LINEAR REGRESSION

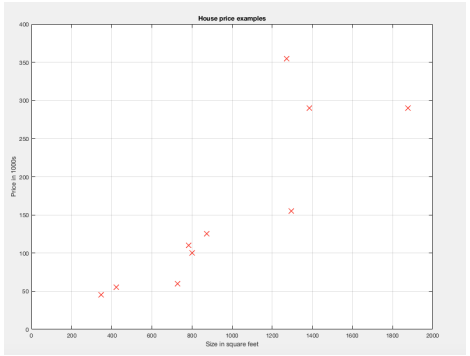
1

Katharine Beaumont
@katharineCodes

	Size in square feet	Price in thousands of £
example 1	1272	355
example 2	1385	290
example 3	1877	290
example 4	1294	155
example 5	873	125
example 6	784	110
example 7	801	100
example 8	729	60
example 9	422	55
example 10	346	45

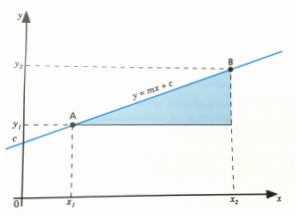
2

We're going to look at some house price examples from back when I lived in an affordable area.
There are 10 examples.



3

If we plot them on a graph, they look like this. You can see that there *might* be a relationship between the size in square feet, and the price of the property, and it *might* be linear.



y = mx + c

c = where the line intercepts the y-axis
m = the gradient of the line

gradient = rate of change = $\frac{\text{change in } \mathbf{y}}{\text{change in } \mathbf{x}}$

4

Reminder: this is the equation of a straight line. We describe it as: some scaling of x (making it bigger/smaller by a factor of m) plus c, where it intercepts the y-axis. This results in the y value.

The hypothesis

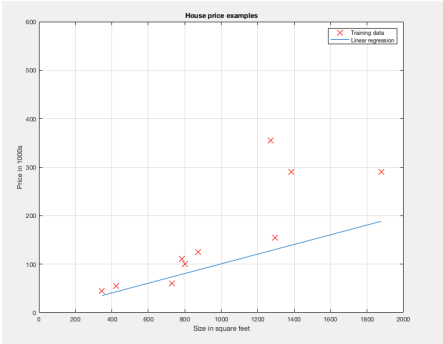
$$h(x) = mx + c$$

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

5

In the example data, we have the x and y values. We're trying to find some m, and some c, that give us a reasonable approximation of y.
m = the gradient
c = the y-intercept
h(x) = the hypothesis. The reason it is written with brackets is that it is saying - there is some function on x, that produces an outcome based on those values. You might see *f(x)* written in maths or some programming examples - some function, f, is operating on x. x is your input parameter.
Another way of writing this is with 'theta', θ, values - they are the 'weights' - just m and c with different notation. θ0 is the y intercept here, and is sometimes called 'the bias'.

We want to learn how to chose the theta values.



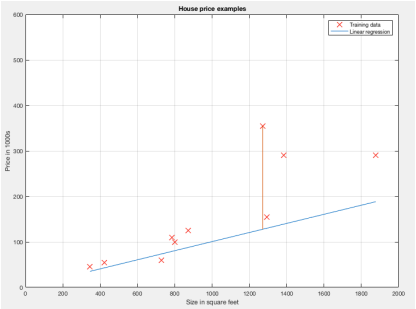
'Guess' **m** is **0.1** and **c** is **0.7**, so

y = 0.1x + 0.7

6

So we could just start with a guess.

7



Prediction = **£127,900**, Price = **£355,000**

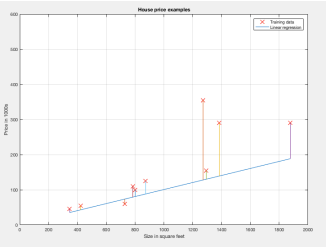
But from one example, we can see that this isn't a good approximation - it's not giving us an answer that is close to the 'answer' we have from the data.

8

What's the error?

h(x)	y	Absolute Error
127.9	355	227.1
139.2	290	150.8
188.4	290	101.6
130.1	155	24.9
88	125	37
79.1	110	30.9
80.8	100	19.2
73.6	60	13.6
42.9	55	12.1
35.3	45	9.7

Absolute error = difference between h(x) and y

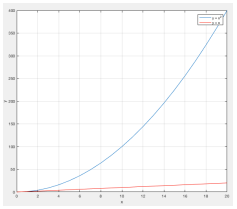


One way to measure the success of our guess, is to measure the difference between what the guess would output, and what the actual answer is.

Squared error (or the Cost Function)

For each example, work out the difference between the actual value and the predicted value, and square it.

$$(h(x) - y)^2$$



Add them together, and average over the number of examples:

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

9

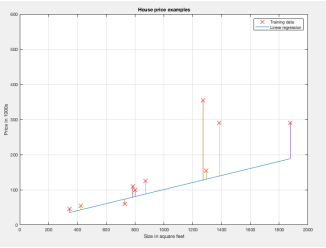
We're going to look at the squared error instead of the absolute error. Why? It emphasises large errors (because they are squared - see the graph. x is 10, x squared is 100, etc).

Why are we dividing by 2m instead of just m? This makes some maths easier later.

What's the error?

h(x)	y	Squared Error
127.9	355	51,574.41
139.2	290	22,740.64
188.4	290	10,322.56
130.1	155	620.01
88	125	1369
79.1	110	954.81
80.8	100	368.64
73.6	60	184.96
42.9	55	146.41
35.3	45	94.09

Squared error =
(h(x) - y) * (h(x) - y)



10

Here we are working out the squared error for each data point.

The Cost Function

$(h(x)-y)^2$

51,574.41
22,740.64
10,322.56
620.01
1369
954.81
368.64
184.96
146.41
94.09

$\sum (h(x)-y)^2$

= 88,376

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

= 88,376 / (2 * 10)

= **4418.78**

11

Now we add them all together, and then divide by 2m. We're basically getting the average error.

How theta changes the Cost Function

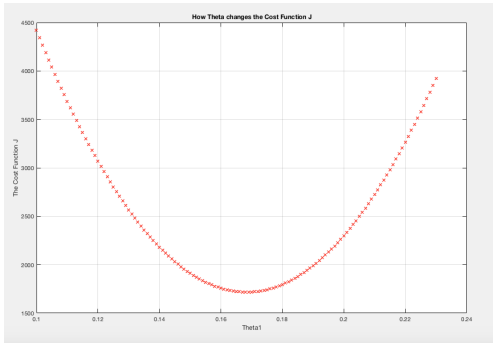
- For now, ignore the 'c' y-intercept, and just look at how changing 'm' the gradient affects the hypothesis
- We tried **y = 0.1x + 0.7**
- Let's try changing '**m**', $\boldsymbol{\theta_1}$, between 0.1 and 0.23 to see how it affects the cost function.
- We will keep '**c**', $\boldsymbol{\theta_2}$, at 0.7

$h(x) = mx+c$

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

12

What next? Now we have a way of seeing how 'right' or 'wrong' the answer is, with the cost, we can see what happens to the cost if we change one the weights, like θ1.

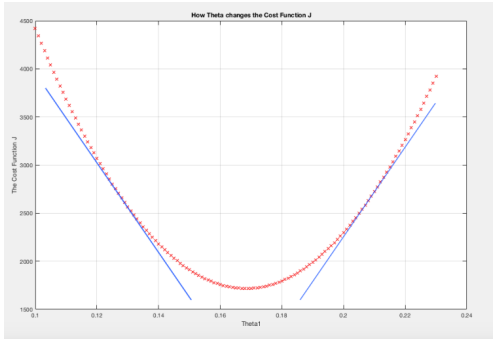


Optimum value = **0.16890**

13

If we plot how the Cost Function changes as θ1 changes, we can see that it is convex. There is a lowest point, or an ‘optimum’ value of theta where the cost function is minimised.

For linear regression with one variable, the cost function is always convex (has a single minimum) - one global optima.



14

We can also see that as θ1 increases towards this optimum value (from 0.1 to 0.17), the slope is going down. The cost is decreasing, and the gradient of the line is negative.
As θ1 increases away from the optimum value, the cost is increasing (the gradient of the line is positive).

Gradient Descent

- One way of 'optimising' the **m** and **c**, or **θ** values in order to minimise the Cost Function. Looks at the gradient of the Cost Function, which is *the rate of change*.
- For a step:
  - Is the Cost Function *increasing* as **θ** increases?

    Reduce **θ**.
  - Is the Cost Function *decreasing* as **θ** increases?

    Increase **θ**.

  Repeat steps.
- When a minimum has been reached, the gradient will be zero, so it will stop changing.

15

Gradient Descent starts with a value of theta. It asks - will the Cost Function decrease if I increase θ, or if I decrease it? It does this by looking at the gradient, the slope of the line.

$$\theta = \theta - \alpha \frac{d(J)}{d\theta}$$

- Adjust the **θ** value by a 'learning rate' (alpha, α) times the rate of change of the Cost Function for the current value of **θ**
- If the Cost Function is increasing, the rate of change will be positive, so we reduce **θ**
- If the Cost Function is decreasing, the rate of change will be negative, so we increase **θ**

16

If you are interested in how we get to the Gradient Descent equation from the Cost Function, see this blog: http://mccormickml.com/2014/03/04/gradient-descent-derivation/

Mathematicians - yes it should be partial derivatives! Please excuse the simplification.

Gradient Descent

$$\theta_0 := \theta_0 - \alpha \frac{d}{d\theta_0} J(\theta_0, \theta_1)$$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_0, \theta_1)$$

17

We simultaneously change θ values (don't set new values until the changes have been worked out. Why? Otherwise θ0 is worked out using the cost function for original θ0, original θ1, but θ1 is worked out with changed θ0, original θ1).

Gradient Descent

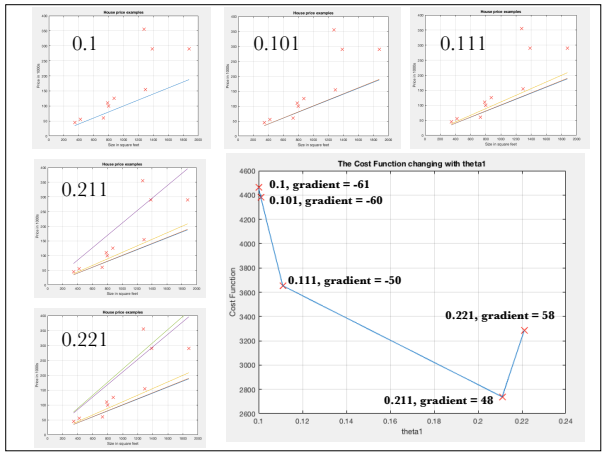
$$\frac{d}{d\theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)$$

$$\frac{d}{d\theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) \times x^i$$

18

For your reference, here is the derivation.

There are different ways of doing it - looking at the gradient for θ across all example data each time, or in batches.



19

Here are some examples of θ1 changing the line of best fit

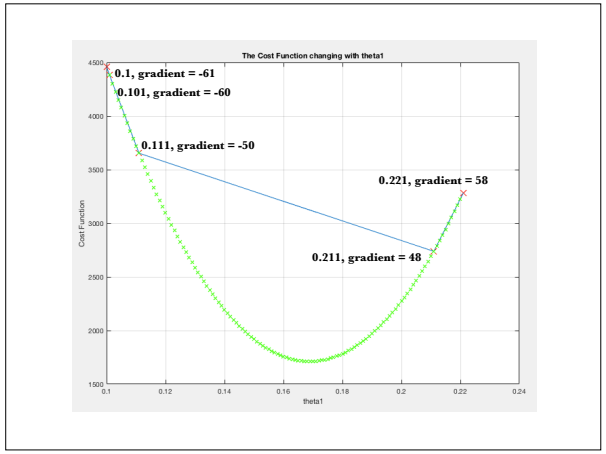
For θ1 = 0.1, the cost function is high and the gradient is negative - so we should increase θ.

Increasing it a little bit to 0.101, it is still negative.

So we can try a bigger step. This takes us to 0.111, where the gradient is still negative.

Another step to 0.211 and we've gone to far, the gradient is positive so the Cost Function is increasing.

We can prove this by going a step further, to 0.221.

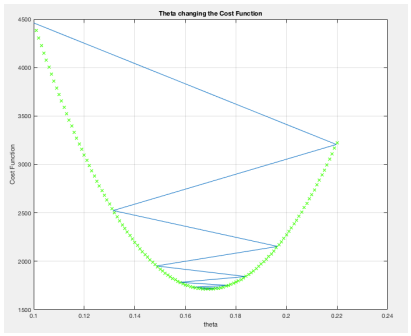


20

Remember alpha, α, the learning rate? [If not, see slide 16]. This determines the size of the step we take each time.

If we take little steps (green examples, we're making a change of 0.001 each time), it takes longer to get to the minimum, so you can take larger steps

However if the step is big, like 0.111 to 0.221, you can go from decreasing the Cost Function, to increasing it, missing the minimum.



21

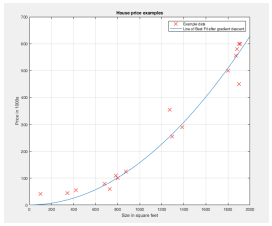
Here we can see Gradient Descent taking big steps towards the minimum. The green crosses are there as a reference.

So - it's like walking down a hill. If you take big steps (think, a giant walking down hill) you could miss the bottom. If you take small steps (think, pixie), you will take a long time to get there.

Polynomial Regression

- Fit complicated, non-linear functions.
- Create extra features by creating polynomials from existing data, e.g. as well as size in square feet, use

  $(\text{size in square feet})^2$

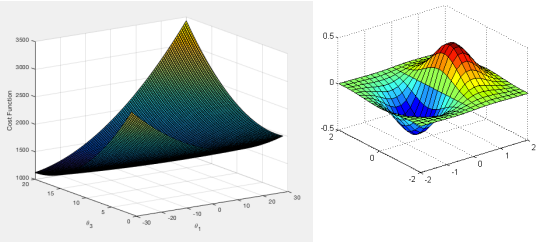


22

Now, we don't just have to use a straight line. We could look at polynomials.

Problems: it's possible to 'overfit' the data - when a new example comes in, it might not work very well - it might not generalise.
Also we have to look at 'feature scaling'.

Gradient Descent with multiple variables



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

23

When there is more than one variable, and we are changing multiple thetas, weights, gradient descent is a lot harder. This is because there isn't necessarily one minimum, where we can get the lowest error. The algorithm 'walks down the hill' and stops at the bottom. What if it chooses the wrong hill, and there is a lower valley somewhere else?

LINEAR REGRESSION

24

Katharine Beaumont
@katharineCodes