**There are a million tutorials on this online but my attempt is really to explain the mathematical concepts to non-math people.**

* **What does gradient descent do?**

**It’s a way of:**

* + **minimizing a Cost Function, which is to say:**
  + **reducing a Cost Function’s error**

**So first we need to clarify:**

* **What is a Cost Function?**
  + **A cost function is a way of calculating the error of a function, i.e. how good or bad that function is about telling us the relationship between our variables**

**What we ultimately want to do in the Linear Regression context is draw a straight line and then minimize the (vertical) distance between that line and a bunch of data points. That will tell us how good a fit the function (and therefore the line) is.**

**Let’s begin with the formula for a straight line, which is what we want to draw:**

**Fig.1**

* **m is the slope of the line. Slope is a number denoting the *direction* and the *steepness* of a line**
* **c is the intercept, i.e. the point at which a line passes through an axis (usually the y-axis).**

**We are going put a bunch of x,y values into this function.**

**To get the best y we can, and in the process draw the best line, we have the option to adjust (update) *m* and *c*.**

**To do this we need a cost function that penalizes the function for making bad predictions.**

**Looks a bit scary, right?**

**Fig.2**

**This is the sigma symbol ∑.**

**It denotes the sum of all values in a particular range. The range here is *n*, which we are going to set ourselves in this program. *n* is going to be our number of x** **values that we put into our function. *i* is going to denote our number of iterations. For the first iteration, *i* will equal 1. For the second iteration it will be 2 etc. will be a real y value that we provide as input. (‘y-hat’) will be a prediction made using our function. We square so as to remove negative values.**

**Remember that the function we’re going to use to draw our line is**

**Fig.3**

**So let’s ‘plug’ that into our cost function:**

**Fig.4**

**Now we have inserted our straight line function into our cost function to try and penalize our function for creating bad straight lines.**

**The next part is where Gradient Descent comes in. This is basically a piece of calculus that minimizes our cost function. You will remember differential calculus from high school. Well, some basic calculus will allow us to reduce the penalty assigned to our model for making bad predictions.**

**You need to know the chain rule for the calculus part.**

**Now let’s differentiate:**

**Fig. 5**