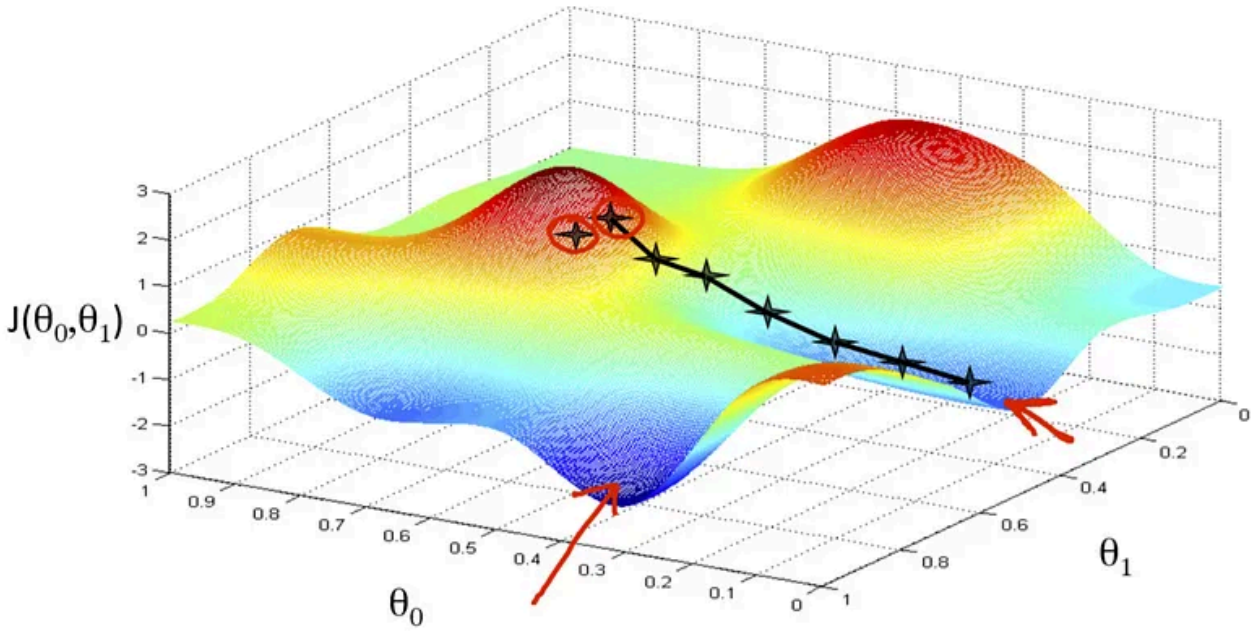
Member-only story

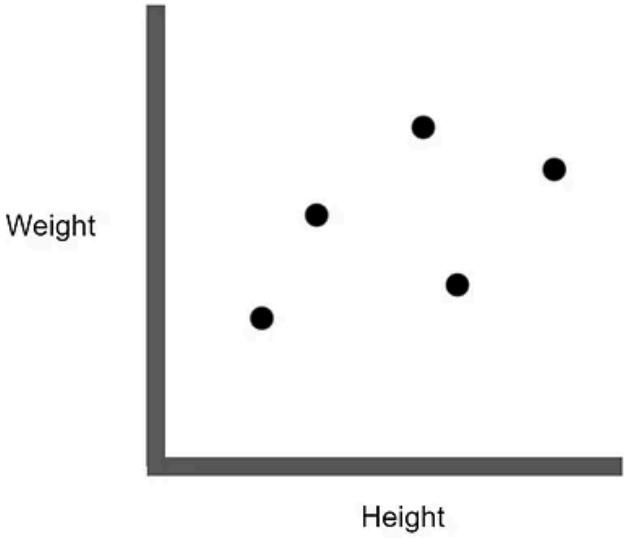
An Intuitive Explanation of Gradient Descent

One of the most widely used machine learning algorithms explained in 5 minutes

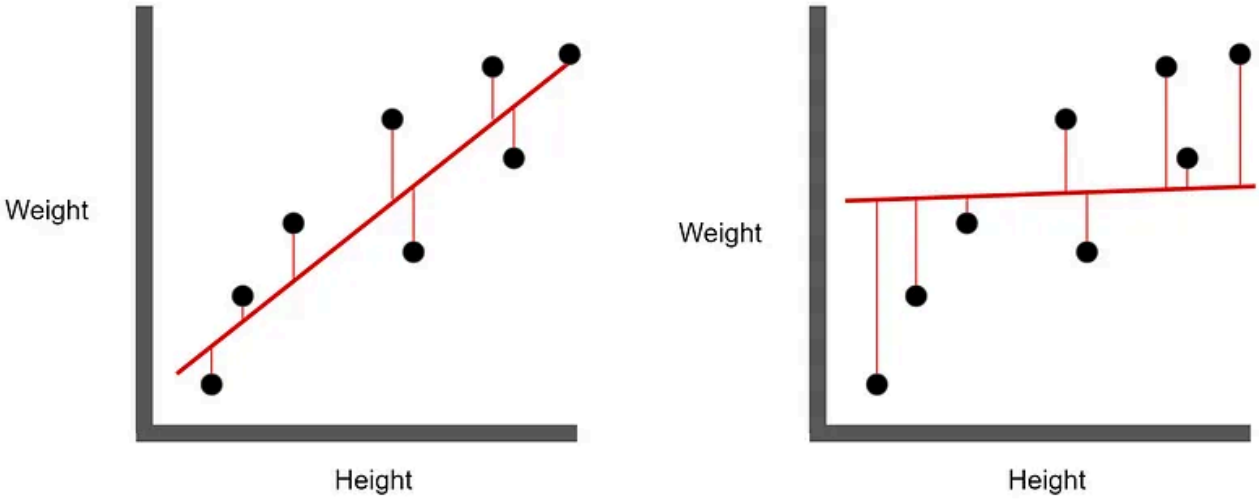
Terence Shin, MSc, MBA · Follow
Published in Towards Data Science
5 min read · Jan 18, 2020



Gradient Descent is widely used in the machine learning world and is essentially an optimization algorithm used to find the minimum of a cost function. In data science, gradient descent is used to refine the parameters (coefficients) of our model to minimize error. We'll go through an example below to understand how gradient descent works.



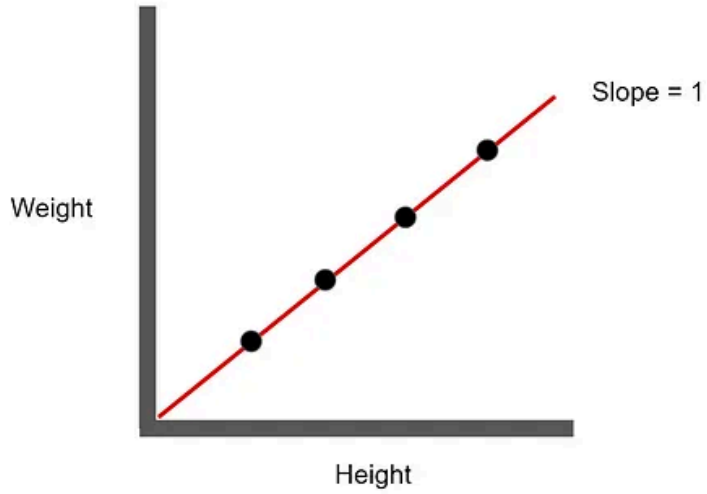
Imagine that the graph above represents the relationship between people's height and weight; taller people generally weigh more than shorter people. Suppose we wanted to find the line of best fit, that is the equation that best represents the data shown above.



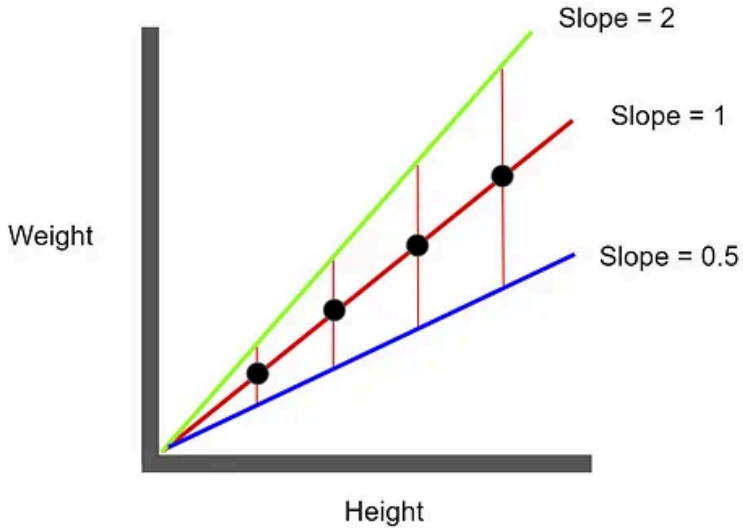
The image above shows that the (red) line of best fit on the left is much more precise than the one on the right. This can be seen by looking at the distance between each point and the line of best fit. The less distance there is (the less error), the better the line of best fit. The line of best fit is represented by the equation that we've all learned in high school, $y = mx + b$.

To understand gradient descent, we first need to understand what the cost function is.

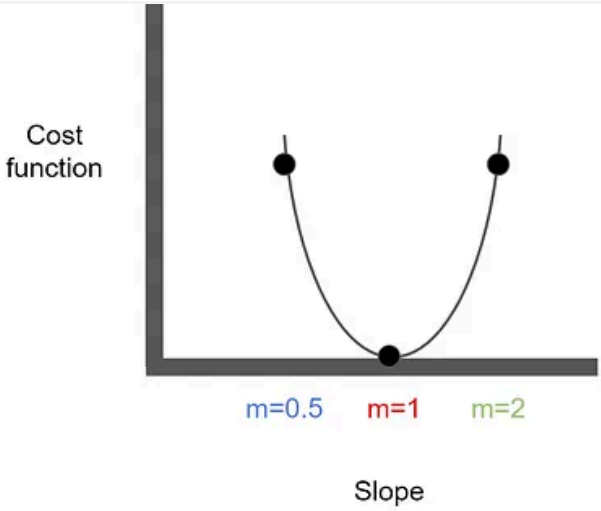
A cost function is essentially something that we want to minimize; in this case, we want to minimize the distance between the data points and the line of best fit. You can think of the cost function as an error function if that makes more sense intuitively. To understand the cost function, let's use an even simpler example.



Suppose we have data points that fall right on the line of best fix, y = x, where the slope is equal to 1. In this case, the distance between the data points and the line of best fit, *aka the cost function,* would be equal to zero.



Now imagine that we move the line of best fit to the green line so that the slope is equal to 2 (green line). Now, the distance between the data points and the line of best fit has increased. **This means that the cost function has increased.** Similarly, if we moved the line of best fit so that the slope was equal to ½ (blue line), the cost function would have equally increased as the green line.



If we plot the slope of the line of best fit with the corresponding cost function, we can see that a slope of one minimizes the cost function, aka minimizes the distance between the data points and the line of best fit. And if you look back at our graph with a slope equal to 1, this is true!

Now onto the fun stuff.

Gradient Descent is an algorithm that is used to essentially minimize the cost function; in our example above, gradient descent would tell us that a slope of one would give us the most precise line of best fit.

$$\text{Repeat until convergence } \{$$

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

$$\}$$

While it looks daunting, it's rather simple. Let me explain:

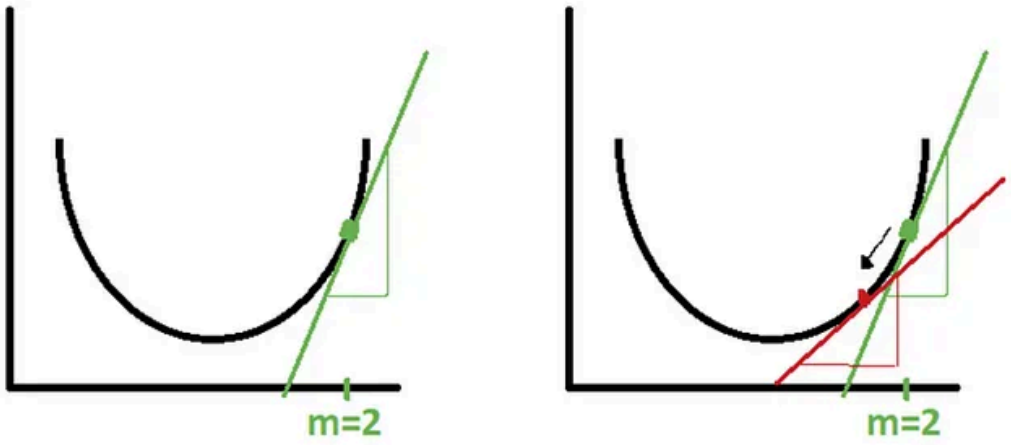
- θj is equal to m (the slope)
- α is equal to the learning rate (I'll explain this more later)
- (∂/ ∂θj)*J(θ) is equal to the partial derivative of the cost function at point m (slope of the cost function, **not the line of best fit)**

- *Repeat until convergence* simply means repeating the algorithm until it reaches a global optimum (minimum in this case).

In words, this algorithm means:

Repeatedly subtract the slope of the line of best fit by the slope of the cost function at point m times a coefficient (α) until the slope of the partial derivative is zero.

> New slope = old slope of line of best fit — alpha * partial derivative of the cost function at point m



To show an example, let's say we start with a line of best fit that has a slope of 2. If we look at the point with a slope of 2 on the cost function (the green point), we can take the partial derivative of that point aka the slope of the green line that is tangent to the curve. Here, we can see that the partial derivative is positive. Looking back at our equation,

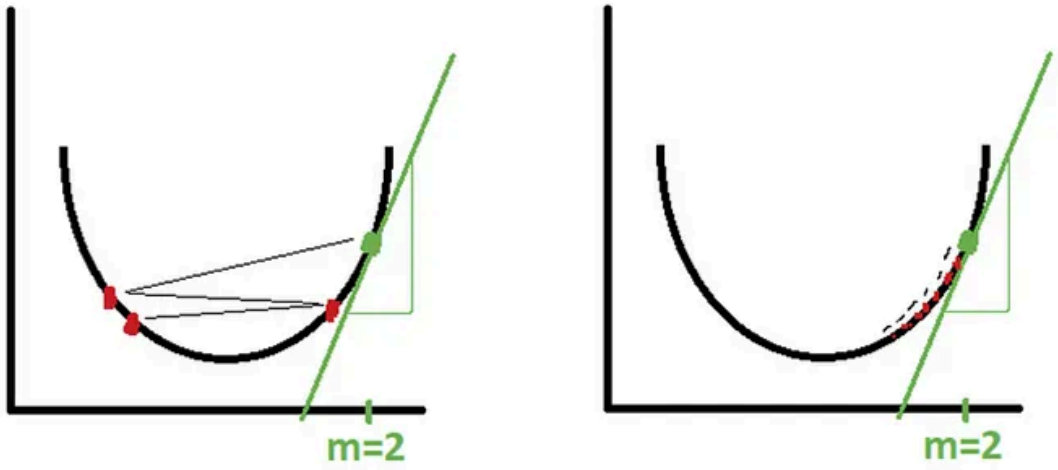
*New slope = old slope of line of best fit — alpha * **partial derivative of the cost function at point m***

The new slope (m) would be equal to 2 **minus a positive number** (since the partial derivative is positive), which would bring us to the red point (shown above).

If the partial derivation was negative aka, if the point started on the left side of the graph, then the new slope would be equal to the old slope minus a negative number, which is equivalent to the old slope **plus a positive number.**

We would repeat this process until we reach convergence, aka we reach a minimum, and thus we would have determined the slope of the line of best fit that minimizes the cost function!

Just a quick note on alpha (α) in the gradient descent algorithm,



If α is too big, then the algorithm will overshoot each iteration (as shown in the left graph), which may inhibit it from reaching the minimum. Conversely, if α is too small, it will take too long to reach the minimum. Thus, α must be in between the two so that neither of these cases occur.

And that's gradient descent in a nutshell!

Data Science | Machine Learning | Artificial Intelligence | Programming | Technology

Follow

Written by Terence Shin, MSc, MBA