Lecture 3

Logistic Regression

Dr. Le Huu Ton

Hanoi, 2020

Outline

- Logistic Regression
- Gradient Descent
- Newton’s Method

Outline

- Logistic Regression
- Gradient Descent
- Newton’s Method

Classification

<u>Example:</u>

Given data of prices of houses with size from 25-30 m^2 and their location. Predict if a house is on Thanh Xuan district or Hoan Kiem district base on it price.

Price (b.VND)	*Location*
2.5	*Thanh Xuan*
3.5	*Thanh Xuan*
5.6	*Hoan Kiem*
2.2	*Thanh Xuan*
6.9	*Hoan Kiem*
9.6	*Hoan Kiem*

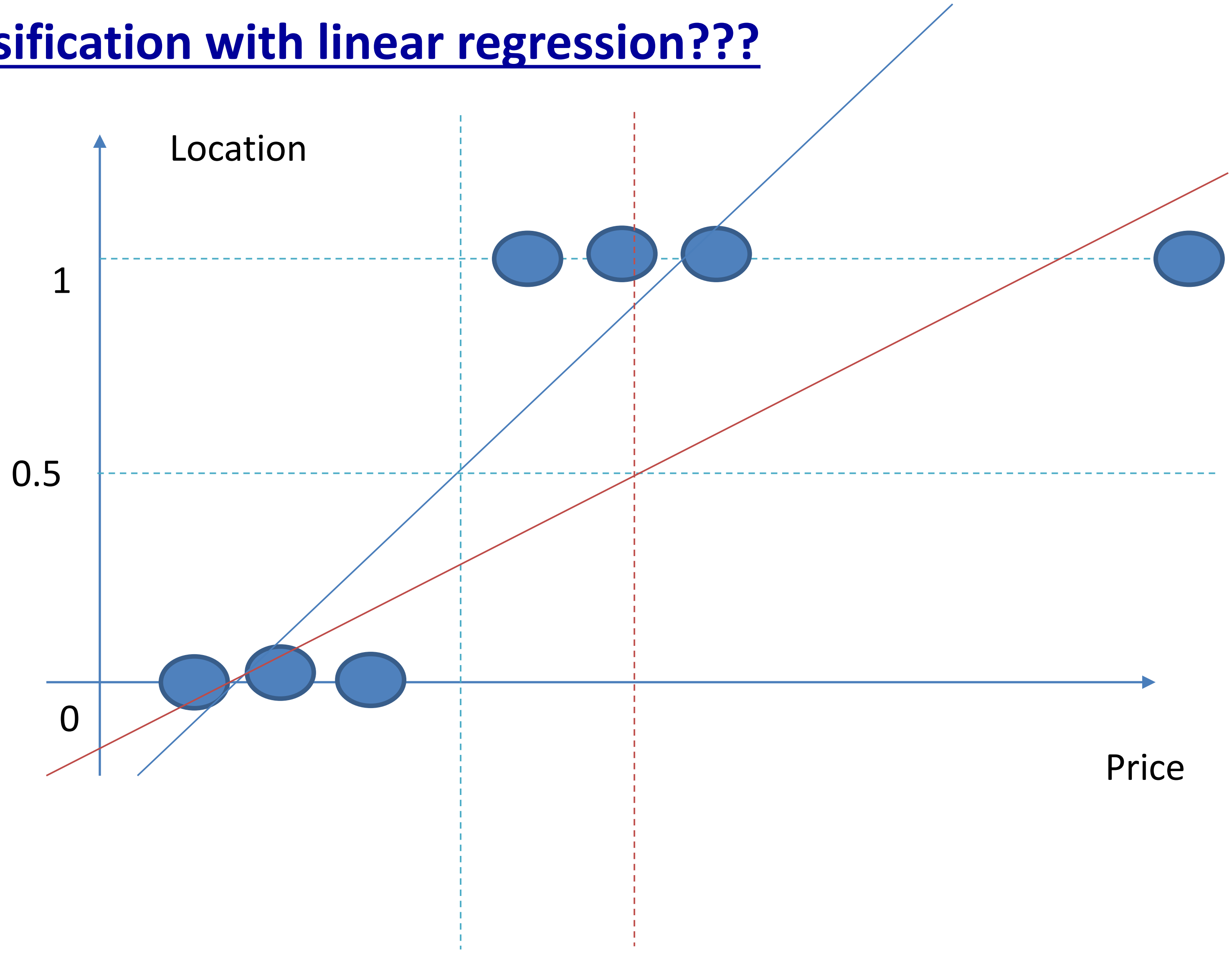
$$y \in \{0,1\}$$

y=0 => Thanh Xuan (negative)

y=1 => Hoan Kiem (positive)

Classification

Classification with linear regression???



Classification

$\Rightarrow$ **Linear regression is not a good choice for classification problem**

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

Need a more suitable hypothesis such as:

$$0 \leq h(x) \leq 1$$

=>

$$h(x) = g(\theta^T x) \quad where \quad g(z) = \frac{1}{1+e^{-z}}$$

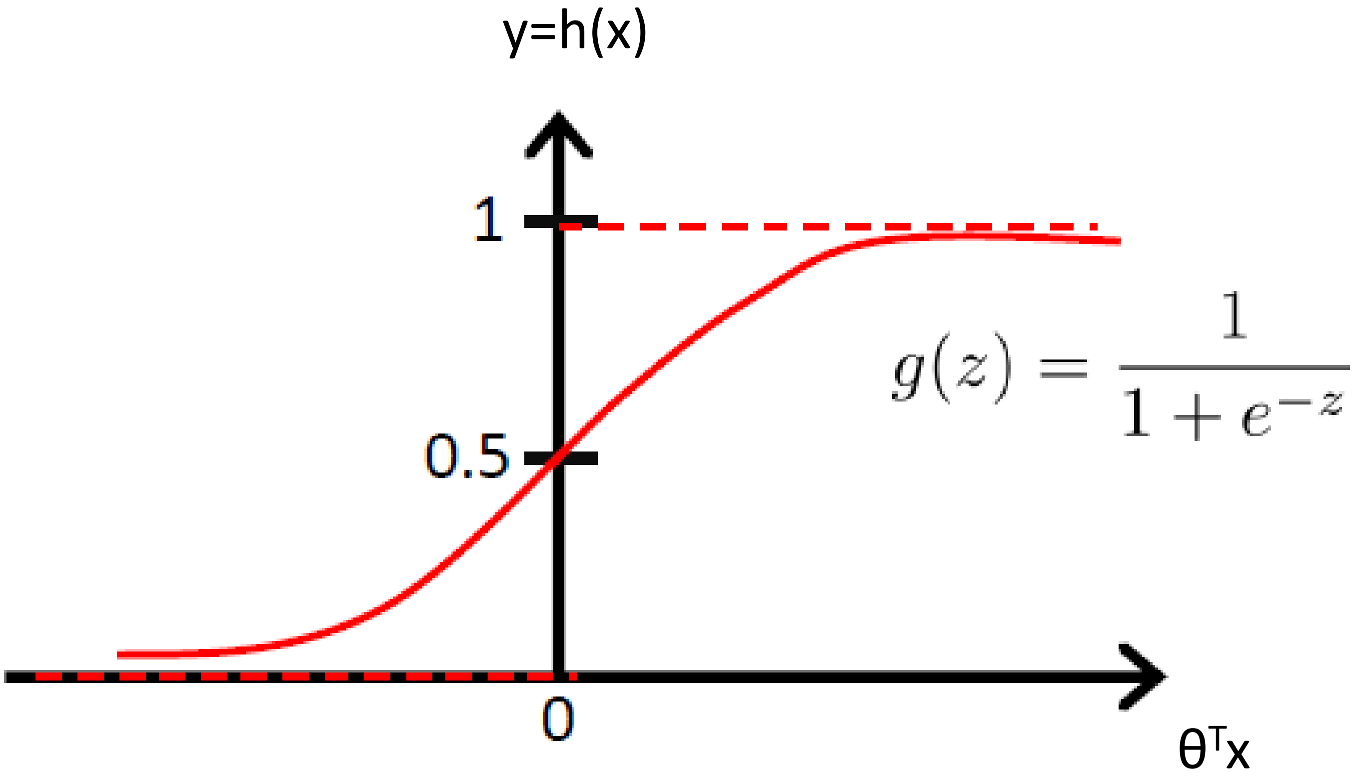
$$h(x) = \frac{1}{1+e^{-\theta^T x}}$$

g(z): sigmoid function or logistic function

Classification

Classification => Logistic Regression

Logistic Function



Classification

Interpretation of Hypothesis output

h(x) can be considered as the probability that output y =1 with a given value of input x

h(x)=0.65 => there is 65% chance that the house is locate at Hoan kiem district

Classification

Example of Image Classification using Caffe

http://demo.caffe.berkeleyvision.org/

Maximally accurate | Maximally specific

structure	0.7642
housing	0.39733
building	0.39136
wheeled vehicle	0.38885
vehicle	0.38175

Example:

Calulate the output value with following coefficient

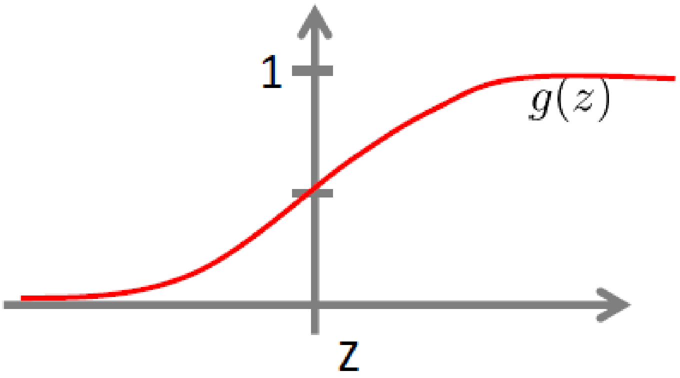
$\Theta_0 = \Theta_1 = 0$;

$\Theta_0 = 0.5 \quad \Theta_1 = 0.7$;

Price (b.VND)	Location
2.5	Thanh Xuan
3.5	Thanh Xuan
5.6	Hoan Kiem
2.2	Thanh Xuan
6.9	Hoan Kiem
9.6	Hoan Kiem

Decision Boundary

Logistic regression



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

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

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

$g(z) \geq 0.5$ When z≥0

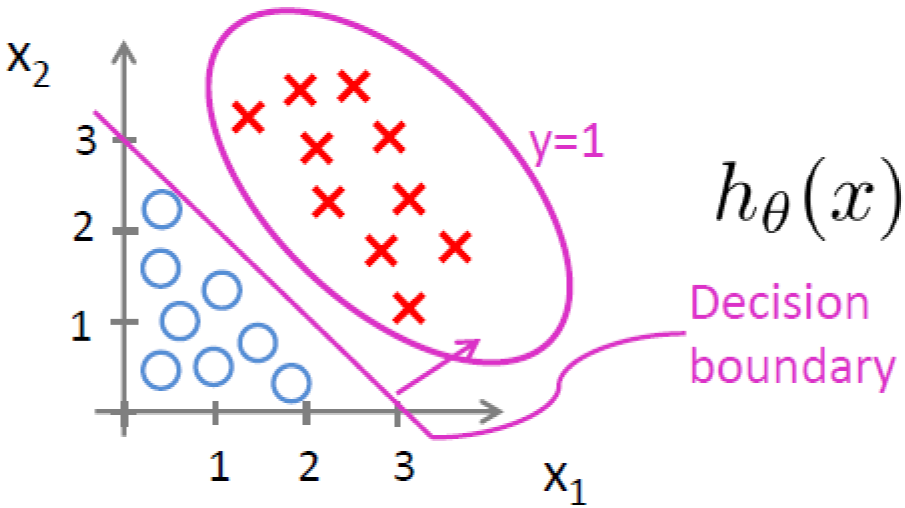
So $h_\theta(x) = g(\theta^T x) \geq 0.5$ When $\theta^T x \geq 0$

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

g(z) ≤ 0.5 When z>0

So $h_\theta(x) = g(\theta^T x) < 0.5$ When $\theta^T x < 0$

Decision Boundary



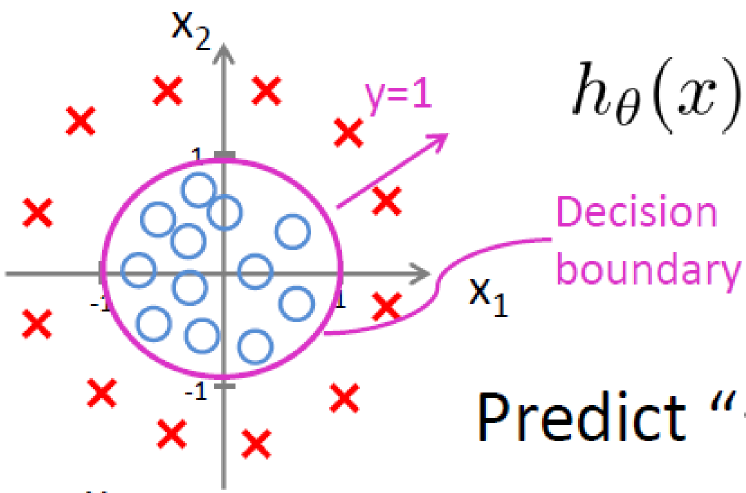
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$\theta_0 = -3$, $\theta_1 = 1$, $\theta_2 = 1$

Predict "$y = 1$" if $-3 + x_1 + x_2 \geq 0$

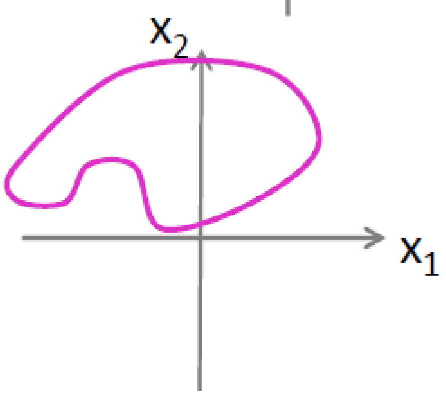
Classification

Decision Boundary



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

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



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

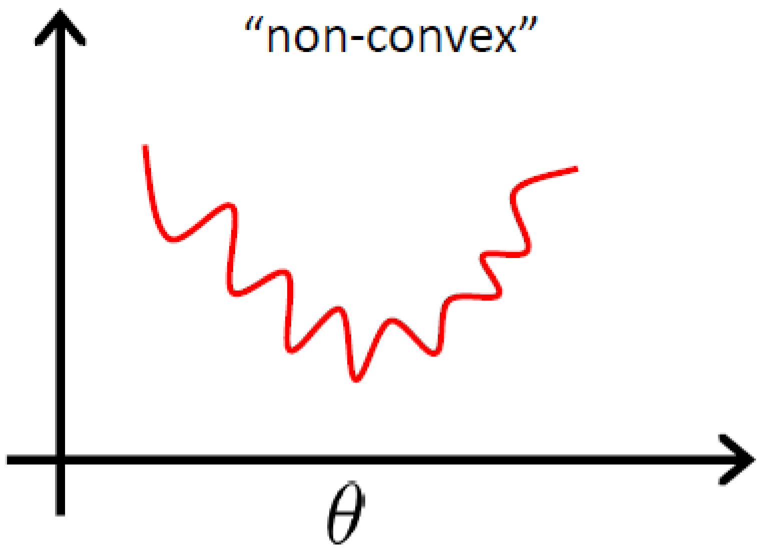
Classification

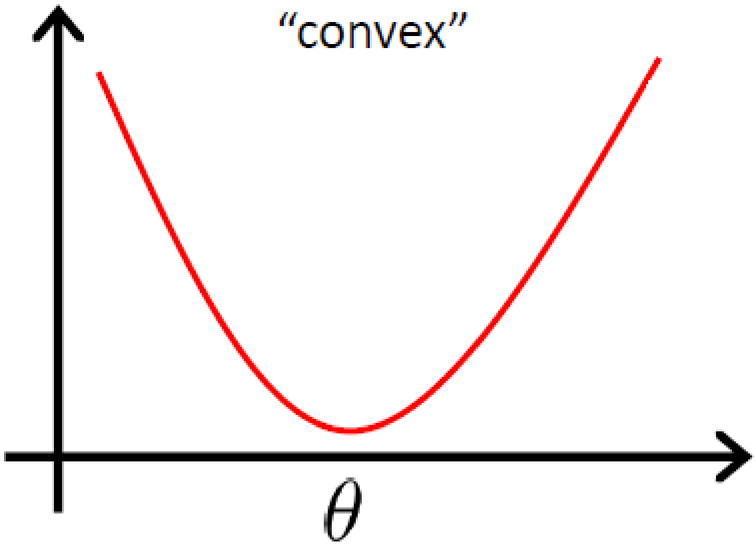
Cost Function

Linear Regression:

$$E(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$$

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

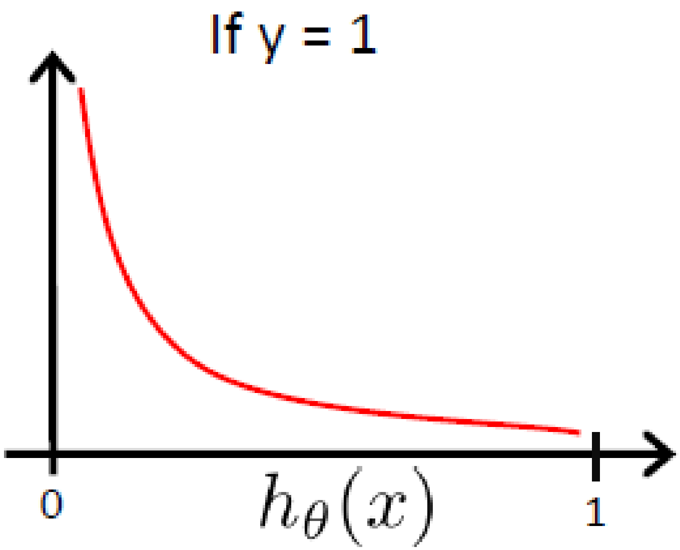


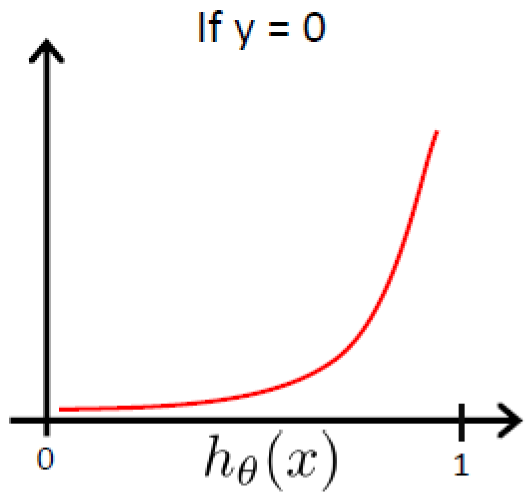


Classification

Logistic Regression Cost function

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





Classification

Logistic Regression Cost function

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

$$E(h_\theta(x), y) = -y \log(h_\theta(x)) - (1 - y)\log(1 - h_\theta(x))$$

Outline

- Logistic Regression
- Gradient Descent
- Newton's Method

Classification

Gradient Descent for logistic regression:

Given the cost function

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

Update θ until convergence:

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

Exercise:

Calculate $\frac{\partial}{\partial \theta_j} E(\theta)$

$$\frac{\partial}{\partial z} g(z) = \frac{\partial}{\partial z} \frac{1}{1+e^{-z}} = \frac{1}{(1+e^{-z})^2}(e^{-z})$$

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

$$\frac{\partial}{\partial_z}\log(z) = \frac{1}{z}$$

$$\frac{\partial}{\partial_z} f(g(z)) = \frac{\partial f(g)}{\partial_g}\frac{\partial_g}{\partial_z}$$

Classification

Solution:

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

Exercise:

Starting with θ_0 and θ_1 equal to 0. α =0.001. Calculate the value of coefficient after first iteration with batch gradient descent

Price	Location	Output Value
2.5	*Thanh Xuan*	0
3.5	*Thanh Xuan*	0
5.6	*Hoan Kiem*	1
2.2	*Thanh Xuan*	0
6.9	*Hoan Kiem*	1
9.6	*Hoan Kiem*	1

Homework

Exercise:

Starting with θ_0 and θ_1 equal to 0. α =0.01, Regularization term lamda =10. Calculate the value of coefficient after first iteration using gradient

Price	Location	Output Value
2.5	*Thanh Xuan*	0
3.5	*Thanh Xuan*	0
5.6	*Hoan Kiem*	1
2.2	*Thanh Xuan*	1
6.9	*Hoan Kiem*	0
9.6	*Hoan Kiem*	1

Outline

- Logistic Regression
- Gradient Descent
- Newton’s Method

Newton's Method

Logistic Regression: Minimize the cost function

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

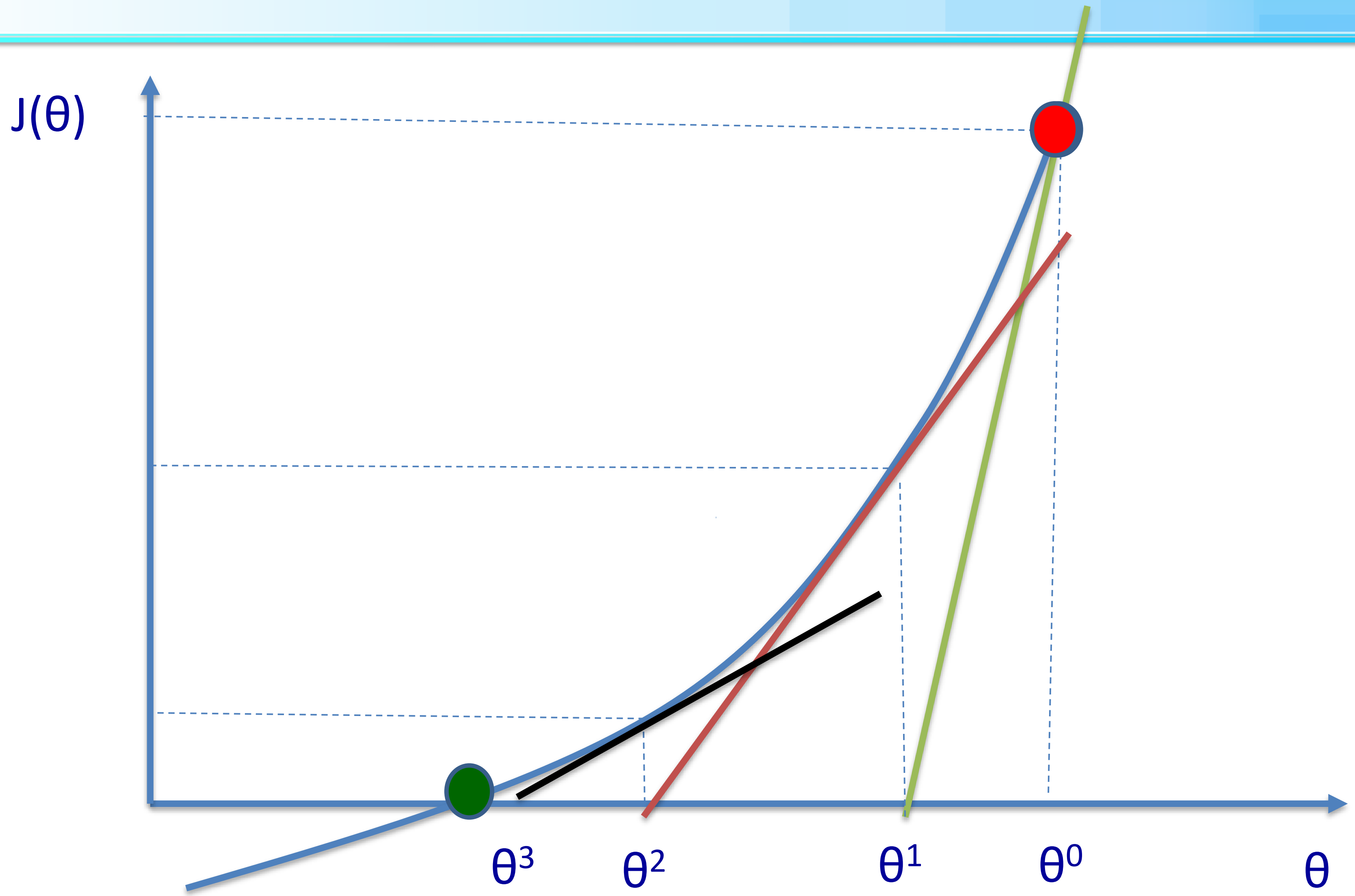
Gradient Descent: step by step modify the coefficients θ such as this modification reduce the cost function

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

Newton's method shares the same idea with normal equation (linear regression): finding the coefficients θ as

$$\frac{\partial}{\partial_\theta}E(\theta) = J(\theta) = 0$$

Newton's Method



Newton's Method

Start with random value of coefficient θ^0 and then step by step update θ, until $E'(\theta)$ reaches 0, or $E(\theta)$ reaches its minimum

While $J(\theta)!=0$

{

- Calculate the tangent line of $J(\theta)$ at θ^t

- **Find the cross point of tangent line with the θ axis, called θ^{t+1}**
- **Update θ^t to θ^{t+1}**

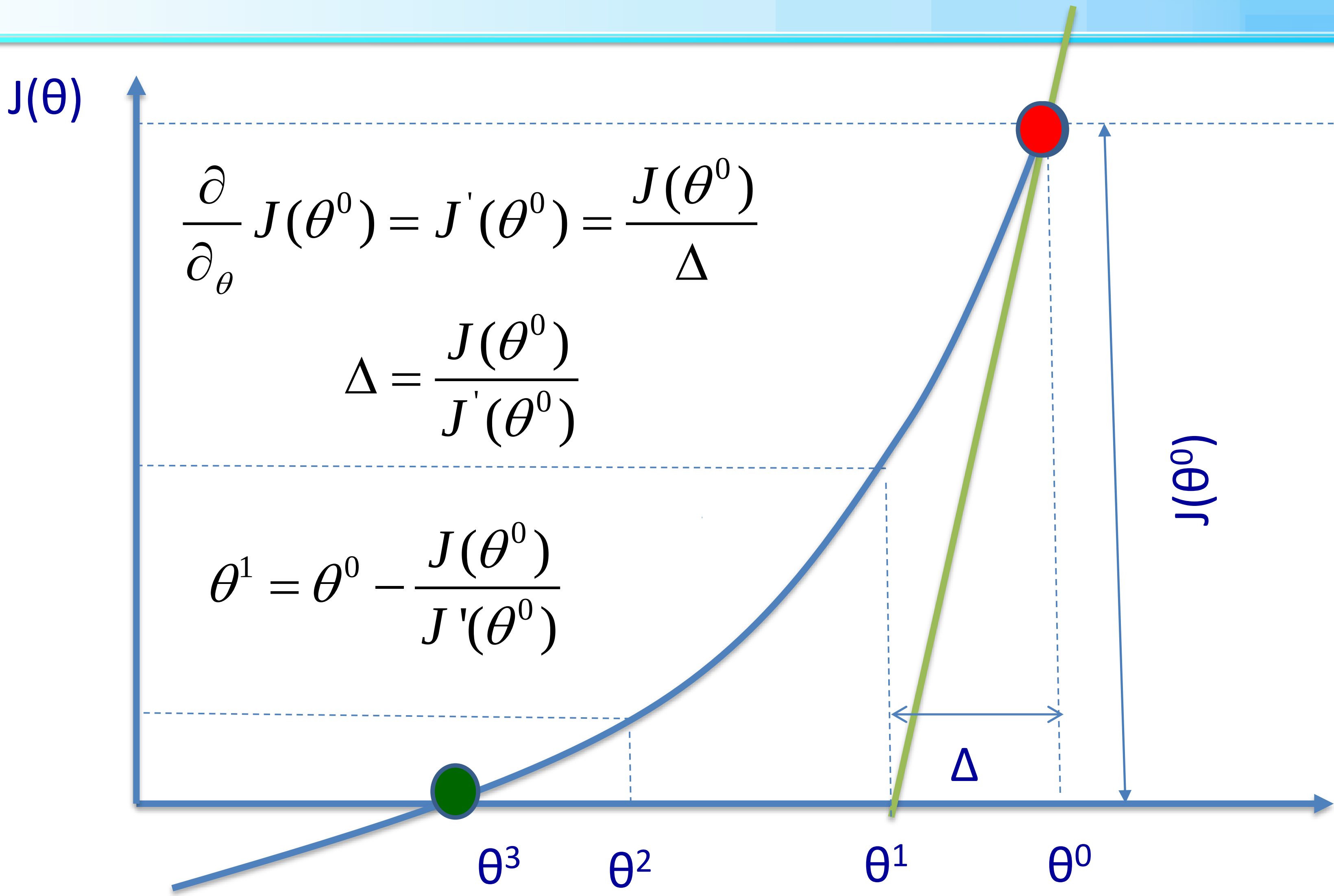
}

Newton's Method

$$\frac{\partial}{\partial_\theta} J(\theta^0) = J'(\theta^0) = \frac{J(\theta^0)}{\Delta}$$

$$\Delta = \frac{J(\theta^0)}{J'(\theta^0)}$$

$$\theta^1 = \theta^0 - \frac{J(\theta^0)}{J'(\theta^0)}$$



Newton's Method

Start with random value of coefficient θ^0 and then step by step update θ

While J(θ)!=0

{

- Calculate the tangent line of J(θ) at θ^t

- **Find the cross point of tangent line with the θ axis, called θ^{t+1}**
- **Update θ^t to θ^{t+1}**

$$\theta^{t+1} = \theta^t - \frac{J(\theta^t)}{J'(\theta^t)} = \theta^t - \frac{E'(\theta^t)}{E''(\theta^t)}$$

}

Newton's Method

$$\theta^{t+1} = \theta^t - \frac{J(\theta^t)}{J'(\theta^t)} = \theta^t - \frac{E'(\theta^t)}{E''(\theta^t)}$$

$$= \theta^t - H^{-1}\Delta_\theta E$$

Where: H is Hessian Matrix, $\Delta_\theta E$ is a derivative vector

$$H = \begin{vmatrix} H_{00} & H_{01} & \dots & H_{0n} \\ H_{10} & H_{11} & \dots & H_{1n} \\ \dots & \dots & \dots & \dots \\ H_{n0} & H_{n1} & \dots & H_{nn} \end{vmatrix} \; where\ H_{ij} = \frac{\partial^2 E}{\partial_{\theta_i}\partial_{\theta_j}} \qquad \Delta_\theta E = \begin{vmatrix} \frac{\partial}{\partial_{\theta_0}} E(\theta) \\ \dots \\ \frac{\partial}{\partial_{\theta_n}} E(\theta) \end{vmatrix}$$

Newton's Method

$$\Delta_\theta E = \left| \begin{matrix} \dfrac{\partial}{\partial_{\theta_0}} E(\theta) \\ \cdots \\ \dfrac{\partial}{\partial_{\theta_n}} E(\theta) \end{matrix} \right|$$

$$\Delta_\theta E = \frac{1}{m} \sum_{i=1}^{m} (h\ (x^{(i)}) - y^{(i)}) x^{(i)}$$

$$H = \frac{1}{m} \sum_{i=1}^{m} \left[h(x^{(i)})(1 - h(x^{(i)}) x^{(i)} (x^{(i)})^T \right]$$

Newton's Method

Which is the best option checking if Newton's method has converged?

1. **Plot h(x) as a function of x, and check if it fits the data well.**
2. **Plot E(θ) as a function of θ and check if it has reach a minimum**
3. **Plot θ as a function of the number of iteration and check if it has stop decreasing (or decreasing only a tiny amount per iteration)**
4. **Plot E(θ) as a function of number of iteration and check if it has stop decreasing (or decreasing only a tiny amount per iteration)**

Newton’s Method

Newton’s Method vs Gradient Descent

	Gradient Descent	Newton’s Method
Implementation	Simpler Need to chose parameter	More complex No
Convergence Speed	Need more Iteration Computation cost of each iteration is cheep 0(n) n:number of features	Less iteration Each iteration is more expensive $0(n^3)$ N:number of features
Application	Use when n is large (n>1000)	Use when n is small

Newton's Method

Exercise:

Given the following data, compute the Hessian Matrix and the derivative vector at $\theta_0 = \theta_1 = 0$

Price (b.VND)	*Location*
2.5	Thanh Xuan
3	Thanh Xuan
6	Hoan Kiem
2	Thanh Xuan
7	Hoan Kiem
10	Hoan Kiem

References

http://openclassroom.stanford.edu/MainFolder/CoursePage.php?course=MachineLearning

Andrew Ng Slides:

https://www.google.com.vn/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&cad=rja&uact=8&sqi=2&ved=0ahUKEwjNt4fdvMDPAhXIn5QKHZO1BSgQFggfMAE&url=https%3A%2F%2Fdatajobs.com%2Fdata-science-repo%2FGeneralized-Linear-Models-%5BAndrew-Ng%5D.pdf&usg=AFQjCNGq37q2uVFcpGhNqH-5KZSlJ_HSxg&sig2=vnCEvyvKQGCuryttAPcokw&bvm=bv.134495766,d.dGo