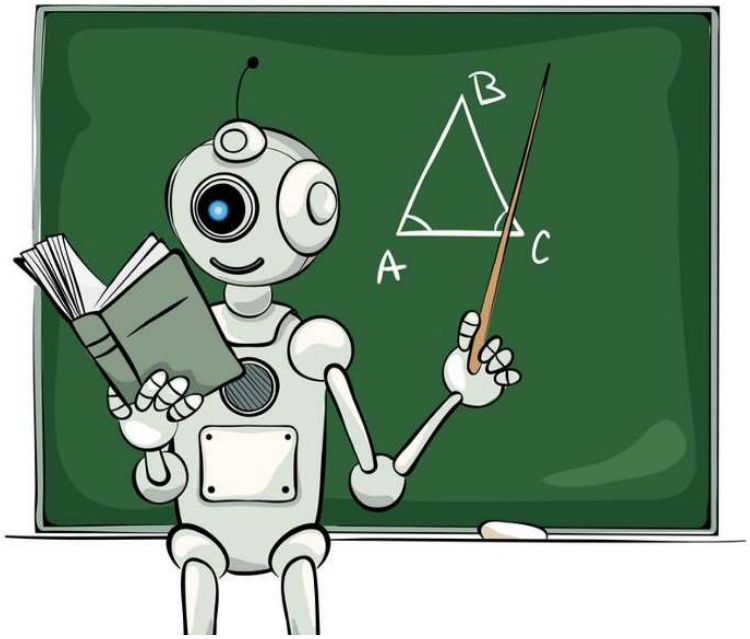
George Seif Follow
Certified Nerd. AI / Machine Learning Engineer.
May 28 · 7 min read

The 4 Machine Learning Skills You Won't Learn in School or MOOCs



Machine Learning (ML) has become massively popular over the last several years. And why… well simply because it works! The latest research has achieved record breaking results, even surpassing human performance on some tasks. Of course as a result many people are rushing to get into this field; and why not. It's well funded, the technology is exciting and interesting, and there's lots of room for growth.

In terms of formal education there are two main paths one might take towards learning Machine Learning: school (universities/colleges) or MOOCs (Massive Open Online Courses). Formal schooling, such as attaining a Masters or PhD can give you very in depth technical and theoretical knowledge, but it's also quite financially expensive. MOOCs are free and you can definitely learn how to practically use many ML algorithms from them, though you won't get the official credentials that you would normally get from going to university/college. Both of these however can be effectively used to gain the technical knowledge you need to work in the Machine Learning field.

But… there's still something missing. At the end of the day, machine learning algorithms are being applied to real-world business problems. Machine learning is simply used as a tool to create value for customers. It's far different from the classroom where assignments are generally done individually and there is little to no business objective. It's not only the accuracy of the ML model that matters, but also its interpretability, speed, memory consumption, and how, at the end of the day, that model fits into the product to create real value.

Today, I'm going to share with you 4machine learning skills you won't learn in school or MOOCs. These are skills that can be learned through real-world experience, using ML to solve a real-world problem and create real value. Hopefully this post gives you insight into how machine learning can be practically used, and what skills you can learn to become more of an expert in your field.

Let's dive in!

Connecting machine learning with business objectives

When you work in any field involving software, including machine learning, there are really two main aspects of understanding: the technical point of view and the business point of view. Knowledge of both is key to becoming an expert in your field.

You learn a lot of the technical part in the classroom. How to code in Python, machine learning and data science algorithms, technical report writing, etc. In the classroom, these things are isolated from business. But when you start to actually work in the field, every single technical aspect of your job is tied to a business objective. Why did your boss as you to increase the accuracy of the current system? Well probably because better accuracy means more value, more value means better product, and better product means more customers and money!

It's important to be able to connect those dots so you can find the best way of tackling the technical problems. You'll know what the important parts are and thus what you should focus on. As you gain seniority in your field, you may be expected to translate business objectives to technical objectives, to find the best way to get the job done. Technology has and still is a tool used to accomplish something greater. Always think in terms of how your technical skills are connected with business objectives to create real value.

Model selection

In the classroom, you learn all about the different ML models. Linear regression, SVMs, decision trees, neural networks… it feels like there's a million of them! So the big question is … which one do you use? You may have used all of those algorithms before, coded them from scratch or used libraries like TensorFlow and Scikit Learn. But why choose one over the other? Lots of people are using deep learning now, so should we just default to that?

All methods will have tradeoffs, that's just the nature of science and engineering! It's important to connect those tradeoffs to your business objectives in order to define what the most important attributes of your models are.

In general, machine learning has something called the "No Free Lunch" theorem which basically states that no one ML algorithm is best for all problems. The performance of different ML algorithms strongly depends on the size and structure of your data. Thus, the correct choice of algorithm can be unclear unless we test out our algorithms directly through plain old trial and error.

But, there are some pros and cons to each ML algorithm that we can use as guidance. Although one algorithm won't always be better than another, there are some properties of each algorithm that we can use as a guide in selecting the best one quickly and tuning hyper parameters. I wrote an article about it for Towards Data Science too. For example, neural networks (and deep learning) often achieve very high accuracies, but aren't very interpretable at all. It probably wouldn't be a very good idea to use them when you need to know exactly where your results come from; but if you only care about the final output, they're perfect! Decision trees on the other hand are quite interpretable and easy to understand since they are by design quite intuitive. They may not achieve the same level of accuracy that a deep network can, but they're great when you want to understand exactly where your results are coming from.

Model deployment

Machine learning education tends to go quite in depth into the ML algorithms themselves, teaching your how they work on a technical level. Once your model is fully trained, it's common practice to then test it on a held-out, unseen test dataset to benchmark the model performance. Once we've verified that our model works well for our task, it must be deployed into the full software product that you have created your model for.

From a high-level, deployment essentially involves "plugging your algorithm in" to the rest of the system. The role of your model is simply to take an input and use it to make some kind of prediction that is useful to the system. It is thus important to understand your software from a higher level, from a systems level.

So far we've talked about how in school you learn the very low level technical things and the business objectives are the very high level goals defining product value delivery. Understanding of your software as a system of connected components to deploy your model lies right in the middle. You must understand the architecture of your system, the pipeline of connected blocks. You can then view your model as just another component, block, or module to plug in to the system.

Getting the most bang for your buck

When we're in school or taking an online course, we have all the time in the world to experiment. We can spend forever doing research, trying to work with the latest and greatest algorithms, or even just experimenting with all of them to find the best one! In the real-world, that might not be the most efficient way of doing things.

Businesses have a finite amount of time and resources. They can't spend all day trying every single possible method to see which one is the best. They need to find the best way of doing things in an efficient manner (read: without burning time and cash).

You need to look at things from the perspective of getting the best bang for your buck. So maybe there's a new state-of-the-art regression algorithm, but it's quite technically challenging and perhaps time consuming to implement. Instead of spending so much time and money trying to use it, you might get more bang for buck simply by getting more data for your current algorithm to train on, or using best practice "hacks" to bump up your accuracy by a few points. There might be certain aspects of your algorithm that adds tons of value, and others that don't. Maybe it doesn't matter if your algorithm is much more accurate than it currently is, but the product would have a lot more value if it was super fast!

In the classroom, it's easy to overlook such things and always aim for the best performing model in one single aspect, forgetting that all paths we take will have their tradeoffs. The key is finding out which tradeoffs are worth it, which ones give you the best bang for buck. And, what you'll usually notice, is that the things that are most important tie back to the business objective.

Machine learning and all software in general is simply a set of tools; getting the most bang for buck in this context means knowing how best to use those tools to get the job done.

Conclusion

Thanks for reading! Hopefully you learned something new and useful about some of the machine learning skills you won't learn in school or MOOCs! If you enjoyed reading, feel free to hit the clap button so other people can see this post and hop on the learning train with us!