Bias correction in exponentially weighted averages

(DESCRIPTION)
Text, deep learning dot AI. Optimization algorithms. Bias correction in exponentially weighted average

(SPEECH)
You've learned how to implement exponentially weighted averages.

There's one technical detail called biased correction that can make you computation of these averages more accurately.

Let's see how that works.

In

(DESCRIPTION)
Graph of daily temperature over the period of a year. Scatter plot. Formula. v of t, equals, beta times v of t minus one, plus, one minus beta, times theta t

(SPEECH)
a previous video, you saw this figure for beta = 0.9.

This

(DESCRIPTION)
Rolling average line traced through

(SPEECH)
figure for beta = 0.98.

But

(DESCRIPTION)
Another rolling average line, this one somewhat smoother and delayed with respect to the data

(SPEECH)
it turns out that if you implement the formula as written here, you won't actually get the green curve when, say, beta = 0.98.

You actually get the purple curve here.

And

(DESCRIPTION)
Similar to the green curve, but starts too low and ends too high

(SPEECH)
you notice that the purple curve starts off really low.

So let's see how it affects that.

When you're implementing a moving average, you initialize it with v0 = 0, and then v1 = 0.98 V0 + 0.02 theta 1.

But V0 is equal to 0 so that term just goes away.

So V1 is just 0.02 times theta 1.

So that's why if the first day's temperature is, say 40 degrees Fahrenheit, then v1 will be 0.02 times 40, which is 8.

So you get a much lower value down here.

So it's not a very good estimate of the first day's temperature.

v2 will be 0.98 times v1 + 0.02 times theta 2.

(DESCRIPTION)
Zero point zero 2, times theta 1

(SPEECH)
And if you plug in v1, which is this down here and multiply it out, then you find that v2 is actually equal to 0.98 times 0.02 times theta 1 + 0.02 times theta 2.

And that 0.0 196 theta1 + 0.02 theta2.

So again, assuming theta1 and theta2 are positive numbers, when you compute this v2 will be much less than theta1 or theta2.

So v2 isn't a very good estimate of the first two days' temperature of the year.

So it turns out that there is a way to modify this estimate that makes it much better, that makes it more accurate, especially during this initial phase of your estimate.

Which is that, instead of taking Vt, take Vt divided by 1- Beta to the power of t where t is the current data here on.

So

(DESCRIPTION)
The denominator is, beta to the power of t, all subtracted from 1

(SPEECH)
let's take a concrete example.

When t = 2, 1- beta to the power of t is 1- 0.98 squared and it urns out that this is 0.0396.

And so your estimate of the tempature on day 2 becomes v2 divided by 0.0396 and this is going to be 0.0196 times theta 1 + 0.02 theta 2.

You notice that these two things adds up to the denominator 0.03 and 6.

And so this becomes a weighted average of theta 1 and theta 2 and this removes this bias.

So you notice that as t becomes large, beta to the t will approach 0 which is why when t is large enough, the bias correction makes almost no difference.

This is why when t is large, the purple line and the green line pretty much overlap.

But during this initial phase of learning when you're still warming up your estimates when the bias correction can help you to obtain a better estimate of this temperature.

And it is this bias correction that helps you go from the purple line to the green line.

So in machine learning, for most implementations of the exponential weighted average, people don't often bother to implement bias corrections.

Because most people would rather just wait that initial period and have a slightly more biased estimate and go from there.

But if you are concerned about the bias during this initial phase, while your exponentially weighted moving average is still warming up.

Then bias correction can help you get a better estimate early on.

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

Let's go on and use this to build some better optimization algorithms.