

mlflow

Platform for Machine Learning Lifecycle

Jules S. Damji
@2twitme



Outline – Introduction to MLflow: Model Registry Workflows Explained – Module 4

- Model Registry
- Concepts and Motivations
 - MLflow Model Registry
 - Model Registry UI & API Workflow
 - Tutorials on local host
 - Jupyter Lab
 - Google Colab
- Q & A

<https://github.com/dmatrix/tmhs-workshop>

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



mlflow.org



github.com/mlflow



twitter.com/MLflow



The Model Management Problem

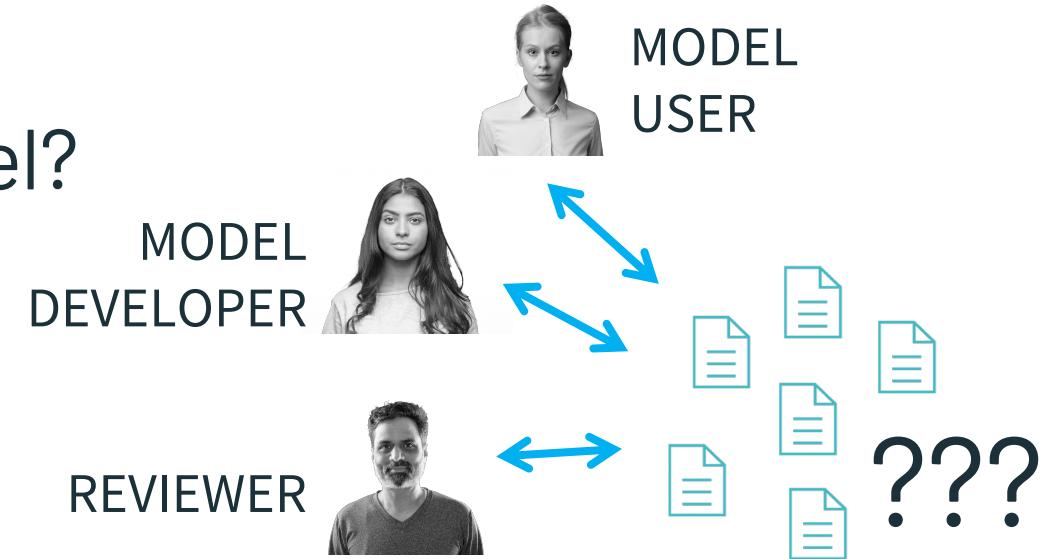
When you're working on one ML app alone, storing your models in files is manageable



The Model Management Problem

When you work in a large organization with many models, many data teams, management becomes a major challenge:

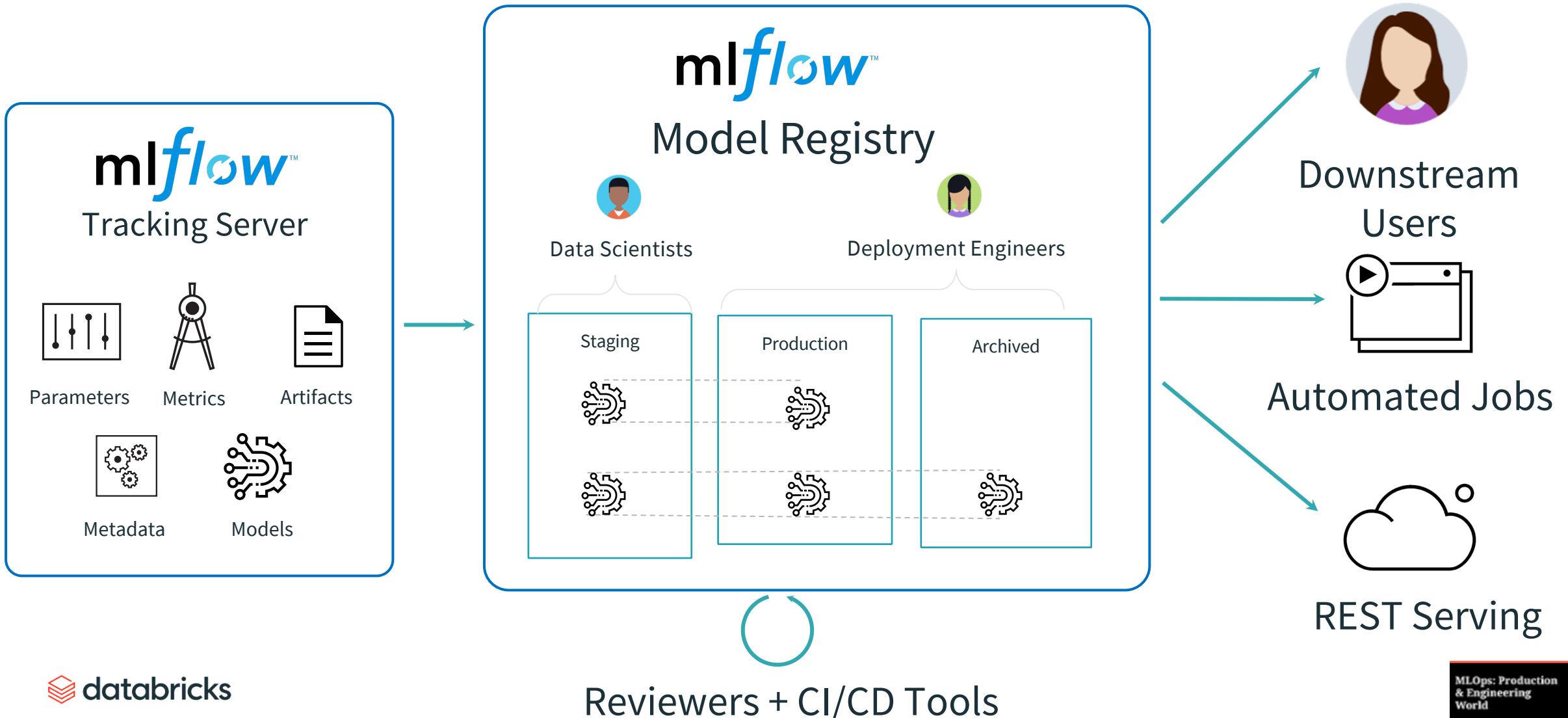
- Where can I find the best version of this model?
- How was this model trained?
- How can I track docs for each model?
- How can I review models?
- How can I integrate with CI/CD?





Model Registry

VISION: Centralized and collaborative model lifecycle management



MLflow Model Registry

- Repository of named, versioned models with controlled Access to Models
- Track each model's stage: none, staging, production, or archived
- Easily inspect a specific version and its run info
- Easily load a specific version
- Provides model description, lineage and activities

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

Version	Registered at	Created by	Stage
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

MLflow Model Registry

The MLflow Model Registry component is a centralized model store, set of APIs, and UI, to collaboratively manage the full lifecycle of an MLflow Model. It provides model lineage (which MLflow experiment and run produced the model), model versioning, stage transitions (for example from staging to production), and annotations.

Table of Contents

- [Concepts](#)
- [Model Registry Workflows](#)

- [UI Workflow](#)

- [Registering a Model](#)
- [Using the Model Registry](#)

- [API Workflow](#)

- [Adding an MLflow Model to the Model Registry](#)
- [Fetching an MLflow Model from the Model Registry](#)
- [Serving an MLflow Model from Model Registry](#)
- [Adding or Updating an MLflow Model Descriptions](#)
- [Renaming an MLflow Model](#)
- [Transitioning an MLflow Model's Stage](#)
- [Listing and Searching MLflow Models](#)
- [Archiving an MLflow Model](#)
- [Deleting MLflow Models](#)

Model Registry CRUD Operations MLflowClient()

```
create_model_version(name, source, run_id, tags=None, run_link=None, description=None) [source]
```

```
create_registered_model(name, tags=None, description=None) [source]
```

```
delete_model_version(name, version) [source]
```

```
get_latest_versions(name, stages=None) [source]
```

```
transition_model_version_stage(name, version, stage, archive_existing_versions=False) [source]
```



Model Registry Workflow UI

This screenshot shows the Model Registry interface. On the left, there's a tree view of artifacts under 'Artifacts'. A folder named 'sklearn-model' is expanded, showing files: 'MLmodel', 'conda.yaml', and 'model.pkl'. The 'Full Path' is listed as '/mirluns/0/55eb2ad528114c68bd354a0568eca327/artifacts/sklearn-model' and the 'Size' is '0B'. Below this, there's a section to 'Select a file to preview' with supported formats: image, text, html, geojson files. A large green arrow points from this screen to the 'MODEL DEVELOPER' section.



MODEL
DEVELOPER

This screenshot shows the Model Registry interface. On the left, there's a 'Tags' section with a table for adding tags. In the center, there's a 'Register Model' dialog box. It has fields for 'Model' (dropdown menu) and 'Model Name' (text input field containing 'SKLearnPowerForecast'). At the bottom are 'Cancel' and 'Register' buttons. A large green arrow points from this screen to the 'OPERATOR' section.

This screenshot shows the Model Registry interface. On the left, there's a 'Tags' section with a table for adding tags. In the center, there's a 'Register Model' dialog box. It has fields for 'Model' (dropdown menu) and 'Model Name' (text input field containing 'SKLearnPowerForecast'). At the bottom are 'Cancel' and 'Register' buttons. A large green arrow points from this screen to the 'OPERATOR' section.



This screenshot shows the Model Registry interface. On the left, there's a 'Artifacts' section with a tree view of artifacts under 'sklearn-model'. A folder named 'sklearn-model' is expanded, showing files: 'MLmodel', 'conda.yaml', and 'model.pkl'. The 'Full Path' is listed as '/mirluns/0/55eb2ad528114c68bd354a0568eca327/artifacts/sklearn-model' and the 'Size' is '0B'. To the right, there's a summary card for a registered model: 'SKLearnPowerForecast', version 'v1', registered on '2020/05/04'. Below this, there's a section to 'Select a file to preview' with supported formats: image, text, html, geojson files. A large green arrow points from this screen to the 'OPERATOR' section.

Model Registry Workflow UI

This screenshot shows the mlflow Model Registry UI. It displays a model version detail page for 'SKLearnPowerForecast' Version 1. The page includes fields for 'Registered At' (2020-05-04 11:38:47), 'Creator' (None), 'Stage' (None), and 'Source Run' (Random Forest Regressor: Power Forecasting Model). A large green arrow points from this screen to the 'MODEL REVIEWER' section.

Registered Models > SKLearnPowerForecast > Version 1

Registered At: 2020-05-04 11:38:47 Creator: Stage: None

Last Modified: 2020-05-04 11:38:47 Source Run: Random Forest Regressor: Power Forecasting Model

Description

This version forecasts the power

Save Cancel



MODEL
REVIEWER

This screenshot shows the mlflow Model Registry UI. It displays a model version detail page for 'SKLearnPowerForecast' Version 1. The 'Stage' dropdown is set to 'None'. Below it, there are three buttons: 'Transition to → Staging' (orange), 'Transition to → Production' (green), and 'Transition to → Archived' (grey). A large green arrow points from the 'MODEL REVIEWER' section to this screen.

Registered Models > SKLearnPowerForecast > Version 1

Registered At: 2020-05-04 11:38:47 Creator: Stage: None

Last Modified: 2020-05-04 11:38:47 Source Run: Random Forest Regressor: Power Forecasting Model

Description

This version forecasts the power

Transition to → Staging

Transition to → Production

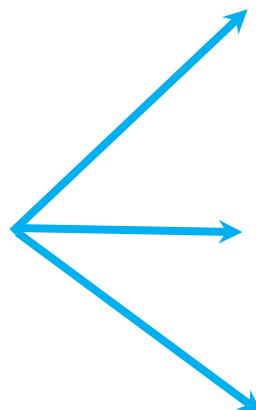
Transition to → Archived



DOWNSTREAM
USERS

This screenshot shows the mlflow Model Registry UI. It displays a list of model versions for 'SKLearnPowerForecast'. The table includes columns for 'Version', 'Registered at', 'Created by', and 'Stage'. Three versions are listed: Version 1 (Staging), Version 2 (Staging), and Version 3 (Production). A large green arrow points from the 'DOWNSTREAM USERS' section to this screen.

Version	Registered at	Created by	Stage
Version 1	2020-05-04 11:38:47		Staging
Version 2	2020-05-04 11:41:37		Staging
Version 3	2020-05-04 11:42:07		Production



AUTOMATED JOBS



REST SERVING



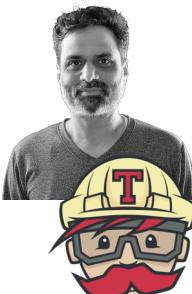
Model Registry Workflow API

```
mlflow.register_model(model_uri, "WeatherForecastModel")  
  
mlflow.sklearn.log_model(model,  
    artifact_path="sklearn_model",  
    registered_model_name= "WeatherForecastModel")
```

MODEL
DEVELOPER



REVIEWERS,
CI/CD TOOLS



```
client = mlflow.tracking.Mlflowclient()  
client.transition_model_version_stage(name="WeatherForecastModel",  
    version=5,  
    stage="Production")
```



```
model_uri= "models:/{{model_name}}/production".format(  
    model_name="WeatherForecastModel")  
model_prod = mlflow.sklearn.load_model(model_uri)  
model_prod.predict(data)
```

DOWNSTREAM
USERS



AUTOMATED JOBS



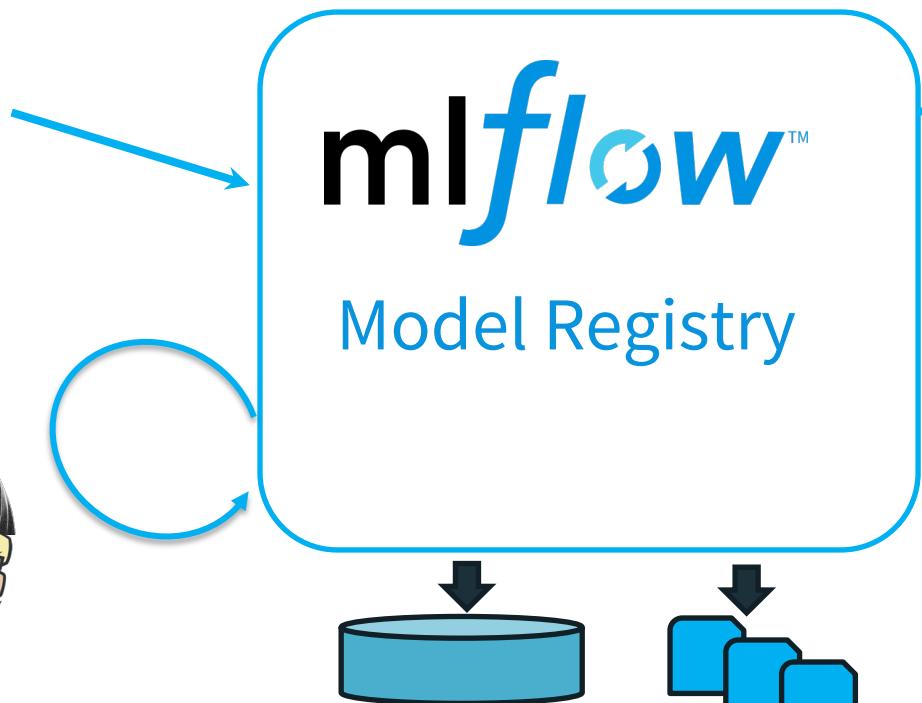
REST SERVING



Model Registry Workflow API

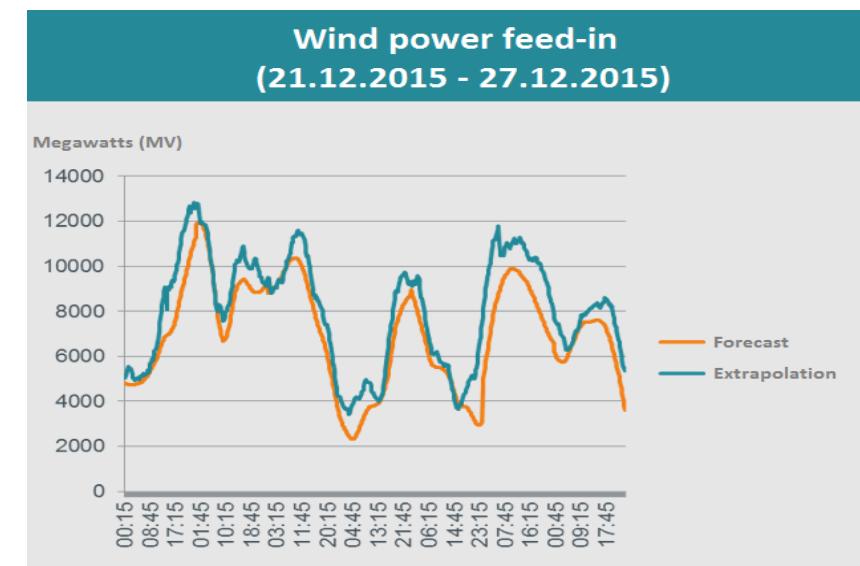
```
mlflow.register_model(model_uri, "WeatherForecastModel")  
  
mlflow.sklearn.log_model(model,  
    artifact_path="sklearn_model",  
    registered_model_name= "WeatherForecastModel")
```

MODEL
DEVELOPER



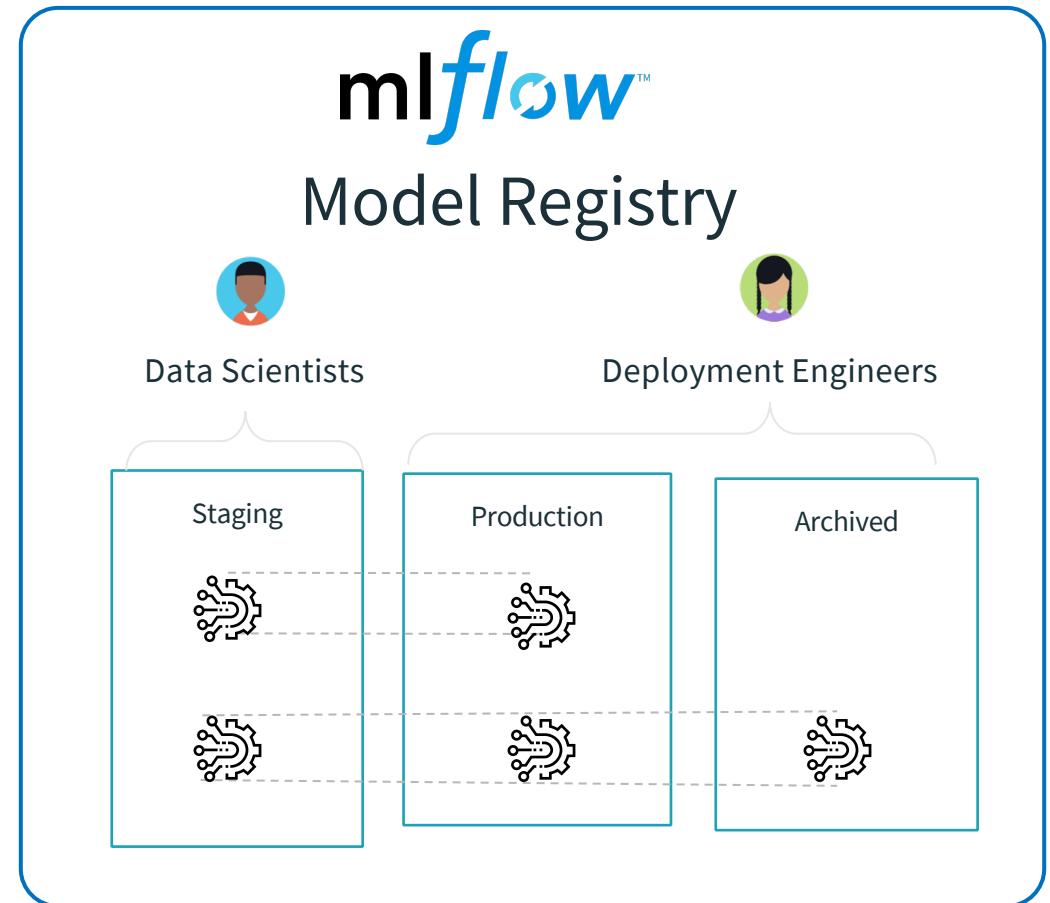
```
client = mlflow.tracking.MlflowClient()  
client.transition_model_version_stage(name="WeatherForecastModel",  
    version=5,  
    stage="Production")
```

```
model_uri = "models:/{}{}/production".format(  
    model_name="WeatherForecastModel")  
model_prod = mlflow.pyfunc.load_model(model_uri)  
model_prod.predict(data)
```



MLflow Model Registry Recap

- **Central Repository:** Unique named registered models for discovery across data teams
- **Model Registry Workflow:** Provides UI and API for registry operations
- **Model Versioning:** Allow multiple versions of model in different stages
- **Model Stages:** Allow stage transition: none, staging, production, or archived
- **CI/CD Integration:** Easily load a specific version for testing and inspection
- **Model Lineage:** Provides model description, lineage and activities



mlflow Model Registry: Tag and Search APIs

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

Search API to automate model management and MLOps actions

The screenshot shows the MLflow Model Registry interface. At the top, there are navigation links for 'Experiments' and 'Models'. Below that, the path 'Models > KNN > Version 12' is displayed. On the right, there are details: 'Created at: 2018-12-04 17:11:06', 'User: test@example.com', 'Stage: Staging', 'Last Modified: 2018-12-04 17:11:06', and 'Source: Run 123'. A red arrow points to the delete icon in the 'Actions' column for the tag 'passed-gdpr-review'. Below the tags table, there is an 'Add Tag' button and a section for 'Activity' showing two recent stage transition requests.

Name	Value	Actions
passed-gdpr-review	true	
passed-performance-test	true	

Add Tag

Activity

Alice requested a stage transition None → Staging . 10 hours ago

Carol approved a stage transition None → Staging . 10 hours ago

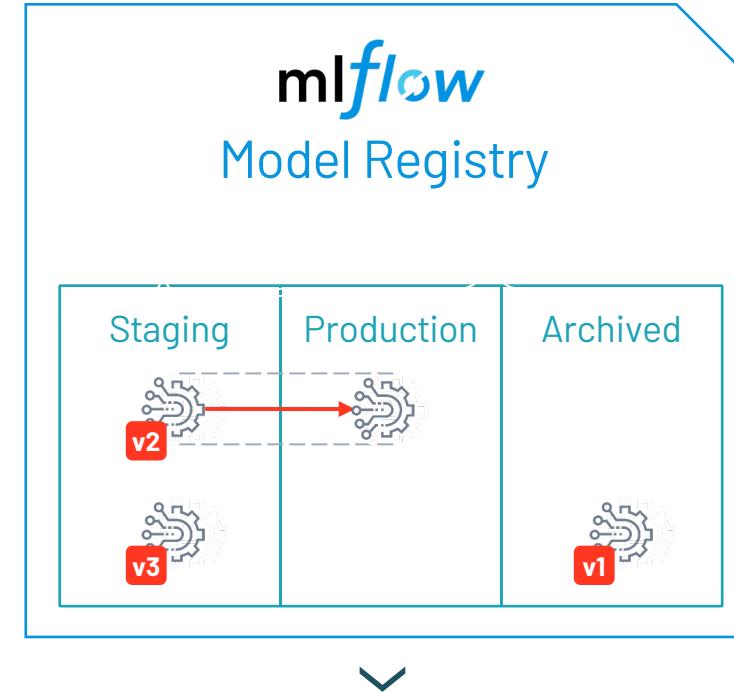
mlflow Model Registry: Webhooks

Databricks Webhooks allow setting callbacks on registry events like stage transitions to run CI/CD tools

MLflow Model Registry on Databricks
Simplifies MLOps With CI/CD Features



by Sue Ann Hong, Ankit Mathur, Jules Damji and Mani Parkhe
Posted in ENGINEERING BLOG | November 19, 2020



just launched

Human
Reviewers

CI/CD Tools

Batch
Scoring

Real-time
Serving

`VERSION_REGISTERED: MyModel, v2`

`TRANSITION_REQUEST: MyModel, v2, Staging→Production`

`TAG_ADDED: MyModel, v2, BacktestPassed`



mlflow Model Registry: Comments

just launched

Comments in the Databricks workspace can now be used to discuss changes on models

The screenshot shows a user interface for the MLflow Model Registry. At the top, there's a section titled "Activities". Below it, a comment is displayed from an account named "ci-pipeline@databricks.com" posted 2 hours ago (edited). The comment text is "Tests failed - see output [here](#)". At the bottom of the comment area, there is a button labeled "Add a comment".

MLflow Model Registry on Databricks Simplifies MLOps With CI/CD Features



by Sue Ann Hong, Ankit Mathur, Jules Damji and Mani Parkhe
Posted in ENGINEERING BLOG | November 19, 2020



What Did We Talk About?



- Modular Components greatly simplify the ML lifecycle
- Easy to install & Great Developer experience
- Develop & Deploy locally; track locally or remotely
- Available APIs: Python, Java & R (Soon Scala)
- REST APIs and CLI tools
- Visualize experiments and compare runs
- Centrally register and manage model lifecycle

Thank you! 😊

Q & A

jules@databricks.com
@2twitme

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