MLflow: A Platform for Productionizing Machine Learning

Matei Zaharia
Databricks and Stanford University

databricks

Outline

- Challenges using ML in industry
- ML Platforms as an emerging abstraction
- MLflow overview and interesting use cases

About Databricks

Data and ML cloud platform used by >7000 customers

- Millions of VMs and 100,000s of users

Some of our ML customers:



ML is Being Adopted for Critical Applications

Price insurance policies based on data and ML
(core business for >90 years!)

Manage inventory and supply chains
(affects company's entire cost/revenue)

HSBC

Fraud detection & personalization on 170 PB data
(directly impacts profitability & compliance)

But ML is Different from Traditional Software

Traditional Software

Goal: meet a functional specification

Quality depends only on application code

Pick one software stack

Machine Learning

Goal: optimize a metric (e.g. prediction accuracy)

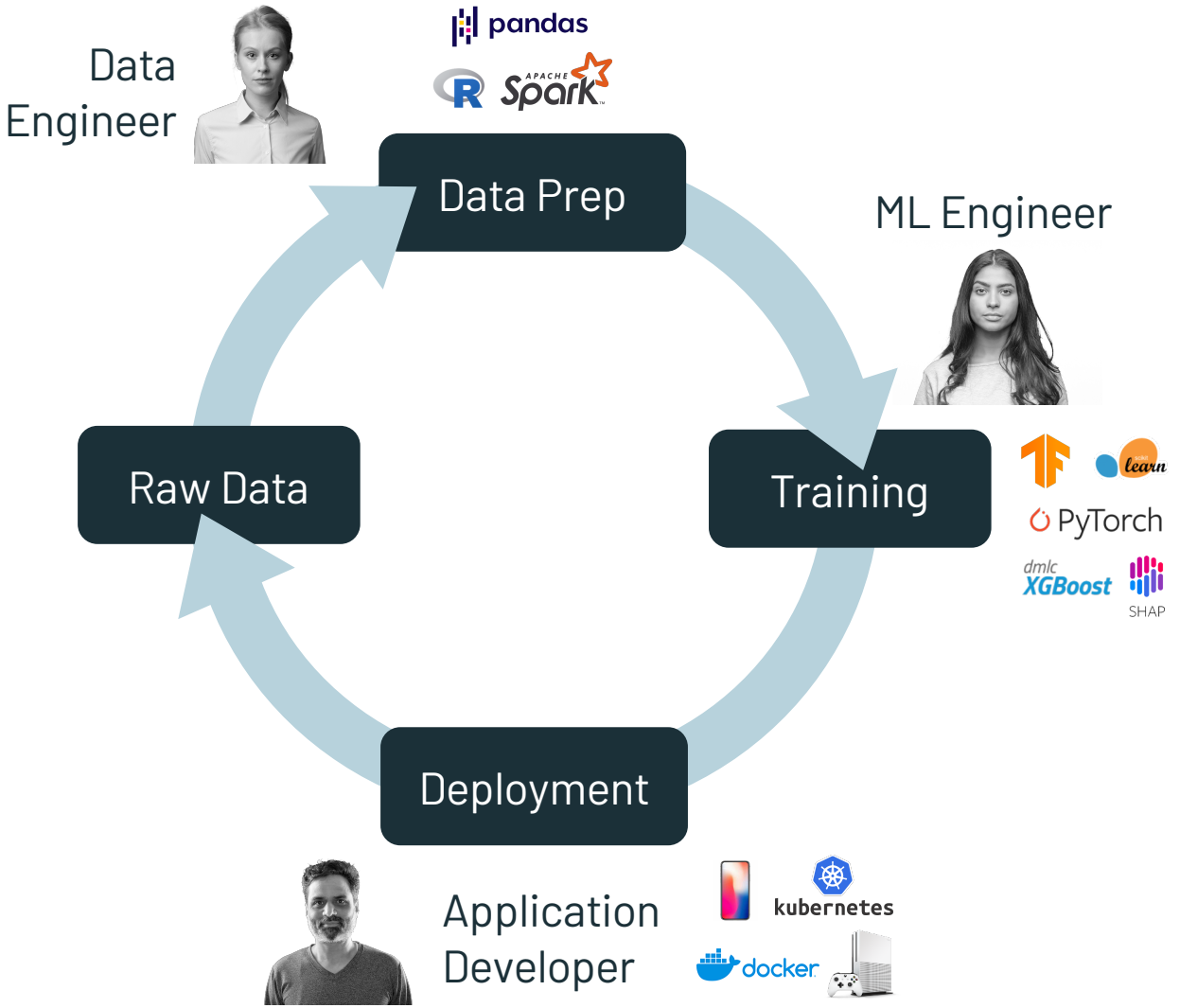
Quality depends on input data and tuning parameters

Compare + combine many libraries and algorithms for the same task

Production ML is Even Harder

ML apps must be fed new data to keep working

Design, retraining & inference often done by different people



databricks

A Solution is Emerging: ML Platforms

Software to manage the ML development and deployment process, from data to experimentation to production

Examples: Google TFX, Facebook FBLearner, Uber Michelangelo

Typical concerns:

- Data management
- Experiment management
- Model management
- Deployment for inference
- Reproducibility
- Testing & monitoring

All through a consistent interface!

ML Platforms in 2018

Each company largely designing its own platform, with limited scope

- Specific libraries (e.g. TensorFlow for TFX)
- Specific deployment environment (e.g. Kubernetes on AWS)

ML Platform team often becomes a bottleneck in the organization

Can we provide the same benefits with an open platform?

mlflow: An Open Source ML Platform

Based on an **open interface** design philosophy: make it easy to connect arbitrary ML code & tools into the platform

- Simple command-line and REST APIs rather than environment-specific
- Easy to add to existing software

mlflow TRACKING

Experiment and metric tracking

mlflow PROJECTS

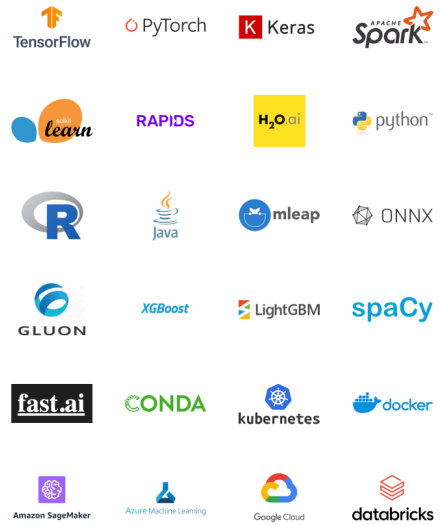
Reproducible execution

mlflow MODELS

Model packaging and deployment

mlflow MODEL REGISTRY

Model management



Built-in connectors

databricks

mlflow Community

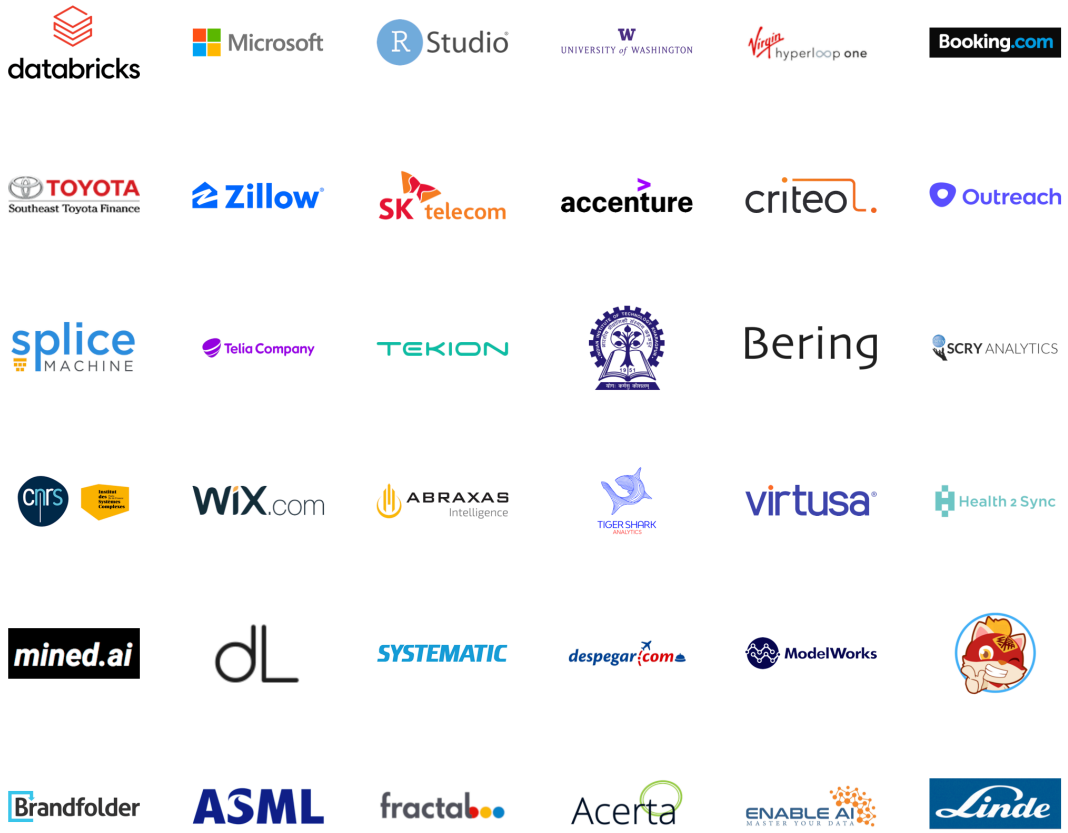
2 million downloads/month on PyPI

260 open source contributors

4x annual growth

1.5M runs/week on Databricks

Users and contributors:

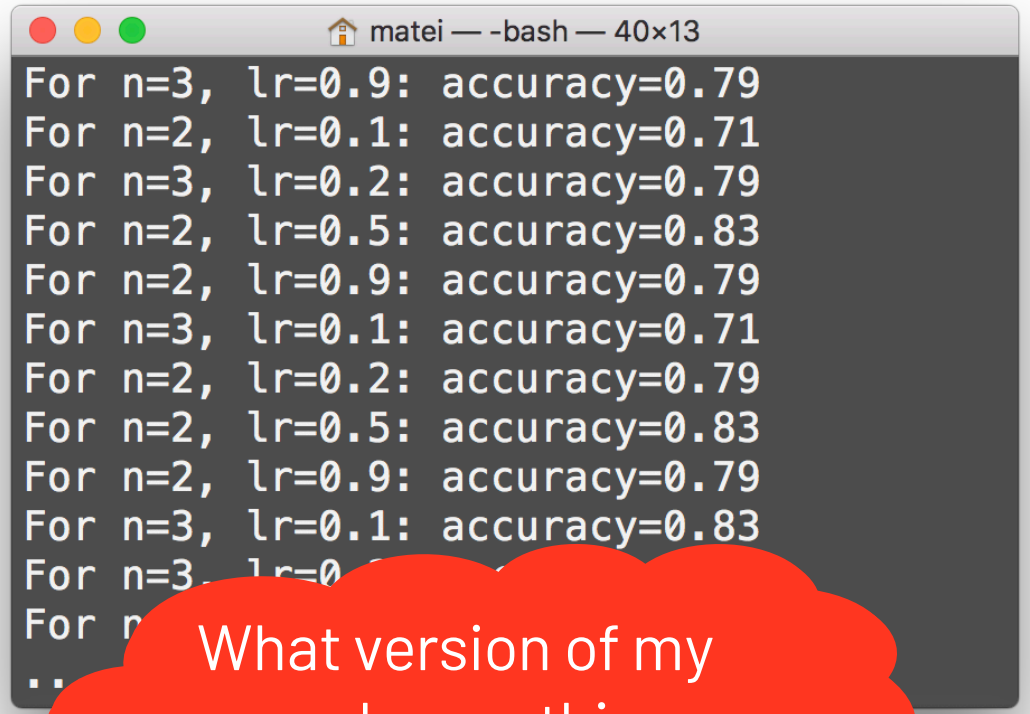


MLflow Components

MLflow Tracking

Get visibility into experiments and production runs

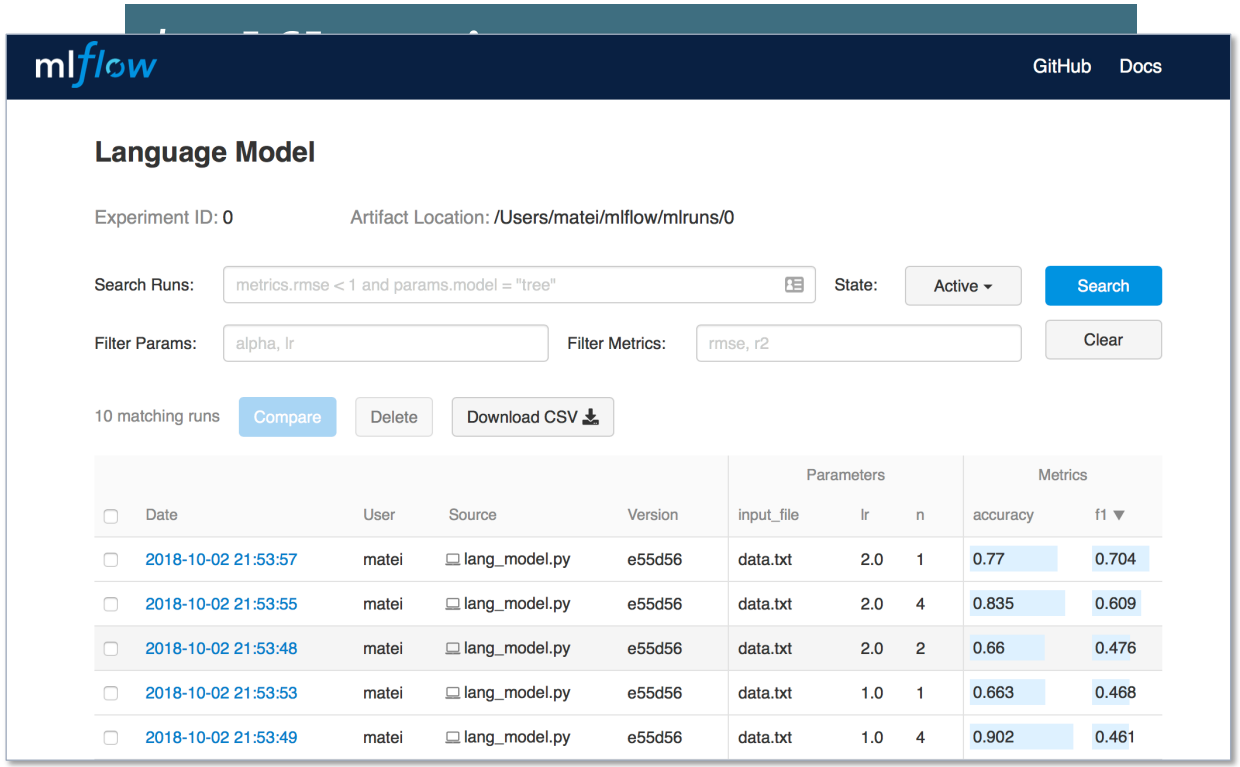
```
data   = load_text(file)
ngrams = extract_ngrams(data, N=n)
model  = train_model(ngrams,
                     learning_rate=lr)
score  = compute_accuracy(model)

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



MLflow Tracking

Get visibility into experiments and production runs

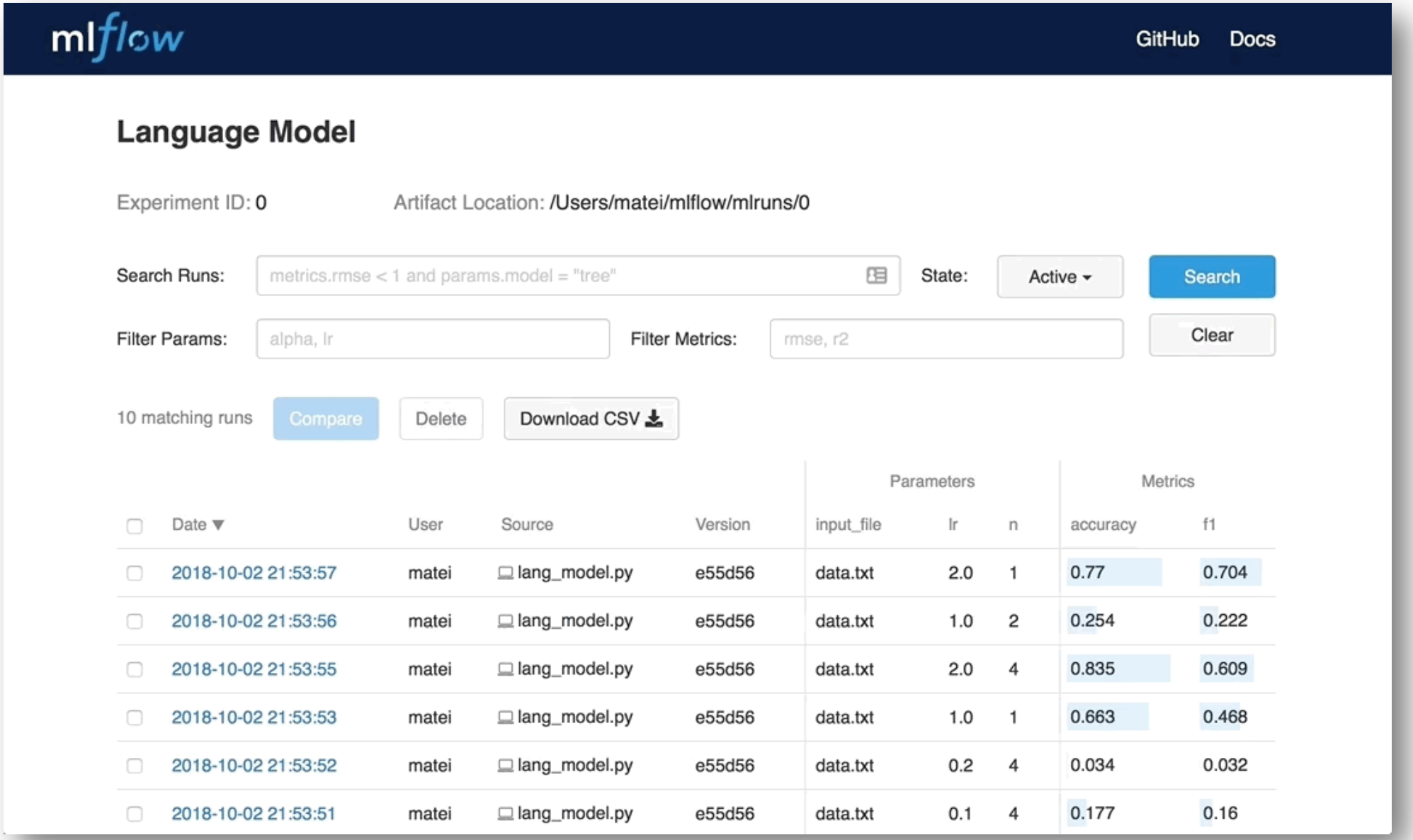
```
mlflow.keras.autolog()

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

# Or log custom info if desired
mlflow.log_param("country", "US")
```



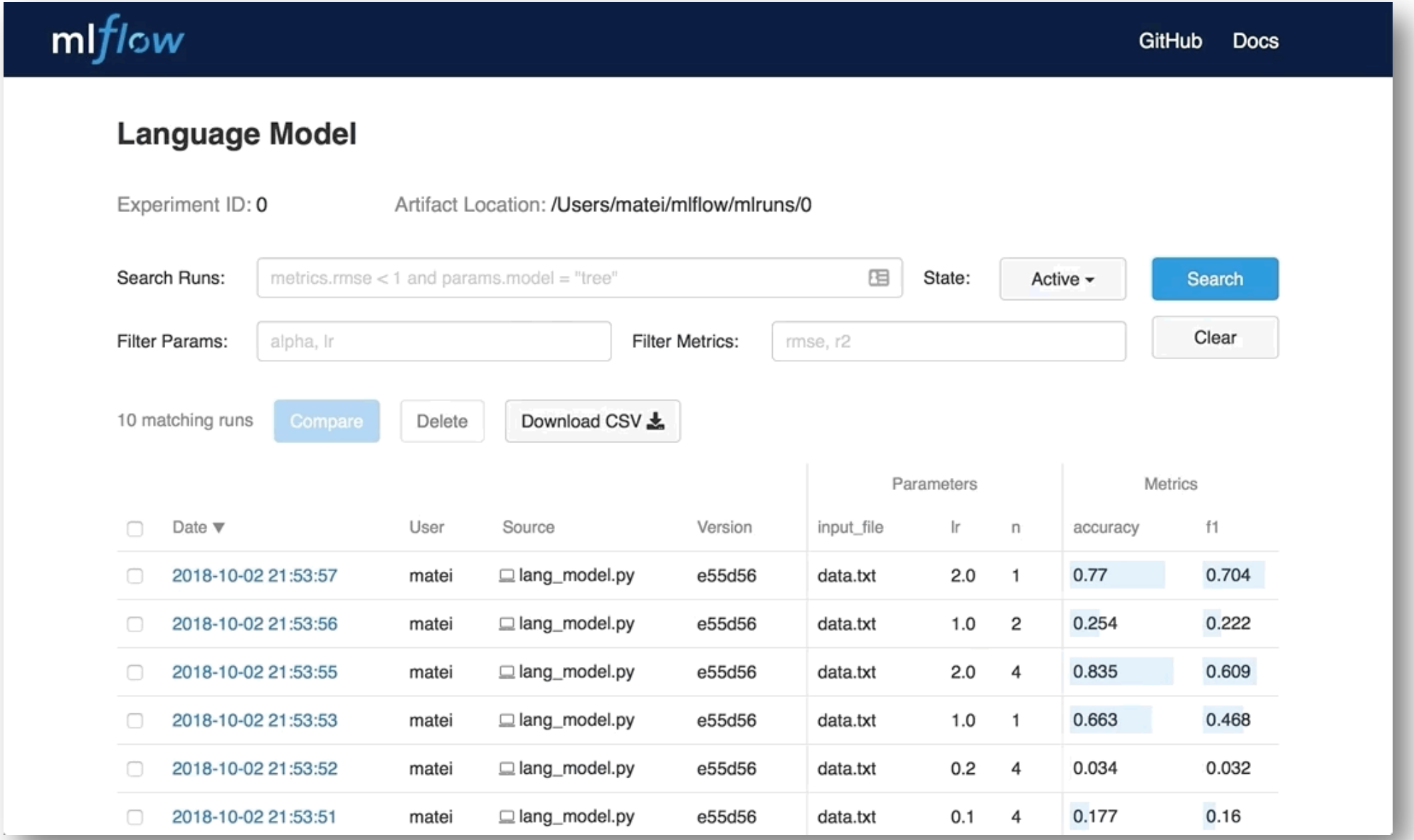
Track parameters, metrics, output files & code version

databricks

Tracking UI: Inspecting Runs



Tracking UI: Comparing Runs



MLflow Projects

Package code + dependencies for reusable workflows

```
my_project/
├── MLproject
│
│
│
│
├── conda.yaml
├── main.py
└── model.py
    ...
```

```
conda_env: conda.yaml

entry_points:
  main:
    parameters:
      training_data: path
      lr: {type: float, default: 0.1}
    command: python main.py {training_data} {lr}
```

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

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

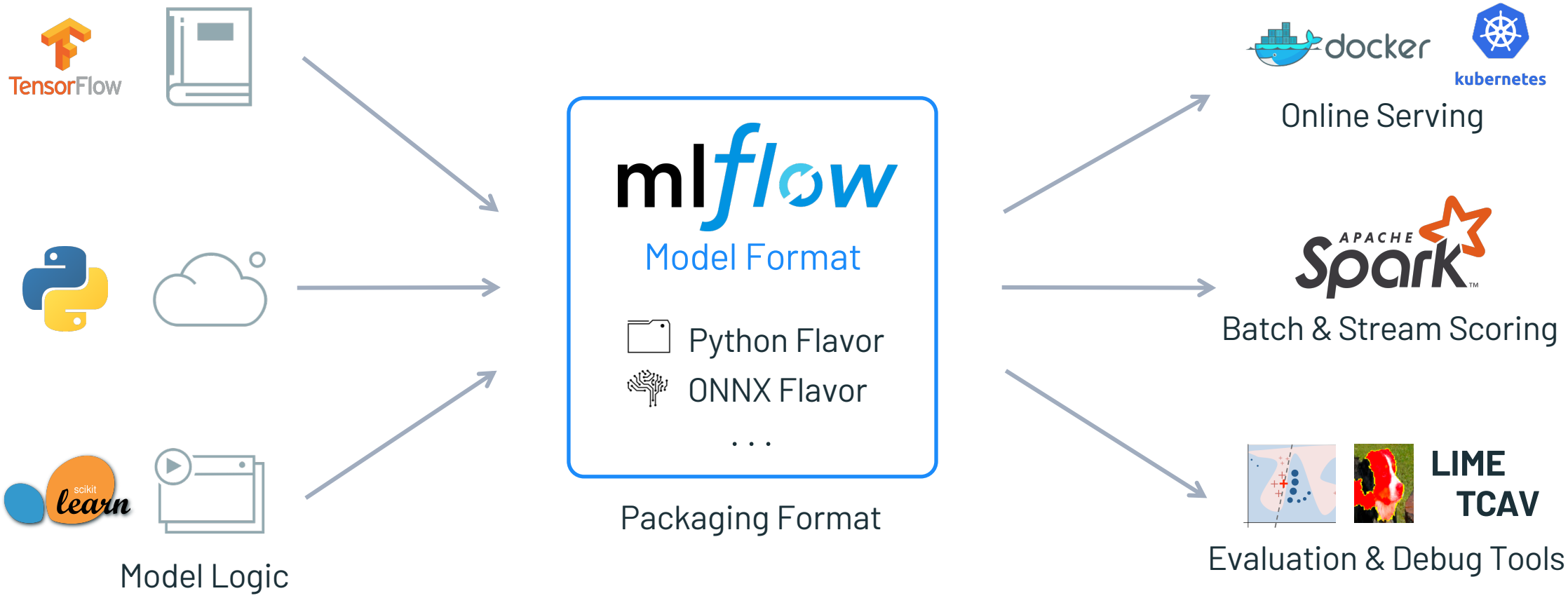
Composing Projects

```
r1 = mlflow.run("ProjectA", params)

if r1 > 0:
    r2 = mlflow.run("ProjectB", …)
else:
    r2 = mlflow.run("ProjectC", …)

r3 = mlflow.run("ProjectD", r2)
```

	Date	User	Parameters		
			batch_size	epochs	lr
⊞	2018-12-07 12:14:08	matei	512		0.05
⊟	2018-12-07 12:14:05	matei	512		0.05
	2018-12-07 12:14:05	matei		20	
	2018-12-07 12:14:05	matei		40	
	2018-12-07 12:14:05	matei		40	
⊞	2018-12-07 12:12:23	matei	512		0.05

MLflow Models

Generic format to package & deploy models from any library



databricks

Example MLflow Model

```
my_model/
├── MLmodel
└── estimator/
    ├── saved_model.pb
    └── variables/
        ...
```

```
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
```

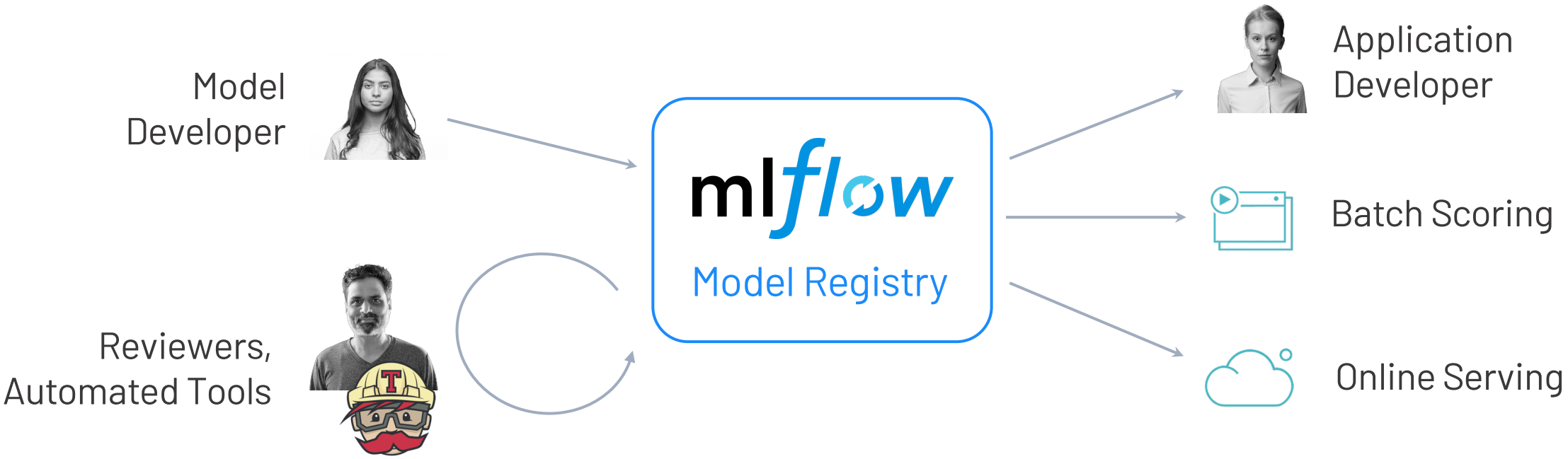
Usable by tools that understand TensorFlow model format

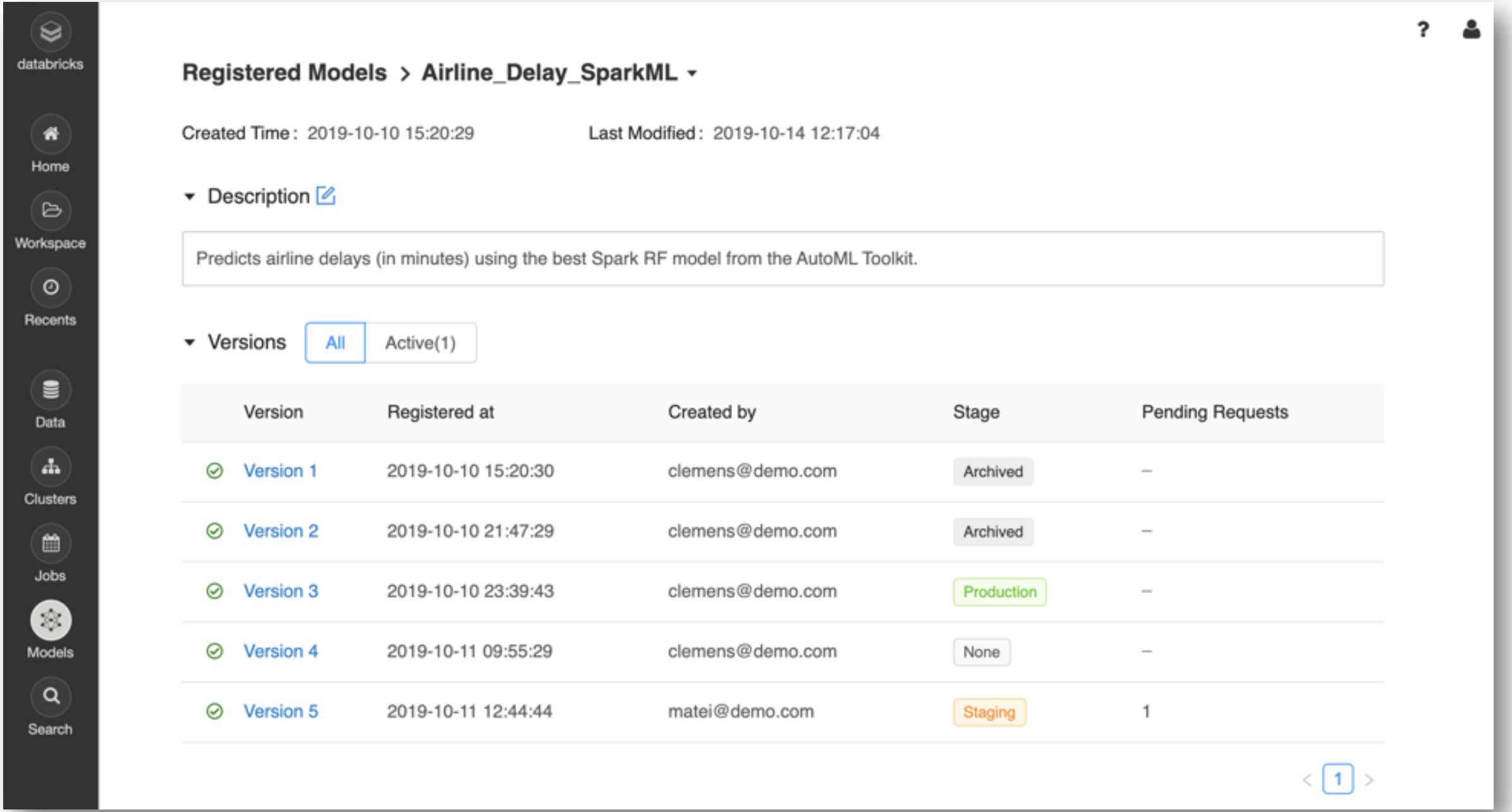
Usable by any tool that can run Python (Docker, Spark, etc)

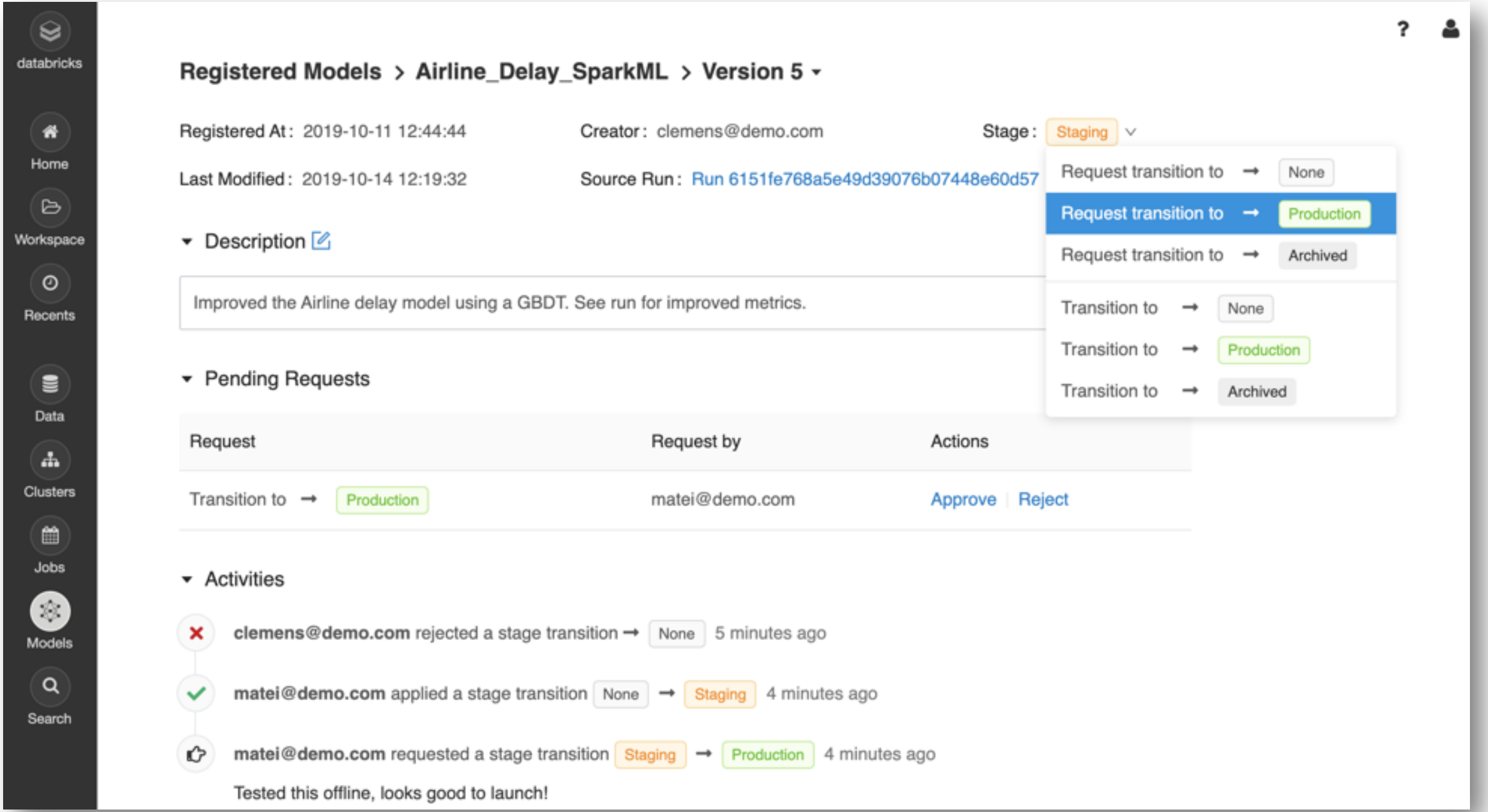
```
$ mlflow pyfunc serve -r <run_id>

spark_udf = pyfunc.spark_udf(<run_id>)
```

MLflow Model Registry

GitHub-like environment for managing and reviewing models



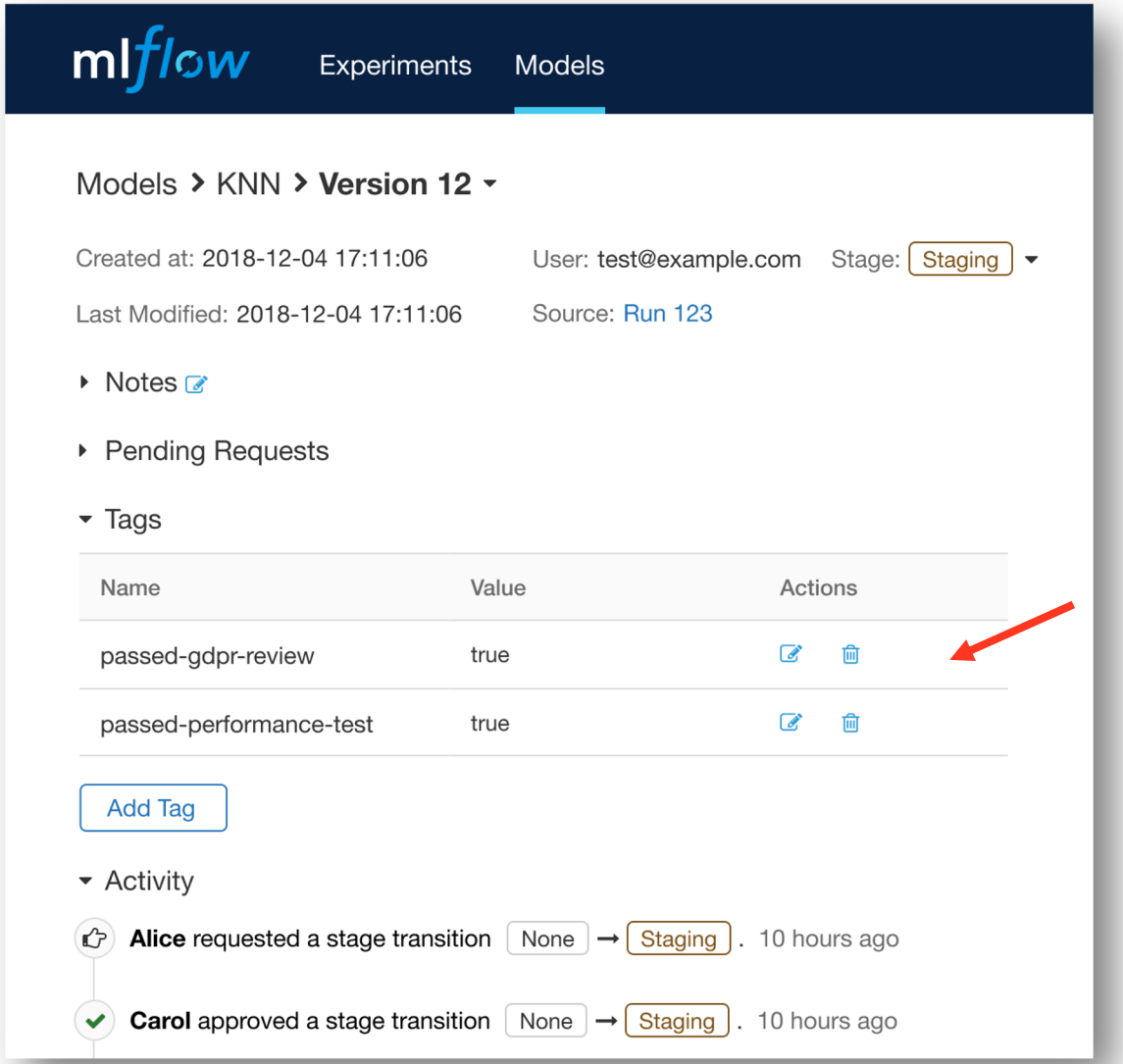

databricks
Home
Workspace
Recents
Data
Clusters
Jobs
Models
Search
Registered Models > Airline_Delay_SparkML
Created Time: 2019-10-10 15:20:29
Last Modified: 2019-10-14 12:17:04
Description
Predicts airline delays (in minutes) using the best Spark RF model from the AutoML Toolkit.
Versions
All
Active(1)
Version
Registered at
Created by
Stage
Pending Requests
Version 1
2019-10-10 15:20:30
clemens@demo.com
Archived
–
Version 2
2019-10-10 21:47:29
clemens@demo.com
Archived
–
Version 3
2019-10-10 23:39:43
clemens@demo.com
Production
–
Version 4
2019-10-11 09:55:29
clemens@demo.com
None
–
Version 5
2019-10-11 12:44:44
matei@demo.com
Staging
1
1


databricks
Home
Workspace
Recents
Data
Clusters
Jobs
Models
Search
Registered Models > Airline_Delay_SparkML > Version 5
Registered At: 2019-10-11 12:44:44
Creator: clemens@demo.com
Stage: Staging
Last Modified: 2019-10-14 12:19:32
Source Run: Run 6151fe768a5e49d39076b07448e60d57
Request transition to None
Request transition to Production
Request transition to Archived
Transition to None
Transition to Production
Transition to Archived
Description
Improved the Airline delay model using a GBDT. See run for improved metrics.
Pending Requests
Request
Request by
Actions
Transition to Production
matei@demo.com
Approve | Reject
Activities
clemens@demo.com rejected a stage transition None 5 minutes ago
matei@demo.com applied a stage transition None Staging 4 minutes ago
matei@demo.com requested a stage transition Staging Production 4 minutes ago
Tested this offline, looks good to launch!

Tag and Search APIs for Automated CI/CD

Tags to track custom metadata for a model version, e.g. test results

Search API to automate model management and MLOps actions



databricks

How are Companies Using MLflow?

“Expected” Use Cases

- Track experiments during model design
- Track performance of continuous training & deployment pipelines
- Deploy the same model for batch and real-time scoring
- Run pipelines deterministically in different environments
- CI/CD using Model Registry stages and APIs

Example Use Cases

Manage the models for ad fraud detection, including monitoring for drift in over 200 metrics.

Let data scientists spend 70%-90% of their time on model development instead of tuning and monitoring.

Automated and consistent deployment of 100+ models, from fraud detection, to marketing, to logistics

Interesting Use Case: Massive # of Models

Company wants to train a separate model for each {facility, customer, chemical processing machine, ...}

- Avoid interference across these entities
- Preserve privacy & compliance

Examples:

ExxonMobil

Predictive maintenance

Quby

Energy grid

Enterprise Software Company

Per-customer models

Interesting Use Case: Massive # of Models

Company wants to train a separate model for each {facility, customer, chemical processing machine, ...}

- Avoid interference across these entities
- Preserve privacy & compliance

Solution: "hands-free" ML with large-scale analytics

- Train millions of models in parallel using an AutoML library on each entity
- Query experiment metrics using analytics tools (e.g. MLflow -> Pandas API)
- Run online or batch inference with the models

Interesting Use Case: Experiments Beyond ML

Systems like MLflow can add structure to other experimentation tasks

Develop, review & publish visualizations for COVID-19 data using the MLflow Model Registry

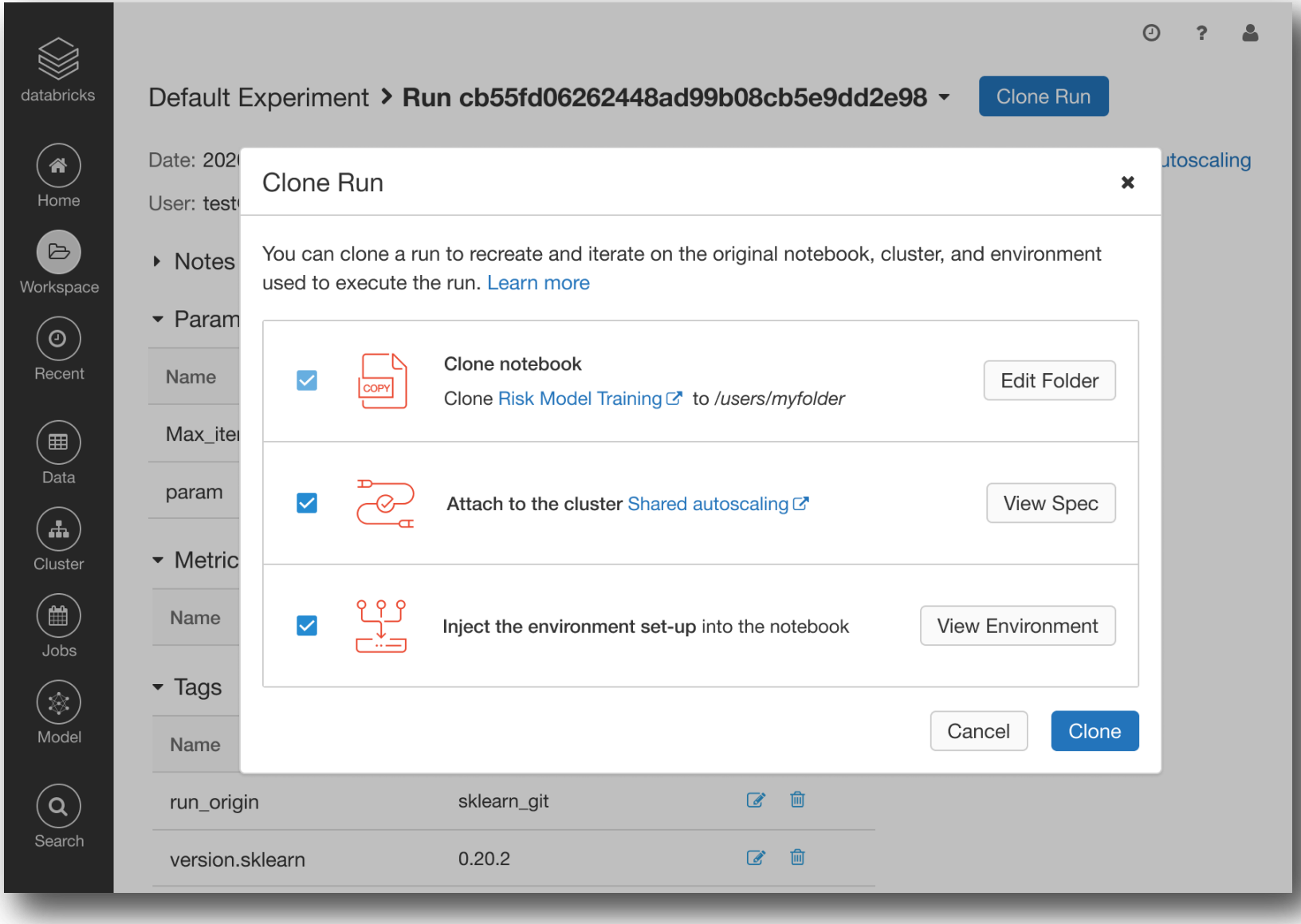
Tune hundreds of Hyperloop engineering design parameters in simulation to optimize efficiency

Interesting Use Case: Reproducibility & Explainability

Government regulators and highly-regulated companies want:

- Documentation of every piece of data and code that went into a result (many have built their own lineage systems for this)
- Explanation of models
- AutoML to demonstrate they employed best practices

Environment Reproducibility on Databricks



Recreate exact configuration of an experiment run

Recently Added Features

mlflow Tracking for Data Versioning

Spark and Delta Lake auto-logging: track data sources read and data versions in Delta Lake

```
with mlflow.start_run(run_name='keras'):
  # log model and datasource
  mlflow.keras.autolog()
  mlflow.spark.autolog()
```

▼ Tags

Name	Value	Actions
sparkDatasourceInfo	path=dbfs:/delta/clemens_windfarm,version=2,format=delta	

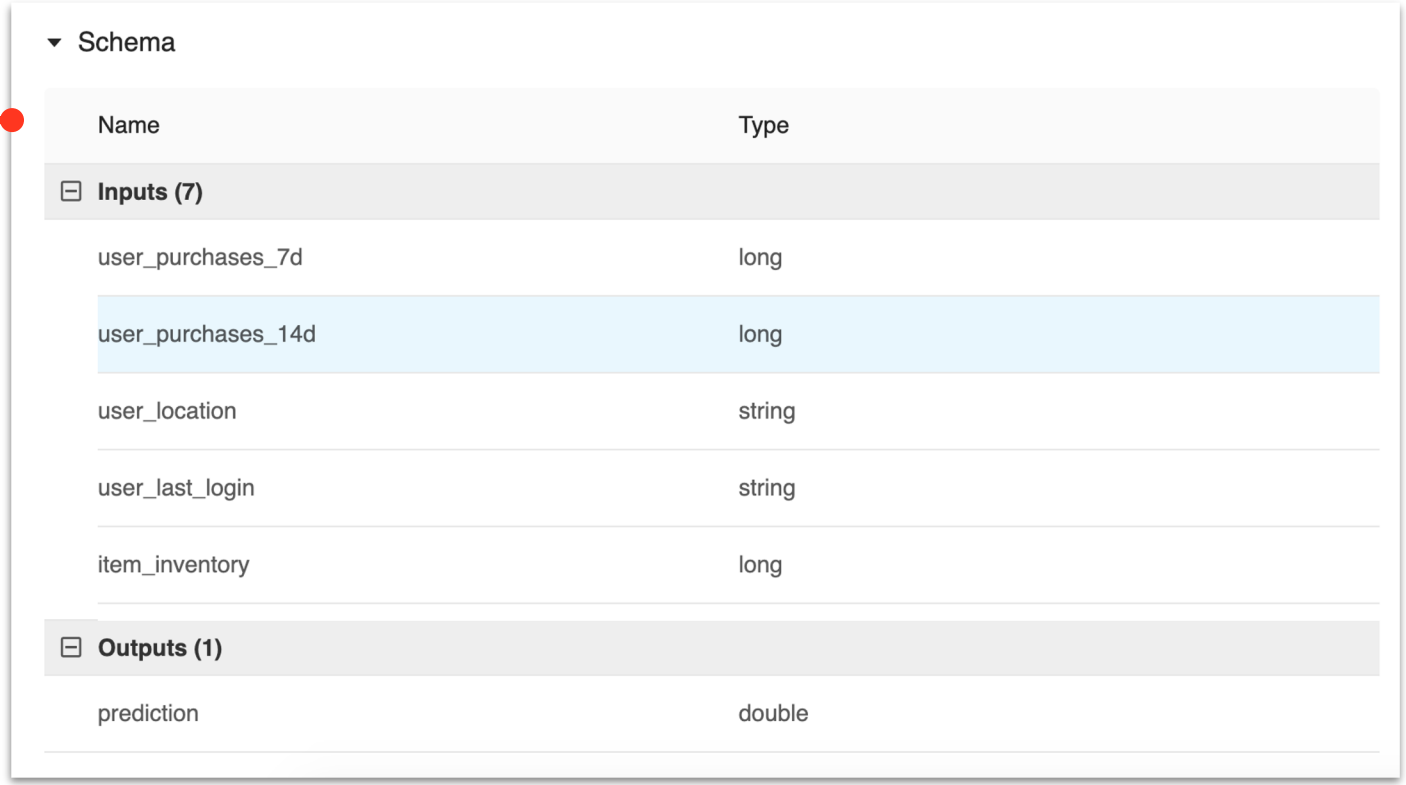
To reload data version:

```
df = spark.read.format("delta") \
                .option("versionAsOf", 2) \
                .load("/delta/clemens_windfarm")
```

mlflow Tracking for Model Schemas

Record what fields are consumed & produced by the model to prevent data mismatches

```
with mlflow.start_run(run_name='keras'):
  # log model and datasource
  mlflow.keras.autolog()
  mlflow.spark.autolog()

  sig = infer_signature(X_train, y_train)
```

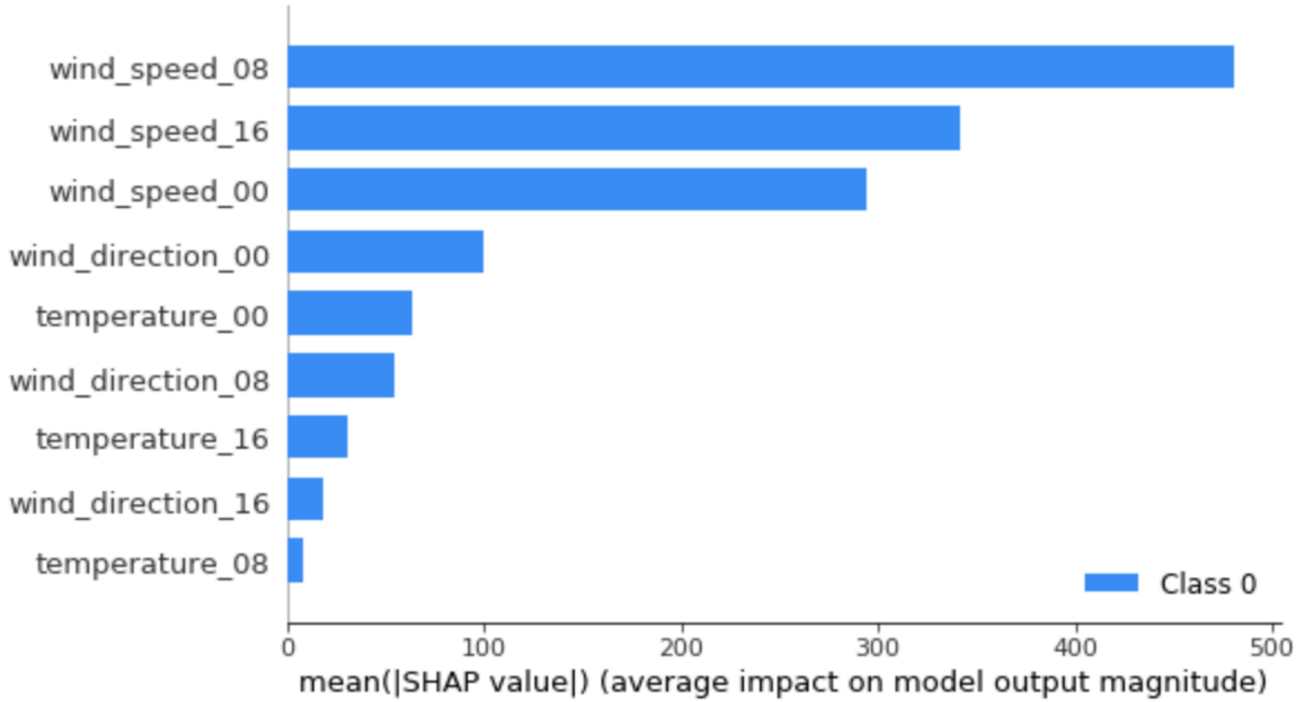


databricks

mlflow Tracking for Interpretability

SHAP library feature importances and visualizations

```
with mlflow.start_run(run_name='keras'):
  # log model and datasource
  mlflow.keras.autolog()
  mlflow.spark.autolog()

  sig = infer_signature(X_train, y_train)

  mlflow.shap.log_explanation(model, X_train[:100])
```



SHAP

databricks

Collaboration between Facebook and Databricks to bring ML platform features to PyTorch

- MLflow autologging for PyTorch Lightning
- TorchScript support for faster packaged models
- Model deployment to TorchServe

Many Open Questions in ML Platforms!

- How to design “feature stores” that can hold fast-changing data about entities to be used in training and inference?
- How to automatically detect performance degradation in models?
- What kind of information should be tracked at inference time?

Conclusion

Machine learning is being applied to critical problems in industry, but requires careful management when the stakes are high

ML Platforms are an emerging abstraction to help with this

- And we believe an “open-interface” design is very important for usability

Many open problems, especially as needs evolve