What is online machine learning?

Online machine learning includes methods to create machine learning models using data which becomes available sequentially in time.In classical machine learning approaches, you use the whole data set to train required models and after that, you make predictions on data points using your trained model when you need. At that point, the data you used for training the model is not needed.Differently, online learning techniques update the trained model when each new data point occurs. Thus, you can make use of any data you have for the accuracy of your machine learning model instantly.
If using online learning approach is not applicable to your case (e.g algorithm you chose), you could re-train model with increasing data periodically as an alternative.Compared to "traditional" ML solutions, online learning is a fundamentally different approach, one that embraces the fact that learning environments can (and do) change from second to second. It's tricky to get right, but when applied correctly, the results you can achieve with online learning are nothing short of remarkable. In this post, I'll give a quick introduction to the technique.

> **Consider the following example. Let's say we run a news website. We personalise our news by collecting data on what was clicked or not clicked, and by whom. Based on this**

information, we predict the types of news different visitors might like, and serve them relevant items.

One day, out of the blue, word gets out that the government is issuing a state of emergency, and will hold a press conference in an hour. Suddenly, *everyone* is interested in domestic affairs — even those who typically only read about sports or look at the funnies. When presented with a news piece about the conference, a huge percentage of the audience clicks it to learn more.

If you had gone the traditional route and batch trained your recommendation engine once a day, it would still be stuck offering the

same type of content, even though the underlying world changed dramatically[1]. You *should* be serving up domestic news right now, but aren't because your system is too slow.

It gets worse: the following day, after the press conference and following a new training cycle, your engine would start actively recommending domestic news which, after 24 hours, isn't necessarily interesting any more. It's made two mistakes, both because it can't react fast enough.

That's the power of online learning: done properly, it can react in minutes or even

seconds. With it, there is no such thing as "yesterday's news".

Bolding for emphasis, implementing real time learning isn't easy. If you place some learning algorithm behind an API and, god forbid, open it up to the Internet, there's an almost limitless number of ways it can go wrong. You might get lots of feedback (examples) from one thing but not another, leading to a skewed classes problem. You might've set your learning rate too high, causing your model to forget everything that happened more than a second ago. You might overfit, or underfit. Someone might DDoS your system, screwing up learning in the process. Online learning is prone to

catastrophic interference **— more so than most other techniques.**

Online learning also requires an entirely different approach in terms of technical architecture. Since a model can, and will, change from second to second, you can't just instantiate several instances like you can with traditional techniques. It's not horizontally scalable. Instead, you are forced to have a single model instance that eats new data as fast as it can, spitting out sets of learned parameters behind an API. And the second that one set in one process gets replaced by a new one, all other processes must follow suit immediately. It's an engineering challenge, because the most

important part (the model) is only vertically scalable. It may not even be feasible to distribute between threads.

Learning immediately also requires fast access to new data. If you're lucky enough to get all the data you need for a single training example as part of an API call, you're good to go. But if something is not available client-side, you need to be able to grab that data from somewhere in milliseconds. Typically, that means using an in-memory store like Redis. "Big data" processing frameworks aren't of much help. If you want to to both batch and online learning, Spark isn't enough. If you do only online learning, Spark is useless.

I could go on for hours about the technical aspects, but the bottom line is this: online learning is an ML solution unlike any other, and it requires a technical approach unlike any other.

Summing up online learning isn't easy. It's not a single learning algorithm: in fact, lots of algorithms can learn online. It's also not fundamentally different in terms of how learning happens: you can use the same optimisation steps you always do. It doesn't even have a bombastic, sci-fi sounding name.

What online learning is is a fundamentally different way of approaching machine learning. It's an approach that embraces

change, no matter how drastic. Its existence is predicated on the belief that since everything is in flux, we should stop seeking stationarity and instead start living in the moment.