Understanding exponentially weighted averages

(SPEECH)
In

(DESCRIPTION)
Text, Optimization Algorithms. Understanding exponentially weighted averages. Website, deep learning, dot, A.I.

(SPEECH)
the last video, we talked about exponentially weighted averages.

This will turn out to be a key component of several optimization algorithms that you used to train your neural networks.

So, in this video, I want to delve a little bit deeper into intuitions for what this algorithm is really

(DESCRIPTION)
New slide, Exponentially weighted averages. Equation is, V, subscript T, equals, Beta, V, subscript, T minus 1, plus, 1 minus Beta, zero, subscript T.

(SPEECH)
doing.

Recall that this is a key equation for implementing exponentially weighted averages.

And so, if beta equals 0.9 you got the red line.

If it was much closer to one, if it was 0.98, you get the green line.

And it it's much smaller, maybe 0.5, you get the yellow line.

Let's look a bit more than that to understand how this is computing averages of the daily temperature.

So here's that equation again, and let's set beta equals 0.9 and write out a few equations that this corresponds to.

So whereas, when you're implementing it you have T going from zero to one, to two to three, increasing values of T. To analyze it, I've written it with decreasing values of T. And this goes on.

So let's take this first equation here, and understand what V100 really is.

So V100 is going to be, let me reverse these two terms, it's going to be 0.1 times theta 100, plus 0.9 times whatever the value was on the previous day.

Now, but what is V99?

Well, we'll just plug it in from this equation.

So this is just going to be 0.1 times theta 99, and again I've reversed these two terms, plus 0.9 times V98.

But then what is V98?

Well, you just get that from here.

So you can just plug in here, 0.1 times theta 98, plus 0.9 times V97, and so on.

And if you multiply all of these terms out, you can show that V100 is 0.1 times theta 100 plus.

Now, let's look at coefficient on theta 99, it's going to be 0.1 times 0.9, times theta 99.

Now, let's look at the coefficient on theta 98, there's a 0.1 here times 0.9, times 0.9.

So if we expand out the Algebra, this become 0.1 times 0.9 squared, times theta 98.

And, if you keep expanding this out, you find that this becomes 0.1 times 0.9 cubed, theta 97 plus 0.1, times 0.9 to the fourth, times theta 96, plus dot dot dot.

So this is really a way to sum and that's a weighted average of theta 100, which is the current days temperature and we're looking for a perspective of V100 which you calculate on the 100th day of the year.

But those are sum of your theta 100, theta 99, theta 98, theta 97, theta 96, and so on.

So one way to draw this in pictures would be if, let's say we have some number of days of temperature.

So this is theta and this is T. So theta 100 will be sum value, then theta 99 will be sum value, theta 98, so these are, so this is T equals 100, 99, 98, and so on, ratio of sum number of days of temperature.

And what we have is then an exponentially decaying function.

So starting from 0.1 to 0.9, times 0.1 to 0.9 squared, times 0.1, to and so on.

So you have this exponentially decaying function.

And the way you compute V100, is you take the element wise product between these two functions and sum it up.

So you take this value, theta 100 times 0.1, times this value of theta 99 times 0.1 times 0.9, that's the second term and so on.

So it's really taking the daily temperature, multiply with this exponentially decaying function, and then summing it up.

And this becomes your V100.

It turns out that, up to details that are for later.

But all of these coefficients, add up to one or add up to very close to one, up to a detail called bias correction which we'll talk about in the next video.

But because of that, this really is an exponentially weighted average.

And finally, you might wonder, how many days temperature is this averaging over.

Well, it turns out that 0.9 to the power of 10, is about 0.35 and this turns out to be about one over E, one of the base of natural algorithms.

And, more generally, if you have one minus epsilon, so in this example, epsilon would be 0.1, so if this was 0.9, then one minus epsilon to the one over epsilon.

This is about one over E, this about 0.34, 0.35.

And so, in other words, it takes about 10 days for the height of this to decay to around 1/3 already one over E of the peak.

So it's because of this, that when beta equals 0.9, we say that, this is as if you're computing an exponentially weighted average that focuses on just the last 10 days temperature.

Because it's after 10 days that the weight decays to less than about a third of the weight of the current day.

Whereas, in contrast, if beta was equal to 0.98, then, well, what do you need 0.98 to the power of in order for this to really small?

Turns out that 0.98 to the power of 50 will be approximately equal to one over E. So the way to be pretty big will be bigger than one over E for the first 50 days, and then they'll decay quite rapidly over that.

So intuitively, this is the hard and fast thing, you can think of this as averaging over about 50 days temperature.

Because, in this example, to use the notation here on the left, it's as if epsilon is equal to 0.02, so one over epsilon is 50.

And this, by the way, is how we got the formula, that we're averaging over one over one minus beta or so days.

Right here, epsilon replace a row of 1 minus beta.

It tells you, up to some constant roughly how many days temperature you should think of this as averaging over.

But this is just a rule of thumb for how to think about it, and it isn't a formal mathematical statement.

Finally, let's talk about how you actually implement this.

(DESCRIPTION)
New slide, Implementing exponentially weighted averages.

(SPEECH)
Recall that we start over V0 initialized as zero, then compute V one on the first day, V2, and so on.

Now, to explain the algorithm, it was useful to write down V0, V1, V2, and so on as distinct variables.

But if you're implementing this in practice, this is what you do: you initialize V to be called to zero, and then on day one, you would set V equals beta, times V, plus one minus beta, times theta one.

And then on the next day, you add update V, to be called to beta V, plus 1 minus beta, theta 2, and so on.

And some of it uses notation V subscript theta to denote that V is computing this exponentially weighted average of the parameter theta.

So just to say this again but for a new format, you set V theta equals zero, and then, repeatedly, have one each day, you would get next theta T, and then set to V, theta gets updated as beta, times the old value of V theta, plus one minus beta, times the current value of V theta.

So one of the advantages of this exponentially weighted average formula, is that it takes very little memory.

You just need to keep just one row number in computer memory, and you keep on overwriting it with this formula based on the latest values that you got.

And it's really this reason, the efficiency, it just takes up one line of code basically and just storage and memory for a single row number to compute this exponentially weighted average.

It's really not the best way, not the most accurate way to compute an average.

If you were to compute a moving window, where you explicitly sum over the last 10 days, the last 50 days temperature and just divide by 10 or divide by 50, that usually gives you a better estimate.

But the disadvantage of that, of explicitly keeping all the temperatures around and sum of the last 10 days is it requires more memory, and it's just more complicated to implement and is computationally more expensive.

So for things, we'll see some examples on the next few videos, where you need to compute averages of a lot of variables.

This is a very efficient way to do so both from computation and memory efficiency point of view which is why it's used in a lot of machine learning.

Not to mention that there's just one line of code which is, maybe, another advantage.

So, now, you know how to implement exponentially weighted averages.

There's one more technical detail that's worth for you knowing about called bias correction.

Let's see that in the next video, and then after that, you will use this to build a better optimization algorithm than the straight forward create