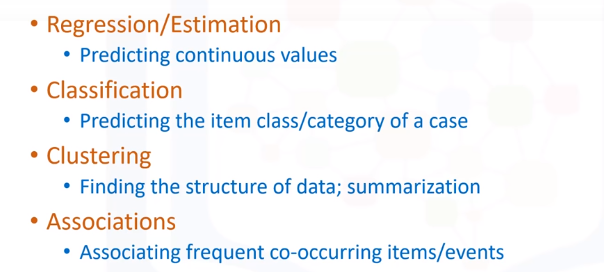
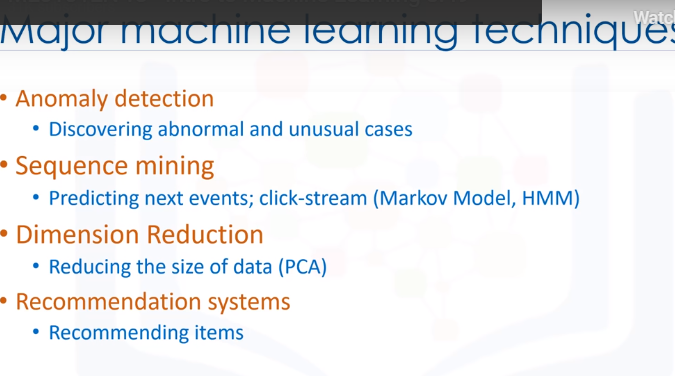
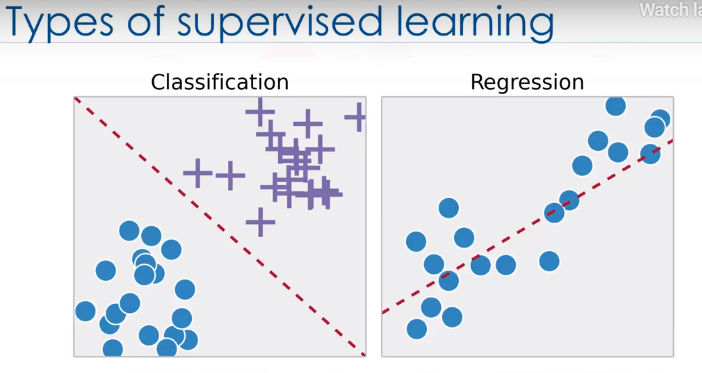
Machine Learning:

1. In this course, you’ll learn how Machine Learning is used in many key fields and industries.
2. For example, in the health care industry, data scientists use Machine Learning to predict
3. whether a human cell that is believed to be at risk of developing cancer, is either benign
4. or malignant.
5. As such, Machine learning can play a key role in determining a person’s health and welfare.
6. You’ll also learn about the value of decision trees and how building a good decision tree
7. from historical data helps doctors to prescribe the proper medicine for each of their patients.
8. You’ll learn how bankers use machine learning to make decisions on whether to approve loan
9. applications.
10. And you will learn how to use machine learning to do bank customer segmentation, where it
11. is not usually easy to run for huge volumes of varied data.
12. In this course, you’ll see how machine learning helps websites such as YouTube, Amazon, or
13. Netflix develop recommendations to their customers about various products or services, such as
14. which movies they might be interested in going to see or which books to buy.
15. Machine learning is the subfield of computer science that gives "computers the ability
16. to learn without being explicitly programmed.”

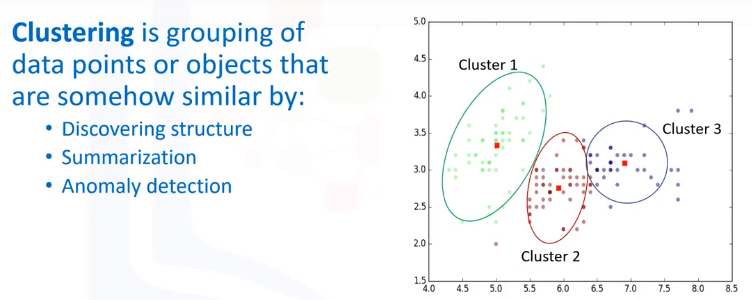
  


* In brief, AI tries to make computers intelligent in order to mimic the cognitive functions of humans.
* So, Artificial Intelligence is a general field with a broad scope including: Computer Vision,
* Language Processing, Creativity, and Summarization.
* Machine Learning is the branch of AI that covers the statistical part of artificial
* intelligence.
* It teaches the computer to solve problems by looking at hundreds or thousands of examples,
* learning from them, and then using that experience to solve the same problem in new situations.
* And Deep Learning is a very special field of Machine Learning where computers can actually
* learn and make intelligent decisions on their own.
* Deep learning involves a deeper level of automation in comparison with most machine learning algorithms.

1. You can write your machine learning algorithm using python, and it works very well.
2. However, there are a lot of modules and libraries already implemented in python that can make
3. **your life much easier.**
4. We teach the model by training it with some data from a labeled dataset.
5. It’s important to note that the data is labeled.
6. When dealing with machine learning, the most commonly used data is numeric.
7. **The second is categorical… that is, it’s non-numeric, because it contains characters**
8. rather than numbers.
9. **Classification is the process of predicting a discrete class label or category.**
10. Regression is the process of predicting a continuous value as opposed to predicting
11. a categorical value in Classification.

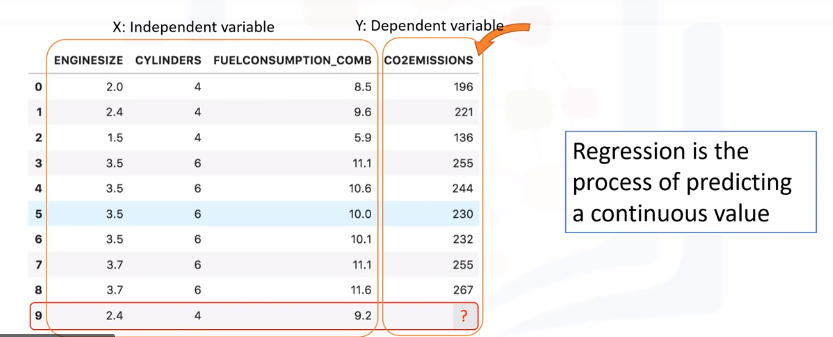


1. Unsupervised Learning is exactly as it sounds.
2. We do not supervise the model, but we let the model work on its own to discover information
3. that may not be visible to the human eye.
4. It means, The Unsupervised algorithm trains on the dataset, and draws conclusions on UNLABELED
5. data.
6. Generally speaking, unsupervised learning has more difficult algorithms than supervised
7. learning, since we know little to no information about the data, or the outcomes that are to
8. **be expected.**





Regression:

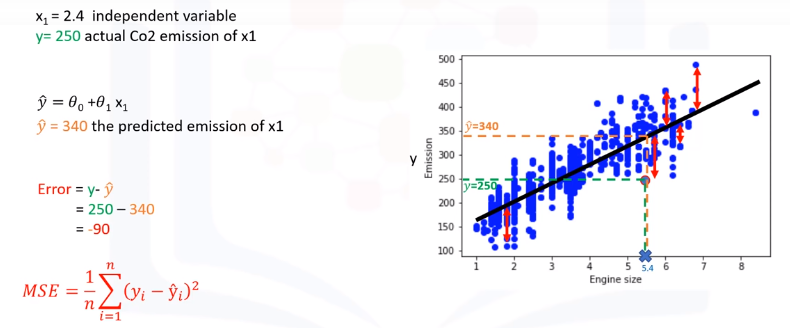


1. The dependent variable can be seen as the "state", "target" or "final goal" we study
2. and try to predict, and the independent variables, also known as explanatory variables, can be
3. seen as the "causes" of those "states".
4. The independent variables are shown conventionally by x; and the dependent variable is notated by y.
5. A regression model relates y, or the dependent variable, to a function of x, i.e., the independent
6. variables.
7. The key point in the regression is that our dependent value should be continuous, and
8. cannot be a discreet value.
9. However, the independent variable or variables can be measured on either a categorical or
10. continuous measurement scale.
11. Basically there are 2 types of regression models: simple regression and multiple regression.
12. Simple regression is when one independent variable is used to estimate a dependent variable.
13. It can be either linear on non-linear.
14. For example, predicting Co2emission using the variable of EngineSize.
15. Linearity of regression is based on the nature of relationship between independent and dependent
16. variables.
17. When more than one independent variable is present, the process is called multiple linear
18. **Again, depending on the relation between dependent and independent variables, it can be either**
19. linear or non-linear regression.
20. egression.



SIMPLE LINEAR REGRESSION:

1. The key point in the linear regression is that our dependent value should be continuous
2. and cannot be a discreet value. However, the independent variable(s) can be
3. **measured on either a categorical or continuous measurement scale.**

****

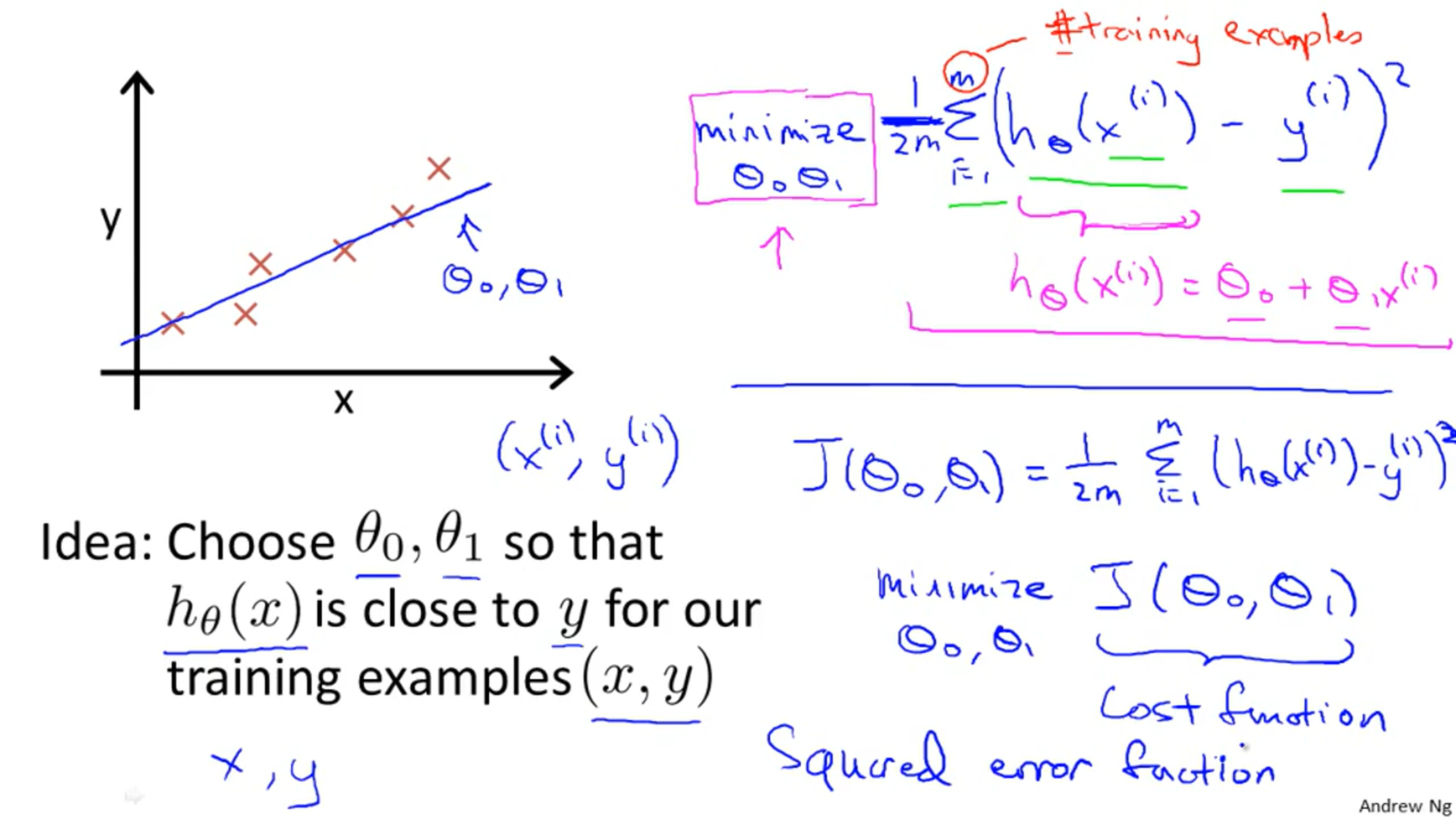
MULTIPLE LINEAR REGRESSION :

We can’t add categorical vehicle to an equation. So wherever we get a categorical variable, we need to create dummy variables.

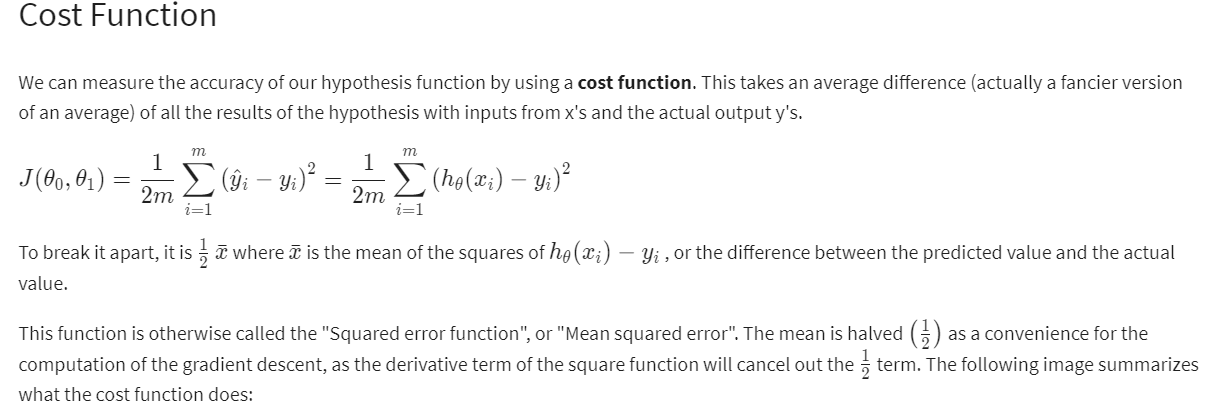
First you need to go through column and find all the different categories you have. So for every single category that you follow, you need to create a new column.

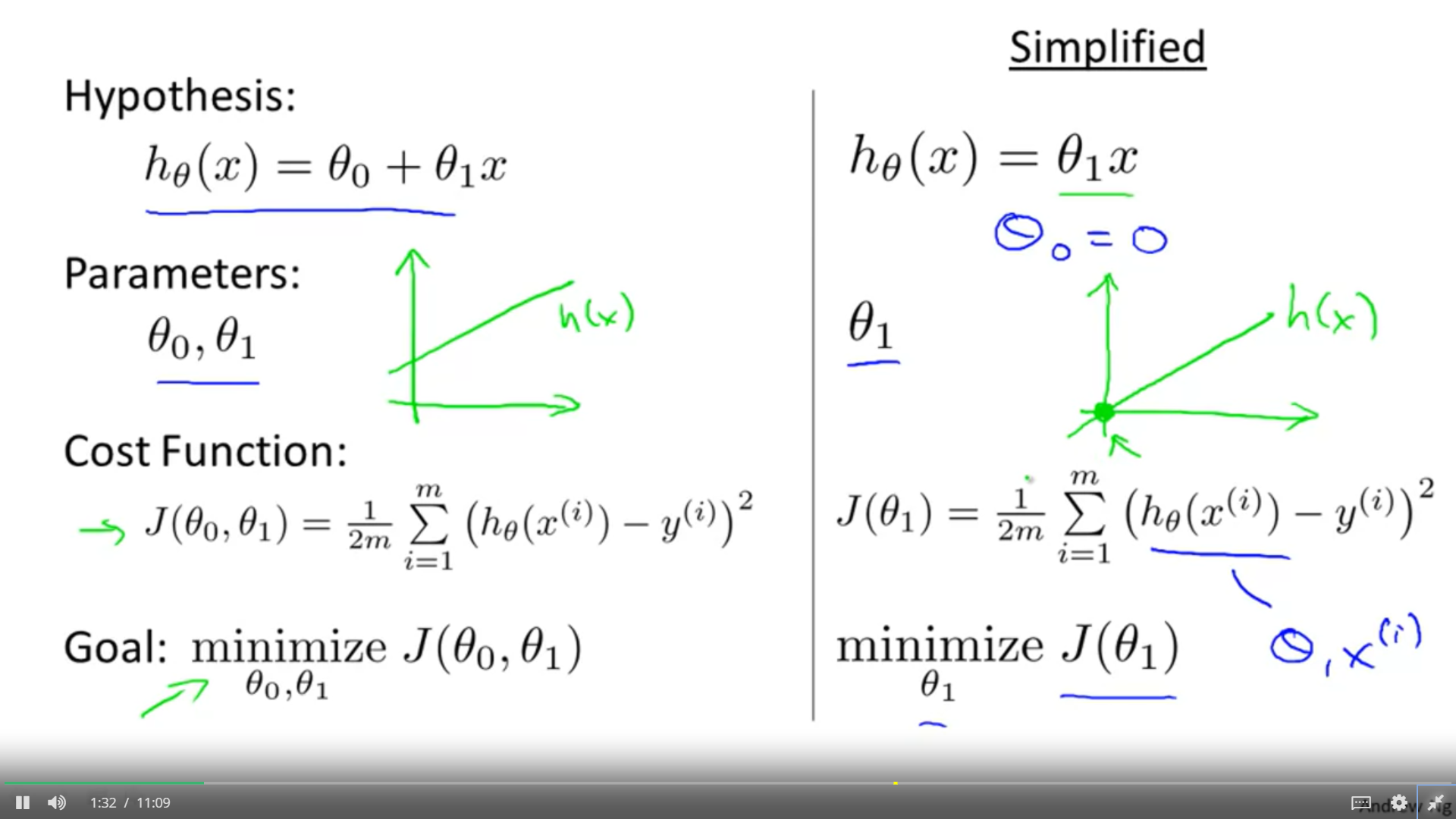


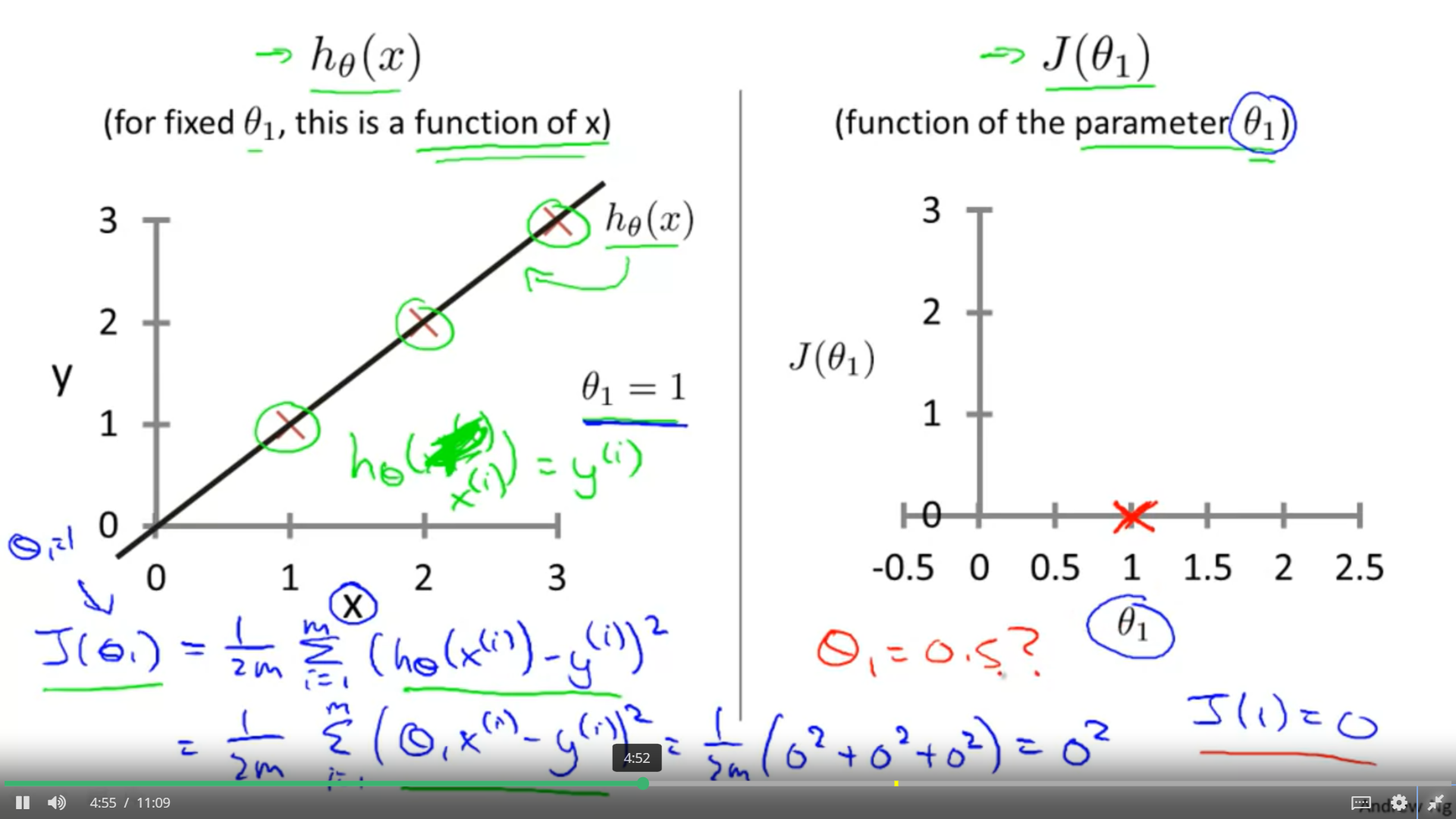


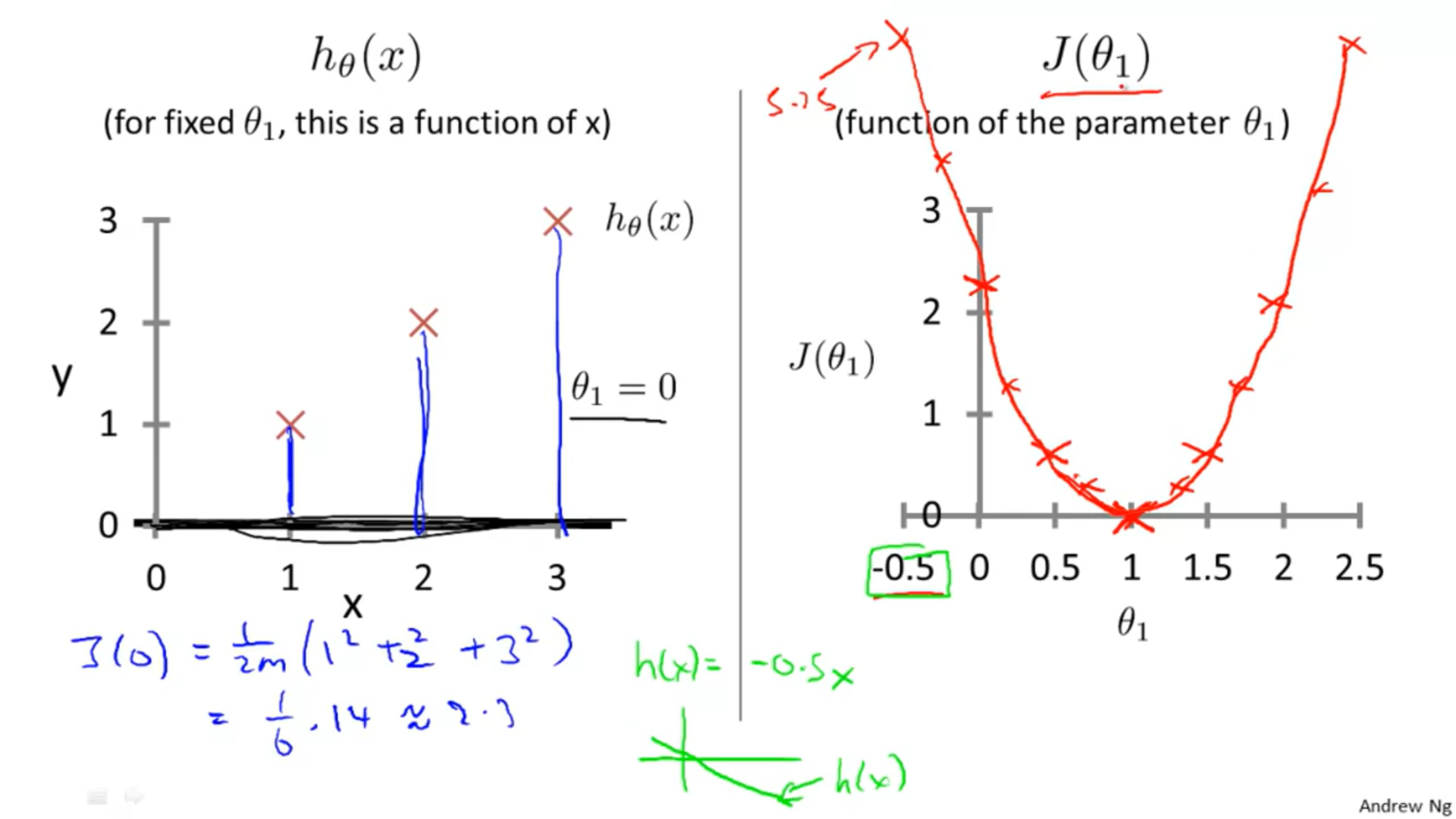


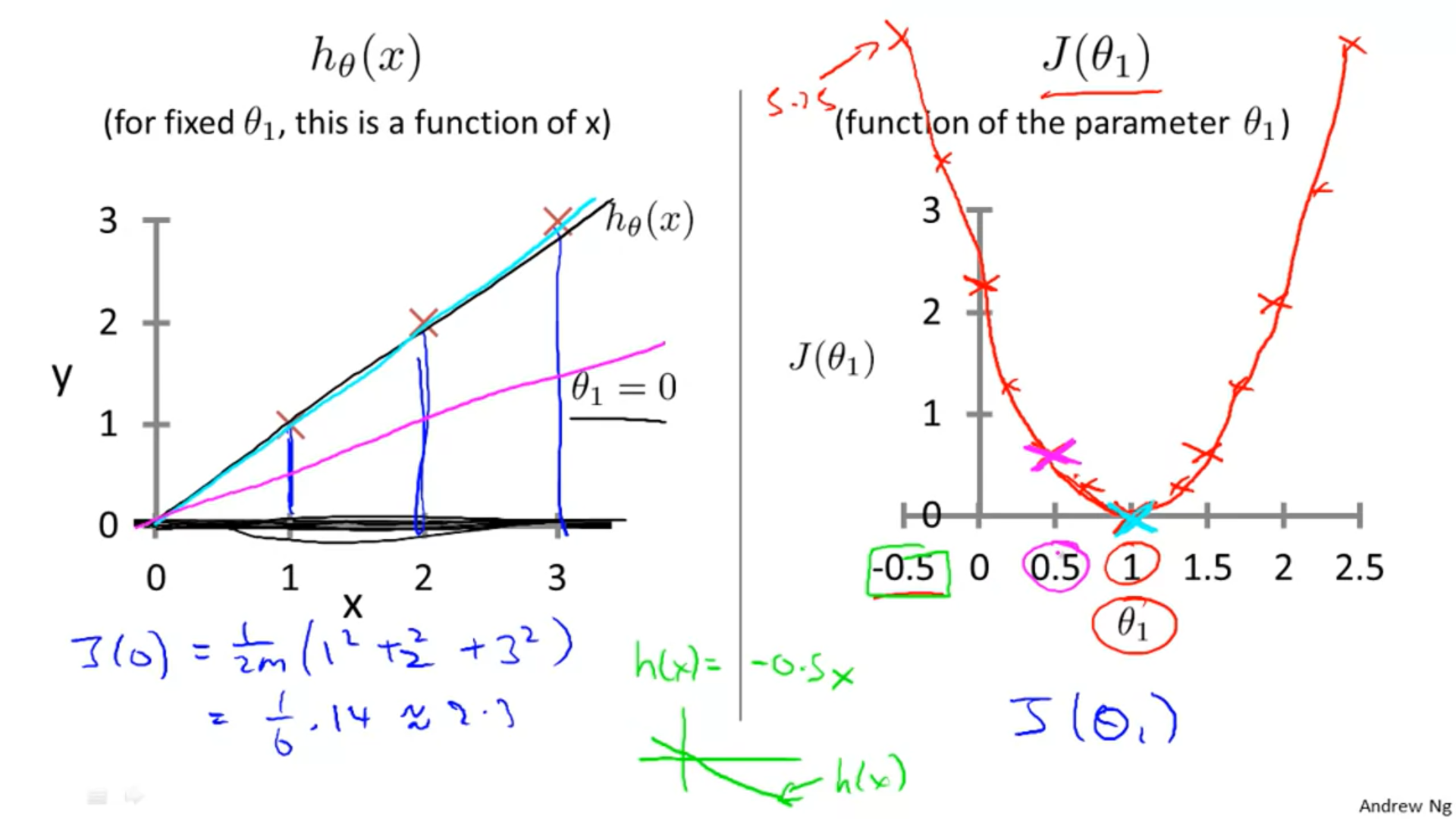
The square of the difference of predicted vs actual y value should be minimum to find the values of theta zero and theta one.







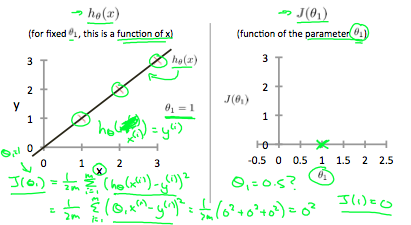




Cost Function - Intuition I

If we try to think of it in visual terms, our training data set is scattered on the x-y plane. We are trying to make a straight line (defined by h\_\theta(x)*hθ*​(*x*)) which passes through these scattered data points.

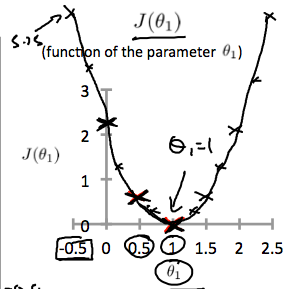
Our objective is to get the best possible line. The best possible line will be such so that the average squared vertical distances of the scattered points from the line will be the least. Ideally, the line should pass through all the points of our training data set. In such a case, the value of J(\theta\_0, \theta\_1)*J*(*θ*0​,*θ*1​) will be 0. The following example shows the ideal situation where we have a cost function of 0.



When \theta\_1 = 1*θ*1​=1, we get a slope of 1 which goes through every single data point in our model. Conversely, when \theta\_1 = 0.5*θ*1​=0.5, we see the vertical distance from our fit to the data points increase.



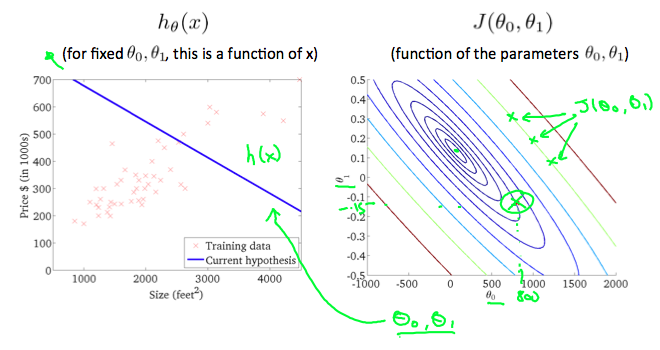
This increases our cost function to 0.58. Plotting several other points yields to the following graph:



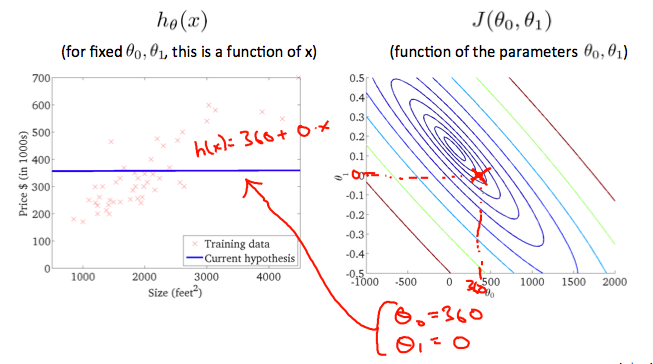
Thus as a goal, we should try to minimize the cost function. In this case, \theta\_1 = 1*θ*1​=1 is our global minimum.

Cost Function - Intuition II

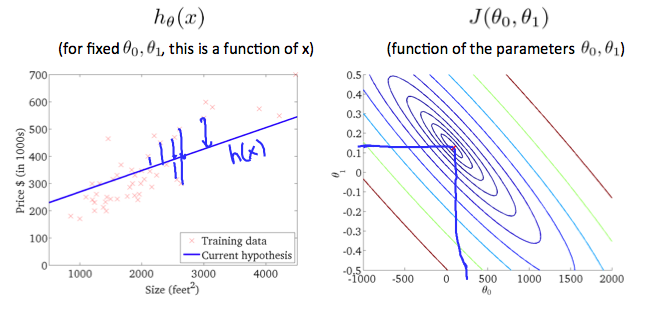
A contour plot is a graph that contains many contour lines. A contour line of a two variable function has a constant value at all points of the same line. An example of such a graph is the one to the right below.



Taking any color and going along the 'circle', one would expect to get the same value of the cost function. For example, the three green points found on the green line above have the same value for J(\theta\_0,\theta\_1)*J*(*θ*0​,*θ*1​) and as a result, they are found along the same line. The circled x displays the value of the cost function for the graph on the left when \theta\_0*θ*0​ = 800 and \theta\_1*θ*1​= -0.15. Taking another h(x) and plotting its contour plot, one gets the following graphs:



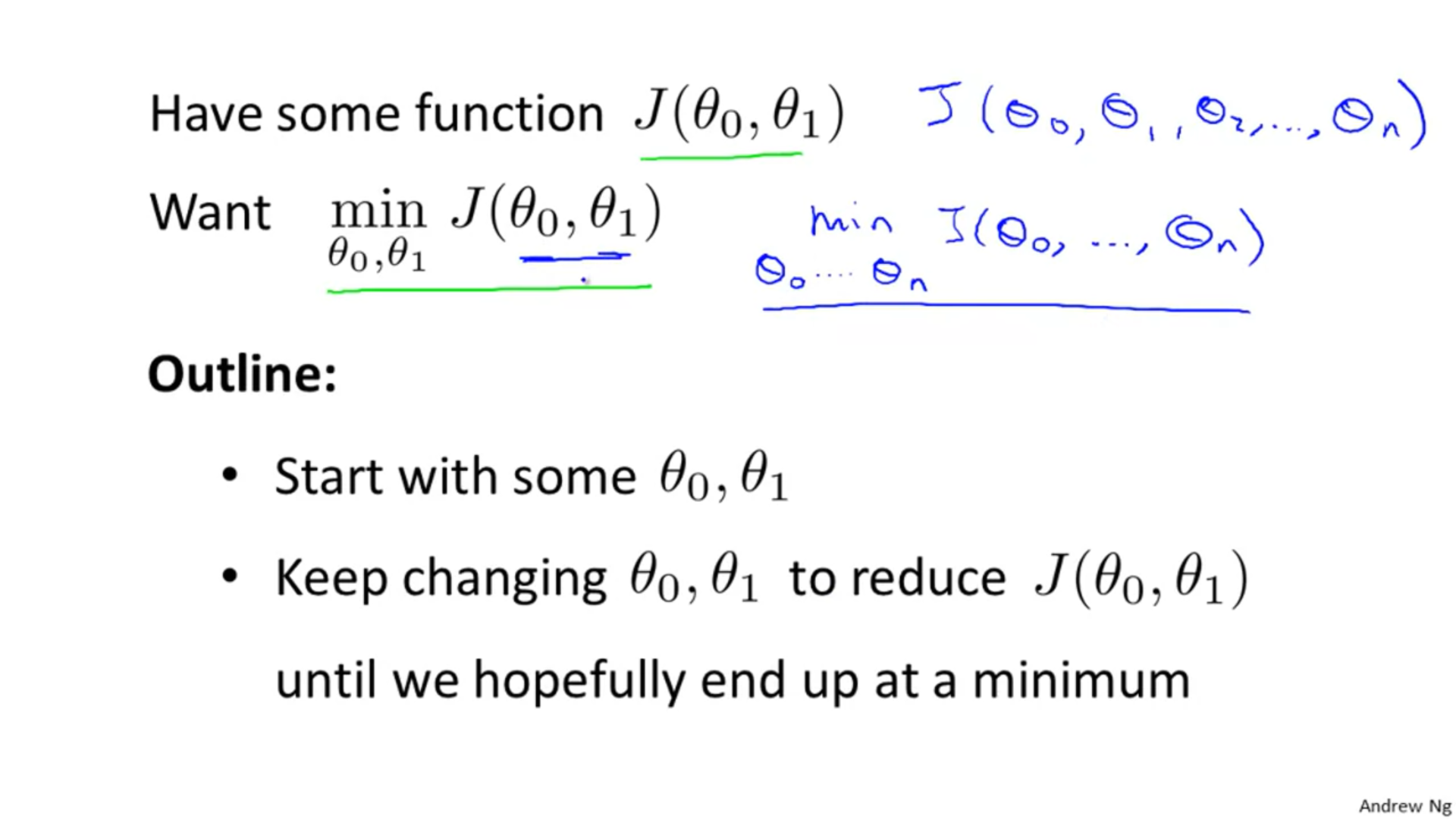
When \theta\_0*θ*0​ = 360 and \theta\_1*θ*1​ = 0, the value of J(\theta\_0,\theta\_1)*J*(*θ*0​,*θ*1​) in the contour plot gets closer to the center thus reducing the cost function error. Now giving our hypothesis function a slightly positive slope results in a better fit of the data.



The graph above minimizes the cost function as much as possible and consequently, the result of \theta\_1*θ*1​ and \theta\_0*θ*0​ tend to be around 0.12 and 250 respectively. Plotting those values on our graph to the right seems to put our point in the center of the inner most 'circle'.

In future lecs we will see videos about an algorithm for automatically finding that value of theta zero and theta one that minimizes the cost function J.

It turns out gradient descent is a more general algorithm, and is used not only in linear regression. It's actually used all over the place in machine learning.



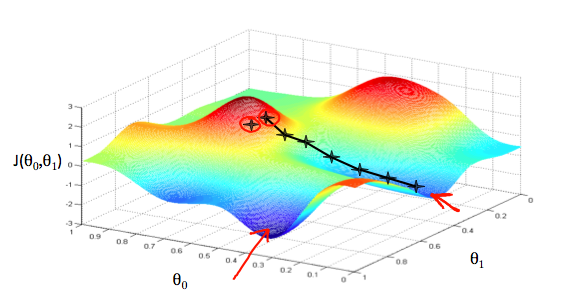
## Gradient Descent

# Gradient Descent

So we have our hypothesis function and we have a way of measuring how well it fits into the data. Now we need to estimate the parameters in the hypothesis function. That's where gradient descent comes in.

Imagine that we graph our hypothesis function based on its fields \theta\_0*θ*0​ and \theta\_1*θ*1​ (actually we are graphing the cost function as a function of the parameter estimates). We are not graphing x and y itself, but the parameter range of our hypothesis function and the cost resulting from selecting a particular set of parameters.

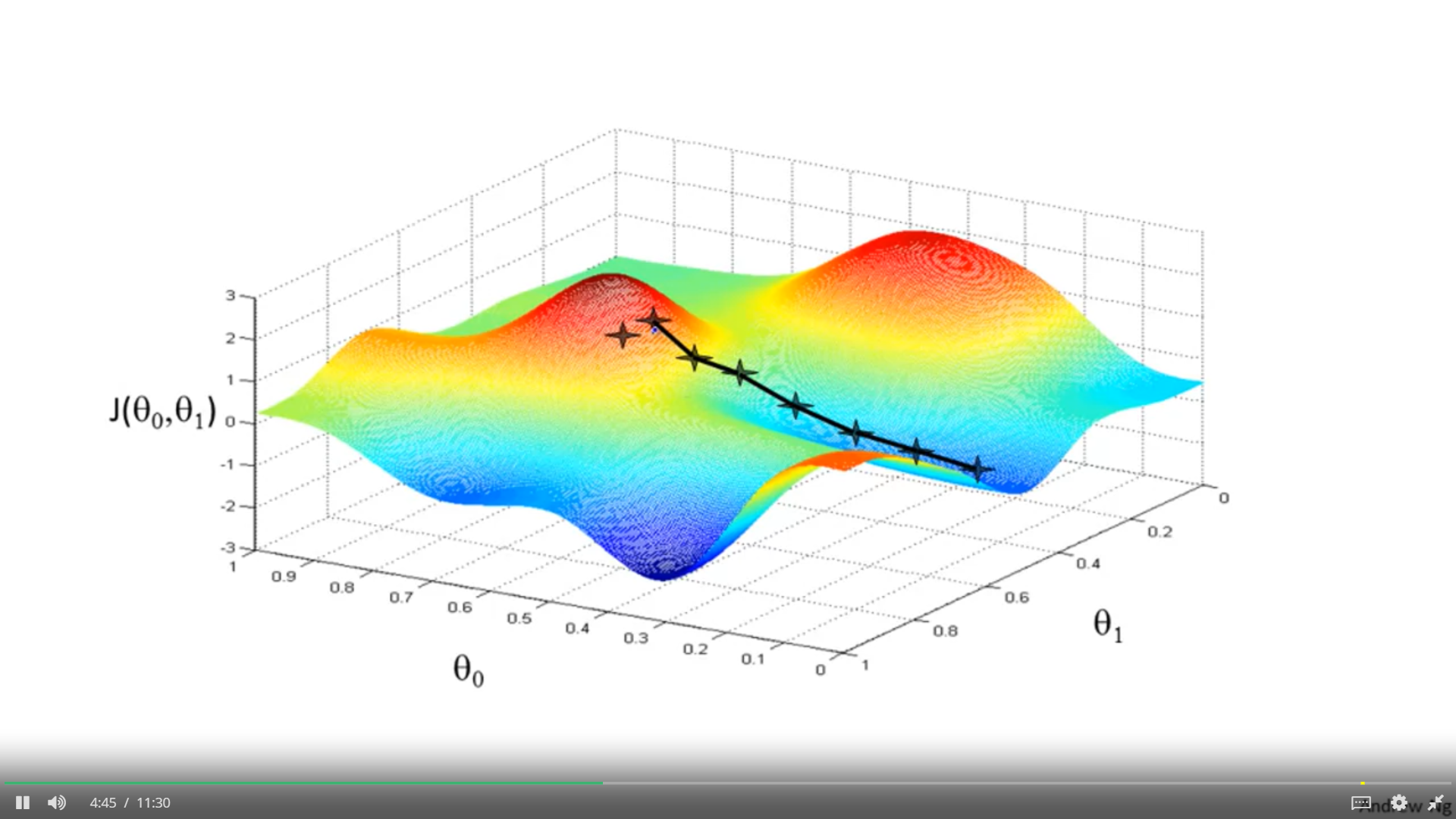
We put \theta\_0*θ*0​ on the x axis and \theta\_1*θ*1​ on the y axis, with the cost function on the vertical z axis. The points on our graph will be the result of the cost function using our hypothesis with those specific theta parameters. The graph below depicts such a setup.



We will know that we have succeeded when our cost function is at the very bottom of the pits in our graph, i.e. when its value is the minimum. The red arrows show the minimum points in the graph.

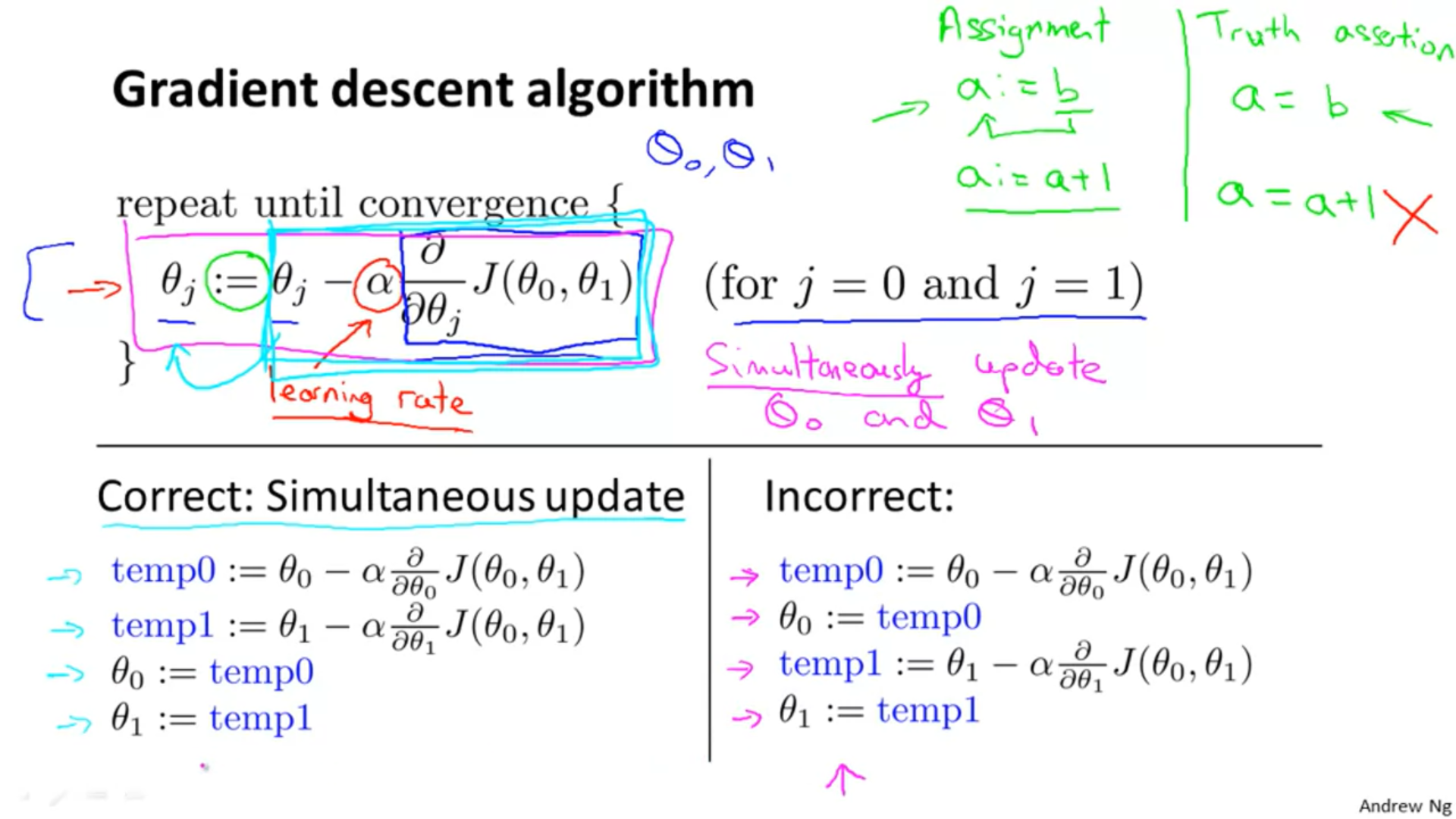
The way we do this is by taking the derivative (the tangential line to a function) of our cost function. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α, which is called the learning rate.

For example, the distance between each 'star' in the graph above represents a step determined by our parameter α. A smaller α would result in a smaller step and a larger α results in a larger step. The direction in which the step is taken is determined by the partial derivative of J(\theta\_0,\theta\_1)*J*(*θ*0​,*θ*1​). Depending on where one starts on the graph, one could end up at different points. The image above shows us two different starting points that end up in two different places.



In gradient descent we are going to update theta 0 and theta 1.

And the subtlety of how you implement gradient descent is for this expression, for this update equation, you want to simultaneously update theta 0 and theta 1.



It turns out that the way gradient descent is usually implemented, which I'll say more about later, it actually turns out to be more natural to implement the simultaneous updates. And when people talk about gradient descent, they always mean simultaneous update. If you implement the non simultaneous update, it turns out it will probably work anyway. But this algorithm wasn't right. It's not what people refer to as gradient descent, and this is some other algorithm with different properties.

One of the issues we saw with gradient descent is that it can be susceptible to local optima. So when I first explained gradient descent I showed you this picture of it going downhill on the surface, and we saw how depending on where you initialize it, you can end up at different local optima. You will either wind up here or here.

But, it turns out that that the cost function for linear regression is always going to be a bow shaped function like this. The technical term for this is that this is called a convex function.

And I'm not gonna give the formal definition for what is a convex function, C, O, N, V, E, X. But informally a convex function means a bowl shaped function and so this function doesn't have any local optima except for the one global optimum. And does gradient descent on this type of cost function which you get whenever you're using linear regression it will always converge to the global optimum. Because there are no other local optimum, global optimum. So now let's see this algorithm in action.

Finally, by convention, usually

when writing matrices and vectors,

most people will use upper

case to refer to matrices.

So we're going to use

capital letters like

A, B, C, you know,

X, to refer to matrices,

Play video starting at :8:16 and follow transcript8:16

and usually we'll use lowercase,

like a, b, x, y,

Play video starting at :8:21 and follow transcript8:21

to refer to either numbers,

or just raw numbers or scalars or to vectors.

This isn't always true but

this is the more common

notation where we use

lower case "Y" for referring

to vector and we usually

use upper case to refer to a matrix.