MLOps Fundamentals

MLOps Tools and Technologies

Overview

MLOps involves multiple tools that help manage the **ML lifecycle, automation, and deployment**. This lesson provides a hands-on introduction to **MLflow**, a key tool for **experiment tracking, model management, and deployment**.

Learning Objectives

By the end of this microlesson, you will:

- **Understand** the role of different MLOps tools.
- **Use MLflow** to track and compare ML experiments.
- **Log metrics, parameters, and models** in MLflow.

1. Common MLOps Tools

MLOps solutions fall into different categories:

Category	Tools
Experiment Tracking	MLflow, Weights & Biases, Neptune.ai
Model Versioning	MLflow Model Registry, DVC, Pachyderm
CI/CD for ML	GitHub Actions, Jenkins, Kubeflow Pipelines
Model Deployment	MLflow, TensorFlow Serving, Seldon Core, Kubernetes
Monitoring & Drift Detection	EvidentlyAI, WhyLabs, Prometheus, Grafana

2. Hands-On: Tracking Experiments with MLflow

Step 1: Setup MLflow

1. Install MLflow if you haven't already:

Copy

```
pip install mlflow
```

2. Start the MLflow tracking server:

Copy

```
mlflow ui
```

Open `http://localhost:5000` to access the MLflow dashboard.

Step 2: Log an ML Experiment

Modify your training script to log parameters, metrics, and models.

Before (No Experiment Tracking):

Copy

```
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(f"Accuracy: {accuracy_score(y_test, y_pred)}")
```

After (Using MLflow):

```
import mlflow
import mlflow.sklearn

mlflow.start_run():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)

    mlflow.log_param("model_type", "RandomForest")
    mlflow.log_metric("accuracy", acc)
    mlflow.sklearn.log_model(model, "model")
```

Run the script and check **MLflow UI** to see logged parameters and metrics.

3. Key Takeaways