



mlflow Platform for Complete Machine Learning Lifecycle

Jules S. Damji
[@2twitme](https://twitter.com/2twitme)

San Francisco | April 29, 2020: Part 1 of 4 Series

\$ whoami



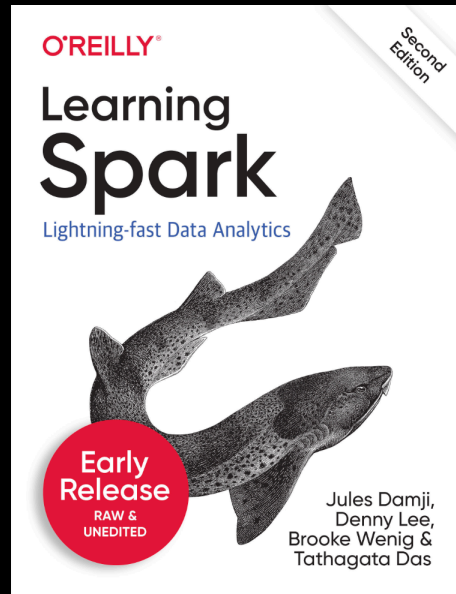
Apache Spark Developer & Community Advocate @ Databricks

Developer Advocate @ Hortonworks

Software engineering @ Sun Microsystems, Netscape, @Home, Excite@Home, VeriSign, Scalix, Centrify, LoudCloud/Opware, ProQuest

Program Co-chair Spark + AI Summit

[@2twitme](https://www.linkedin.com/in/dmatrix)





VISION

Accelerate innovation by unifying data science, engineering and business to solve data problems

SOLUTION

Unified Data Analytics Platform

WHO WE ARE

- Original creators of  **APACHE Spark**™
 -  **DELTA LAKE**
 -  **mlflow**™
 -  **Koalas**
- 2000+ global companies use our platform across big data & machine learning lifecycle

Outline – Part 1

- Overview of ML development challenges
- Concepts and Motivations
- How MLflow tackles these
- MLflow Components
 - MLflow Tracking
 - Build and Track metrics, params, runs
 - User MLflow UI to compare runs
- Q & A

Machine Learning Development is Complex

Traditional Software

Goal: Meet a functional specification

Quality depends only on code

Typically pick one software stack

Machine Learning

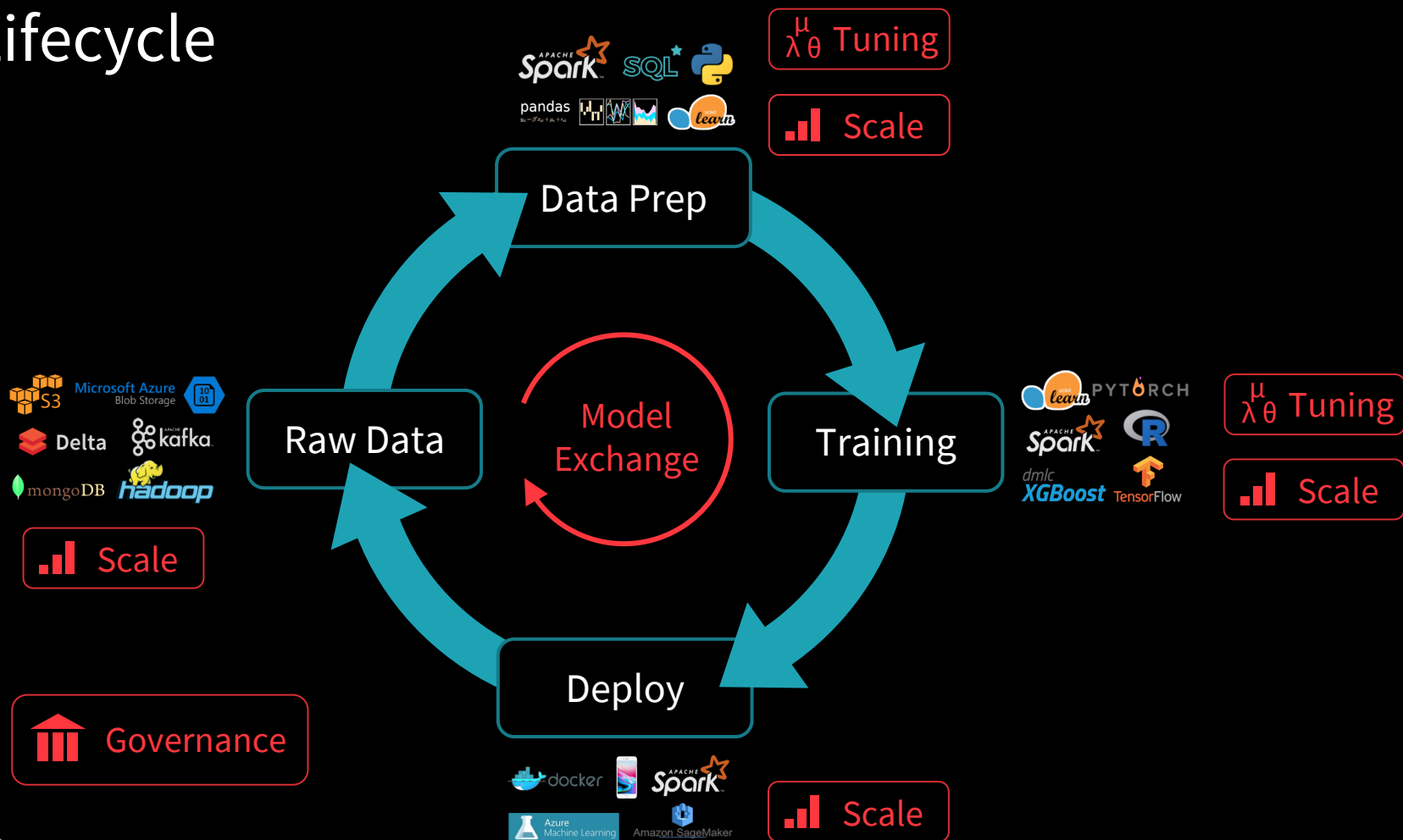
Goal: Optimize a metric (e.g., accuracy)
Constantly experiment to improve it

Quality depends on input data
and tuning parameters



Compare + combine many libraries,
models & algorithms for the same task

ML Lifecycle



Custom ML Platforms

Some Big Data Companies

- + **Standardize the data prep / training / deploy loop:**
if you work with the platform, you get these!
- Limited to a few algorithms or frameworks
- Tied to one company's infrastructure
- Out of luck if you left the company....

Can we provide similar benefits in an **open manner?**

Introducing mlflow

Open machine learning platform

- Works with popular ML library & language
- Runs the same way anywhere (e.g., any cloud or locally)
- Designed to be useful for 1 or 1000+ person orgs
- *Simple. Modular. Easy-to-use.*
- *Offers positive developer experience to get started!*

MLflow Design Philosophy

“API-first”

- Submit runs, log models, metrics, etc. from popular library & language
- Abstract “model” lambda function that MLflow can then deploy in many places (Docker, Azure ML, Spark UDF)
- Open interface allows easy integration from the community

**Key enabler: built around
Programmatic APIs, REST APIs & CLI**

Modular design

- Allow different components individually (e.g., use MLflow’s project format but not its deployment tools)
- Not monolithic
- But Distinctive and Selective

**Key enabler: distinct components
(Tracking/Projects/Models/Registry)**

MLflow Components

mlflow Tracking

Record and query experiments: code, data, config, and results

mlflow Projects

Package data science code in a format that enables reproducible runs on any platform

mlflow Models

Deploy machine learning models in diverse serving environments

new

mlflow Model Registry

Store, annotate and manage models in a central repository

[databricks.com/
mlflow](https://databricks.com/mlflow)



mlflow.org



github.com/mlflow



twitter.com/MLflow

Key Concepts in MLflow Tracking

Parameters: key-value inputs to your code

Metrics: numeric values (can update over time)

Tags and Notes: information about a run

Artifacts: files, data, and models

Source: what code ran?

Version: what of the code?

Model Development without MLflow

```
data    = load_text(file)
ngrams  = extract_ngrams(data, N=n)
model   = train_model(ngrams,
                      learning_rate=lr)
score   = compute_accuracy(model)

print("For n=%d, lr=%f: accuracy=%f"
      % (n, lr, score))

pickle.dump(model, open("model.pkl"))
```

```
For n=2, lr=0.1: accuracy=0.71
For n=2, lr=0.2: accuracy=0.79
For n=2, lr=0.5: accuracy=0.83
For n=2, lr=0.9: accuracy=0.79
For n=3, lr=0.1: accuracy=0.83
For n=3, lr=0.2: accuracy=0.82
For n=4, lr=0.5: accuracy=0.75
...
```

What version of
my code was this
result from?

MLflow Tracking API: *Simple & Pythonic!*

mlflow
Tracking

Record and query
experiments: code,
configs, results,
...etc

```
import mlflow
import mlflow.tensorflow

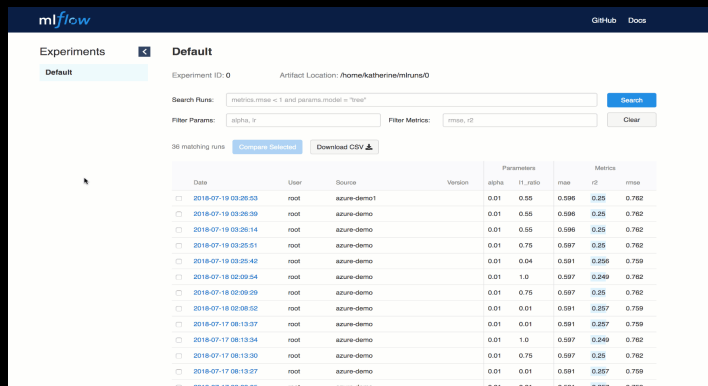
# log model's tuning parameters
with mlflow.start_run() as run:
    mlflow.log_param("layers", layers)
    mlflow.log_param("alpha", alpha)

# log metrics and model
mlflow.log_metric("mse", model.mse())
mlflow.log_artifact("plot", model.plot(test_df))
mlflow.tensorflow.log_model(model)
```

Model Development *with* MLflow is *Simple*!

```
data    = load_text(file)
ngrams  = extract_ngrams(data, N=n)
model   = train_model(ngrams,
                      learning_rate=lr)
score   = compute_accuracy(model)
with mlflow.start_run() as run:
    mlflow.log_param("data_file", file)
    mlflow.log_param("n", n)
    mlflow.log_param("learn_rate", lr)
    mlflow.log_metric("score", score)
    mlflow.sklearn.log_model(model)
```

```
$ mlflow ui
```



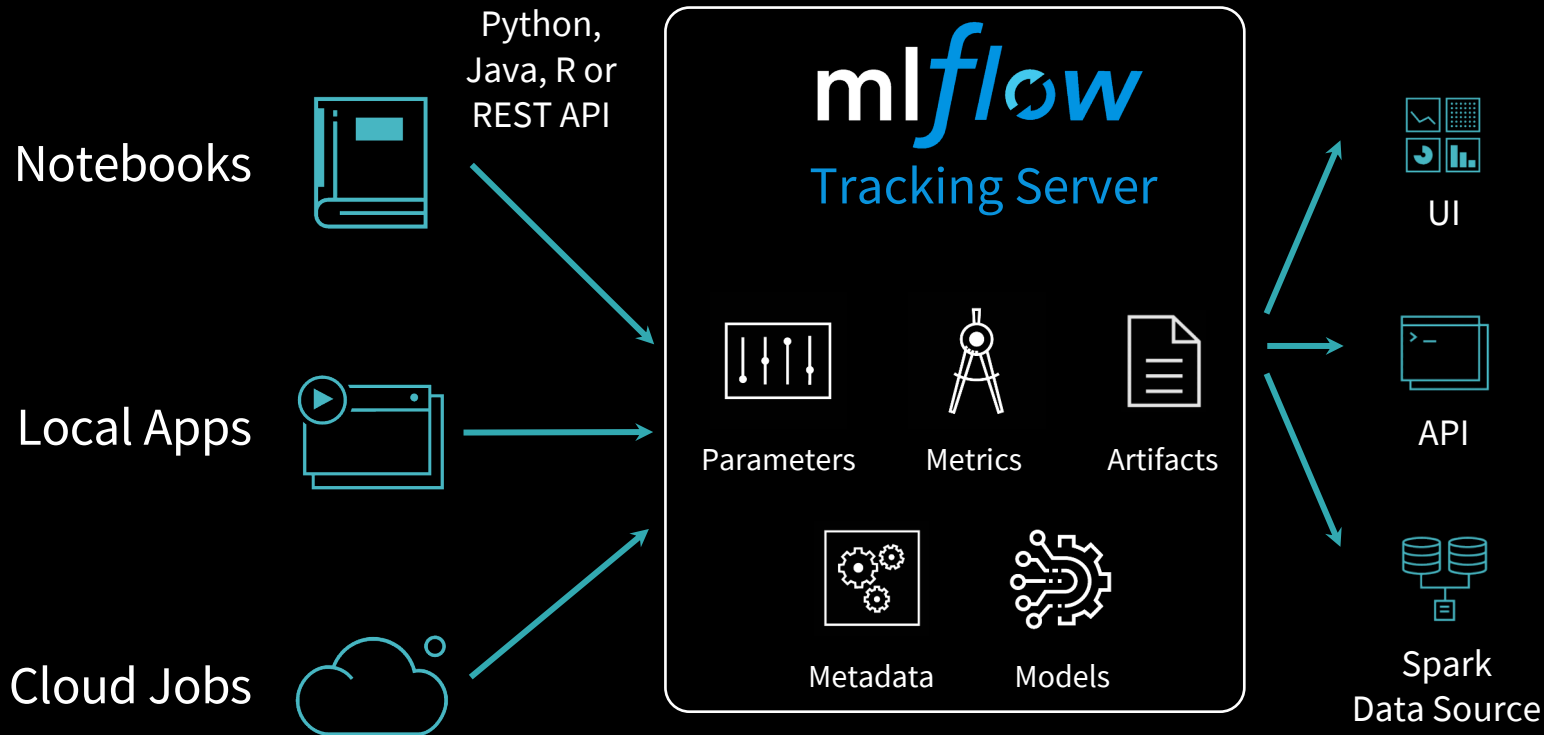
MLflow Experiments page showing a table of runs. The table has columns: Date, User, Source, Version, Parameters, and Metrics. The metrics column shows 'max' and 'min' values. The table lists 16 runs, all with a score of 0.25.

Date	User	Source	Version	Parameters	Metrics
2018-07-19 03:26:53	root	azure-demo1	0.01	0.55	0.595 0.25 0.762
2018-07-19 03:26:59	root	azure-demo	0.01	0.55	0.595 0.25 0.762
2018-07-19 03:26:14	root	azure-demo	0.01	0.55	0.595 0.25 0.762
2018-07-19 03:25:51	root	azure-demo	0.01	0.75	0.587 0.25 0.762
2018-07-19 03:25:42	root	azure-demo	0.01	0.04	0.591 0.255 0.759
2018-07-19 02:09:54	root	azure-demo	0.01	1.0	0.597 0.249 0.762
2018-07-18 02:09:29	root	azure-demo	0.01	0.75	0.597 0.25 0.762
2018-07-18 02:08:52	root	azure-demo	0.01	0.01	0.591 0.257 0.759
2018-07-17 08:13:27	root	azure-demo	0.01	0.01	0.591 0.257 0.759
2018-07-17 08:13:34	root	azure-demo	0.01	1.0	0.597 0.249 0.762
2018-07-17 08:13:30	root	azure-demo	0.01	0.75	0.597 0.25 0.762
2018-07-17 08:13:27	root	azure-demo	0.01	0.01	0.591 0.257 0.759
2018-07-17 08:08:05	root	azure-demo	0.01	0.01	0.591 0.257 0.759

Track parameters, metrics,
output files & code version

Search using UI or API

MLflow Tracking



```
$ export MLFLOW_TRACKING_URI <URI>  
mlflow.set_tracking_uri(URI)
```


MLflow Tracking Backend Stores

1. Entity (Metadata) Store

- FileStore (local filesystem)
- SQLStore (via SQLAlchemy)
 - PostgreSQL, MySQL, SQLite

2. Artifact Store

- S3 backed store
- Azure Blob storage
- Google Cloud storage
- DBFS artifact repo

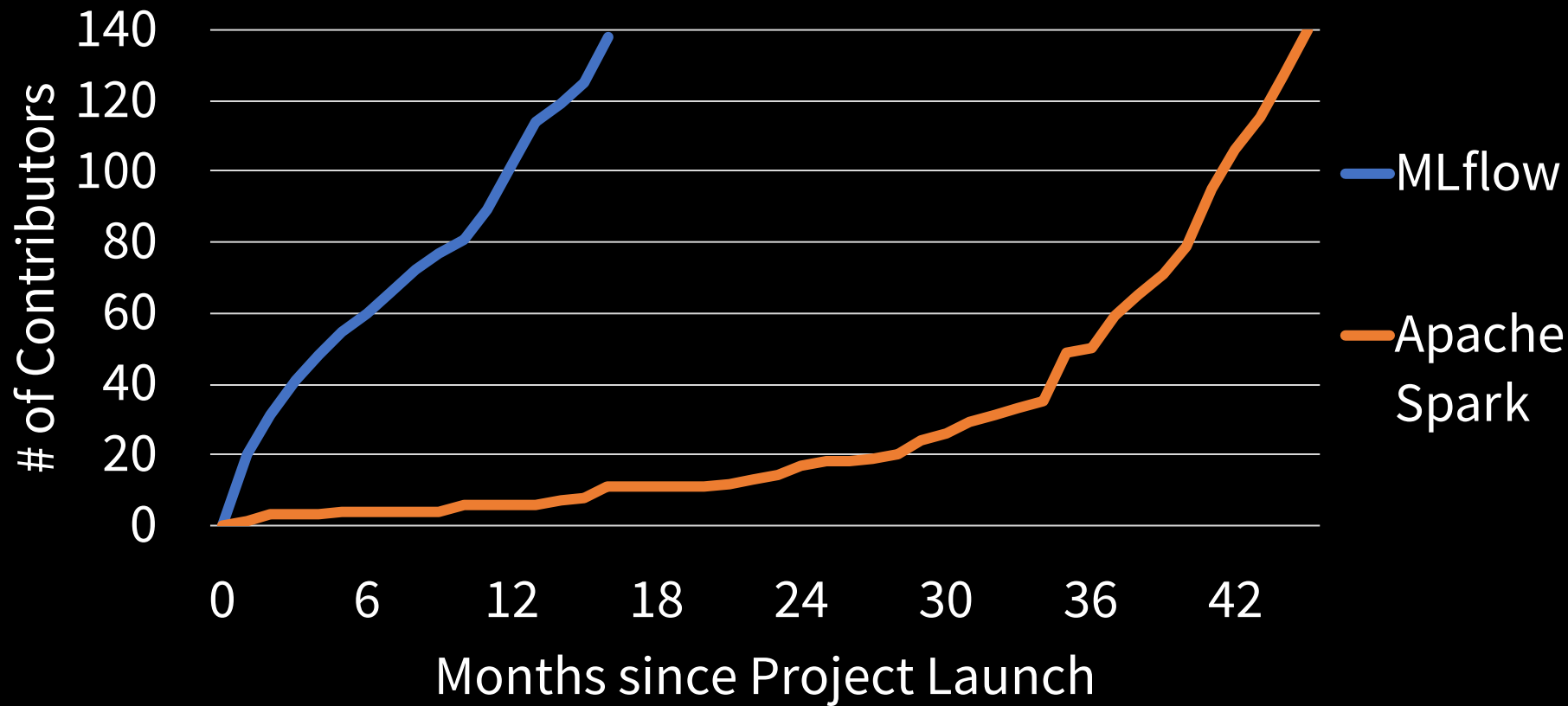
What Did We Talk About?

Modular Components greatly simplify the ML lifecycle

- Available APIs: Python, Java & R (Soon Scala)
- Easy to install and use
- Develop & Deploy locally and track locally or remotely

Project Contributors over Time

Today at 191 contributors



Learning More About MLflow

- `pip install mlflow` to get started
- Find docs & examples at mlflow.org
- <https://github.com/mlflow/mlflow>
- tinyurl.com/mlflow-slac
- dbricks.co/mlflow-tutorials

MLflow Tracking Tutorials

<https://github.com/dmatrix/mlflow-workshop-part-1>



Thank you! ☺

Q & A

jules@databricks.com

[@2twitme](#)

<https://www.linkedin.com/in/dmatrix/>