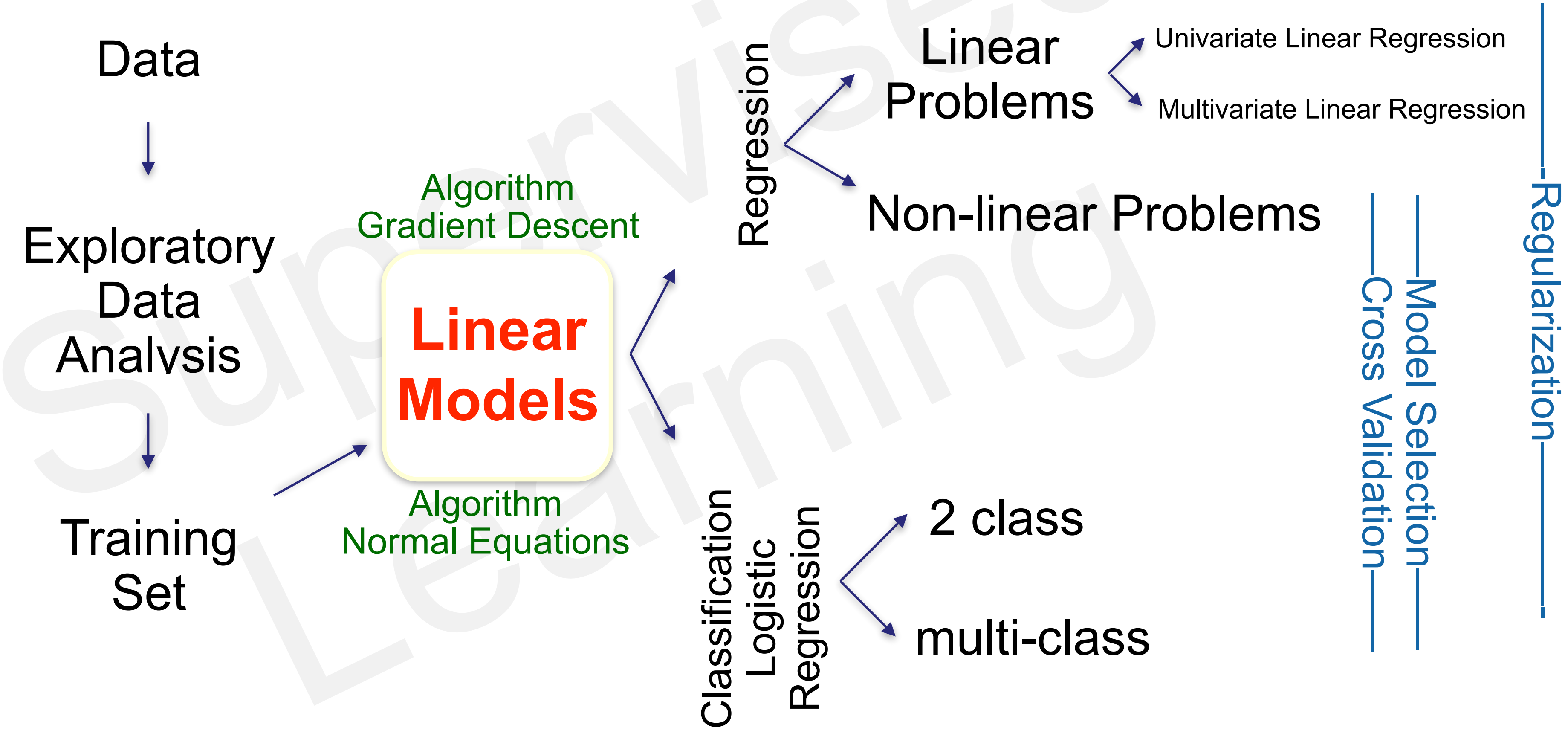
GENERAL ASSEMBLY

INTRO TO DATA SCIENCE
LECTURE 4: LINEAR MODELS & GRADIENT DESCENT

Where are we on the data science road-map?



What do we mean by the term Linear Model?

A model consists of

- input features
  - generally denoted by x
- targets or outputs
  - generally denoted by y
- model parameters
  - generally denoted by theta (θ)

A Linear Model is any model where the output is defined by a linear combination of inputs and model parameters

A simple example of a linear model is the equation for a straight line

$$y = \theta_1 x + \theta_0$$

θ_1 is called the slope of the line

θ_0 is called the intercept

KEY CONCEPTS - LINEAR MODEL EXAMPLE

Given training data how do you “fit” the model - meaning how do you derive values for the parameters of the model?

Once you have the parameters you have an operational model in which, given new values for x, you can predict corresponding values for y

Define 2 functions:

1. The Hypothesis Function, *h(x)*

   The hypothesis function is just your linear model.
   How you transform inputs x, into outputs y
   $y = \theta_1 x + \theta_0$

2. The Cost Function, *J(θ)*

   This tells you how well your model is working to fit the data
   If the parameters are poor then the cost is high
   If the parameters provide a good fit then the cost is low

What is meant by "the cost is high", or "the cost is low"?
What does *$J(\theta)$ actually measure?*

There are many cost functions by the one often used is:

The Sum of Squared Errors

Let's simplify the model even more!

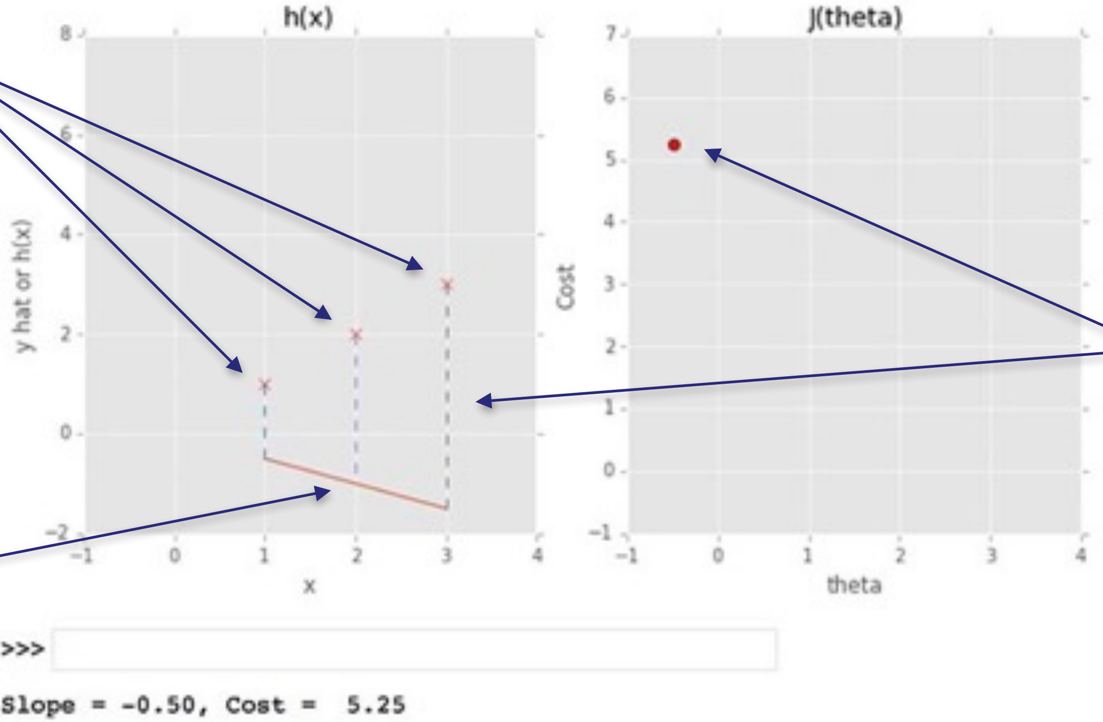
Assume your model will go through the origin, so the intercept is 0

$\theta_0 = 0$

$y = \theta_1 x$

KEY CONCEPTS - LINEAR MODEL EXAMPLE

Training Data

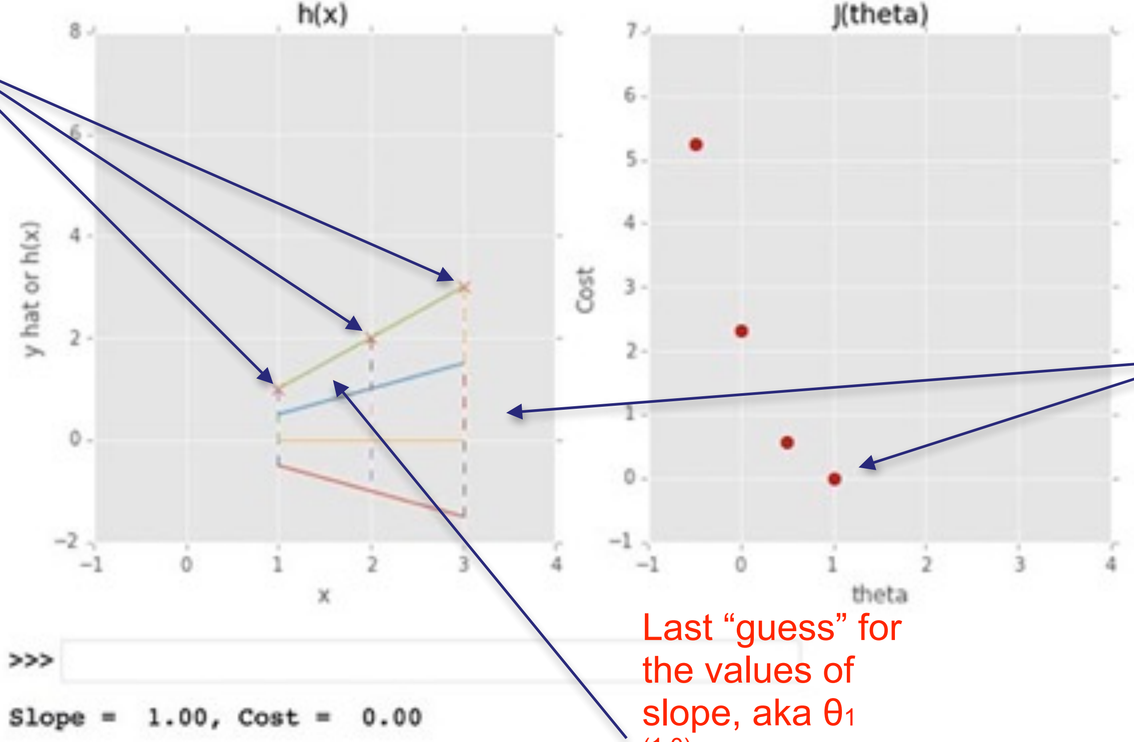


Cost is the differences between what the model predicts and the training data, squared and averaged

First "guess" for the value of the slope, aka θ_1 (-0.5)

KEY CONCEPTS - LINEAR MODEL EXAMPLE

Training Data



When the optimal values for theta are found the cost is minimized. 0 in this case

Last “guess” for the values of slope, aka θ_1 (1.0)

Finding the minimum of the cost function determines values for θ that optimally (in the sum of squared errors sense) model the training set

One Algorithm for finding the minimum of the cost function is gradient descent

Let's now have the full univariate model, slope and y-intercept

The model: $y = \theta_1 x + \theta_0$

The cost function now has two parameters, θ_1 and θ_0: $J(\theta_{0,} \theta_1)$

Algorithm:
Step 1: set $\theta_{0,}$ and θ_1 to initial random values
Step 2: change $\theta_{0,}$ and θ_1 in such a way as to reduce $J(\theta_{0,} \theta_1)$ to its minimum

Gradient descent has "finished" when the cost is at or close to it's minimum

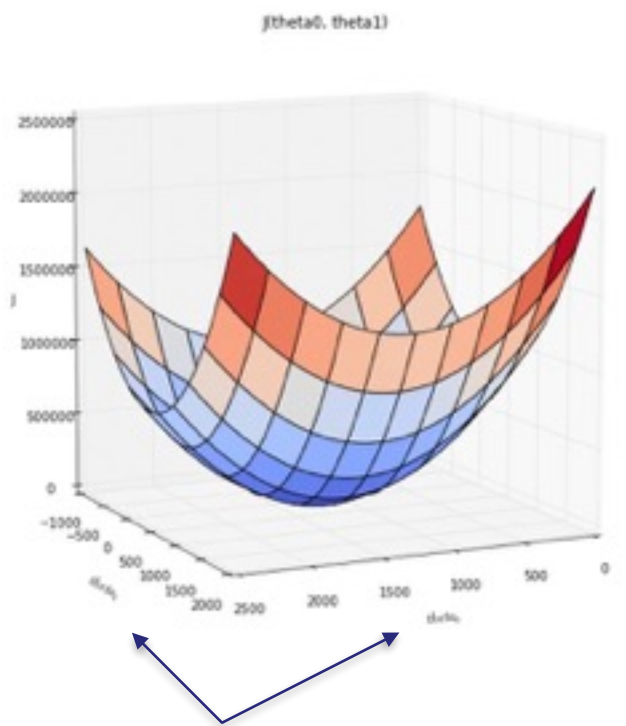
Gradient Descent requires feature scaling

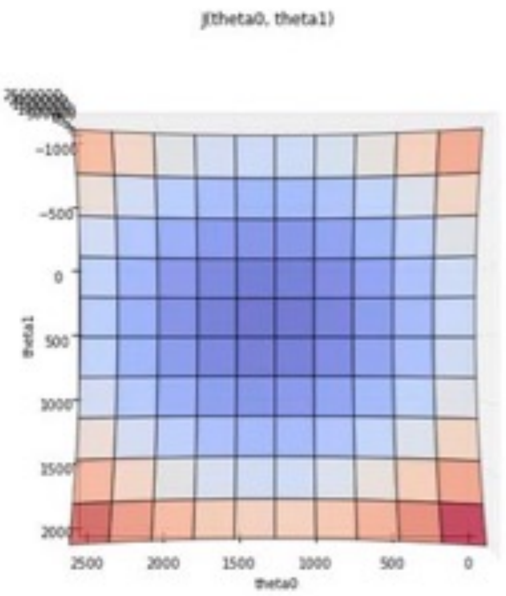
Gradient descent has a single hyper-parameters, called α, alpha. This is also referred to as the learning rate. Alpha alters the amount that $\theta_{0,}$ and θ_1 are changed at each step

Setting the value for alpha affects the performance of the algorithm. Generally 0.01, 0.001 are good starting values

Too small and the algorithm converges slowly, too large and the algorithm may not converge…

KEY CONCEPTS - LINEAR MODEL EXAMPLE





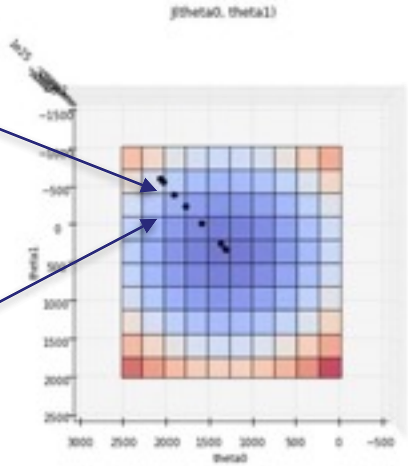
We need to find the values of theta0 AND theta1 so as to minimize the cost

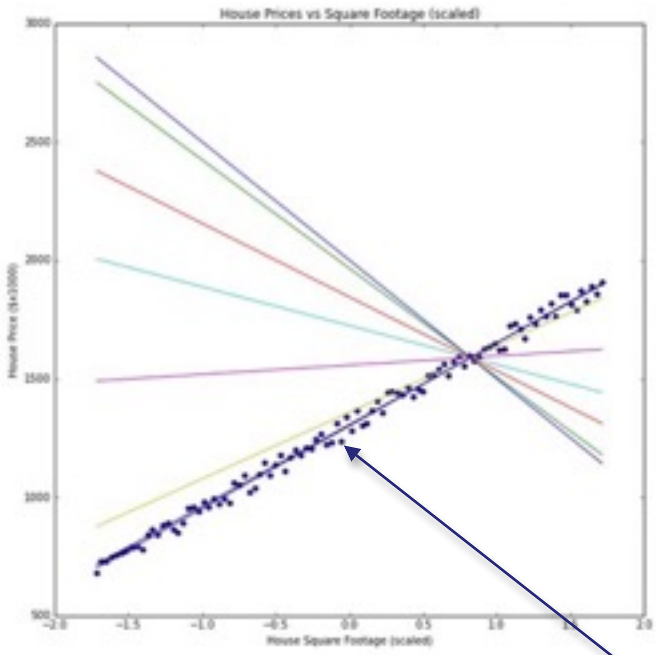
Looking from above 'the bowl”

KEY CONCEPTS - LINEAR MODEL EXAMPLE

Alpha dictates how the bowl is traversed. Must not to be too big.

As gradient descent follow the shape of the bowl towards the minimum, so the model fit to the training data improves





The lines generated by the model parameters as gradient descent works to find more optimal values

Training Data

Input Features have been scaled

Key Concepts - Feature Scaling

- For gradient descent to work optimally all features need to be on a similar scale
- Similarly scaled features make the bowl circular rather than elliptical, so convergence is faster
- As a general rule features should take values between -1 and 1
- As a general rule mean normalization is a good idea
  - mean of the transformed data is zero
- I personally use zero mean, unit standard deviation
  - note however, that this is a non-linear transformation

KEY CONCEPTS - OTHER PRACTICAL CONSIDERATIONS

- Finding a good value for alpha is as simple as testing values and seeing what works well
  - usually, 0.1, 0.01, 0.001 are good initial options
  - if the cost function goes up, or oscillates α is too high
- In general gradient descent is working when the cost function decreases with every iteration
  - Sklearn - set the verbose=True option for the SGDRegressor
- You don’t know the number of iterations you will need for convergence in advance. Monitor the MSE for convergence

There are 2 flavors of gradient descent:

1. Batch Gradient Descent
   i. Uses the entire training set before updating the model parameters

2. Stochastic Gradient Descent
   i. Uses a single training example only before making an update to the model parameters