# Week 1

**Introduction**

**Welcome to Machine Learning**

* What is machine learning
  + Rank pages – Google, Bing
  + Recognize photos
  + Spam emails
  + Robots to tidy up the house
  + Neural networks is learning algorithms

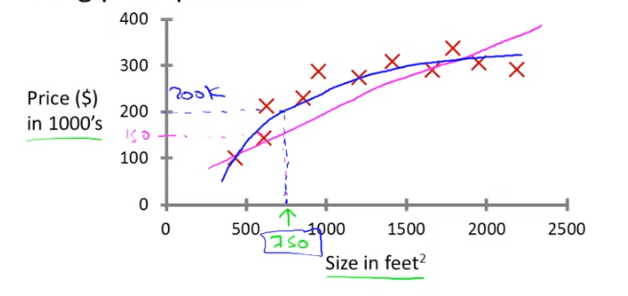
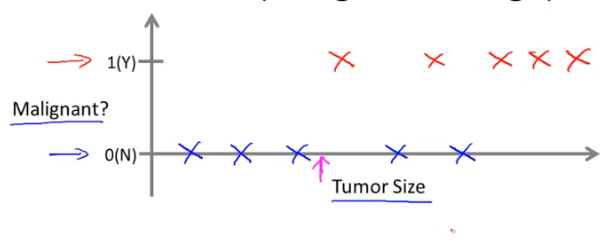
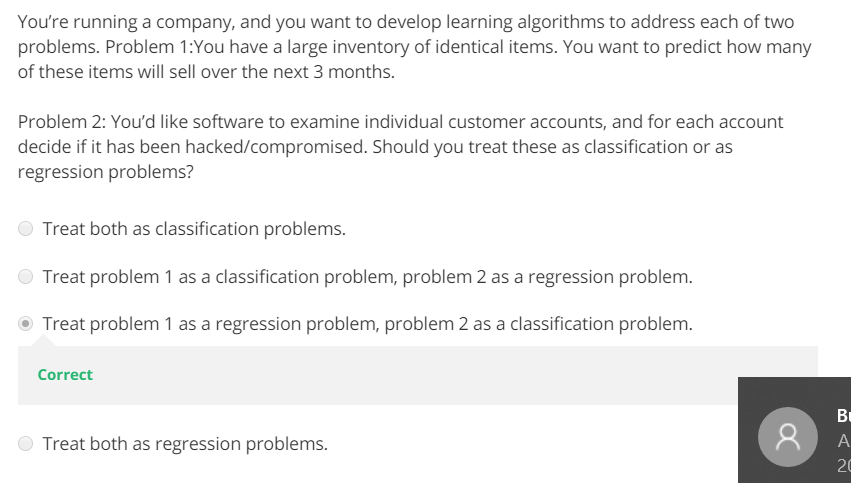
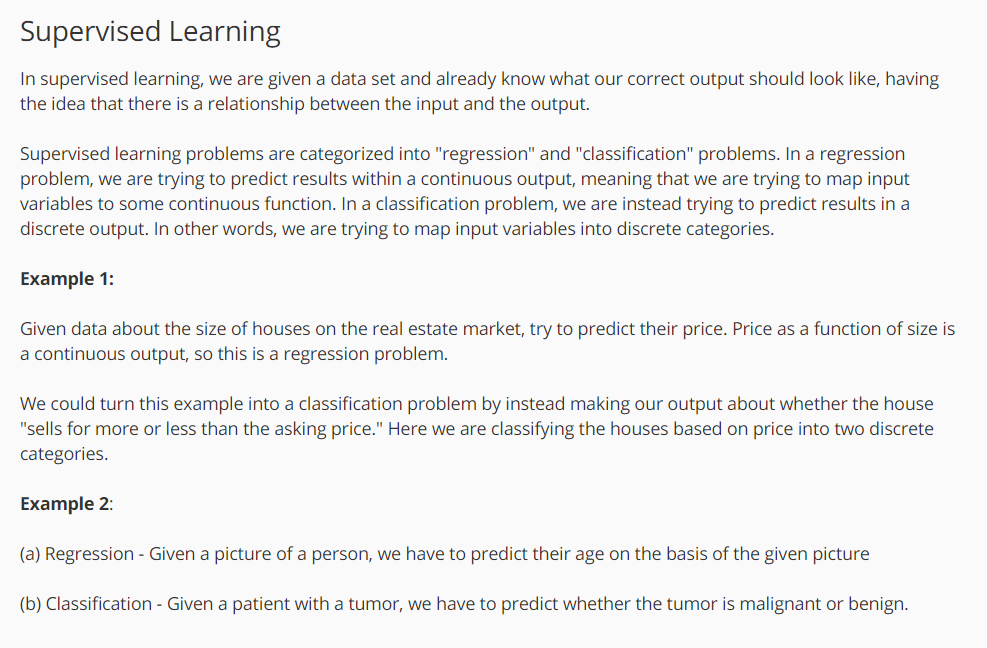
**Welcome**

* Examples
  + Database mining: web click data, medical records, biology, engineering
  + Applications can’t program by hands: autonomous helicopter, handwriting recognition, most of natural language processing, computer vision
  + Self-customizing programs: amazon, Netflix product recommendations
  + Understanding human learning (brain, real AI)

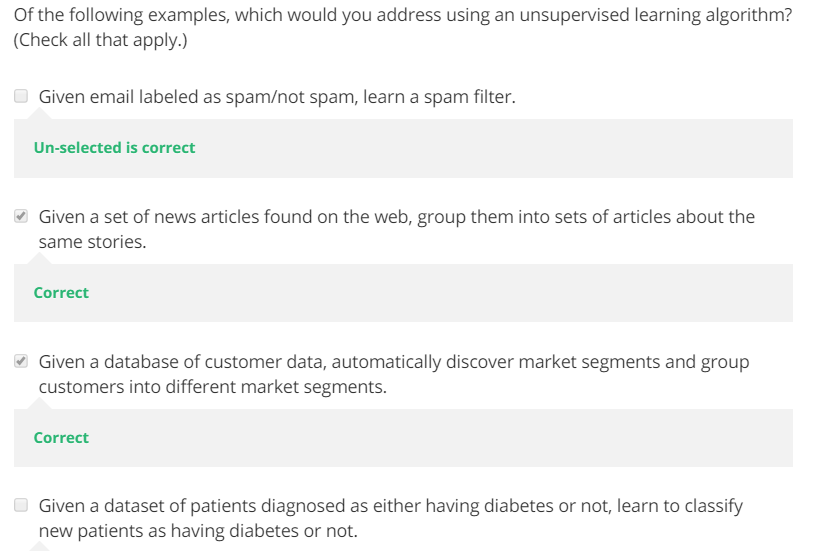
**What is Machine Learning?**

* Definition:
  + field of study that gives computers the ability to learn without being explicitly programmed.
  + "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E."
    - T: classifying emails as spam or not spam.
    - E: watching you label emails as spam or not spam.
    - P: the number (or fraction) of emails correctly classified as spam/not spam.
* Machine Learning Algorithms
  + Supervised learning – we’re going to teach machine
  + Unsupervised learning – machine learns it by itself
  + Reinforcement learning
  + Recommender systems
  + Practical advice for applying learning algorithms

**Supervised Learning**

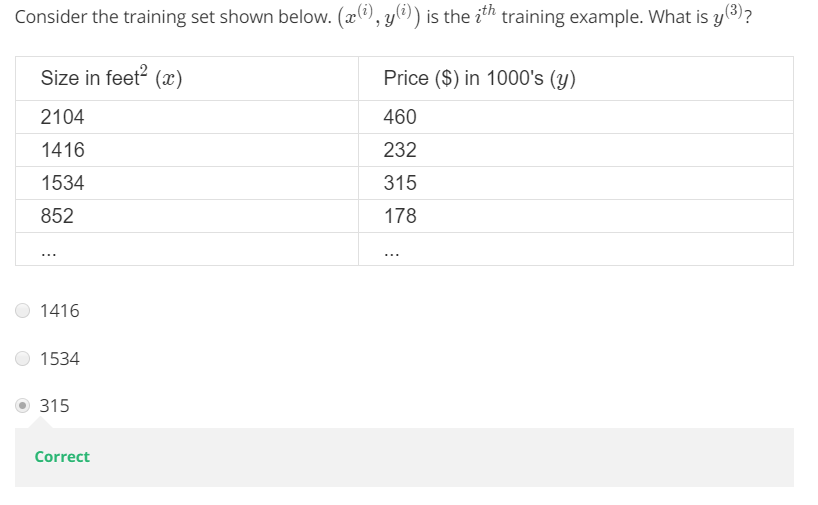
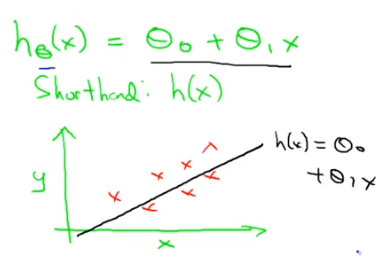
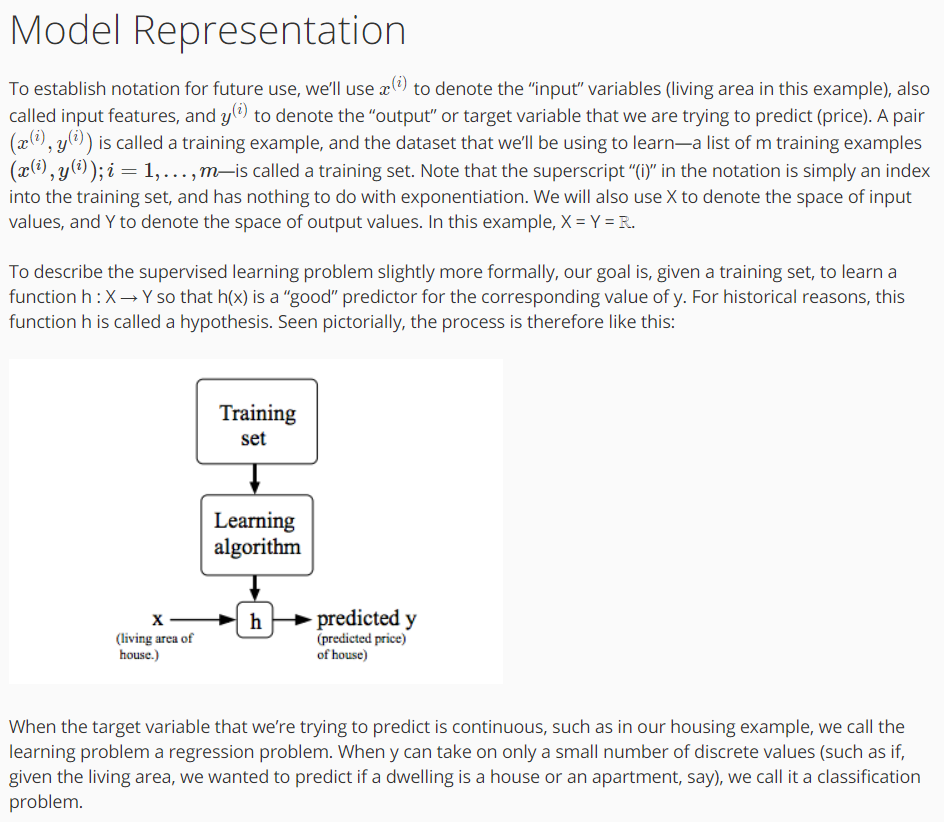
* Right answers given.
* Regression(回归): predict continuous & real valued output  
  
* Classification: discrete valued output (0 or 1), or maybe more than 2 output values.  
    
    
  

**Unsupervised Learning**

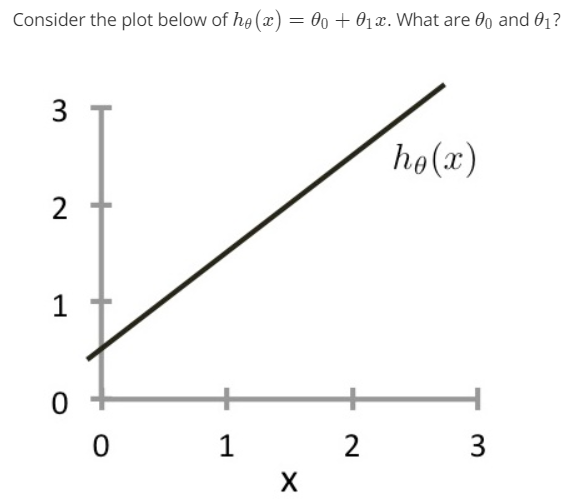
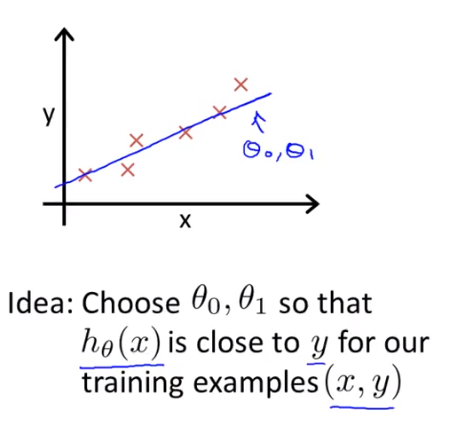
* No label. Do not know what x or y-axis is.
* Examples
  + Organize computing clusters
  + Social network analysis
  + Market segmentation
  + Astronomical data analysis
* Cocktail party problem algorithms
  + 
* Octave / Matlab
*   
  
* Quiz 1

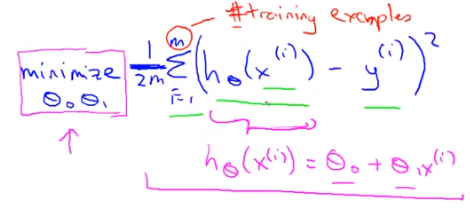
**Model and Cost Function**

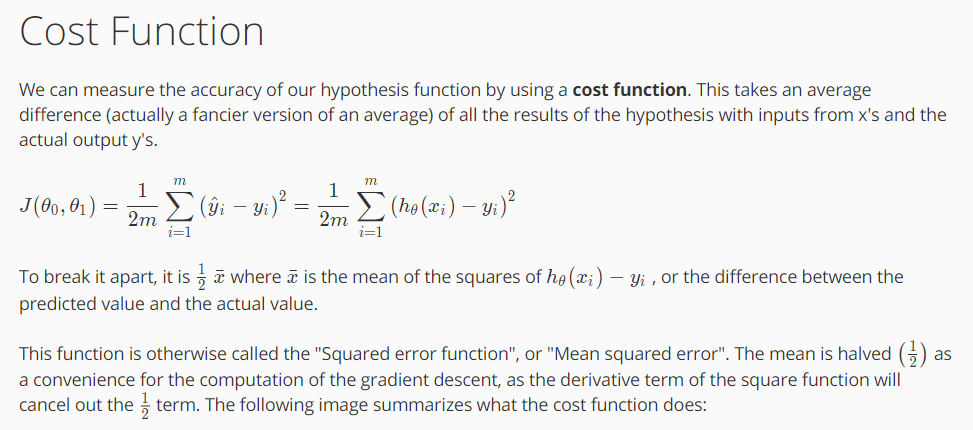
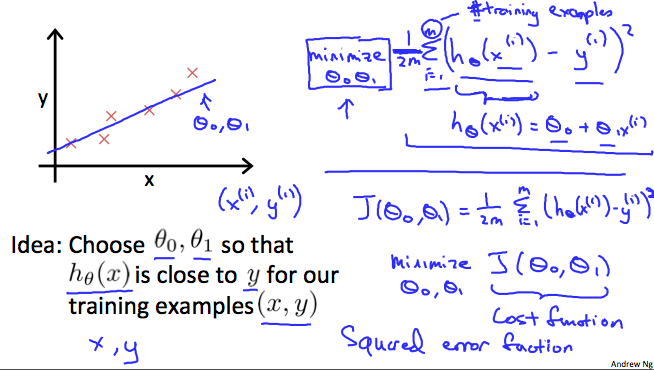
Model Presentation

* (x,y): one training example  
  (x(i), y(i)): ith training example  
  
* Training set 🡪 learning algorithm 🡪 hypothesis (it is a function, take input, output the result)
* Linear regression with one variable / Univariate linear regression  
  univariate = one variable  
    
  why linear? Because it’s the simple build block. Some complicated problem may can not use linear to solve.  
  

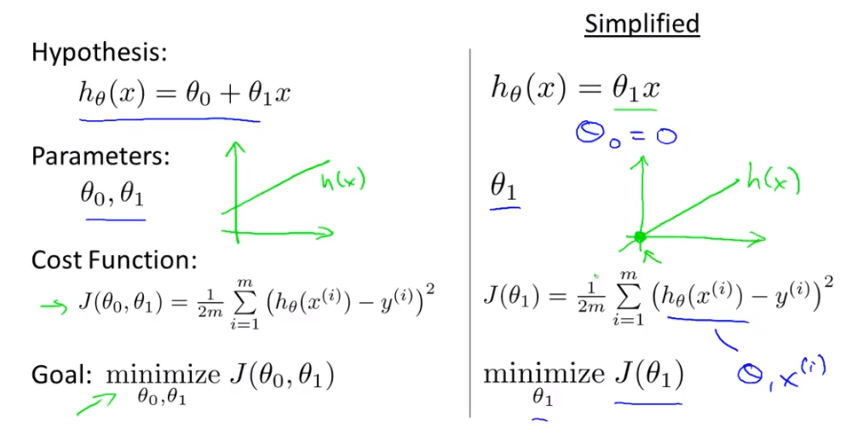
**Cost Function**

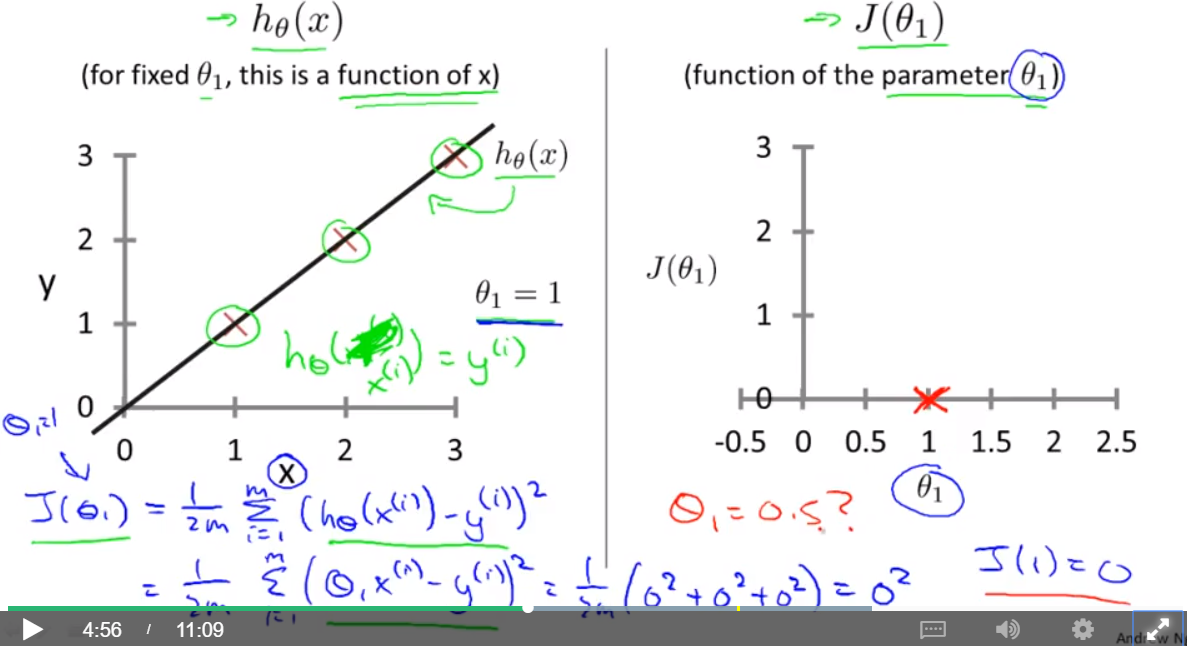
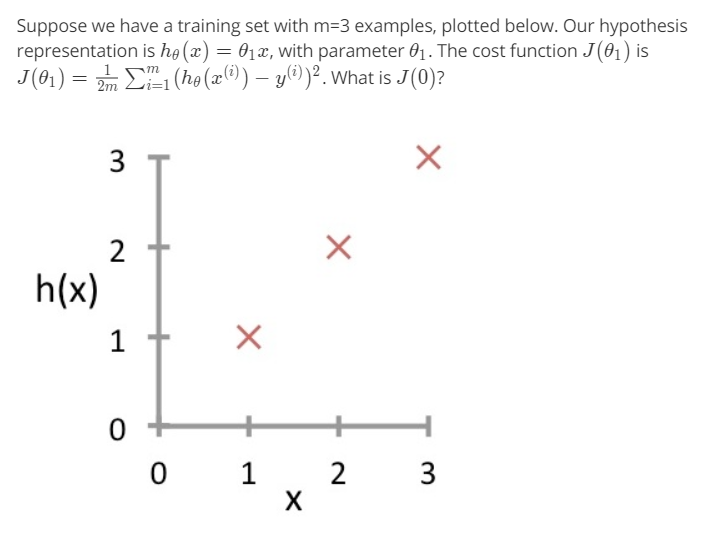
*   
  
* Linear Regression  
  
* Cost function / Squared error function – the most common one used for regression problem



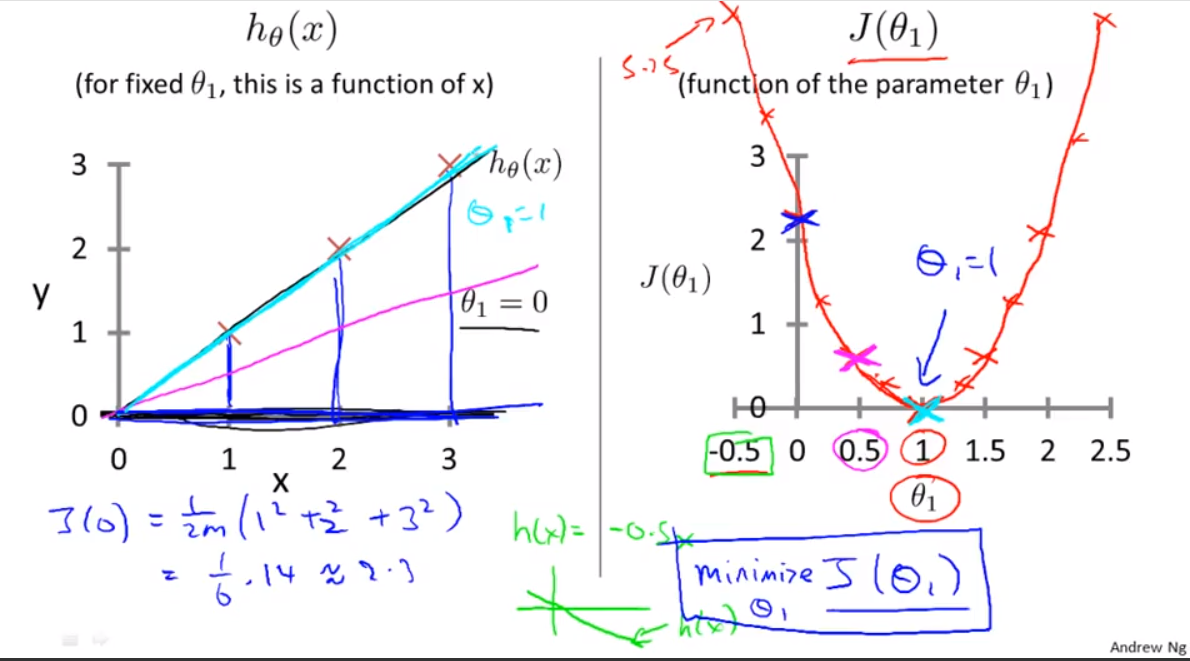
Putting the 2 at the constant one half in front, it may just sound the math probably easier so minimizing one-half of something, right, should give you the same values of the process, theta 0 theta 1, as minimizing that function.  
  


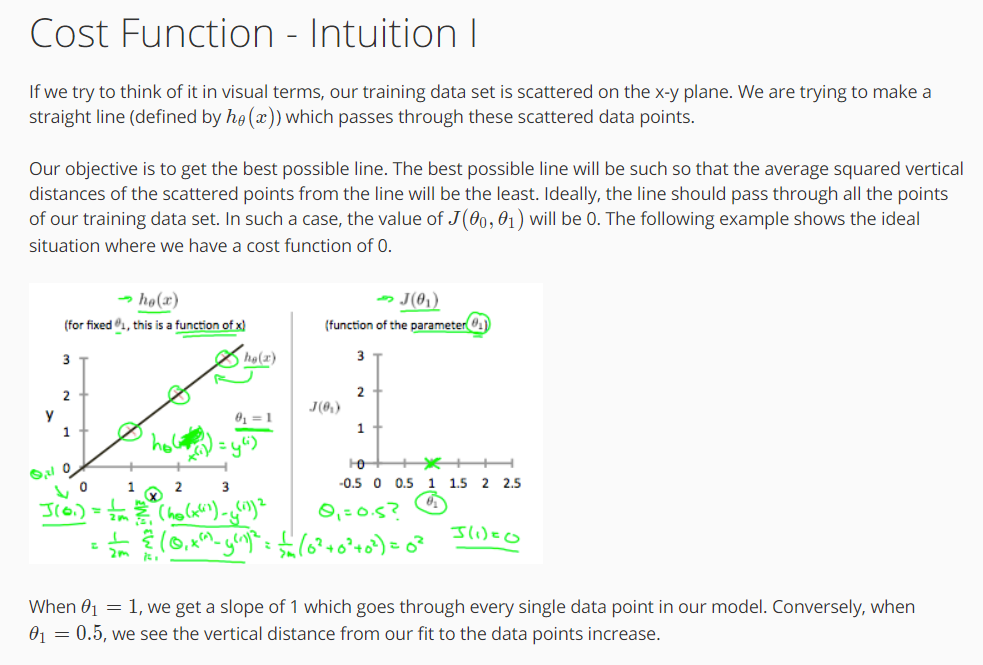
**Cost Function – Intuition I**

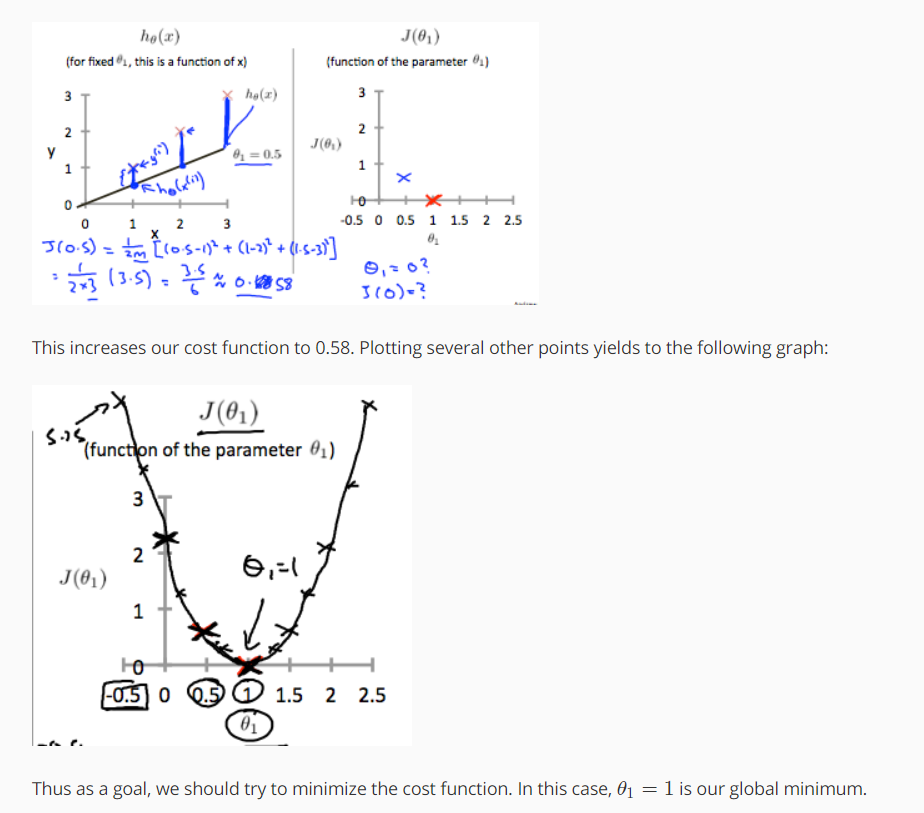


  
  
answer:

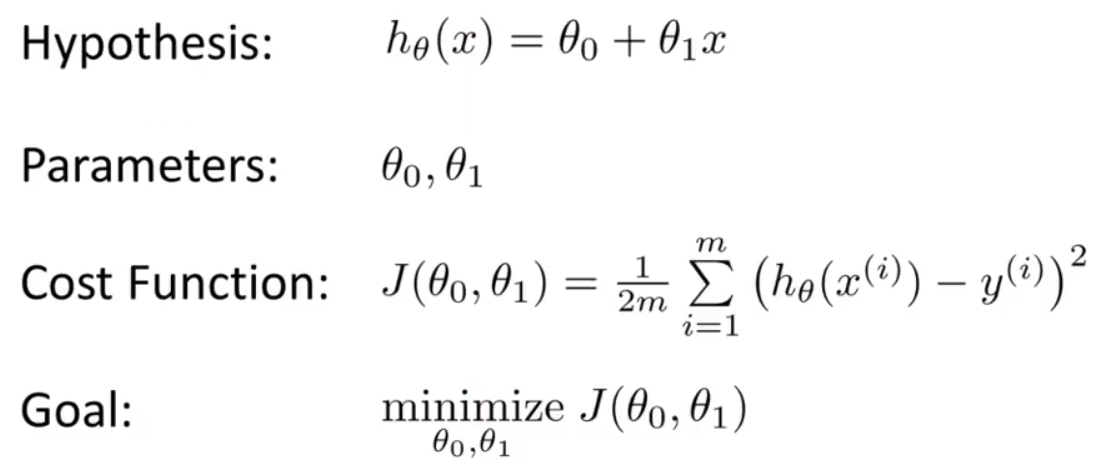
Goal: find the data of theta1 can minimize J(theta1).

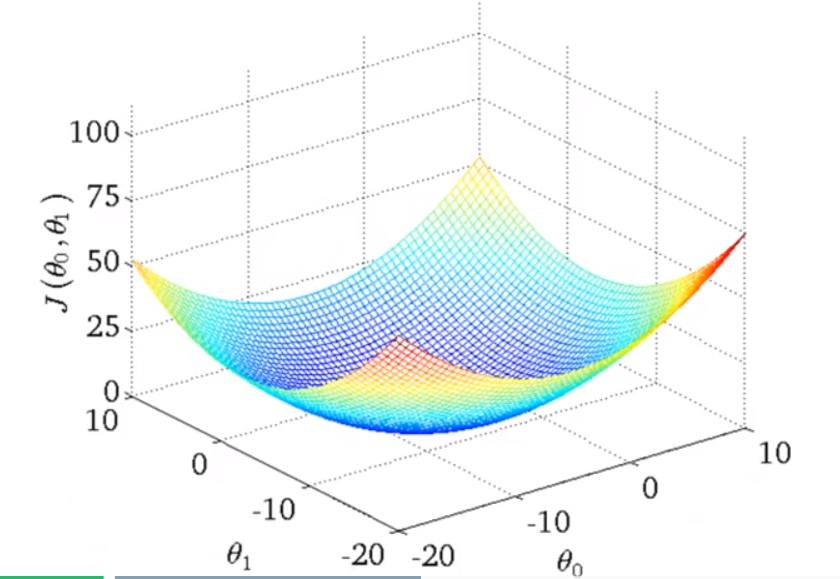
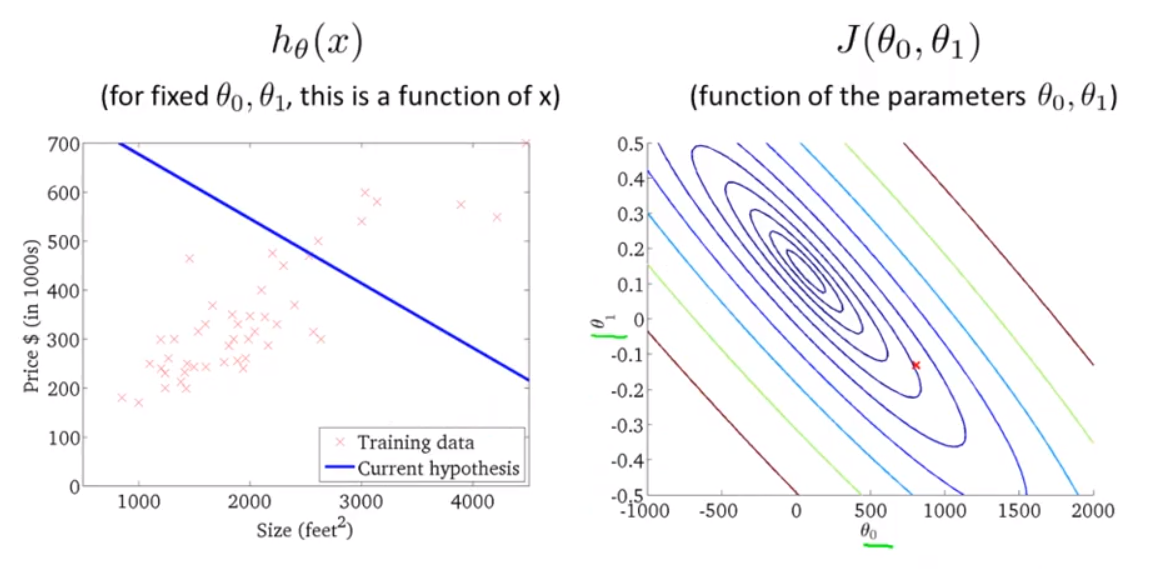
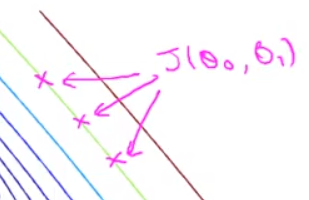


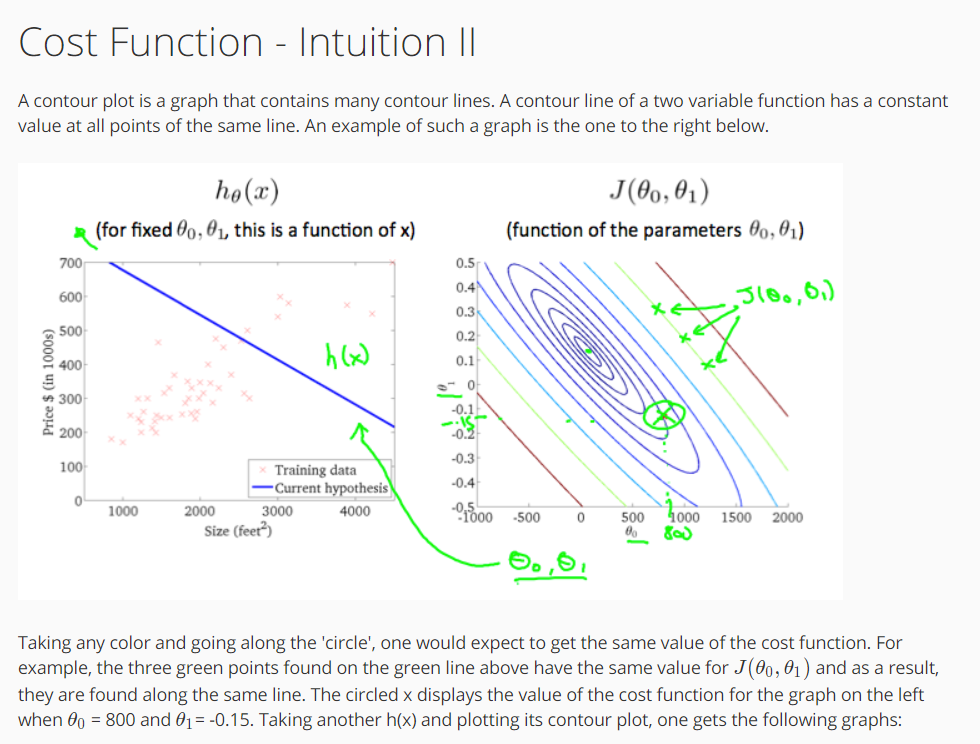
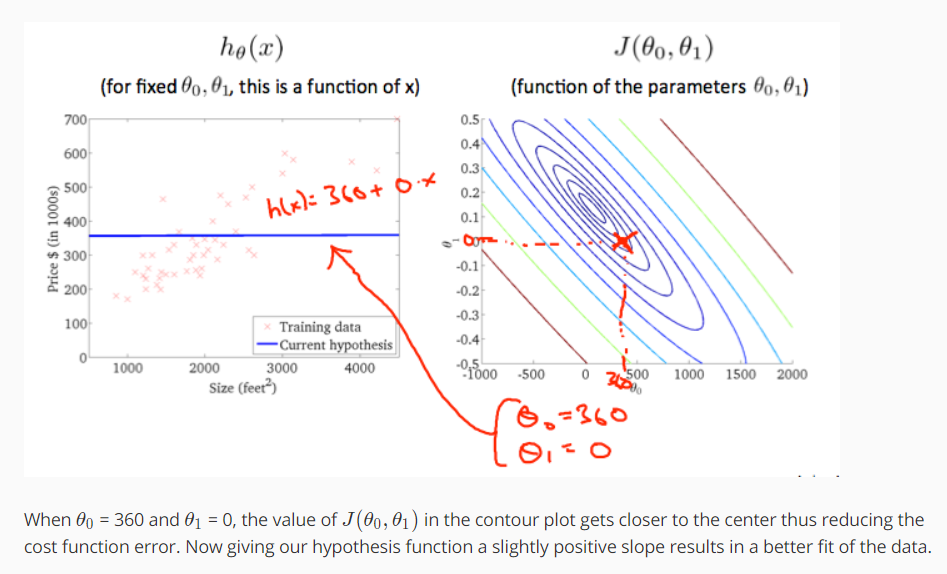
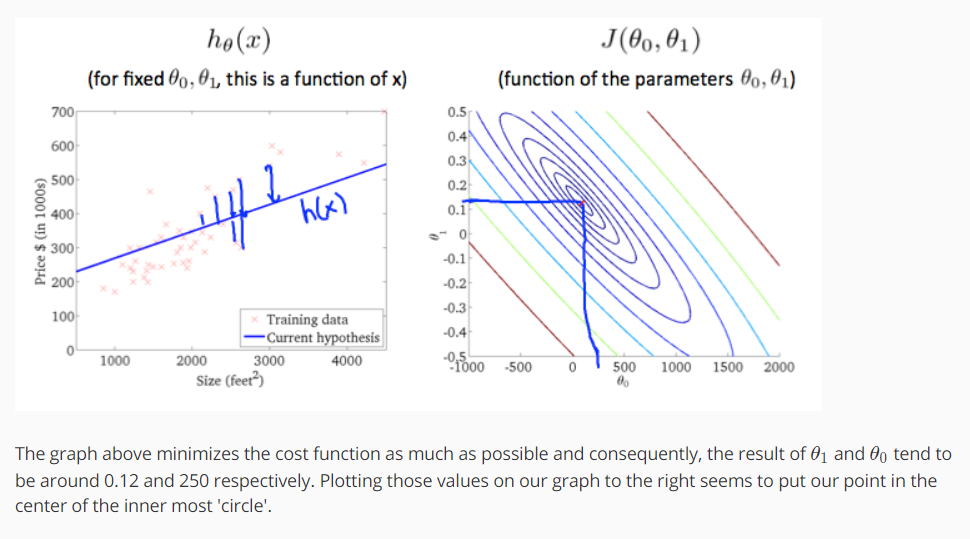




**Cost Function – Intuition II**

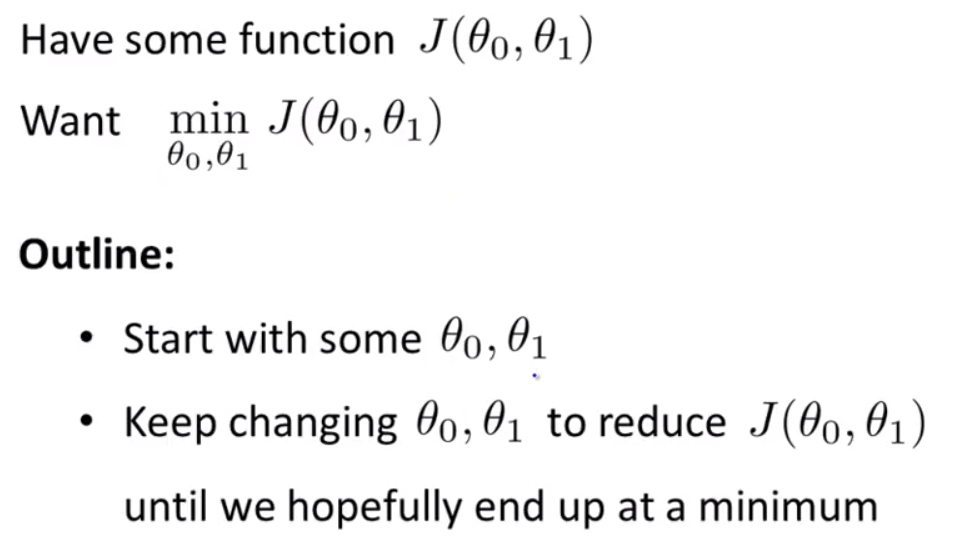


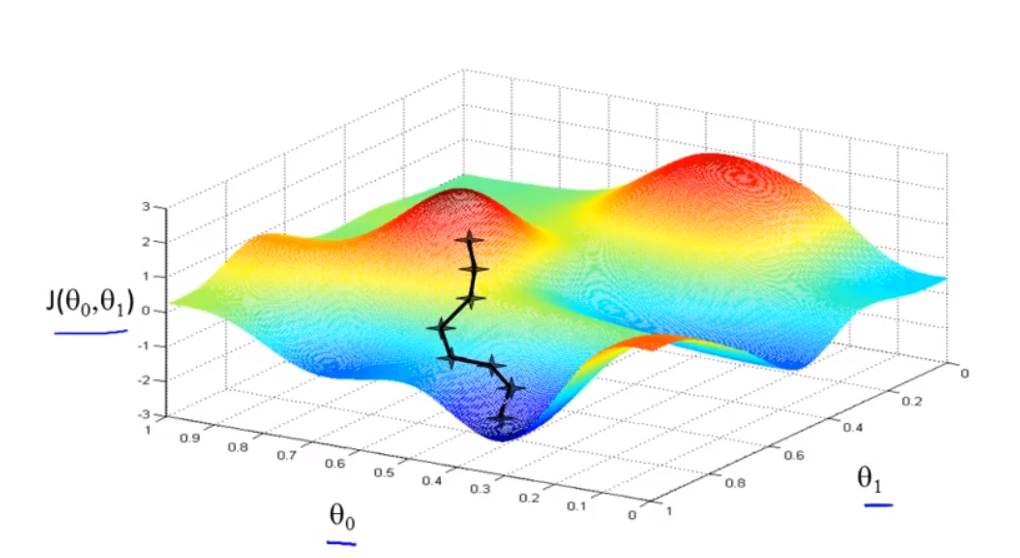
* When we only have theta1, the cost function will look like:  
    
  When we have theta0 and theta1, the cost function will look like:  
    
  contour(轮廓) plot  
  contour figure  
    
  in the diagram above, the contour figure is on the right.  
  i.e. these three plots, have same values of J(theta0, theta1).

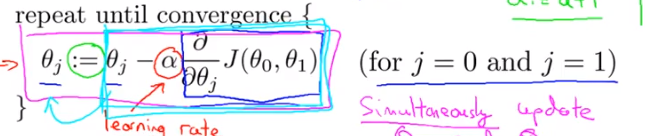
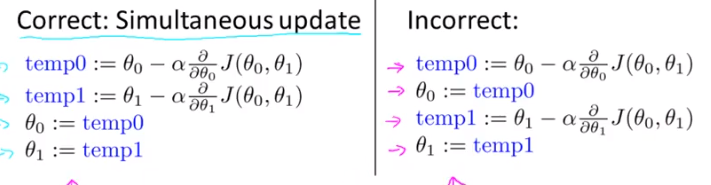
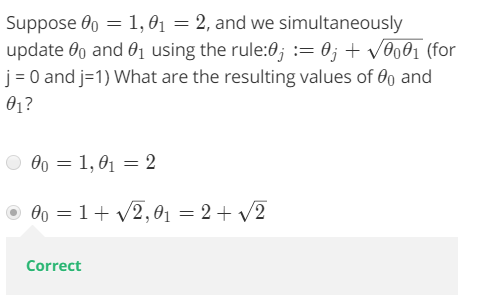
  
  


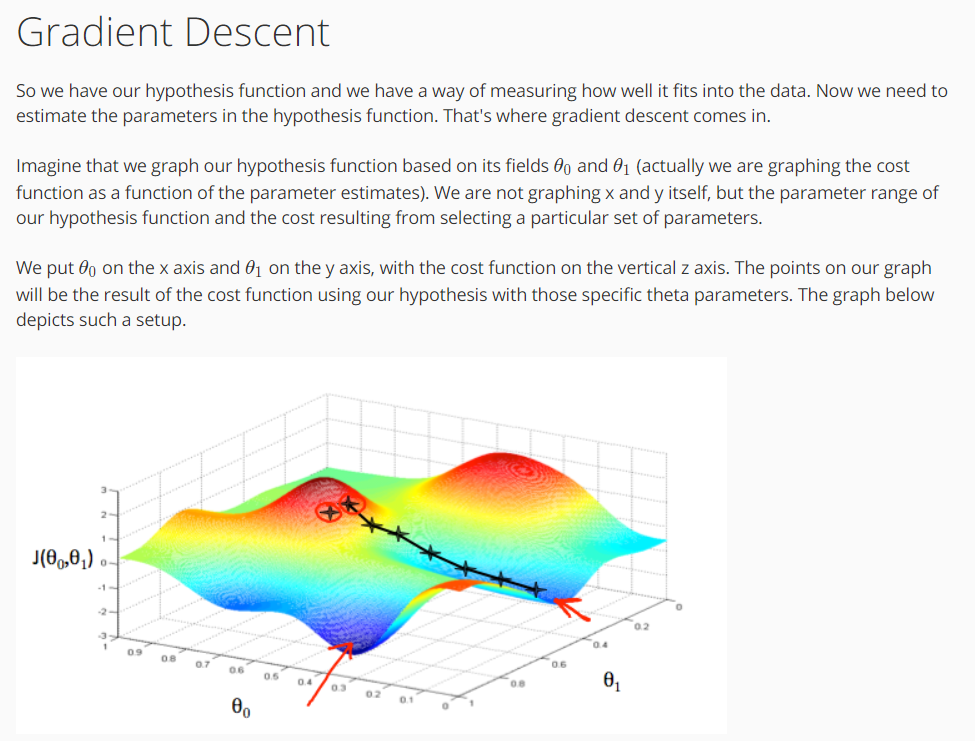
**Parameter Learning**

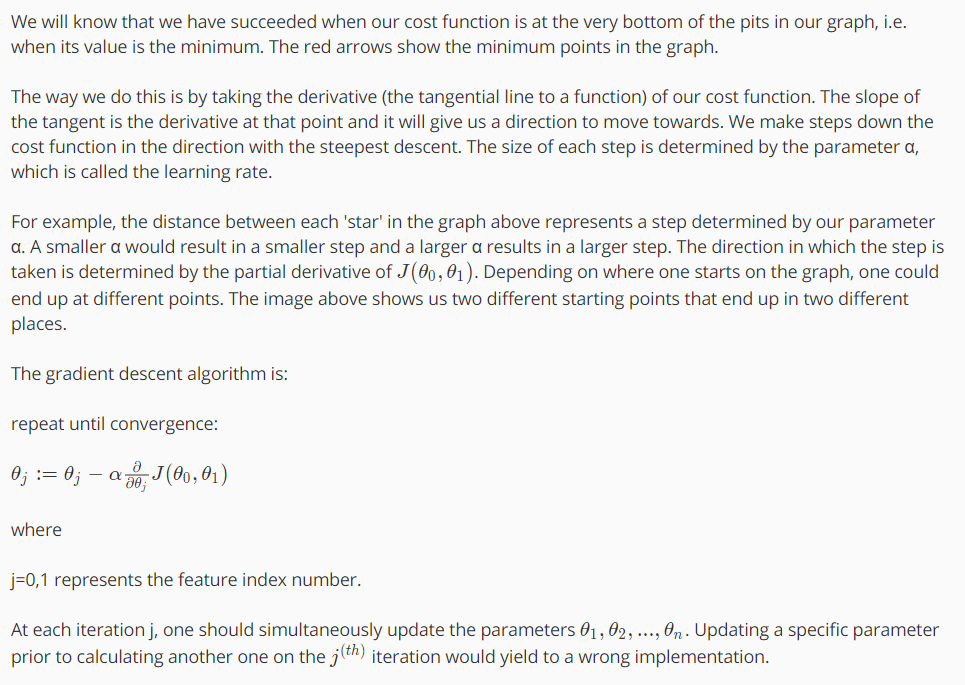
**Gradient Descent (梯度下降) – an algorithm for linear regression with one variable, also could use for others in machine learning to minimize the cost function J.**

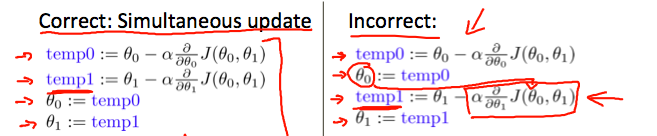


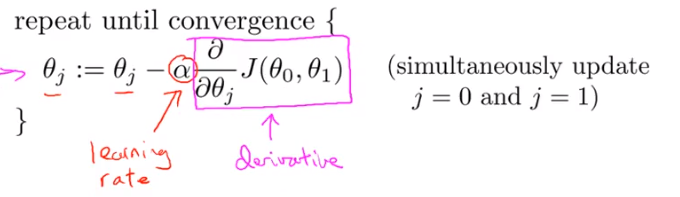
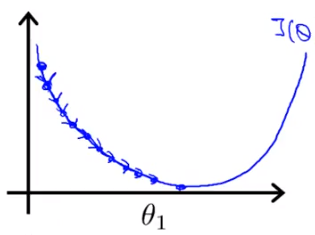
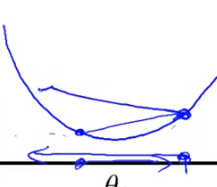


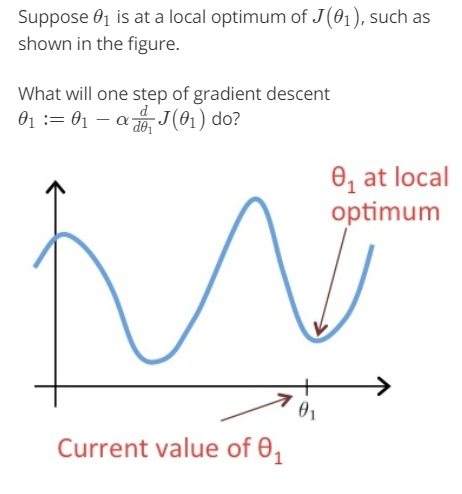
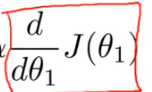
* find the direction that downhill the fastest.
* The different start points will lead to different end points.
* Derivative (导数) can tell the slope of each point.
* Gradient descent algorithm  
  
* a := b 🡪 Computer operation, set the a value to equal to b  
  a = b 🡪 the value of a equal to the value of b
* α 🡪 learning rate to control how big a step we take downhill with creating descent. i.e. if α is really small, we take a baby step downhill.
* Need to update θ0 and θ1 at the same time.  
    
  the differences between left and right:   
   - on the right, if you assign θ0 first, then you are going to use the new value of θ0 for the next step to calculate the temp1. Thus, you get two different values of temp1 on both ways.   
    
  Example:  
   

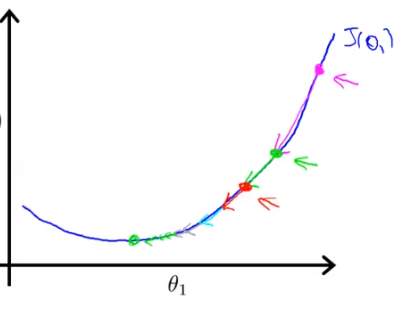


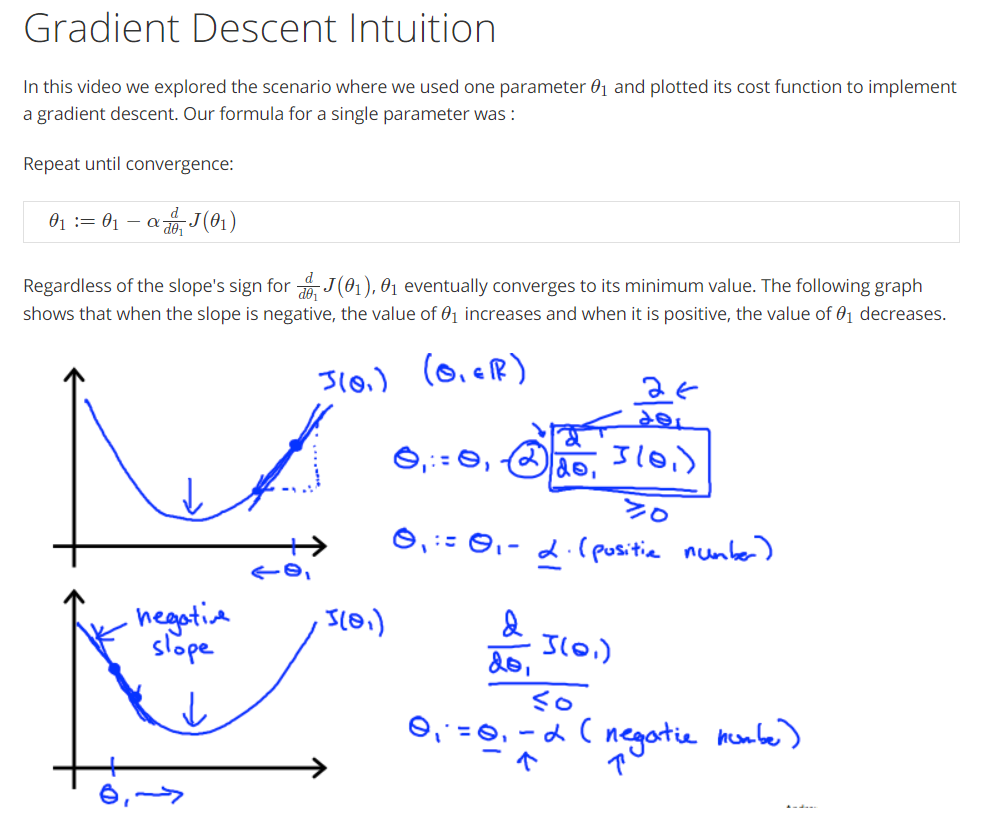
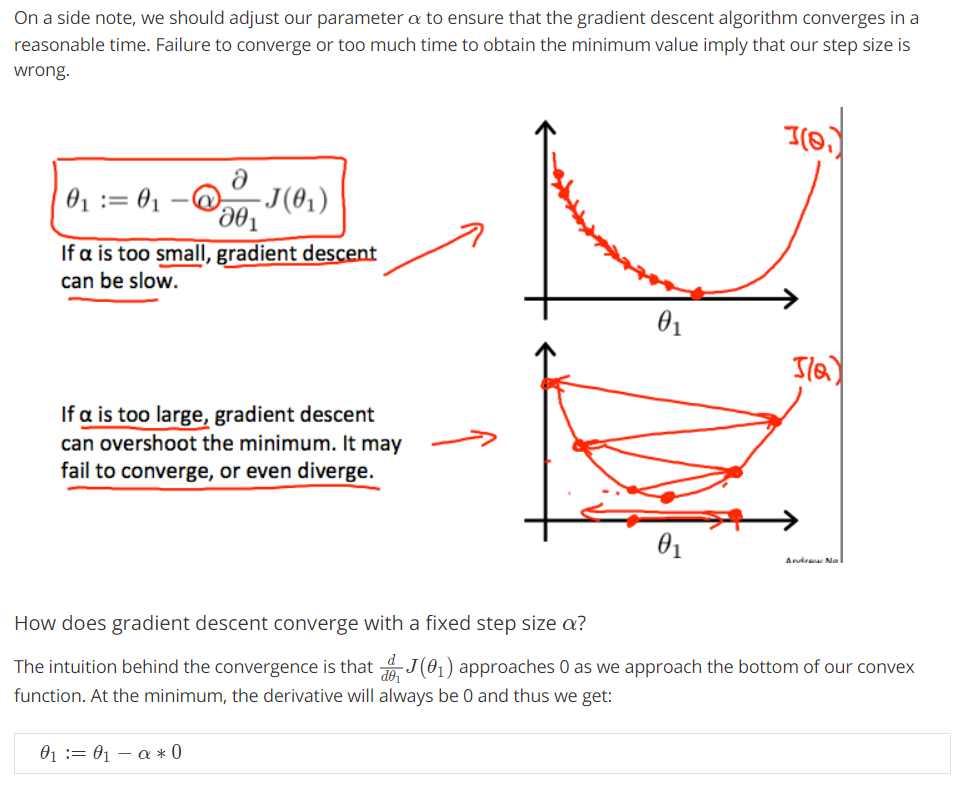
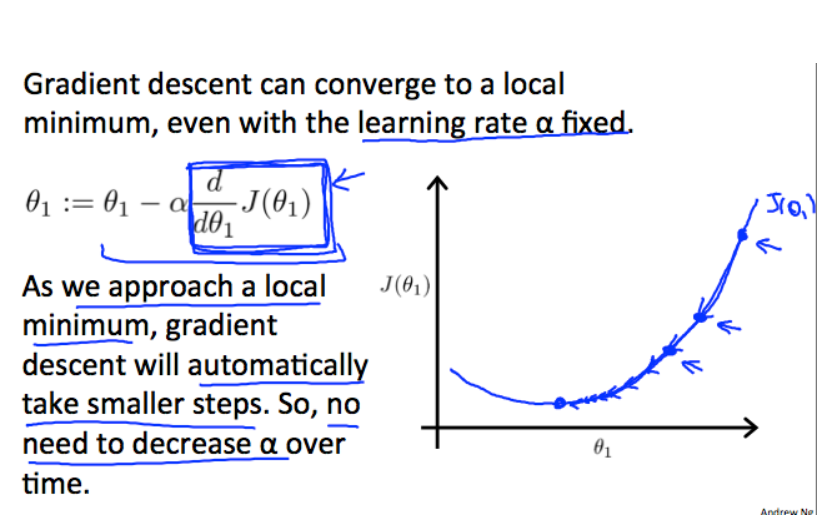


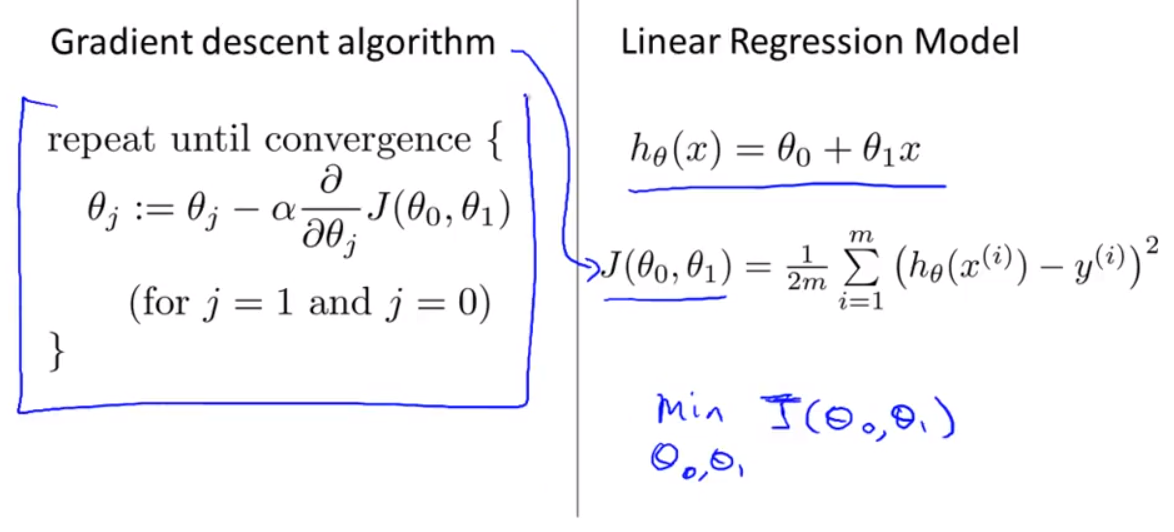
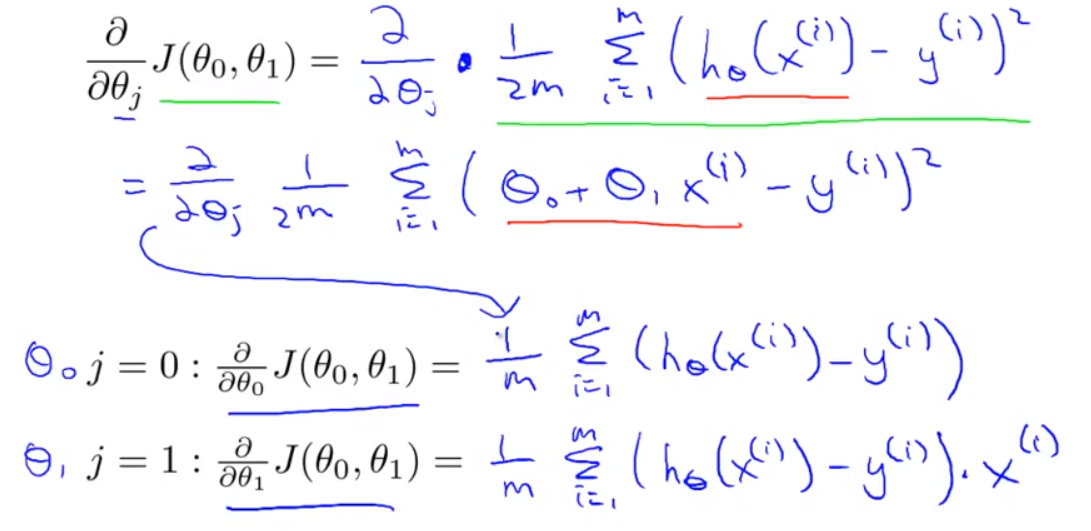
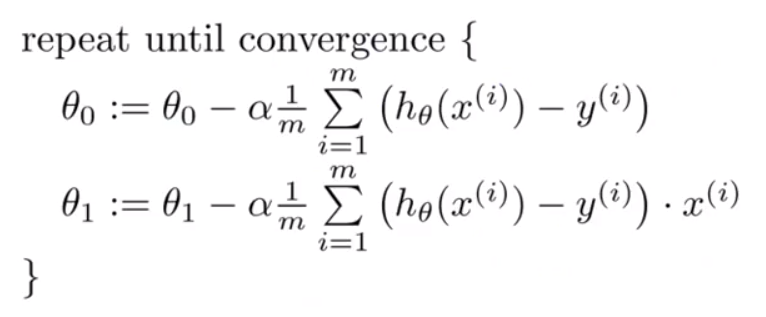
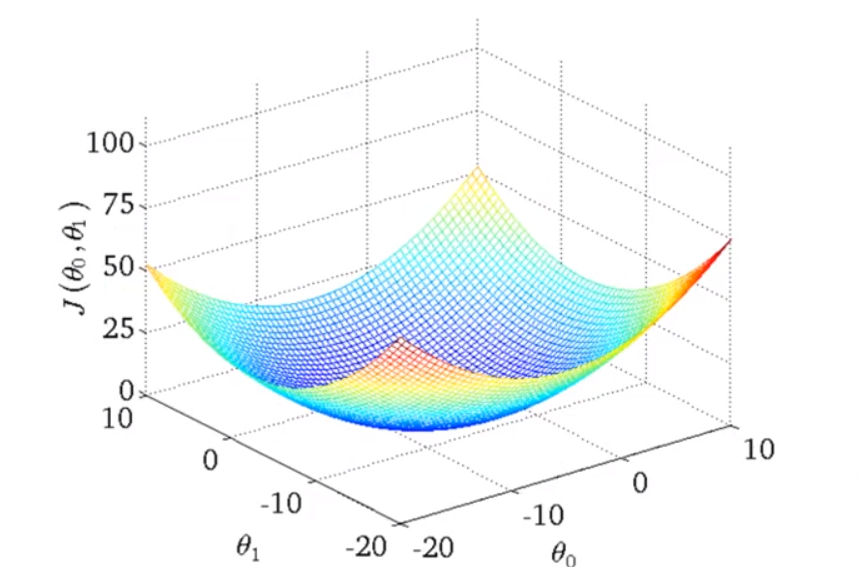
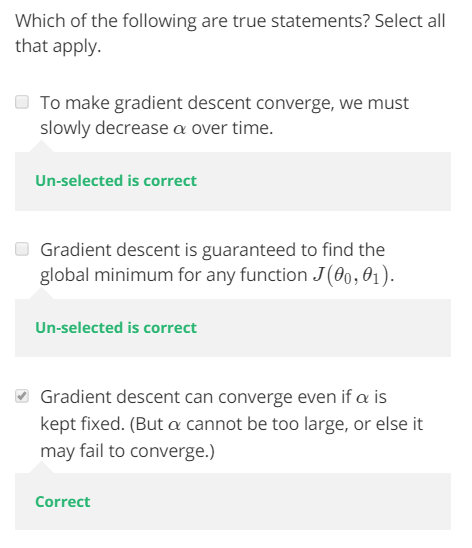
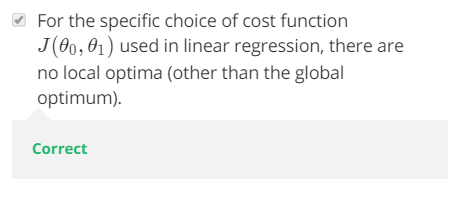


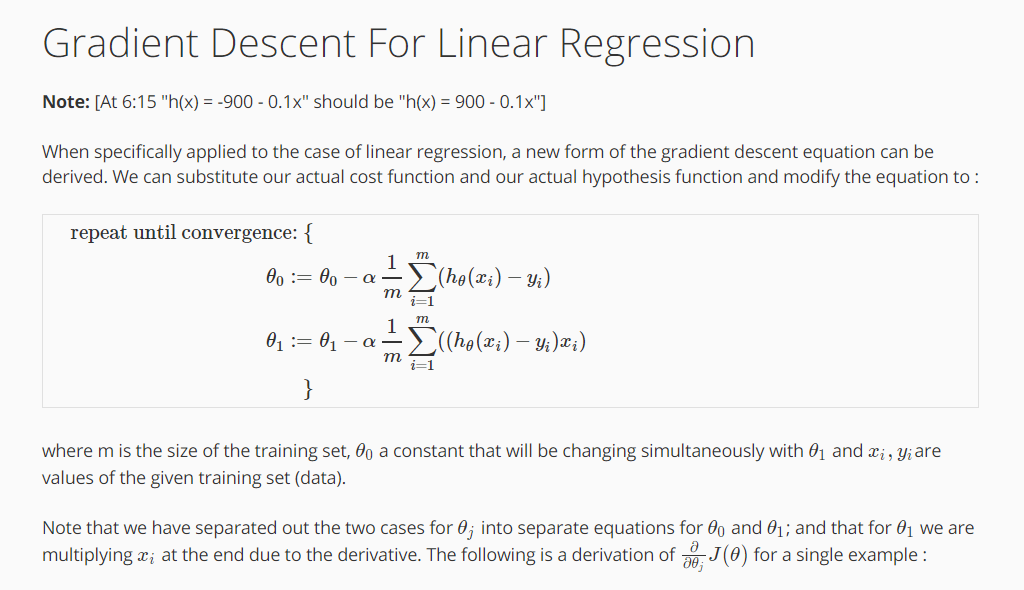
* Gradient descent Intuition
* 
* Derivative and Partial Derivative
  + 一元函数,一个y对应一个x,导数只有一个.  
    二元函数,一个z对应一个x和一个y,那就有两个导数了,一个是z对x的导数,一个是z对y的导数,称之为偏导.
  + Derivative symbol: d  
    Partial derivative symbol: 
  + If the slope >= 0, then the derivative is >= 0.
* α
  + if α is too small, gradient descent can be slow.  
    
  + if α is too large, gradient descent can overshoot(越过) the minimum. It may fail to converge, or even diverge.  
    converge(收敛): 向某一值靠近  
    diverge(发散): 在某一点不收敛  
    
* Exercise:

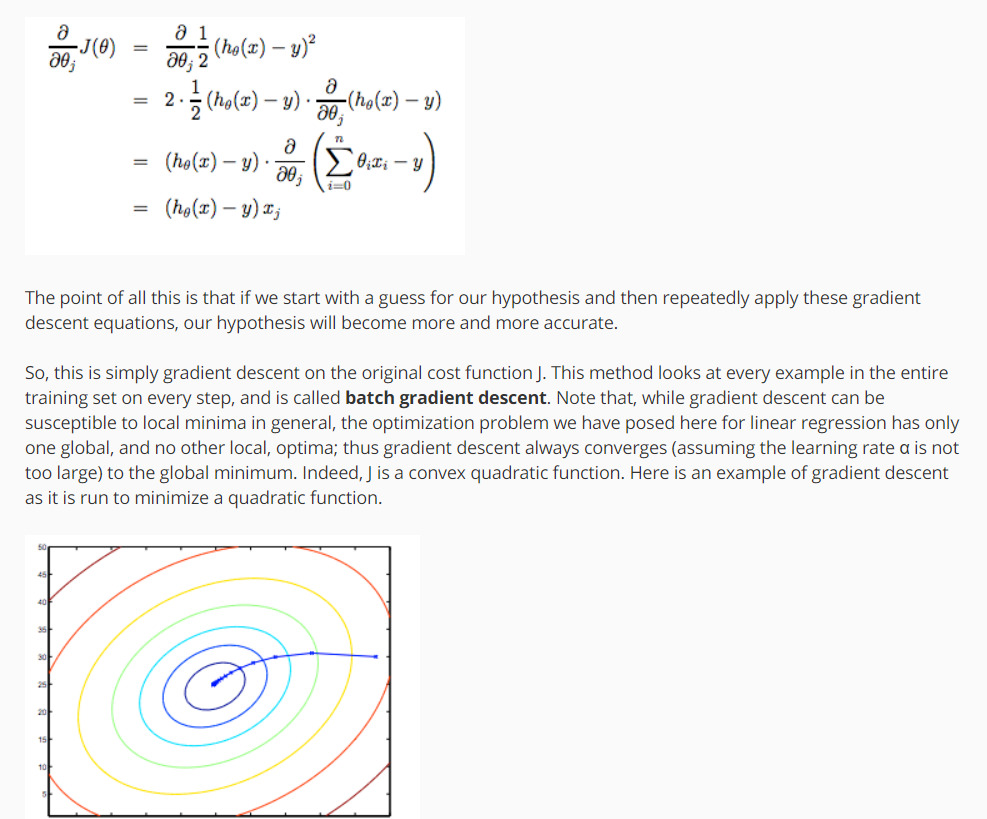
 optimum: 极点; optima:最优  
answer:   
analysis: the slope at θ1 is 0. Thus, the derivative term  is 0. Which means . So, when you use gradient descent intuition to update θ1, actually the θ1’s value would never be changed if θ1 is at local optimum. Which is what you want: get the local optimum.

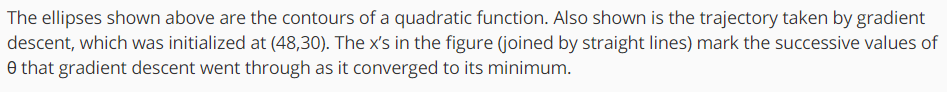
* Gradient descent can converge to a local minimum, even with the learning rate α fixed. As we approach a local minimum, gradient descent will **automatically** take smaller steps because the derivative term is getting smaller. So, no need to decrease α over time.  
  

* Gradient Descent for Linear Regression / batch gradient descent
* Batch: each step of gradient descent uses all the training examples. In computing, the formula will calculate the sum of all data. (∑)
*   
  combine two formulas together by derivative term.  
    
  Then, put them back to formulas:  
  
* Cost function for linear regression is always going to be a bowl-shaped function, which we call convex function. It only has one global optima.  
  
* Exercise  
    
  







* S
* S
* S
* S
* S
* S

S

S

S

S

s

## To learn more and get OneNote, visit [www.onenote.com](http://go.microsoft.com/fwlink/?LinkID=523891).