edquest

Day 6 Agenda

Experiment Tracking & Model Registry

- Recap
- MLFlow/Weights & Biases for tracking experiments
- Logging metrics, parameters
- Model registry and promotion
- Workshop
- Quiz

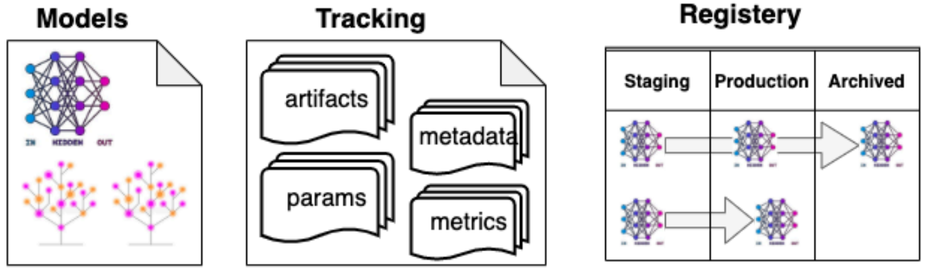
edquest

MLflow Introduction

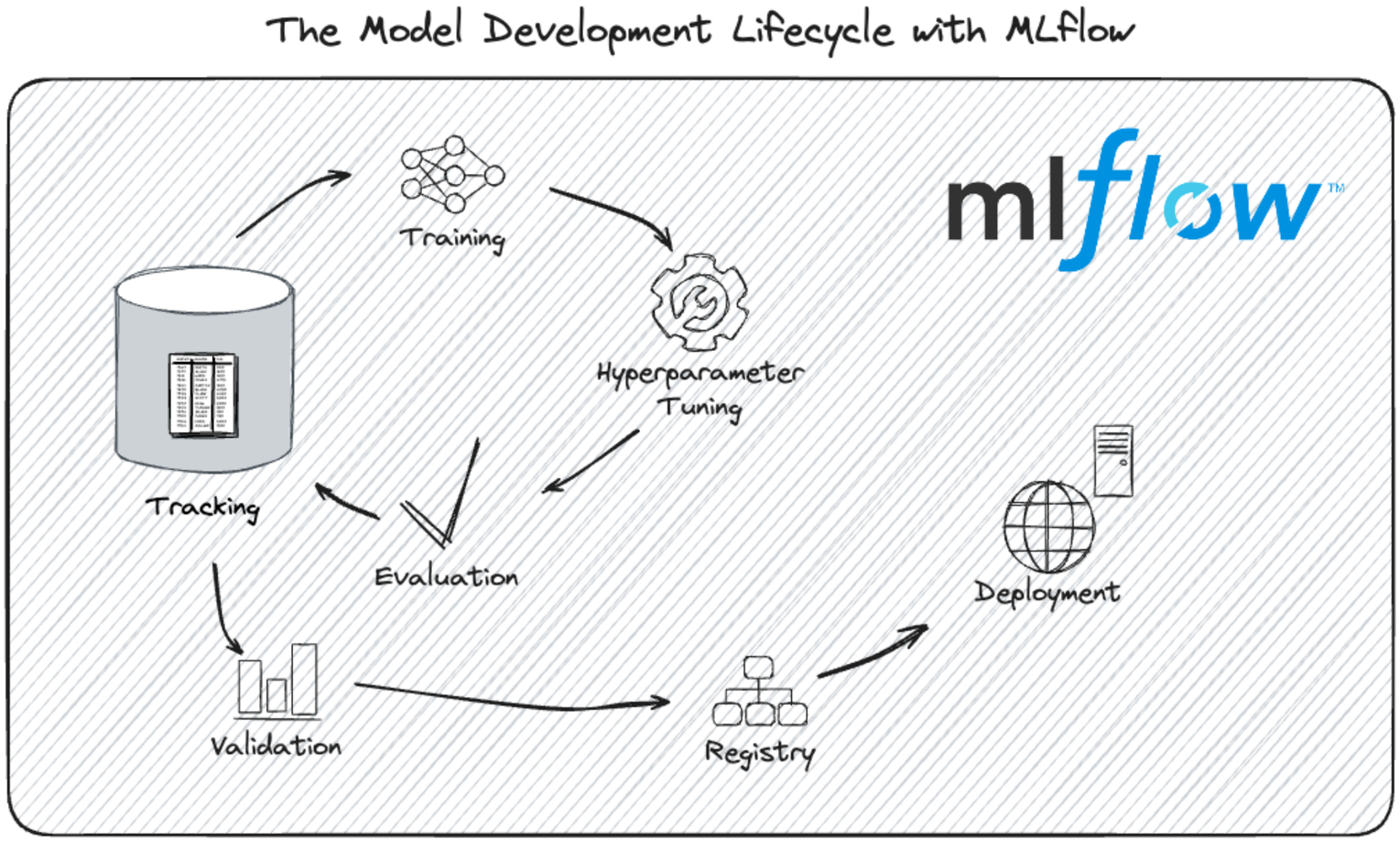
MLflow is an open-source platform for managing the end-to-end machine learning lifecycle.

It provides tools for:

- MLflow Tracking
- MLflow Models
- MLflow Model Registry



MLflow



src: https://mlflow.org/docs/latest/ml/

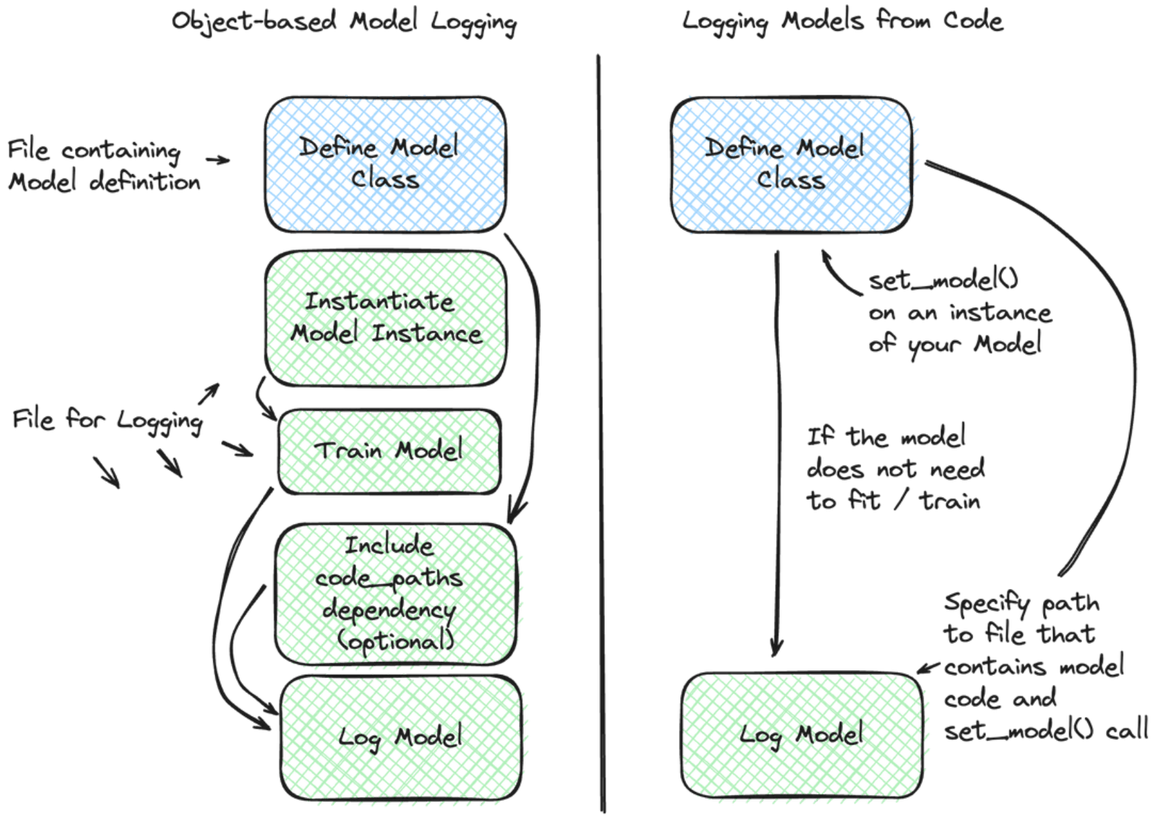
MLflow Tracking

MLflow Tracking is an API and UI for logging parameters, code versions, metrics, and output files when running your machine learning code and for later visualizing the results.
MLflow Tracking provides Python , REST , R, and Java APIs.

src: https://mlflow.org/docs/latest/ml/

MLflow Model

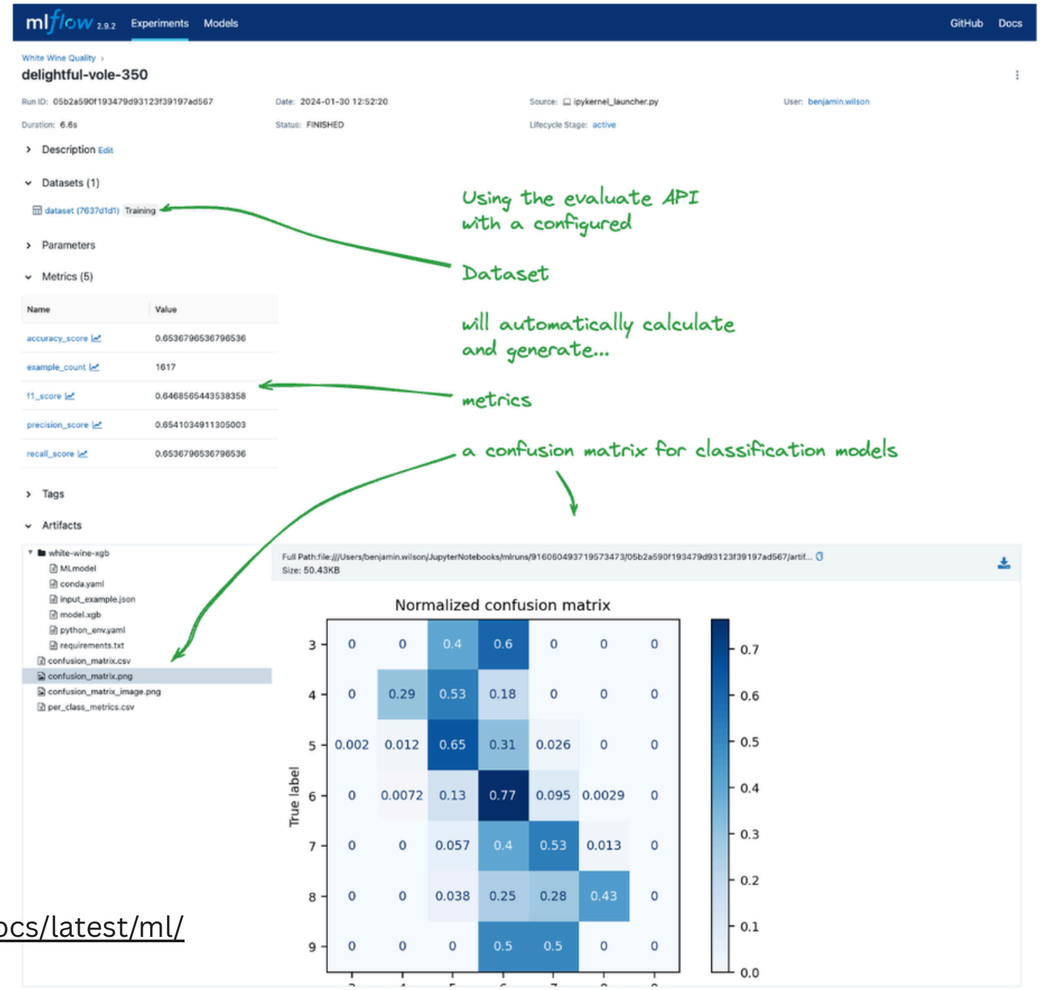
MLflow Model is a standard format for packaging machine learning models that can be used in a variety of downstream tools for example, real-time serving through a REST API or batch inference on Apache Spark.



src: https://mlflow.org/docs/latest/ml/

MLflow Dataset

The mlflow.data module is a comprehensive solution for dataset management throughout the machine learning lifecycle. It enables you to track, version, and manage datasets used in training, validation, and evaluation, providing complete lineage from raw data to model predictions.

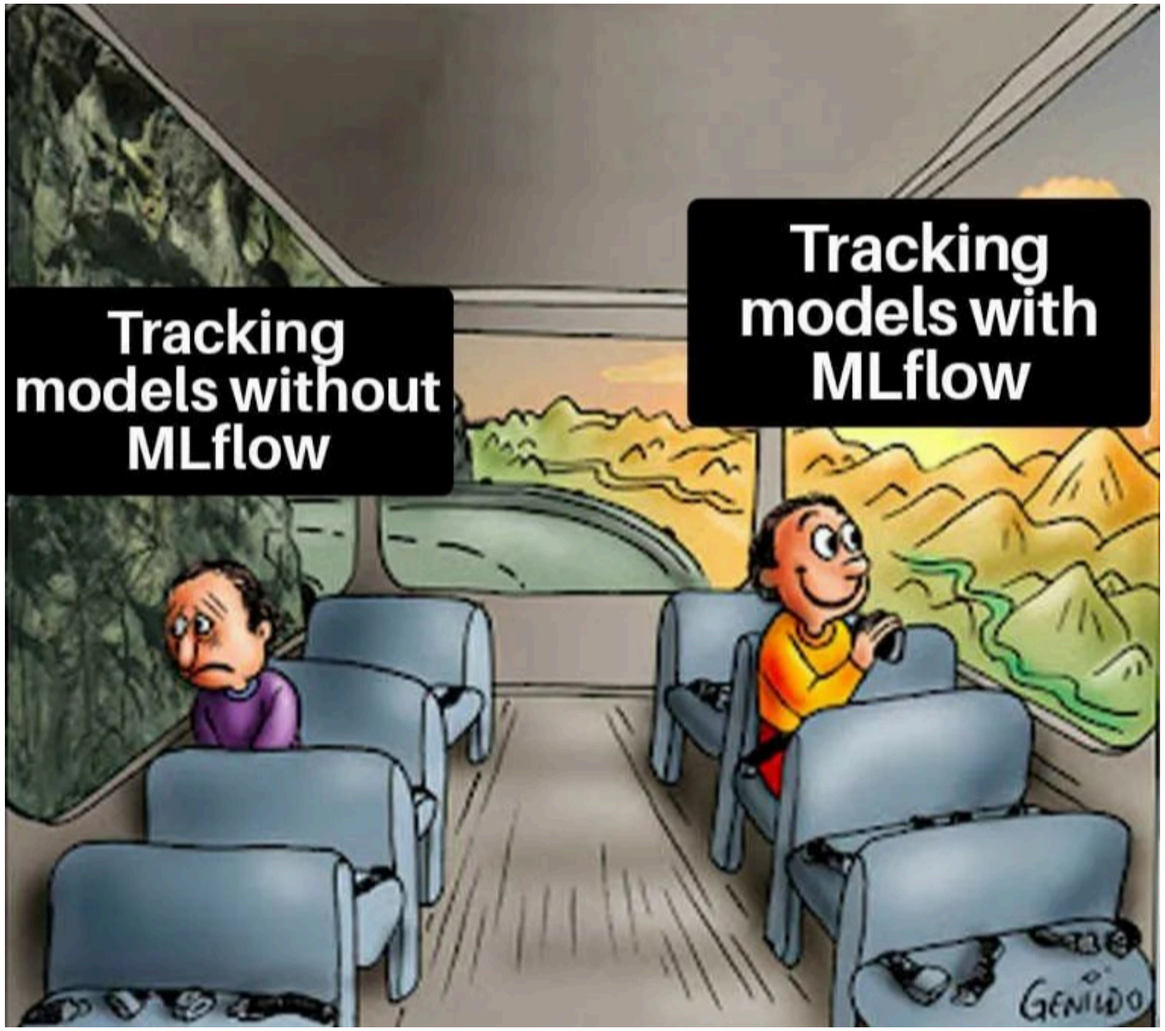


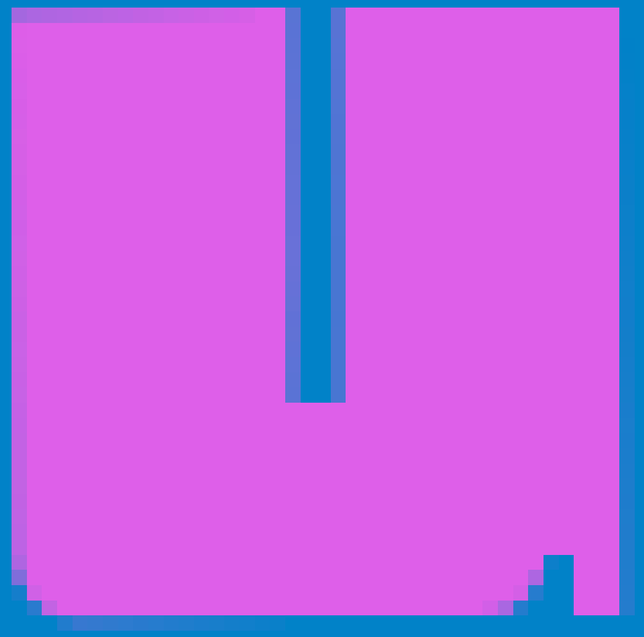
src: https://mlflow.org/docs/latest/ml/

What is MLflow?

- Taking notes on a sticky note or your phone: ‘Strawberry-Mango was sweet. Banana-Spinach? Hmm, not a hit.’

MLflow Tracking?

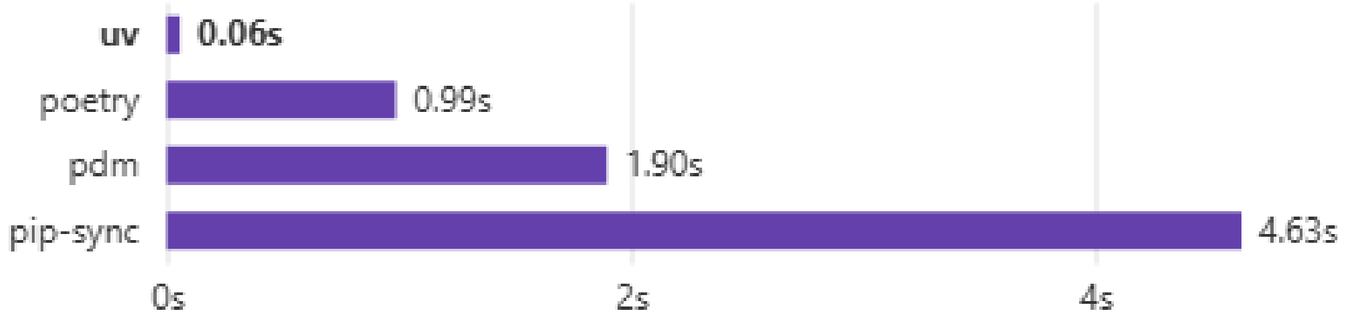




edquest

edquest

An extremely fast Python package and project manager, written in Rust.



edquest

Highlights

- A single tool to replace pip, pip-tools, pipx, poetry, pyenv, twine, virtualenv, and more.
- 10-100x faster than pip.
- Provides comprehensive project management, with a universal lockfile.
- Runs scripts, with support for inline dependency metadata.
- Installs and manages Python versions.
- Runs and installs tools published as Python packages.
- Includes a pip-compatible interface for a performance boost with a familiar CLI.
- Supports Cargo-style workspaces for scalable projects.
- Disk-space efficient, with a global cache for dependency deduplication.
- Installable without Rust or Python via curl or pip.
- Supports macOS, Linux, and Windows.

edquest

UV
&
MLflow

WORKSHOP

Installation

edquest

macOS and Linux

- **Use curl to download the script and execute it with sh:**
  - `curl -LsSf https://astral.sh/uv/install.sh | sh`
- **If your system doesn't have curl, you can use wget:**
  - `wget -qO- https://astral.sh/uv/install.sh | sh`
- **Request a specific version by including it in the URL:**
  - `curl -LsSf https://astral.sh/uv/0.7.19/install.sh | sh`

Installation

Windows

- **Use irm to download the script and execute it with iex:**
  - `powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/install.ps1 | iex"`
- Changing the execution policy allows running a script from the internet.
- **Request a specific version by including it in the URL:**
  - `powershell -ExecutionPolicy ByPass -c "irm https://astral.sh/uv/0.7.19/install.ps1 | iex"`

Creating a new project using uv

edquest

- You can create a new Python project using the uv init command:
  - `uv init hello-world`
  - `cd hello-world`
- Alternatively, you can initialize a project in the working directory:
  - `mkdir hello-world`
  - `cd hello-world`
  - `uv init`

UV will create following files in projects

edquest

uv will create the following files:

```
├── .gitignore
├── .python-version
├── README.md
├── main.py
└── pyproject.toml
```

The `main.py` file contains a simple "Hello world" program. Try it out with `uv run`:

```
$ uv run main.py
Hello from hello-world!
```

Setting Up MLflow

- Installation of Mlflow:
  - `uv add mlflow ipykernel`
- Launching Mlflow UI : $
  - `mlflow server --host 0.0.0.0 --port 5000`
- Access the tracking server at
  - `http://localhost:5000`
- Lightweight and easy to set up locally or on a remote server

edquest

Installing MLflow using docker

- Prerequisites
- Install Docker: https://docs.docker.com/get-docker/
- Pull MLflow Docker Image:

```
docker pull ghcr.io/mlflow/mlflow:v2.22.0
```

- Run MLflow Tracking Server:

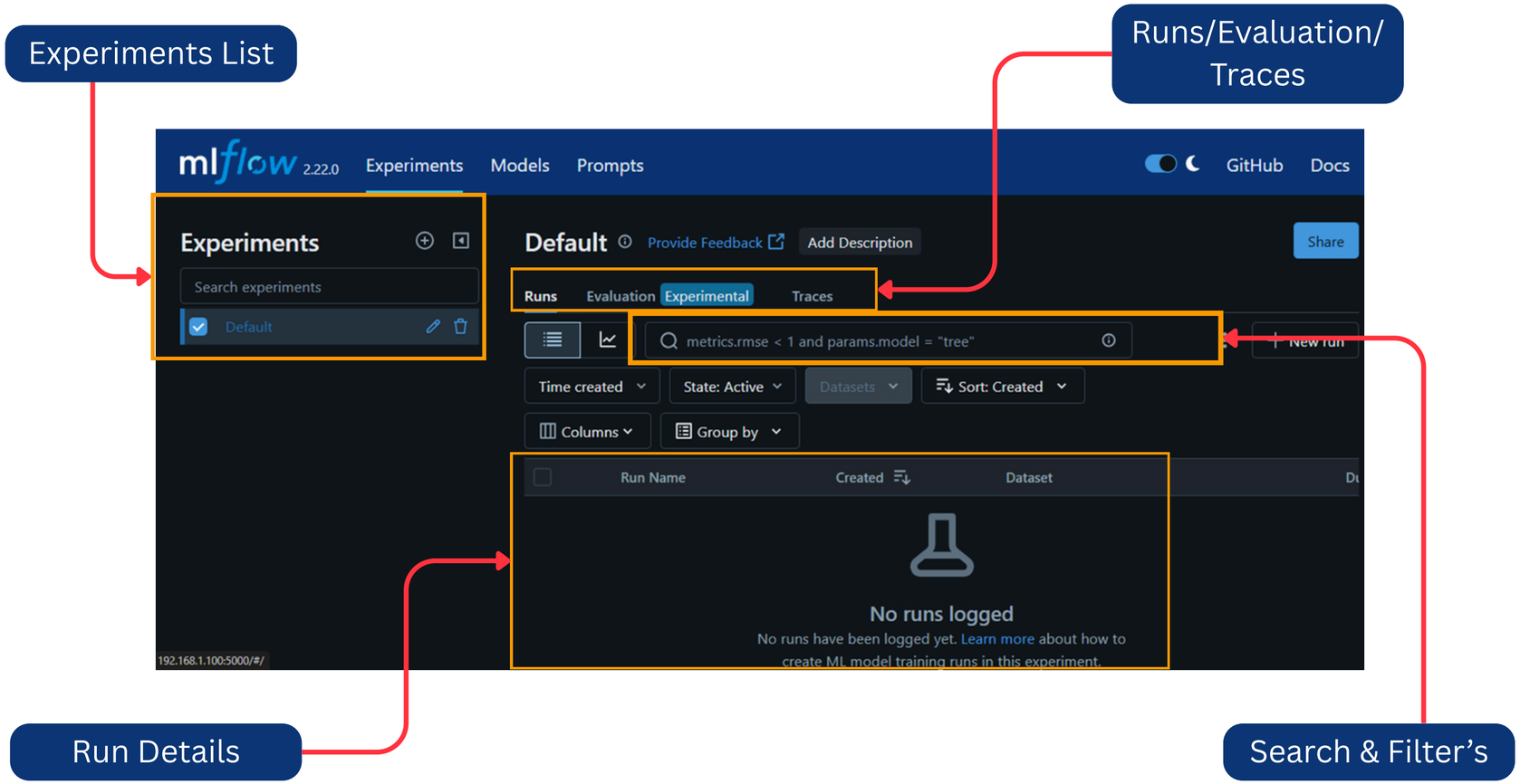
```
docker run -it -p 5001:5000 -v ./mlruns:/apps/mlruns

ghcr.io/mlflow/mlflow:v2.22.0 mlflowui --host 0.0.0.0

--port 5000 --backend-store-uri file:/app/mlruns
```

edquest

MLflow GUI Walkthrough



MLflow Model Registry

The MLflow Model Registry is a centralized model store, set of APIs and a UI designed to collaboratively manage the full lifecycle of a model.

It provides versioning, aliasing, metadata tagging and annotation support to ensure that you have the full spectrum of information at every stage from development to production deployment.

MLflow Model Registry

The MLflow Model Registry is a centralized model store, set of APIs and a UI designed to collaboratively manage the full lifecycle of a model.

It provides versioning, aliasing, metadata tagging and annotation support to ensure that you have the full spectrum of information at every stage from development to production deployment.

Why Model Registry?

As ML projects grow, managing models manually becomes hard and error-prone. The Model Registry offers a structured way to manage models across their lifecycle.

edquest

Why Model Registry?

edquest

Version Control

- Track all model versions; compare, roll back, or manage multiple versions (e.g., staging vs. production).

Traceability

- Link each model to the exact code, data, and parameters used—ensuring full reproducibility.

Deployment Ready

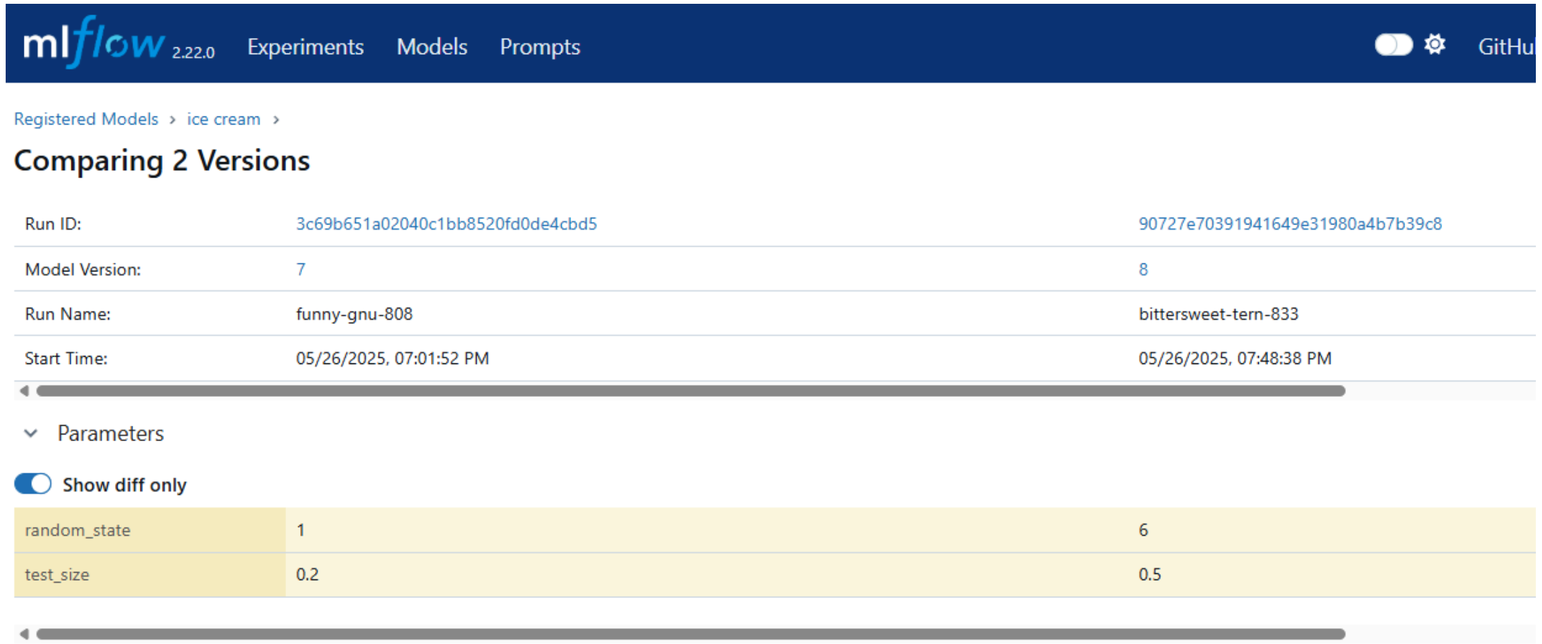
- Use tags and aliases (like @champion) to manage models across stages (experimental → production).

Governance

- Add metadata and access controls to meet security and compliance needs.

STORING A MODEL IN THE MLFLOW MODEL REGISTRY

edquest



edquest

Explain about ML parameters, Artifacts, Evaluation Matrics etc.

ML Parameters

- Parameters are inputs used to configure models
- Examples:
  - Learning rate
  - Number of estimators
  - Test/train split size
- Logged using: mlflow.log_param("param_name", value)

What Are Artifacts?

Artifects in MLflow

- Output files generated during training
- Examples:
  - Trained models (pkl, h5)
  - Plots (e.g., loss curves)
  - Logs, confusion matrices, feature importances
- Logged using: mlflow.log_artifact("file_path")

Evaluation Metrics

- Quantitative measures of model performance
- Examples:
  - MAE (Mean Absolute Error)
  - MSE (Mean Squared Error)
  - RMSE (Root Mean Squared Error)
  - R^2 Score (Coefficient of Determination)
- Logged using: mlflow.log_metric("metric_name", value)