CHAPTER 2: Applying Good Engineering Principles to Machine Learning

Many data science and machine learning projects fail due to preventable issues that have been resolved in software engineering for more than a decade. However, those solutions need to be adapted due to key differences between developing code and training ML models.

- **Expertise, code and data** — With the addition of data, data science and ML, code not only needs to deal with data dependencies but also handle the inherent nondeterministic characteristics of statistical modeling. ML models are not guaranteed to behave the same way when trained twice, unlike traditional code, which can be easily unit tested.
- **Model artifacts** — In addition to application code, ML products and features also depend on models that are the result of a training process. Those model artifacts can often be large — on the order of gigabytes — and often need to be served differently from code itself.
- **Collaboration** — In large organizations, models that are deployed in an application are usually not trained by the same people responsible for the deployment. Handoffs between experimentation, testing and production deployments are similar but not identical to approval processes in software engineering.

The need for standardization

Some of the world's largest tech companies have already begun solving these problems internally with their own machine learning platforms and lifecycle management tools.[2] These internal platforms have been extremely successful and are designed to accelerate the ML lifecycle by standardizing the process of data preparation, model training, and deployment via APIs built for data scientists. The platforms not only help standardize the ML lifecycle but also play a major role in retaining knowledge and best practices, and maximizing data science team productivity and collaboration, thereby leading to greater ROI.

Internally driven strategies still have limitations. First, they are limited to a few algorithms or frameworks. Adoption of new tools or libraries can lead to significant bottlenecks. Of course, data scientists always want to try the latest and the best algorithms, libraries and frameworks — the most recent versions of PyTorch, TensorFlow and so on. Unfortunately, production teams cannot easily incorporate these into the custom ML platform without significant rework. The second limitation is that each platform is tied to a specific company's infrastructure. This can limit sharing of efforts among data scientists. As each framework is so specific, options for deployment can be limited.

The question then is: Can similar benefits to these systems be provided in an open manner? This evaluation must be based on the widest possible mix of tools, languages, libraries and infrastructures. Without this approach, it will be very difficult for data scientists to evolve their ML models and keep pace with industry developments. Moreover, by making it available as open source, the wider industry will be able to join in and contribute to ML's wider adoption. This also makes it easier to move between various tools and libraries over time.

2 Facebook has implemented its FBLearner Flow platform. Uber has a service called Michelangelo. And Google has TFX.

CHAPTER 3: Introducing MLflow

MATEI ZAHARIA
Co-founder and Chief Technologist at Databricks

At Databricks, we believe that there should be a better way to manage the ML lifecycle. So in June 2018, we unveiled MLflow, an open-source machine learning platform for managing the complete ML lifecycle.

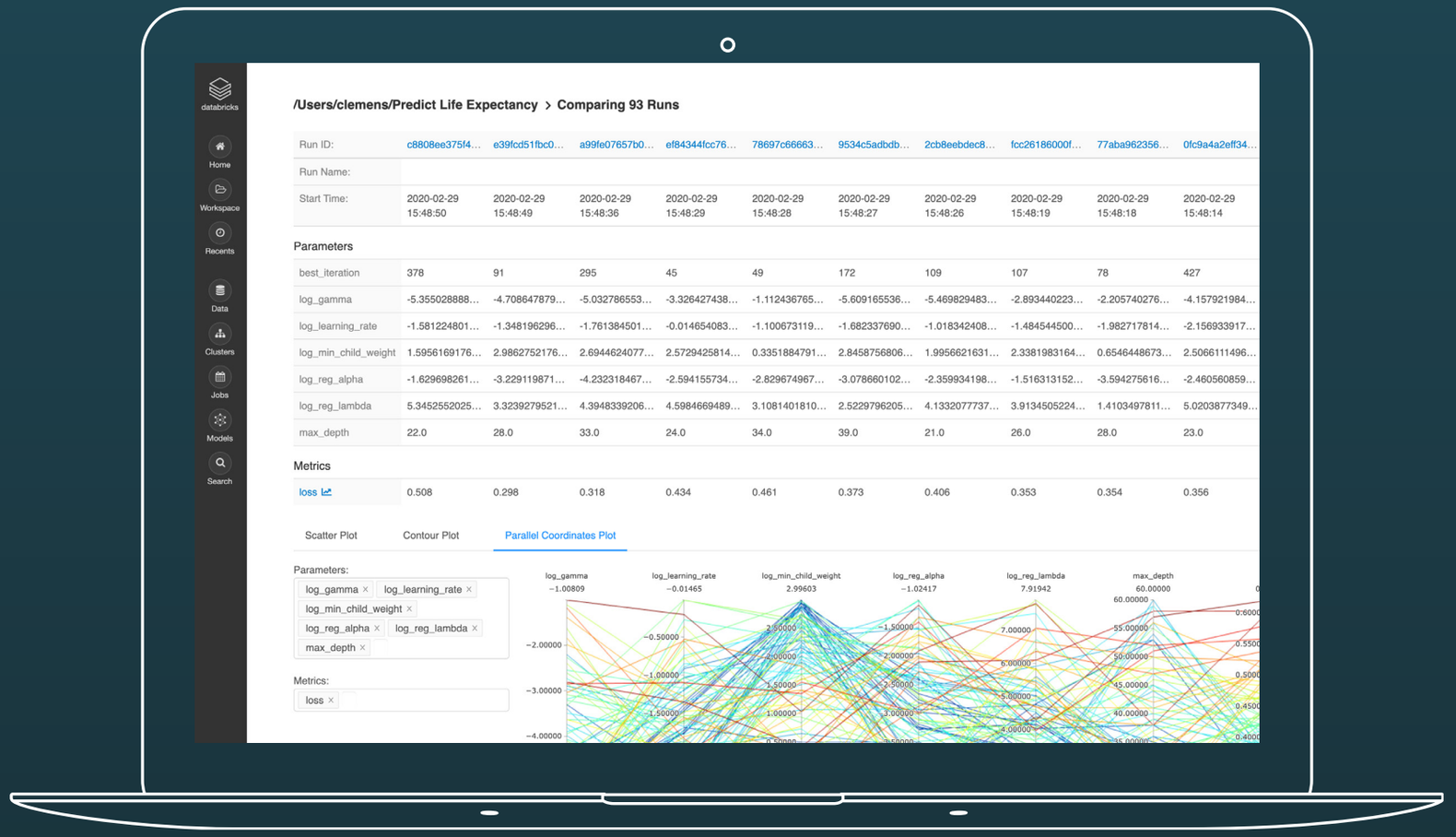
"MLflow is designed to be a cross-cloud, modular, API-first framework, to work well with all popular ML frameworks and libraries. It is open and extensible by design, and platform agnostic for maximum flexibility."

With MLflow, data scientists can now package code as reproducible runs, execute and compare hundreds of parallel experiments, and leverage any hardware or software platform for training, hyperparameter tuning and more. Also, organizations can deploy and manage models in production on a variety of clouds and serving platforms.

"With MLflow, data science teams can systematically package and reuse models across frameworks, track and share experiments locally or in the cloud, and deploy models virtually anywhere," says Zaharia. "The flurry of interest and contributions we've seen from the data science community validates the need for an open-source framework to streamline the machine learning lifecycle."

Key benefits

EXPERIMENT TRACKING As mentioned previously, getting ML models to perform takes significant trial and error, and continuous configuration, building, tuning, testing, etc. Therefore, it is imperative to allow data science teams to track all that goes into a specific run, along with the results. With MLflow, data scientists can quickly record runs and keep track of model parameters, results, code and data from each experiment, all in one place.

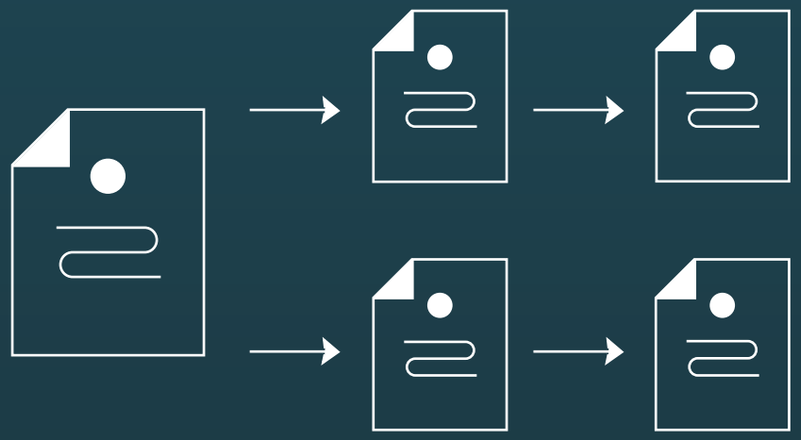


databricks

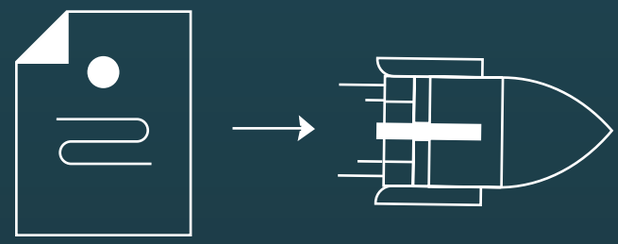
mlflow

Key benefits

REPRODUCIBLE PROJECTS The ability to reproduce a project — entirely or just parts of it — is key to data science productivity, knowledge sharing and, hence, accelerating innovation. With MLflow, data scientists can build and package composable projects, capture dependencies and code history for reproducible results, and quickly share projects with their peers.



FLEXIBLE DEPLOYMENT There is virtually no limit to what machine learning can do for your business. However, there are different ways to architect ML applications for production, and various tools can be used for deploying models, which often lead to code rewrites prior to deploying ML models into production. With MLflow, your data scientists can quickly download or deploy any saved models to various platforms — locally or in the cloud — from experimentation to production.



Key benefits

MODEL MANAGEMENT Use one central place to share ML models, collaborate on moving them from experimentation to online testing and production, integrate with approval and governance workflows, and monitor ML deployments and their performance. This is powered by the latest MLflow component, MLflow Model Registry.