



# Introduction to **mlflow**

Matei Zaharia & Aaron Davidson

June 28, 2018



# Outline

**ML development challenges**

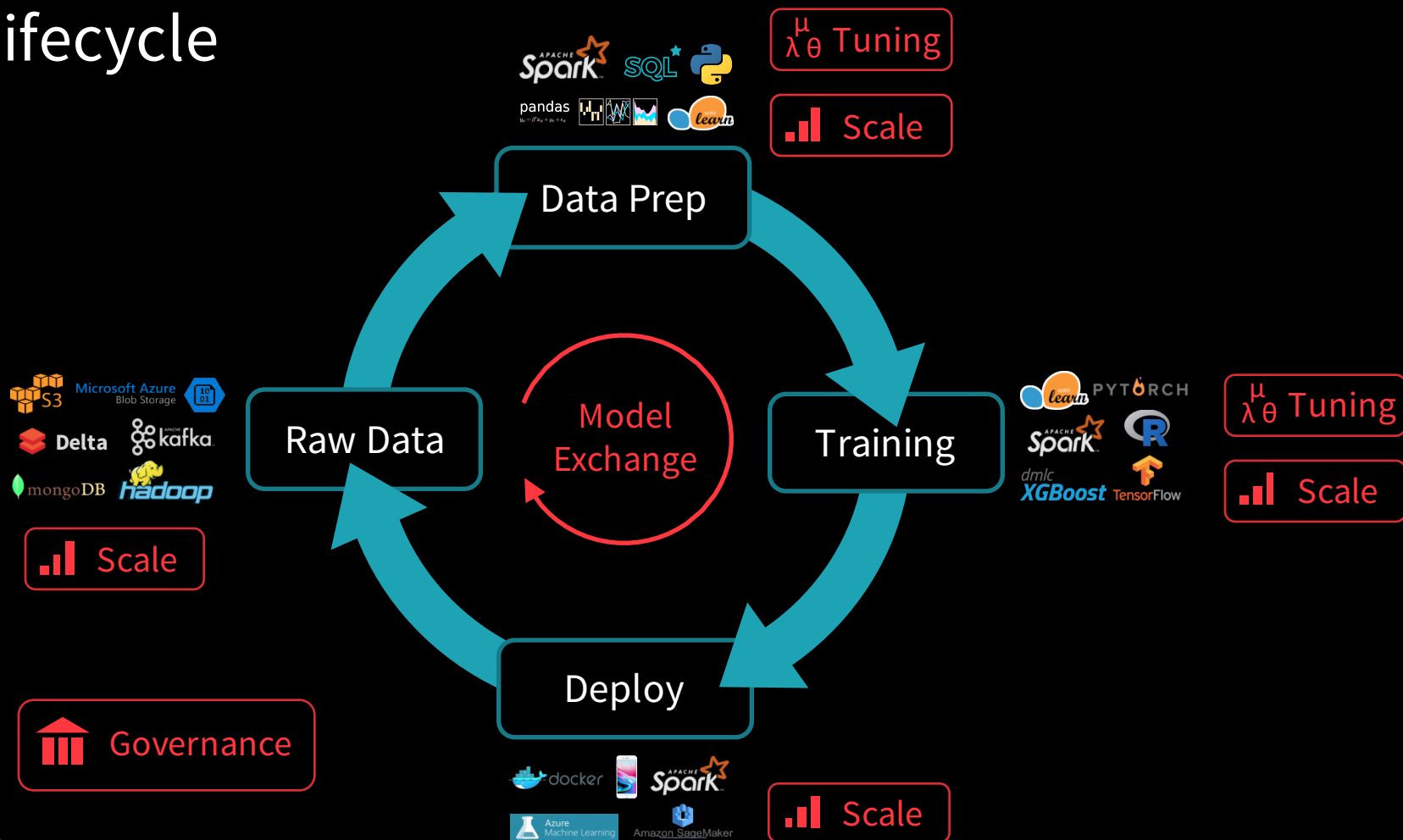
**How MLflow tackles these**

**Demo**

**Roadmap**

# Machine Learning Development is Complex

# ML Lifecycle



# ML Development Challenges

**100s of software tools** to leverage

**Hard to track & reproduce results:** code, data, params, etc

**Hard to productionize models**

**Needs large scale** for best results

# Introducing mlflow

## **Open machine learning platform**

- Works with any ML library & language
- Runs the same way anywhere (e.g. any cloud)
- Designed to be useful for 1 or 100,000 person orgs

# MLflow Design Philosophy

## 1. “API-first”, open platform

- Allow submitting runs, models, etc from any library & language
- Example: a “model” can just be a lambda function that MLflow can then deploy in many places (Docker, Azure ML, Spark UDF, ...)

**Key enabler: built around REST APIs and CLI**

# MLflow Design Philosophy

## 2. Modular design

- Let people use different components individually (e.g., use MLflow's project format but not its deployment tools)
- Easy to integrate into existing ML platforms & workflows

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



# Why Open Source?

**Everyone is solving a similar problem**

**Lots of benefits in having a common API across orgs**

- Can open source & share individual workflow steps
- ML tool developers can easily reach lots of users
  - E.g. a new ML library can use MLflow Models to reach many serving tools

# MLflow Components

## **mlflow** Tracking

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

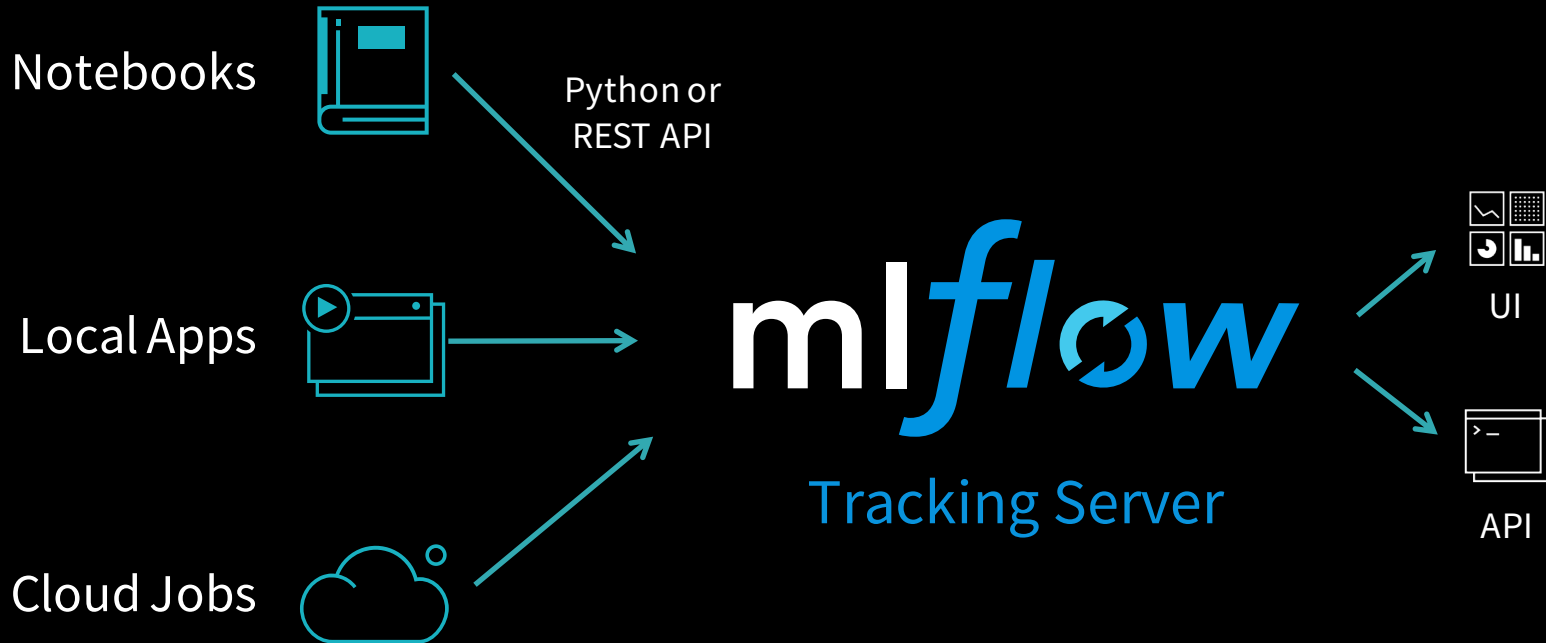
## **mlflow** Projects

Packaging format for reproducible runs on any platform

## **mlflow** Models

General model format that supports diverse deployment tools

# MLflow Tracking



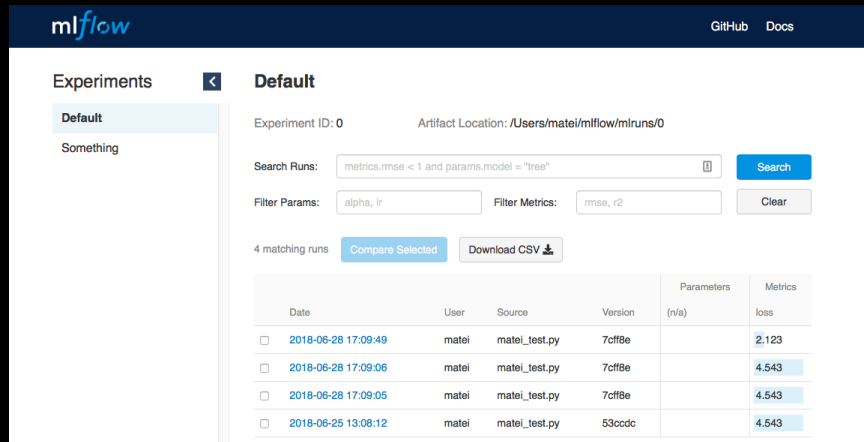
# Key Concepts in Tracking

**Parameters:** key-value inputs to your code

**Metrics:** numeric values (can update over time)

**Artifacts:** arbitrary files, including models

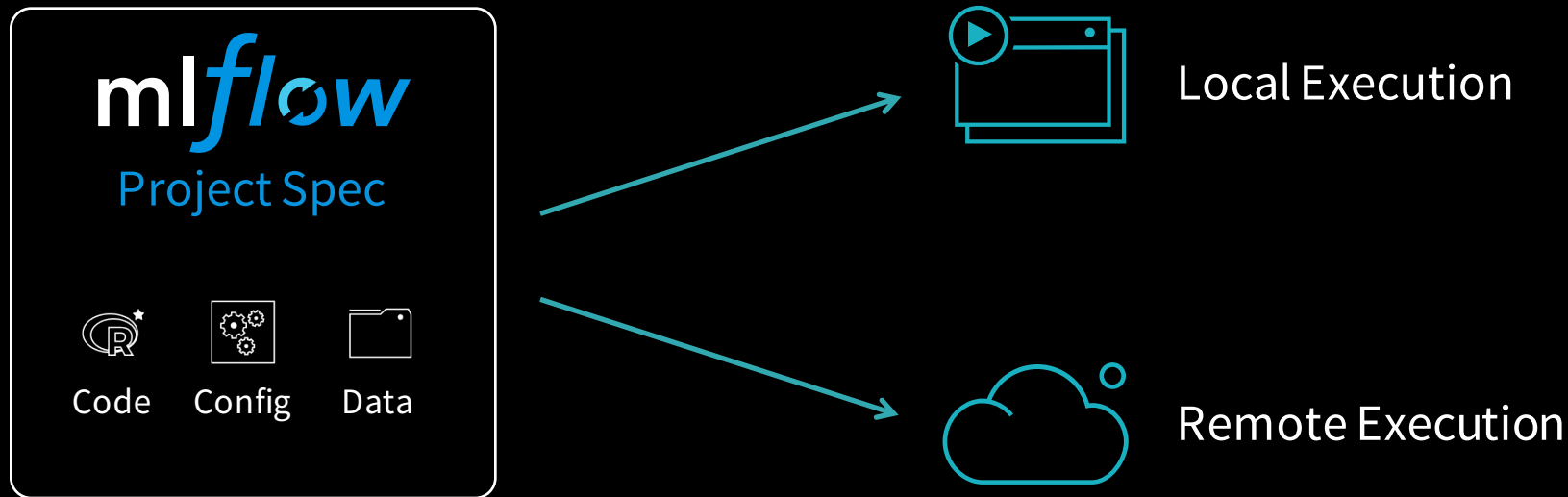
**Source:** what code ran?



The screenshot displays the mlflow web interface. On the left, a sidebar shows 'Experiments' with 'Default' selected. The main panel shows details for the 'Default' experiment, including the Experiment ID (0) and Artifact Location. Search and filter fields are present, with search results showing 4 matching runs. A table lists these runs with columns for Date, User, Source, Version, Parameters, and Metrics.

Date	User	Source	Version	Parameters (n/a)	Metrics loss
<input type="checkbox"/> 2018-06-28 17:09:49	matei	matei_test.py	7cff8e		2.123
<input type="checkbox"/> 2018-06-28 17:09:06	matei	matei_test.py	7cff8e		4.543
<input type="checkbox"/> 2018-06-28 17:09:05	matei	matei_test.py	7cff8e		4.543
<input type="checkbox"/> 2018-06-25 13:08:12	matei	matei_test.py	53ccdc		4.543

# MLflow Projects



# Example MLflow Project

my\_project/  
├── MLproject

├── conda.yaml  
├── main.py  
├── model.py  
└── ...

```
conda_env: conda.yaml
```

```
entry_points:
```

```
  main:
```

```
    parameters:
```

```
      training_data: path
```

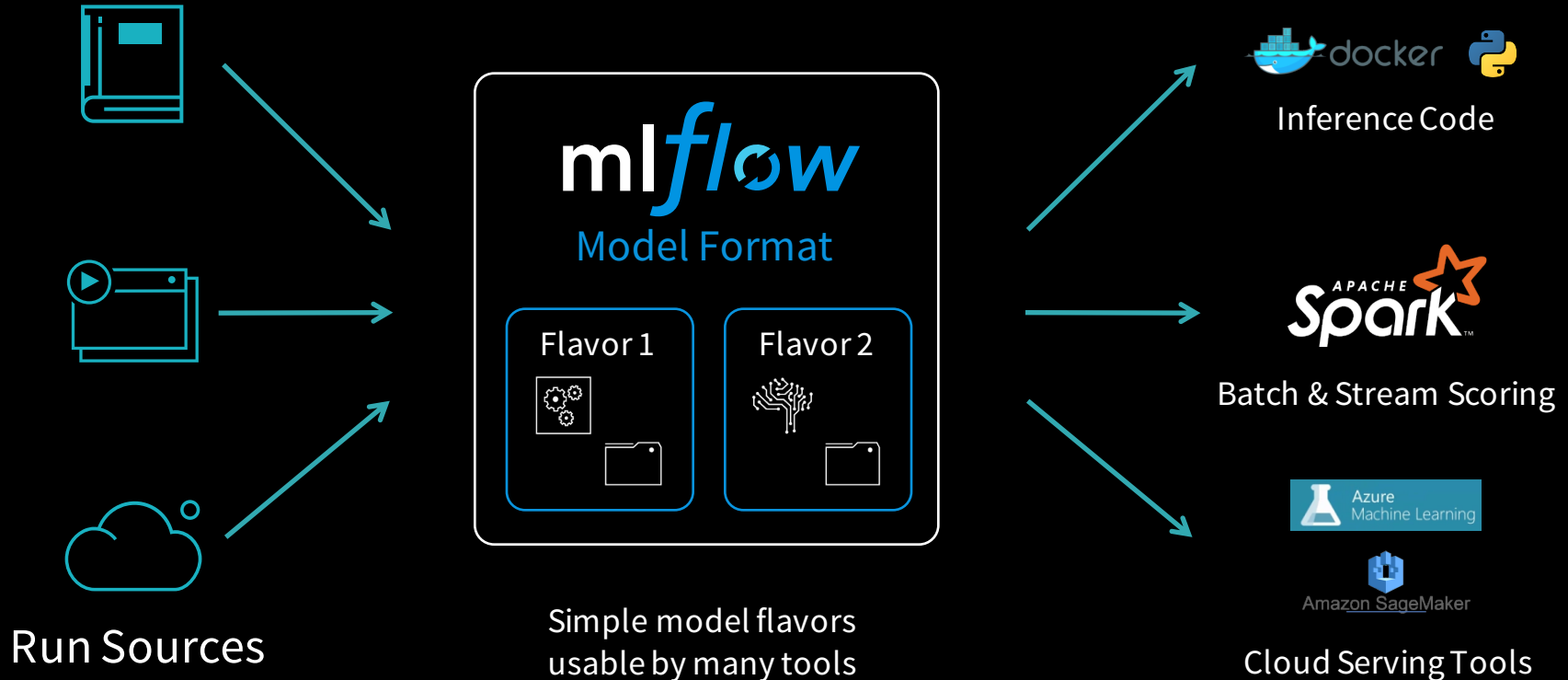
```
      lambda: {type: float, default: 0.1}
```

```
    command: python main.py {training_data} {lambda}
```

```
$ mlflow run git://<my_project>
```

```
mlflow.run("git://<my_project>", ...)
```

# MLflow Models



# Example MLflow Model

my\_model/  
└─ MLmodel

```
run_id: 769915006efd4c4bbd662461  
time_created: 2018-06-28T12:34  
flavors:
```

```
  tensorflow:  
    saved_model_dir: estimator  
    signature_def_key: predict
```

```
  python_function:  
    loader_module: mlflow.tensorflow
```

} Usable by tools that understand TensorFlow model format

} Usable by any tool that can run Python (Docker, Spark, etc!)

└─ estimator/  
 └─ saved\_model.pb  
 └─ variables/  
 ...



# Demo

# Roadmap

# Current Status

**MLflow is still alpha, so expect things to break**

- But send input or patches on GitHub!

**Just made 0.2.1 release**

- TensorFlow integration (model logging & serving)
- More robust server (multi-worker setup and S3 artifact store)
- Doc, example and API improvements

# Longer-Term Roadmap

## 1. Improving current components

- Pluggable execution backends for `mlflow.run`
- Database-backed tracking store (already a pluggable API)
- Model metadata (e.g. required input schema)
- Easier support for multi-step workflows

# Longer-Term Roadmap

## 2. MLflow Data component

- Let MLflow projects load data from diverse formats (e.g. CSV vs Parquet) so you don't have to pick a format in advance
- Will build on Spark's Data Source API

# Longer-Term Roadmap

## 3. Hyperparameter tuning

- Integrate with common hyperparameter tuning libraries
- Make it easier to launch & track many runs in parallel (already possible but kind of awkward)

# Longer-Term Roadmap

## 4. Language and library integrations

- Java and R are high on our list for APIs
- Built-in Spark MLlib and PyTorch integrations
- Demonstrate how to use MLflow with other libraries (it's easy)

**Let us know if you have other roadmap ideas!**

# Contributing to MLflow

## **Submit issues and patches on GitHub**

- We're using it for all our development & issue tracking
- See CONTRIBUTING.rst for how to run dev builds

**Join our mailing list: [tinyurl.com/mlflow-users](https://tinyurl.com/mlflow-users)**

**Join our Slack: [tinyurl.com/mlflow-slack](https://tinyurl.com/mlflow-slack)**



# Conclusion

## **Powerful workflow tools can simplify the ML lifecycle**

- Improve usability for both data scientists and engineers
- Same way that software dev lifecycle tools simplify dev

**MLflow is a lightweight, open platform that integrates easily into existing workflows**

The MLflow logo, featuring the word "ml" in white and "flow" in blue, with a circular arrow icon integrated into the letter "o".