Machine Learning

Week 1: Introduction, Linear Regression, and Linear Algebra

23-12-2021

Introduction

Machine learning is the science of getting computers to learn, without being explicitly programmed.

Machine learning

- grew out of work in AI
- New capability for computers

Examples:

- Database mining
  - Large datasets from growth of automation/web
  - Eg: web click data, medical records, biology, engineering
- Applications can’t program by hand
  - Eg: autonomous helicopter, handwriting recognition, most of Natural Language Processing (NLP), computer vision
- Self customizing programs
  - Eg: amazon, netfilx product recommendations
- Understanding human learning (brain, real AI)

What is Machine Learning?

- Arthur Samuel (1959): Field of study that gives computers the ability to learn without being explicitly programmed.
- Tom Mitchell (1998): Well-posed Learning Problem: 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.
  - Eg: playing chess
    - E – the experience of playing many games of chess
    - T – the task of playing chess
    - P – the probability that the program will win the next game

Machine learning algorithms:

- Supervised learning
- Unsupervised learning

Others: reinforcement learning, recommender systems

Also talk about: practical advice for applying learning algorithm

Supervised Learning

In supervised learning, we are given a data set and already know what our correct output should look like, having the idea that there is a relationship between the input and the output.

Regression: predict continuous valued output

Classification: discrete valued output

Unsupervised Learning

Cocktail party problem algorithm: allows to find structure in a chotic environment

```
[W,s,v] = svd((repmat(sum(x.*x,1),size(x,1),1).*x)*x');
```

Unsupervised learning allows us to approach problems with little or no idea what our results should look like. We can derive structure from data where we don't necessarily know the effect of the variables.

We can derive this structure by clustering the data based on relationships among the variables in the data.

There is no feedback based on the prediction results.

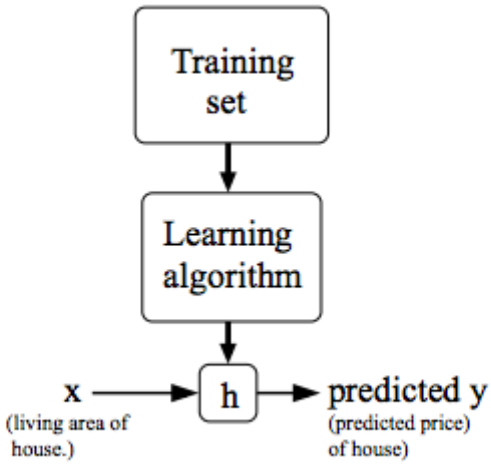
Clustering

Non-clustering

24-12-2021

Linear Regression with One Variable

Model Representation



How do we represent **h** (hypothesis)?: $h(x)$ or $h_\theta(x) = \theta_0 + \theta_1 x$

Linear regression with one variable or Univariant linear regression model is a straight line best-fit for a two variable dataset.

A pair $(x^{(i)}, y^{(i)})$ is called a training example, and the dataset that we'll be using to learn – a list of m training examples $(x^{(i)}, y^{(i)})$; i = 1, . . . , m – is called a training set.

Cost Function

We can measure the accuracy of our hypothesis function by using a cost function. This function is also called Squared error function or Mean squared error.

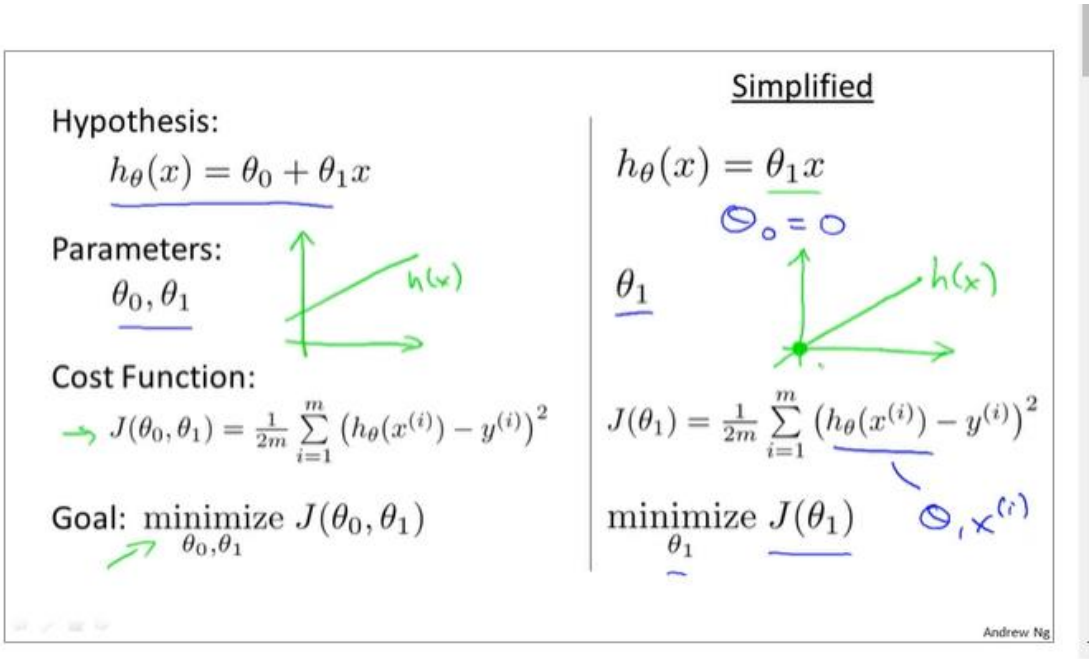
Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

For different θ_0 and θ_1 the hypothesis function changes.

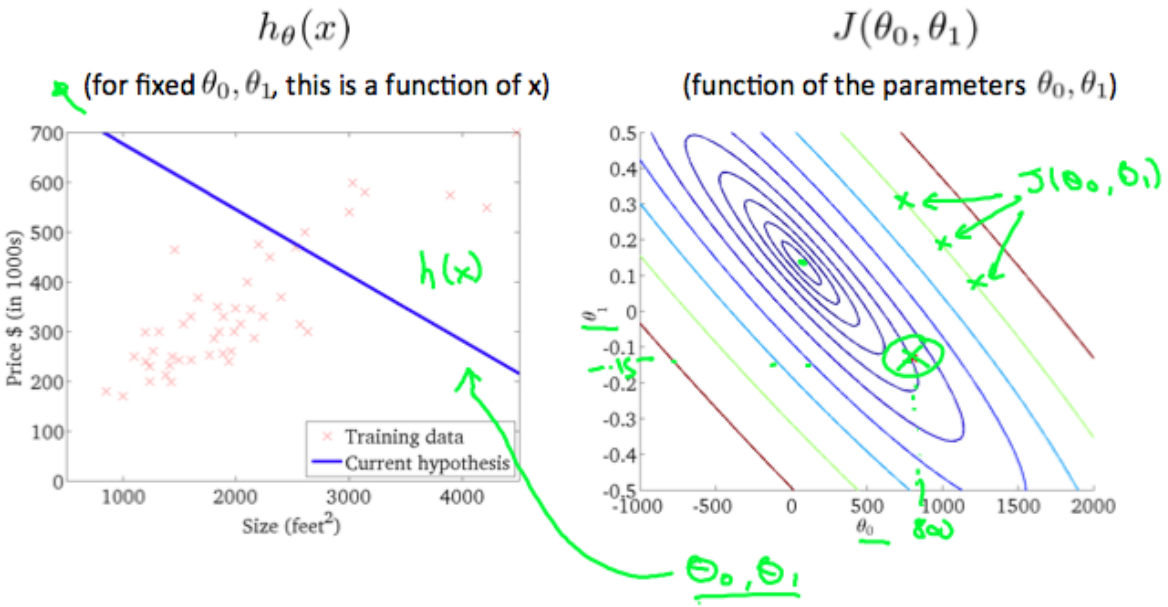
In linear regression we want to find values of θ_0, θ_1 such that the straight line formed is the best fit for the data set.

We would like to minimize the difference between output of the hypothesis and the actual value observed or got.

Minimize error over θ_0, θ_1 = $\frac{1}{2m}\sum_{i=1}^{m}(h_\Theta(x_i) - y_i)^2$

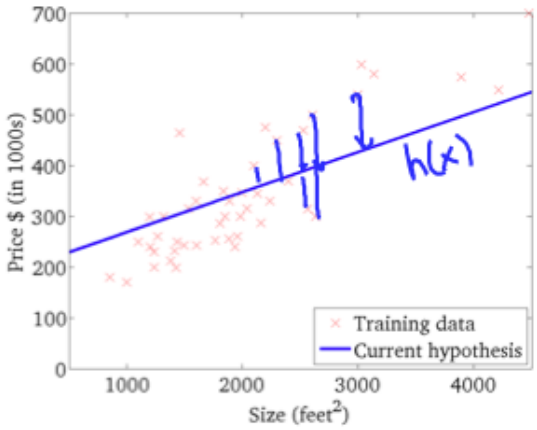


Contour plot: is a graph that contains many contour lines. A contour line of a two variable function has a constant value at all points of the same line.



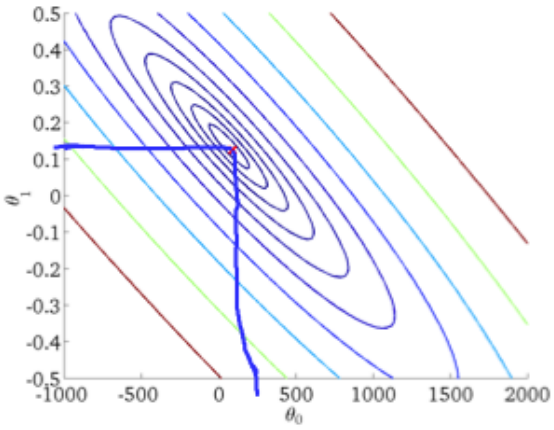
$h_\theta(x)$

(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)

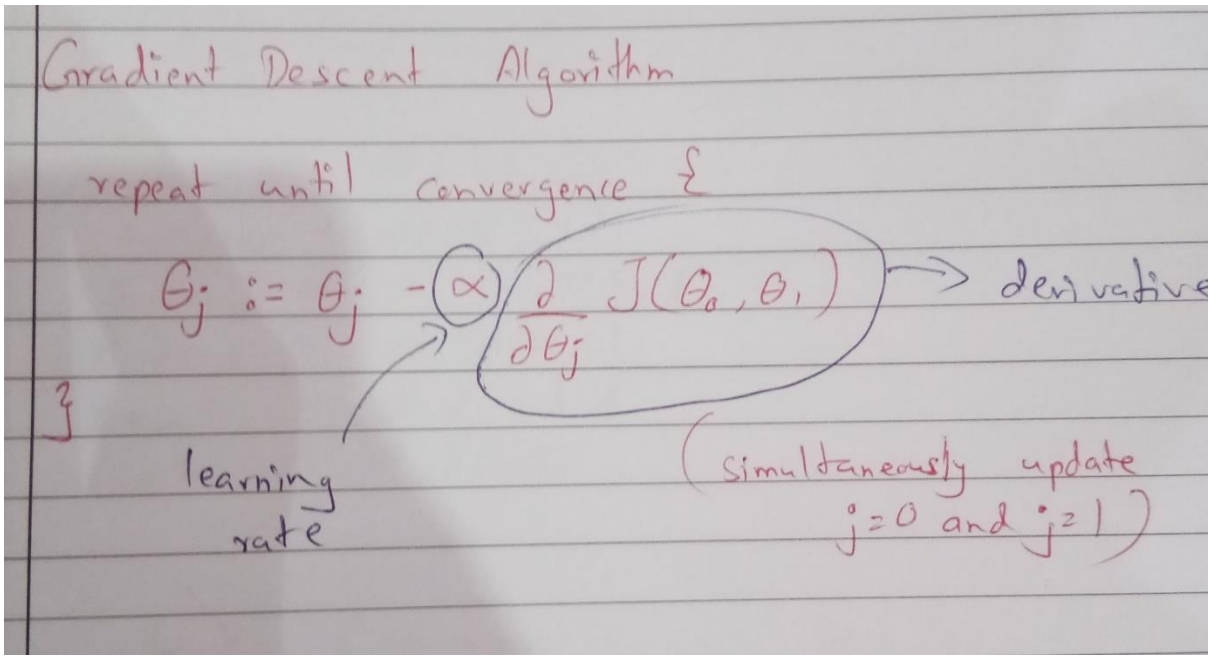


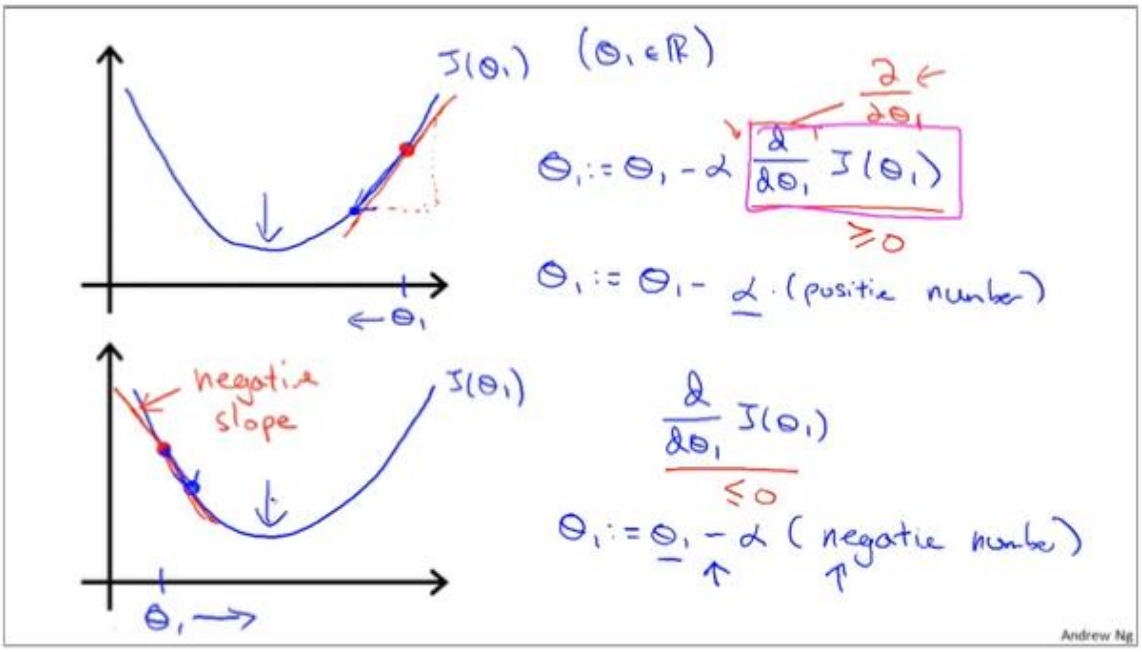
Gradient Descent

For minimizing cost function.

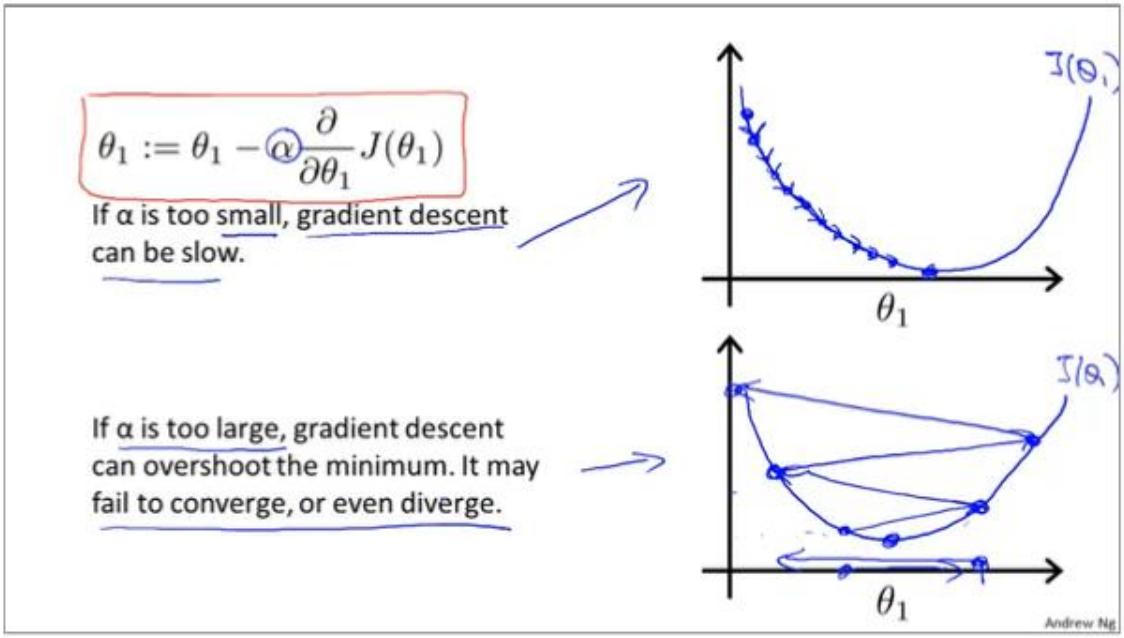
17-01-2022

Gradient Descent Intuition



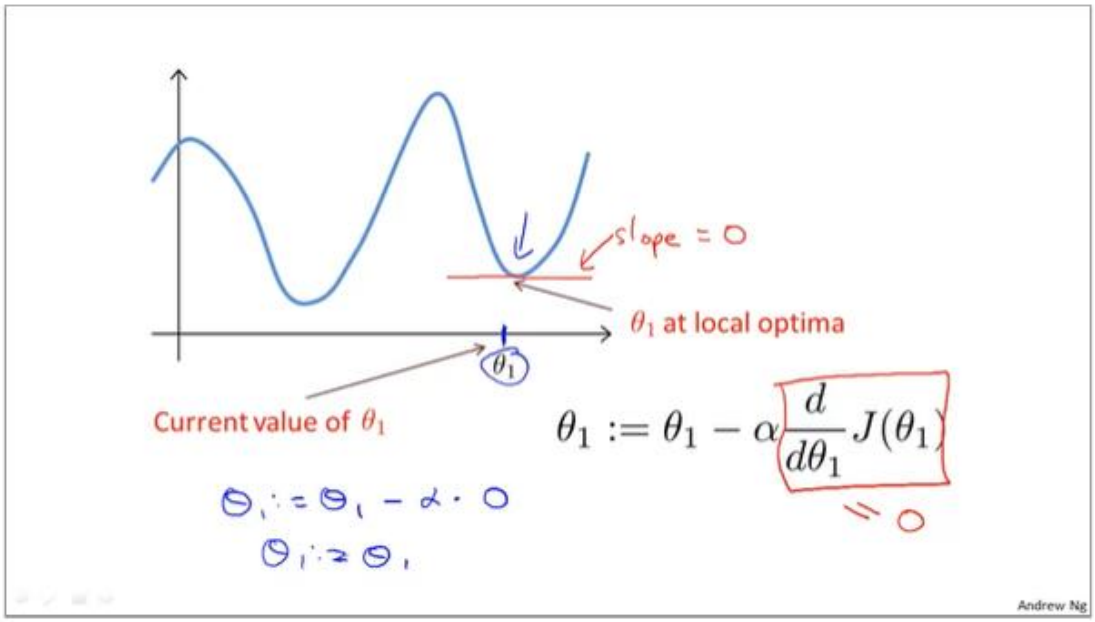


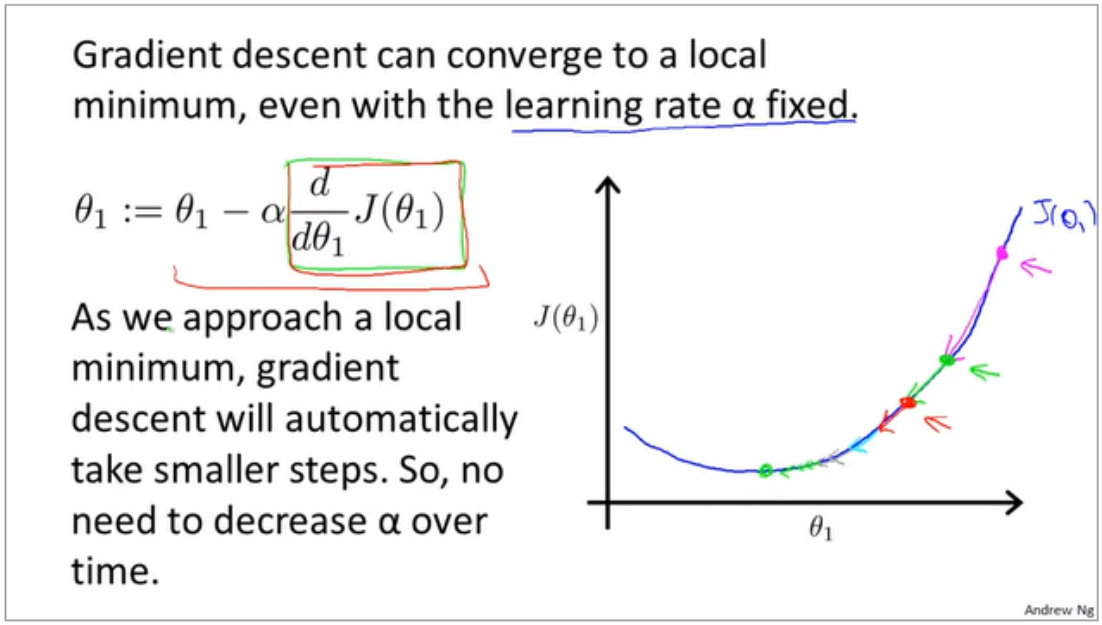
So we can see that the derivative term is shifting our $J(\theta_1)$ term closer to the minimum.



So we see that alpha/learning rate defines how fast we move towards the minimum which if too large can cause completely missing the minimum and shifting forward or backward from it or if too small can be very slow.

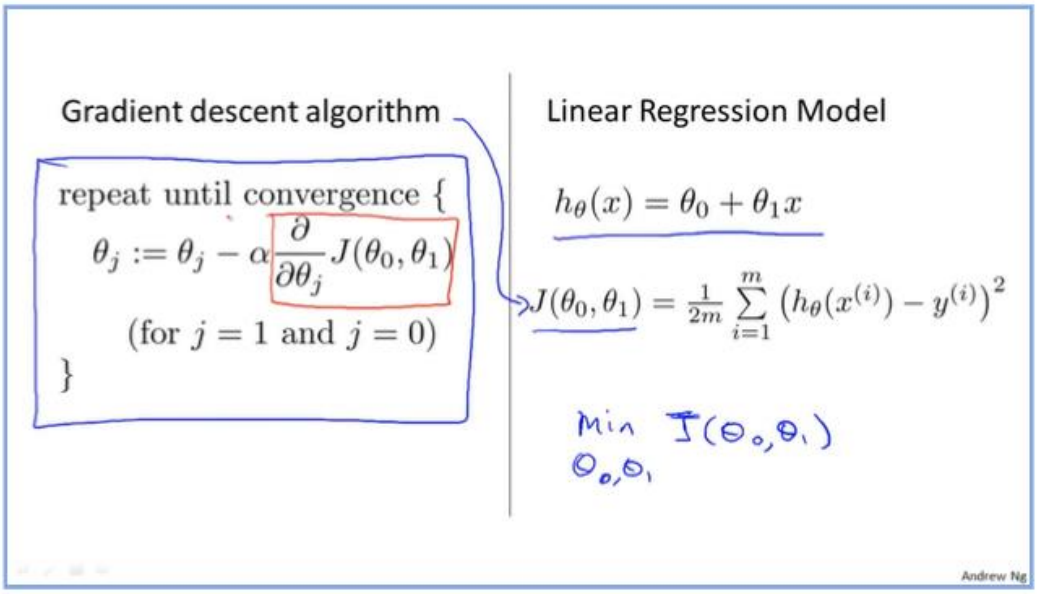
Since derivative for a local minimum will be zero so if we are at a local optima then the value of θ_1 remains unchanged.



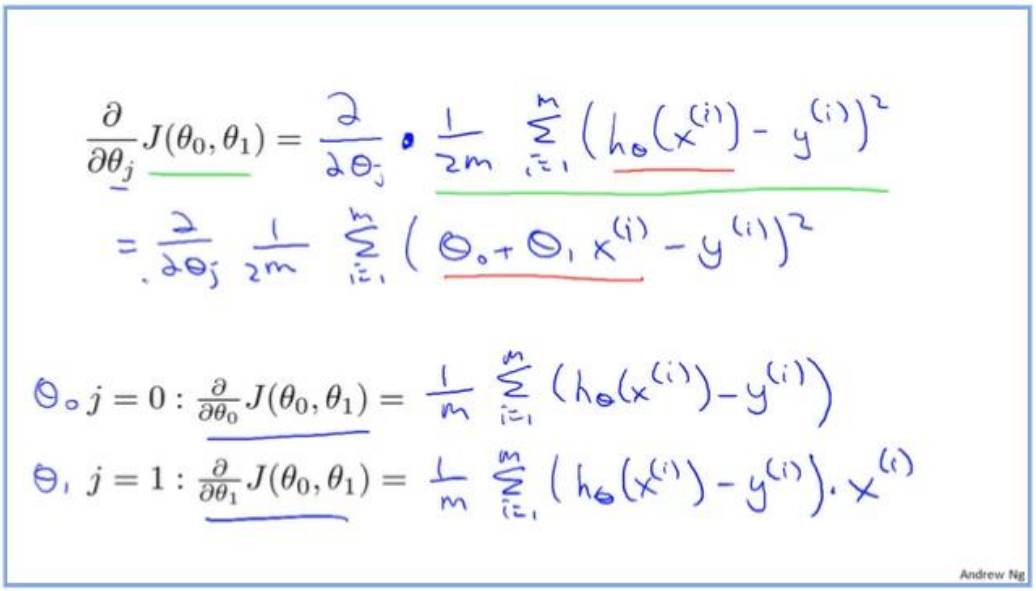


So even if we fix the learning rate the derivative will gradually tend to zero and thus we will approach the local minimum.

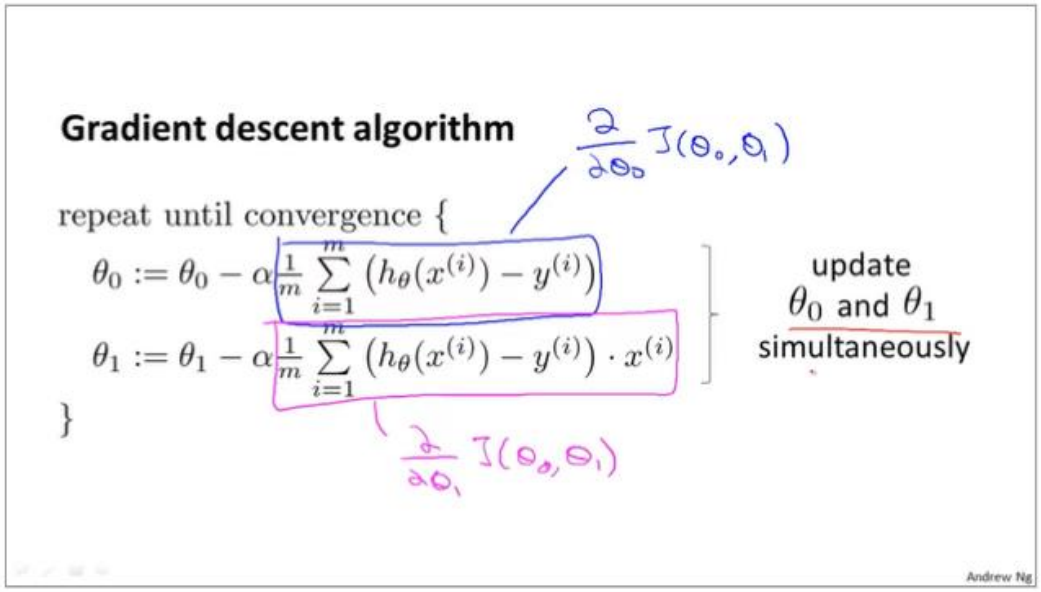
Gradient Descent for Linear Regression



Finding the derivative:



So our gradient descent algorithm converts to:



"Batch" Gradient Descent: Each step of gradient descent uses all the training examples.

Linear Algebra Review

Matrices and Vectors

Matrix: Rectangular array of numbers

Dimension: number of rows * number of columns

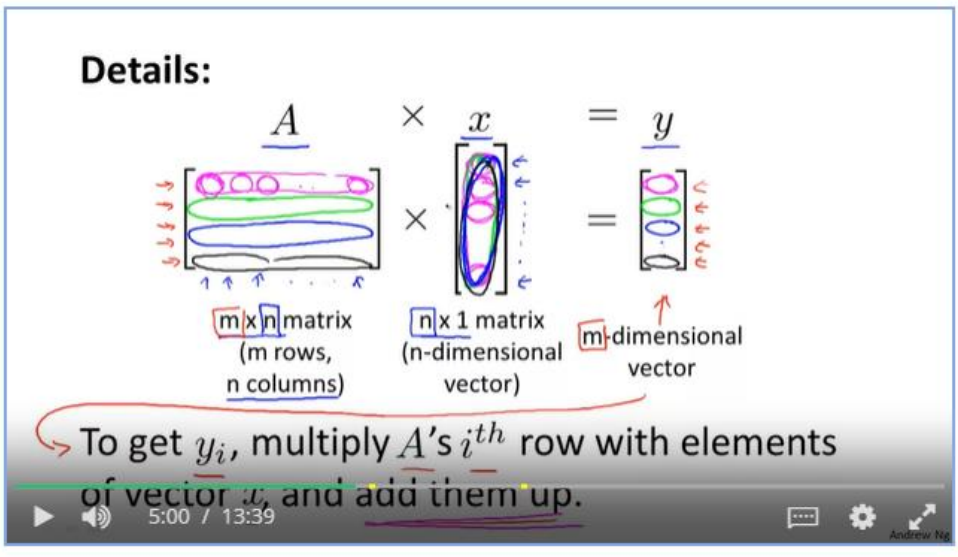
For a matrix A, A_{ij} = "i,j th entery" in the i^{th} row and j^{th} column

Vector: An n*1 matrix

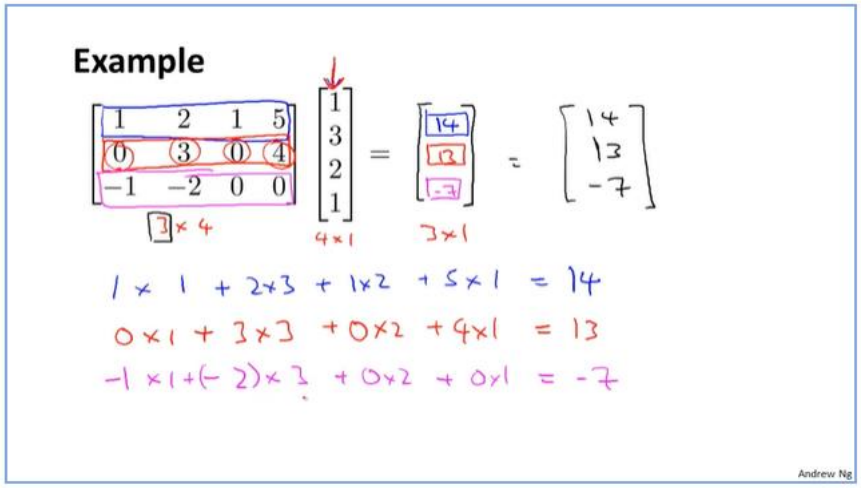
Indexing of vector elements can start from both 1 and 0.

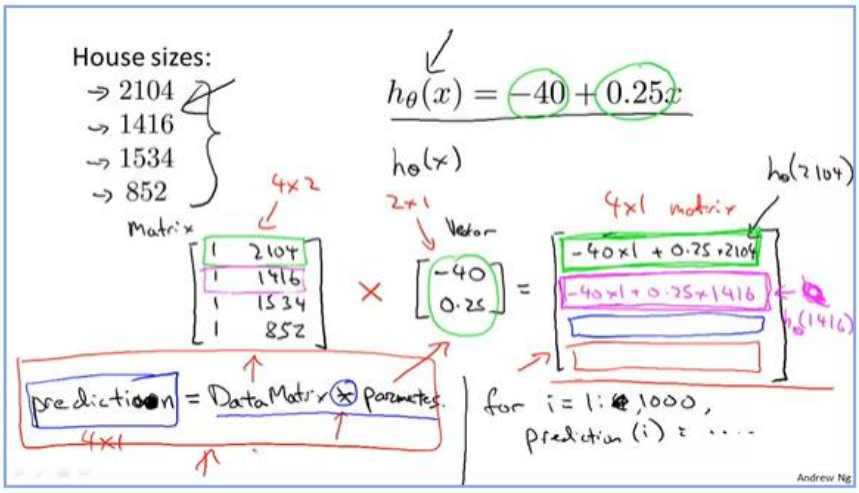
In linear algebra 1 indexed notation is more commonly used and in ML applications 0 indexed.

- Matrix addition works by adding the corresponding elements, given that both the matrices are of same order/dimensions.
- Multiplying a matrix by scalar results in a matrix with all the elements multiplied by the scalar. The dimensions of the matrix does not changes.
- Multiplication of matrix by a vector
  - Multiplication of a matrix by a vector will result in a vector.
  - Multiplying a matrix A with vector V of dimensions:
    - A : n rows * m columns
    - V : m rows * 1 column
    - Resultant matrix: n rows * 1 column
  - No of columns of matrix should be equal to no of rows in the vector
  - Multiplication works by multiplying a row to the vector with corresponding elements multiplying and then their sum is the corresponding result for that element position in resultant vector

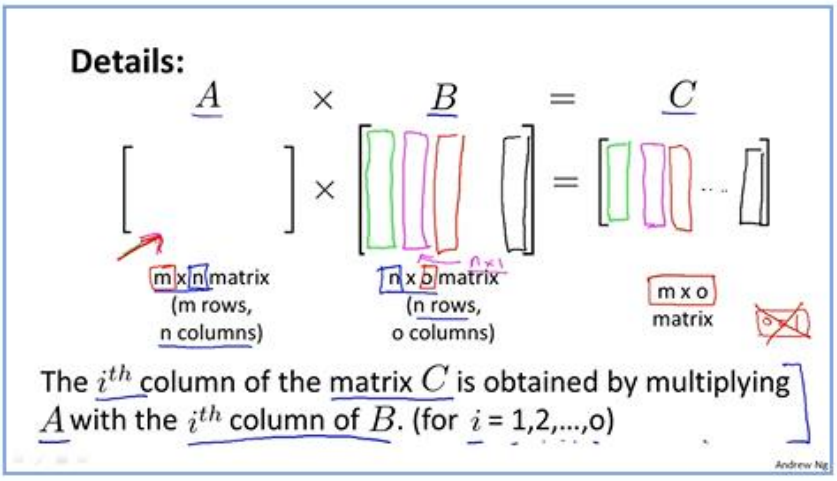


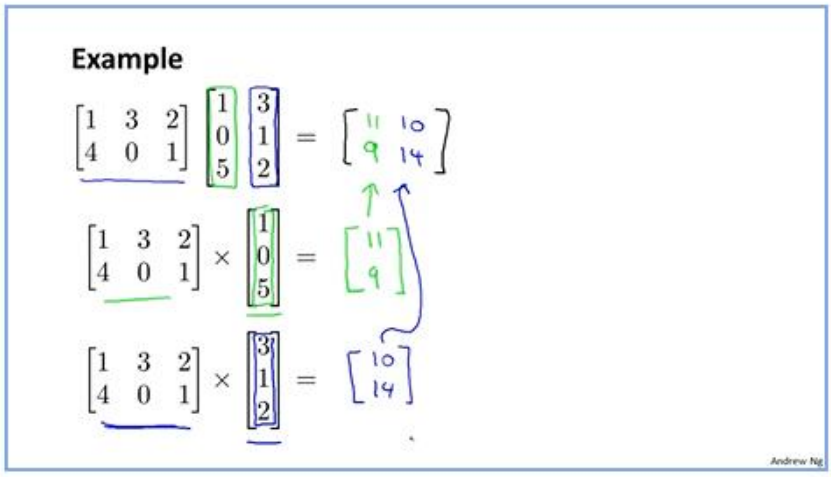
  -

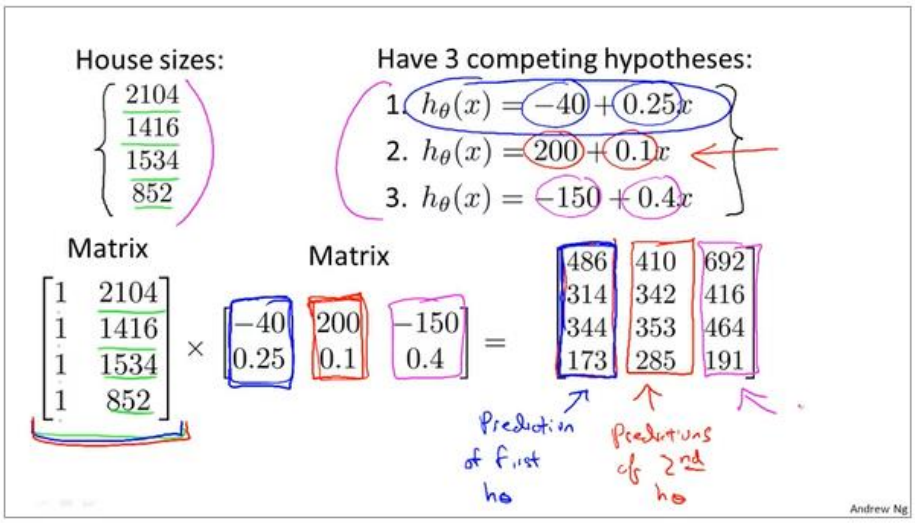




- Matrix-Matrix Multiplication



- 

- 

- Matrix multiplication properties
  - Non-coummtative A x B != B x A
  - Associative A x (B x C) = (A x B) x C
  - Identity Matrix I or I_{nxn} has 1's along the diagonals and all non-diagonal elements are zero.
    - $A_{mxn}.I_{nxn} = I_{mxm}.A_{mxn} = A_{mxn}$
- Inverse and Transpose
  - If A is an m*m matrix, and if it has an inverse, then $A.A^{-1} = A^{-1}.A = I$
  - Matrices that don't have an inverse are "singular" or "degenerate"
  - If A is an n*m matrix, then A^T is the transpose matrix of dimension m*n, where $A^T_{ij} = A_{ji}$