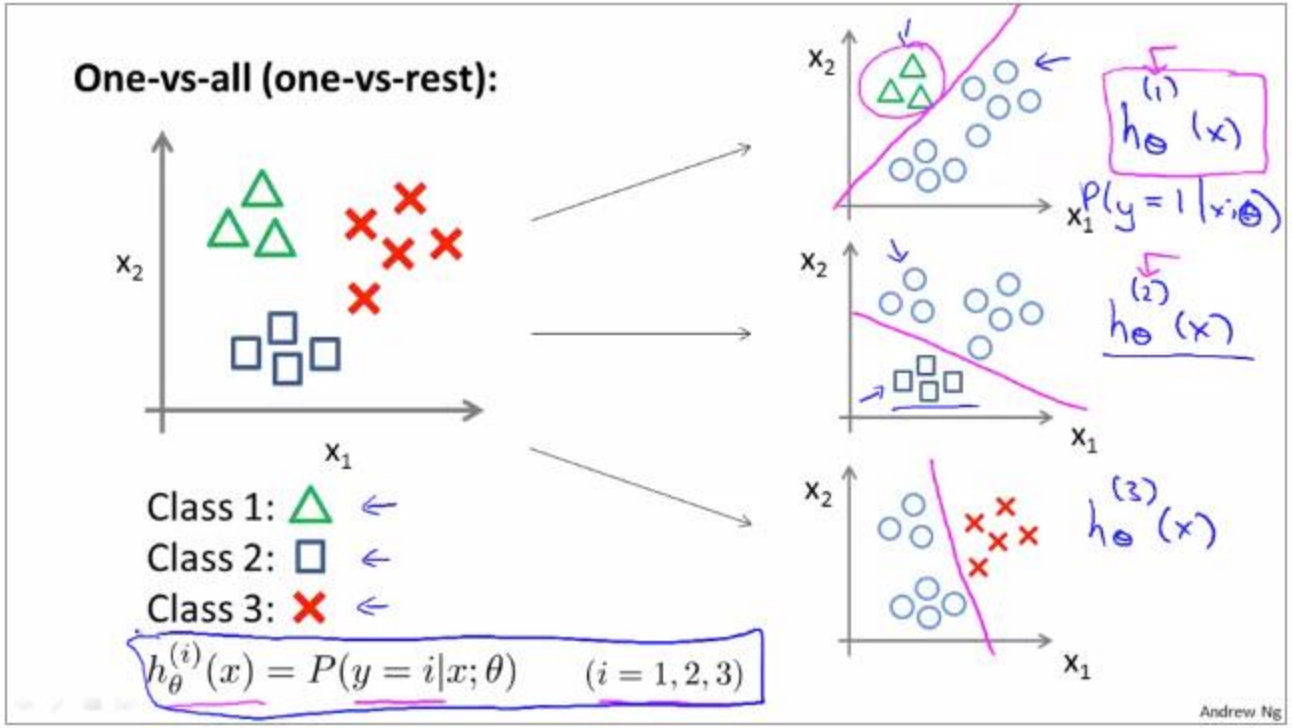
Step 4:

Removing the summation term by converting it into a matrix form for the gradient with respect to all the weights including the bias term.

$$\frac{\partial(J(\theta))}{\partial(\theta)} = \frac{1}{m} X^T [\ h_\theta(x) - y]$$

Multiclass Classification: One-vs-all





Suppose you have a multi-class classification problem with k classes (so $y \in \{1, 2, \ldots, k\}$). Using the 1-vs.-all method, how many different logistic regression classifiers will you end up training?

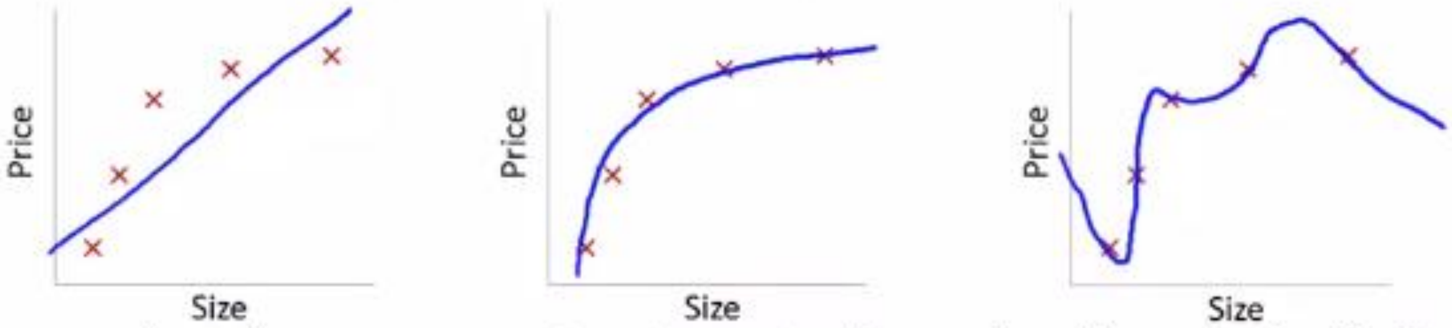
- (selected) k

Regularization

Technique to deal with overfitting

The problem of overfitting

Example: Linear regression (housing prices)



$\rightarrow \theta_0 + \theta_1 x$ "Underfit" "High bias"

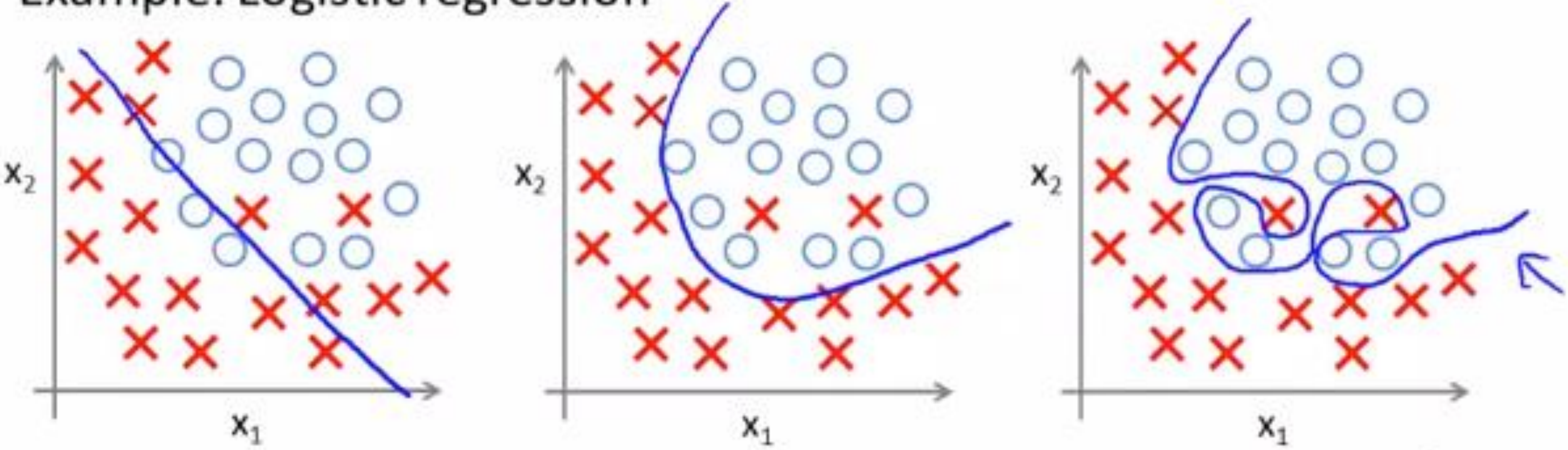
$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2$ "Just right"

$\rightarrow \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$ "Overfit" "High variance"

Overfitting: If we have too many features, the learned hypothesis may fit the training set very well ($J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \approx 0$), but fail to generalize to new examples (predict prices on new examples).

Andrew Ng

Example: Logistic regression



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

(g = sigmoid function)

"Underfit"

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2$
$+\theta_3 x_1^2 + \theta_4 x_2^2$
$+\theta_5 x_1 x_2)$

$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2$
$+\theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2$
$+\theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$

"Over-fit"

Andrew Ng

Consider the medical diagnosis problem of classifying tumors as malignant or benign. If a hypothesis $h_\theta(x)$ has overfit the training set, it means that:

- It makes accurate predictions for examples in the training set and generalizes well to make accurate predictions on new, previously unseen examples.
- It does not make accurate predictions for examples in the training set, but it does generalize well to make accurate predictions on new, previously unseen examples.
- It makes accurate predictions for examples in the training set, but it does not generalize well to make accurate predictions on new, previously unseen examples.
- It does not make accurate predictions for examples in the training set and does not generalize well to make accurate predictions on new, previously unseen examples.

Addressing Overfitting

1. Collect more Data
2. Select Features
3. Reduce size of parameter (Regularization)

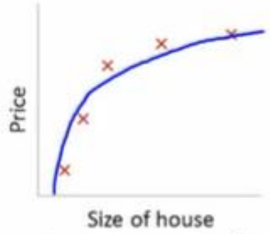
Regularization.

— Keep all the features, but reduce magnitude/values of parameters θ_j.

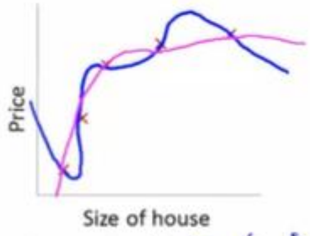
— Works well when we have a lot of features, each of which contributes a bit to predicting y.

Cost Function

Intuition



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



$\theta_0 + \theta_1 x + \theta_2 x^2 + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Suppose we penalize and make θ_3, θ_4 really small.

$$\rightarrow \min_\theta \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + 1000\,\theta_3^2 + 1000\,\theta_4^2$$

$\theta_3 \approx 0$ $\theta_4 \approx 0$

Andrew Ng

Regularization.

Small values for parameters $\boxed{\theta_0, \theta_1, \ldots, \theta_n}$
- "Simpler" hypothesis
- Less prone to overfitting

$\boxed{\theta_3, \theta_4} \approx 0$

Housing:
- Features: $x_1, x_2, \ldots, x_{100}$
- Parameters: $\theta_0, \theta_1, \theta_2, \ldots, \theta_{100}$

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

$\theta_1, \theta_2, \theta_3, \ldots, \theta_{100}$ $\cancel{\theta_0}$

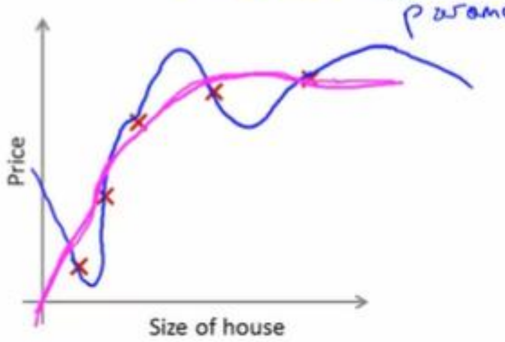
Andrew Ng

Regularization.

$$\rightarrow J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2\right]$$

regularization parameter

$$\min_\theta J(\theta)$$



In regularized linear regression, we choose θ to minimize:

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2\right]$$

What if λ is set to an extremely large value (perhaps too large for our problem, say $\lambda = 10^{10}$)?

- ◯ Algorithm works fine; setting λ to be very large can't hurt it.
- ◯ Algorithm fails to eliminate overfitting.
- ◉ Algorithm results in underfitting (fails to fit even the training set).
- ◯ Gradient descent will fail to converge.

In regularized linear regression, we choose θ to minimize

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \underline{\lambda}\sum_{j=1}^{n}\theta_j^2\right]$$

What if λ is set to an extremely large value (perhaps for too large for our problem, say $\lambda = 10^{10}$)?

$\underline{\theta_1}, \underline{\theta_2}, \underline{\theta_3}, \underline{\theta_4}$

$\theta_1 \approx 0, \theta_2 \approx 0$

$\theta_3 \approx 0, \theta_4 \approx 0$

$\boxed{h_\theta(x) = \theta_0}$

$h_\theta(x)$

$\theta_0 + \cancel{\theta_1 x} + \cancel{\theta_2 x^2} + \cancel{\theta_3 x^3} + \cancel{\theta_4 x^4}$

Andrew Ng

Regularized Linear Regression

$$J(\theta) = \frac{1}{2m}\left[\underbrace{\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2}_{} + \underbrace{\lambda \sum_{j=1}^{n}\theta_j^2}_{}\right]$$

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

Gradient descent θ_0 $\theta_1, \theta_2, \dots, \theta_n$

Repeat {

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

$$\theta_j := \theta_j - \boxed{\alpha}\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$$(j = \cancel{0}, 1, 2, 3, \dots, n)$$

}

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

$$1 - \alpha\frac{\lambda}{m} < 1 \qquad 0.99 \qquad \theta_j \times 0.99 \qquad \boxed{\theta_j^2}$$

Gradient descent θ_0 $\theta_1, \theta_2, \dots, \theta_n$

Repeat {

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

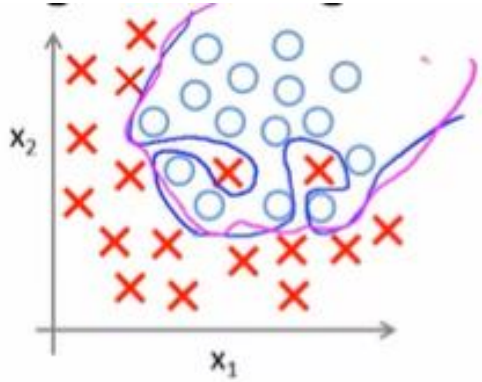
$$\theta_j := \theta_j - \alpha\left[\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} + \frac{\lambda}{m}\theta_j\right]$$

$$(j = \cancel{0}, 1, 2, 3, \dots, n)$$

}

We don’t penalize theta0

Regularized Logistic Regression



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \ldots)$$

Cost function:

$$J(\theta) = -\left[\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right)\right] + \frac{\lambda}{2m}\sum_{j=1}^{n} \theta_j^2$$

$$\boxed{\theta_1, \theta_2, \ldots, \theta_n}$$

Andrew Ng

Reasons for Underfitting:

1. High bias and low variance
2. The size of the training dataset used is not enough.
3. The model is too simple.
4. Training data is not cleaned and also contains noise in it.

Techniques to reduce underfitting:

1. Increase model complexity
2. Increase the number of features, performing feature engineering
3. Remove noise from the data.
4. Increase the number of epochs or increase the duration of training to get better results.