



# MLOps

## Von der Idee zum Deployment

KI Summer Summit Stuttgart

Hands-on Workshop • 24. Juli 2025



**MLflow**

Experiment Tracking



**Iris Dataset**

Praxisbeispiel



**Deployment**

REST-API & Docker

**Kaywan Barazani / Ahmad Salah**

DevOps/MLOps Engineer @ SprintEins Stuttgart

# Workshop-Überblick

## Ziele & Zielgruppe

### Was Sie lernen werden:

- ML-Lebenszyklus verstehen und strukturieren
- MLflow sicher in eigene Projekte integrieren
- Unterschiede zu klassischer Softwareentwicklung erkennen

### Für wen:

- DevOps- & Data-Teams
- Data Scientists
- ML-Neugierige & Einsteiger im MLOps-Umfeld

## Ablauf (2 Stunden)

15 min

### Einführung MLOps

Warum MLOps? Typische Stolpersteine

30 min

### MLflow Basics

Tracking, Model Registry, Hands-on

60 min

### Praxisbeispiel: Iris-Datensatz

Training, Logging, Auswertung

15 min

### Deployment

REST-API, Docker (optional)

## Was Sie brauchen



**Laptop**  
mit Internet



**Python-Umgebung**  
Jupyter Notebooks



**Paket-Installation**  
via pip



**Docker Desktop**  
optional für Deployment

# Was ist MLOps?

## MLOps ist die Brücke zwischen ML-Entwicklung und Produktion

Machine Learning Operations kombiniert bewährte Methoden aus DevOps, DataOps und Machine Learning, um den gesamten ML-Lebenszyklus zu optimieren.

**MLOps = DevOps + DataOps + ModelOps**

**ModelOps**

CI/CD + Continuous Training + Continuous Monitoring



### DevOps

- Continuous Integration
- Continuous Deployment
  - Automatisierung
  - Versionskontrolle



### DataOps

- Datenqualität
- Daten-Pipelines
- Datenversionierung
- Datenvalidierung



### ModelOps

- Modell-Training
- Modell-Deployment
- Modell-Monitoring
- Modell-Governance

### Hauptziele

- Schnellere Time-to-Market
- Höhere Modellqualität
- Bessere Skalierbarkeit
- Reduzierte Risiken





### Kernidee

Struktur und Automatisierung in den ML-Lebenszyklus bringen - von der Entwicklung bis zur kontinuierlichen Überwachung in Produktion.


# Warum MLOps?


## Typische Stolpersteine von ML-Projekten

 **Chaotische Notebooks**  
Unstrukturierter Code, keine Versionierung


 **Verlorene Experimente**  
"Welche Parameter waren das nochmal?"


 **Deployment-Probleme**  
"Auf meinem Laptop funktioniert es!"

 **Keine Reproduzierbarkeit**  
Ergebnisse nicht nachvollziehbar

 **Modell-Versionierung**  
Welches Modell läuft in Produktion?


## MLOps als Lösung

 **Strukturierte Workflows**  
Klare Prozesse und Standards

 **Experiment Tracking**  
Alle Parameter und Metriken dokumentiert

 **Automatisiertes Deployment**  
Konsistente Umgebungen

 **Reproduzierbarkeit**  
Jedes Experiment nachvollziehbar

 **Model Registry**  
Zentrale Modellverwaltung

## Der Unterschied zu klassischer Softwareentwicklung



### Datenabhängigkeit

Code + Daten + Parameter = Modell



### Experimenteller Charakter

Trial & Error, Hyperparameter-Tuning



### Performance-Drift

Modelle verschlechtern sich über Zeit



# MLOps-Tools Landschaft

Die richtige Auswahl für jeden Anwendungsfall



## Vendor End-to-End

### AWS SageMaker

Vollständige ML-Plattform von Amazon

### Azure ML

Microsoft's Cloud ML-Service

### Google Vertex AI

Google's einheitliche ML-Plattform

✓ Vollständig integriert



## Open Source End-to-End

### Kubeflow

ML-Workflows auf Kubernetes

### ClearML

Experiment Management & AutoML

### ZenML

Portable ML-Pipeline Framework

✓ Flexibel & kostenlos



## Spezialisierte Tools

### MLflow

Experiment Tracking & Model Registry

### DVC

Data Version Control

### Weights & Biases

Experiment Tracking & Visualization

✓ Best-of-Breed



## Wie wähle ich die richtigen Tools?



### Teamgröße & Expertise:

Kleine Teams → spezialisierte Tools, große Teams → Plattformen



**Budget:** Open Source vs. kommerzielle Lösungen



**Komplexität:** Einfache Projekte vs. Enterprise-Anforderungen



**Compliance:** Datenschutz und Sicherheitsanforderungen



# MLflow als praktische Lösung

Open-Source-Plattform für den gesamten ML-Lebenszyklus



**MLflow**

Das Toolkit für reproduzierbare, skalierbare und kollaborative ML-Projekte



## Tracking

- Parameter loggen
- Metriken verfolgen
- Artefakte speichern
- Experimente vergleichen



## Models

- Einheitliches Format
- Multi-Framework
- Deployment-ready
- Reproduzierbar



## Registry

- Zentrale Verwaltung
  - Versionierung
- Staging/Production
- Governance



## Deployment

- REST APIs
- Docker Container
- Cloud Platforms
- Batch Inference

## ✔ Vorteile

- **Open Source:** Kostenlos und erweiterbar
- **Framework-agnostisch:** Funktioniert mit allem
- **Einfache Integration:** Wenige Zeilen Code
- **Skalierbar:** Von Prototyp bis Enterprise

## 💡 Perfekt für heute

MLflow löst genau die Probleme, die wir in unserem Hands-on erleben werden:  
Experimente nachverfolgen, Modelle verwalten und bereitstellen.

👁️ **Wir werden alle vier Komponenten praktisch kennenlernen!**

# Hands-on: Was wir gemeinsam bauen

## Unser Praxisbeispiel: Iris-Datensatz



### Iris Flower Classification

150 Datenpunkte • 4 Features • 3 Klassen

#### Warum Iris?

Klassischer ML-Datensatz - jeder kennt ihn  
Klein und überschaubar  
Perfekt für MLOps-Konzepte  
Schnelle Trainingszeiten



**Scikit-learn**  
Random Forest



**Matplotlib**  
Visualisierungen

## Unser Workflow

1

### Datenexploration

Iris-Datensatz laden und verstehen

2

### MLflow Setup

Tracking Server starten, erste Runs

3

### Modelltraining

Random Forest mit verschiedenen Parametern

4

### Experiment Tracking

Parameter, Metriken, Plots loggen

5

### Model Registry

Bestes Modell registrieren und versionieren

6

### Deployment

REST-API erstellen und testen



## Was Sie am Ende haben werden

Strukturiertes Notebook



Experiment History



Registriertes Modell



Deploybare API



# Los geht's!

Zeit für die praktischen Übungen

## Schnelle Checkliste

- ✓ Laptop bereit & Internet verbunden
- ✓ Python-Umgebung funktioniert
- ✓ Jupyter Notebook kann gestartet werden
- ✓ pip install bereit für MLflow & Co.

## Unser Weg

- 1 Jupyter Notebook öffnen
- 2 MLflow installieren & starten
- 3 Iris-Datensatz erkunden
- 4 Erstes Modell trainieren

▶ **Jupyter Notebooks starten!**

Fragen? Probleme? Einfach melden! 

[sprinteins.de](https://sprinteins.de)