KEEP MACHINE LEARNING TEAMS FOCUSED ON DATA SCIENCE, NOT DATA PROCESSING

Mach

ine learning capabilities, use cases, and enterprise demand are exploding. The talent base to fulfill this demand is not. Talent shortages are a real barrier to machine learning initiatives. Organizations need to create an environment where their machine learning teams can focus on data science, not data entry and management. Automation is the key, but it isn't a skeleton key - no single set of automation tools unlocks successful machine learning development. This article presents guidance on selecting the right automation approach so organizations can keep their machine learning talent working on data science, not data processing.

Automation can be applied at every stage of machine learning development and execution. That's good news, but also represents a problem with machine learning automation tools – they tend to be stage-specific and don't integrate well with others. The more stages of the development-test-deploy-execute lifecycle a toolset can address, the faster your teams can deliver production-ready innovations. Here's a look at some of the challenges and tools for different stages of machine learning development and execution.

There are several common challenges to developing machine learning models. Organizations need to work with large volumes of disparate data sources, both structured and unstructured. This requires frequent data transfers, which are often time sensitive. Without automation, basic extract, transform and load (ETL) operations often account for about 70 percent of the time and effort needed to establish a data warehouse – those tasks are not a good use of specialist machine learning resources. Scripting provides a semi-automated way to manage data, but scripts take time to develop and maintain. Again, it is best for specialists to spend their time on data science, not data processing.

The heart of machine learning development is creating test modules. This requires developing the modules themselves plus the workflows that will allow them to execute. Organizations have traditionally used different toolsets, but now developers can create workflows using the same development tools used to create the modules.

Testing follows, and has typically required a lot of manual configuration for the test environment, then debugging and rework of the module itself. These tasks can also be automated. Once testing and debugging are complete it's time to promote the module to the production environment, which usually requires a fair amount of configuration time using another set of tools.

The process above is typical and may need to be repeated several times if the tools used are specific to data types or production environments (e.g. cloud, mainframe).

There are some tools to help, like Azure Machine Learning, Facebook's FBLearner Flow and Amazon Machine Learning. These tools make it easier to get involved with machine learning because they simplify the processes for creating models. However, they do not seriously address the underlying data pipelines that feed machine learning modules. The data pipelines remain complex and often involve coordinating activity among mainframe distributed systems and other systems of record. Unless tools can provide automation that addresses the entire data pipeline for machine learning – ingestion, preparation and processing – scalability will be limited.

There are automation tools available for most steps in the process. Yet the risk of creating silos exists when different tools are used for different environments. Silos can lead to duplication of effort and limited scalability. For example, how well will a tool developed for a public cloud service work for an enterprise mainframe where key workflows and much of the data that power machine learning resides? Silos are a growing concern as hybrid IT architectures are becoming more popular.

It clearly takes a lot of plumbing to keep machine learning modules and insights flowing. If machine learning teams are doing the plumbing, they're not working on fine-tuning the modules and delivering insights.

The way forward

Some development tools and environments are much easier to integrate than others, and the options also vary considerably in how much of the lifecycle they address. When selecting tools for machine learning, their abilities to integrate and automate should be highly valued. Integration and automation save time, which means you can deploy more machine learning modules in less time. Here are some questions to ask when assessing tool options:

- How are data transfers and file transfers managed? Is a separate solution required?
- What happens if data flow to the machine learning module is interrupted? Once resolved, does it have to restart from the beginning?
- Does the component/solution require staff to learn a new environment, or can they work in what they're already familiar with?
- Does it support the technologies and environments you're using, such as Spark, Hive, Scala, R, Docker, microservices, etc.?
- Can the machine learning workloads be scheduled and managed with other enterprise workloads, or is a separate scheduler required?
- What is the process for promoting machine learning modules to production?
- Can execution be monitored through our current workload automation environment, or is a separate system required? If so, who will monitor it – the data science team or operations?

Carefully consider these questions so you can prevent your data scientists from becoming data processors.

Data scientists have been ranked as the top job in the U.S. for two years running[1]. It's a tough job to do, and a tough job to fill. If you're fortunate enough to have the talent, you need to give them the right tools to do the job.

[1] Glassdoor "Glassdoor Reveals The Best 50 Jobs in America For 2017" January 24, 2017. See http://www.prnewswire.com/news-releases/glassdoor-reveals-the-50-best-jobs-in-america-for-2017-300395188.html