Gradient Descent
for optimization.

Topics covered:

- Tangent line slope
- Optimization of 1 variable
- Optimization of 2 and more variables
- Gradient Descent

We're finally here...

Gradient Descent - one of the most essential Machine Learning concepts.

Before diving into Gradient Descent as we know it (on multiple variables)

Let's talk about 1 variable optimization.

Imagine having function $f(x) = x^2$

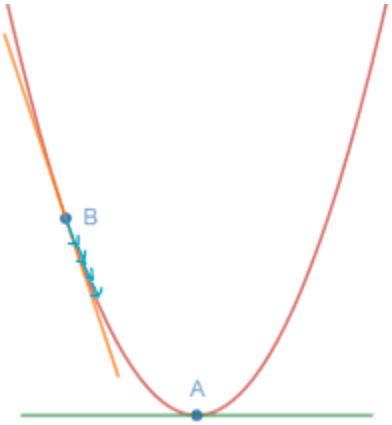
How do we find the minimum of f(x)?

Of course we could solve $2x = 0 \Rightarrow x = 0$

But dealing with bigger equations it's too complicated.

We could just pick random points instead and make random steps to reach values close to minimum.

So optimization with tangent line (slope) makes pretty same thing:



Here we have randomly picked point B and minima point A.

We can see that tangent's line slope is negative, so we can subtract our slope from x value (it was picked randomly) to get new x1. x1 > x (as we subtracted negative) and we will get x to achieve y minima.

Learning Rate

—

Sometimes these steps may be too big, so to make 'em smaller (and control at all) we have a special const value α - learning rate (or alpha coefficient)
If we want to make our steps smaller α must satisfy:

$$0 < \alpha < 1$$

That's it.
Formula for gradient descent x update:

$$x1 = x0 - a\,f'(x0)$$

3 easy steps of tangent line optimization:

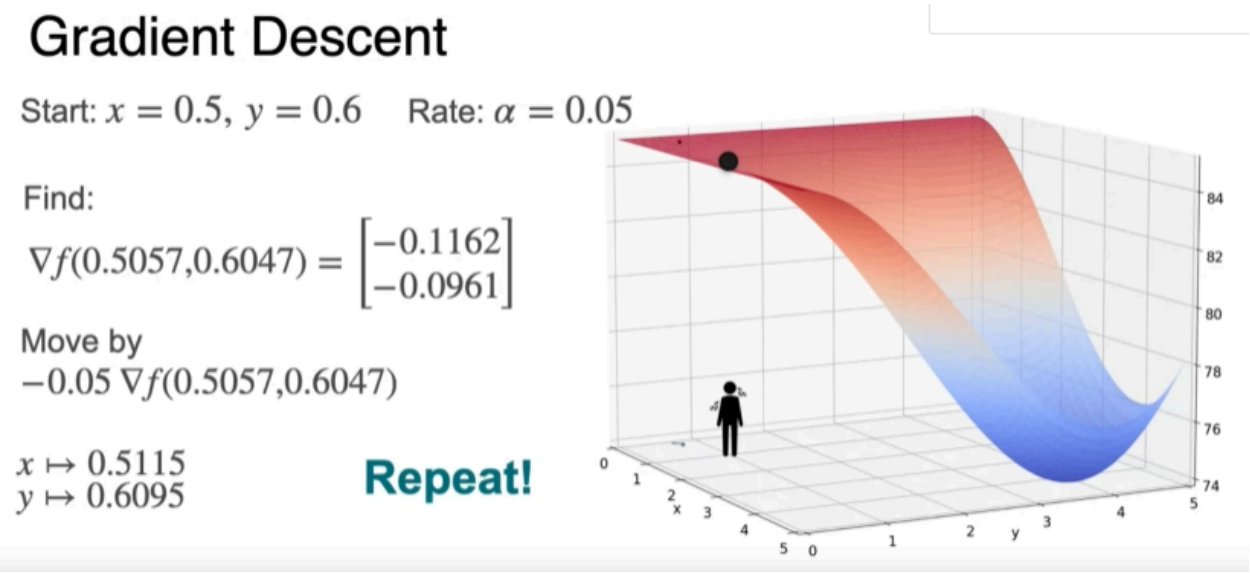
1. Define learning rate and pick some point
2. Find derivative of point and update it
3. Repeat until you reach minimum

Gradient Descent

Final boss...

As we remember gradient is a vector of derivatives for each variable.
So it means that gradient descent is used when we work with 2 and more variables.
Gradient descent follows the same principles as those that were mentioned before for tangent line optimization.

Let me just upload a pic here:



We pick x and y (as we're working in 3d with 2 variables)

We find their gradient

We apply this gradient with Learning Rate

Note: sadly there's no best way for LR choice.

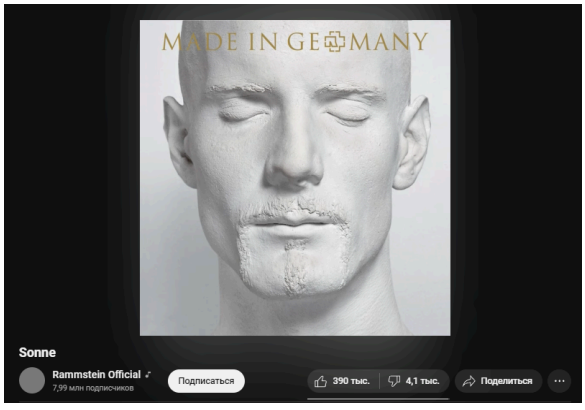
That's pretty much it

Here's a gradient descent formula:

$$(x1,\ y1)\ =\ (x0,\ y0)\ -\ a\bigtriangledown f$$

Fun fact:

I was writing this paper listening to:



This material is free to use, share, and criticize.

Uses materials by DeepLearningAI

DeepLearningAI course - link

Written by Venchislav for the GitHub community❤.

21.03.2024

GoodBye!