Lifecycle management of a machine learning model

WHERE INNOVATION MEETS OPPORTUNITY.

BY
KOLESNIKOV ANDREI
SHVAREV ALEXEY

About This Project

This project studies a real-world domain using open-source data and describes a typical machine learning lifecycle for structured data tasks such as binary classification or regression.

The full ML lifecycle is implemented and demonstrated on a single model, covering data preparation, model development, testing, registration, deployment, and ongoing monitoring.

The project also includes setting up real-time inference pipelines and monitoring processes for production ML systems.

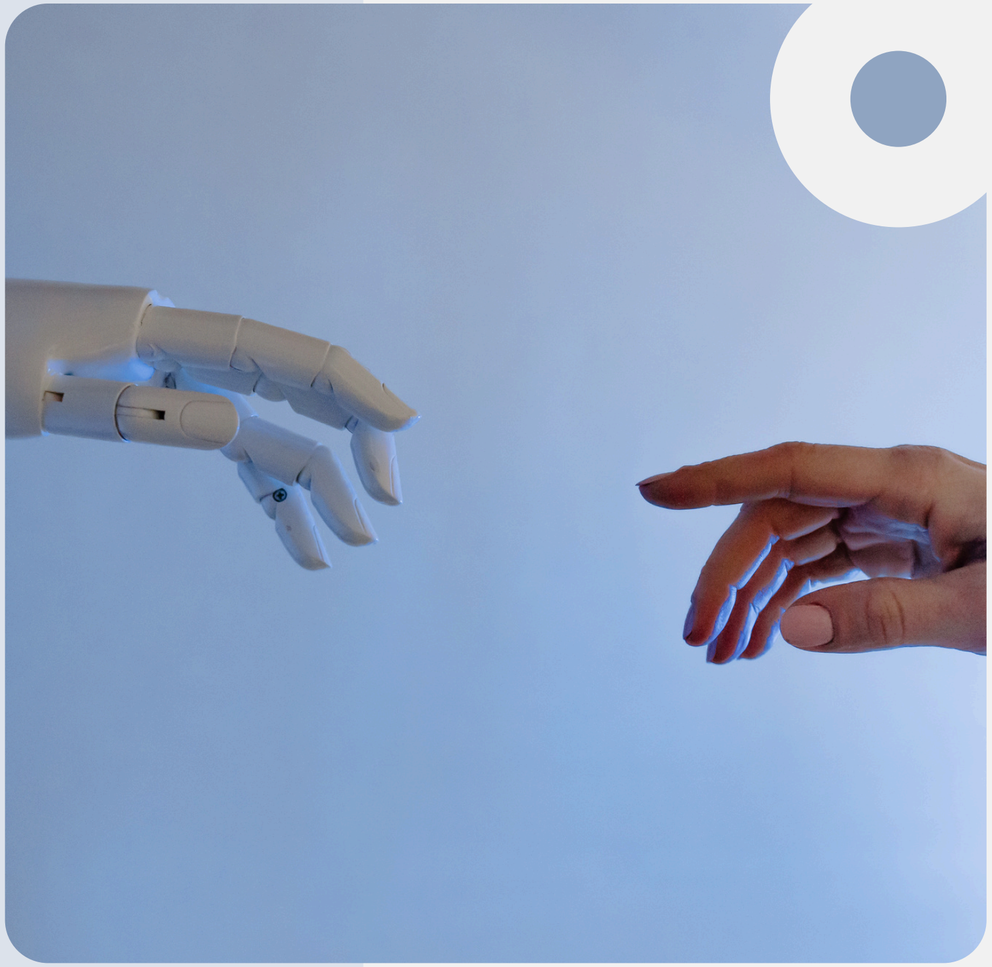
ML Model Lifecycle - Demo Implementation

Demo Scope and Goal

This project demonstrates a managed end-to-end lifecycle of a machine learning model, focusing on operational aspects rather than task-specific performance.
Demo lifecycle includes:

- Data preparation and feature schema
- Model training and validation
- Model versioning and registration
- Deployment for online and batch inference
- Production data logging
- Data monitoring and drift detection

Technology stack:
Python, MLflow, FastAPI, Evidently, Docker



Components of the demo pipeline

Demo Architecture Overview

Training (offline)
- Model training and evaluation
- Baseline (reference) dataset preparation

MLflow (Tracking & Registry)
- Parameters and metrics logging
- Model artifacts and version control

Inference Service (FastAPI)
- Online and batch predictions
- Input schema validation

Monitoring (Evidently)
- Reference vs current data comparison
- Automated monitoring report

Artifacts: model · metrics · production logs · monitoring report

Data stage

Data Preparation and Model Training

Data stage

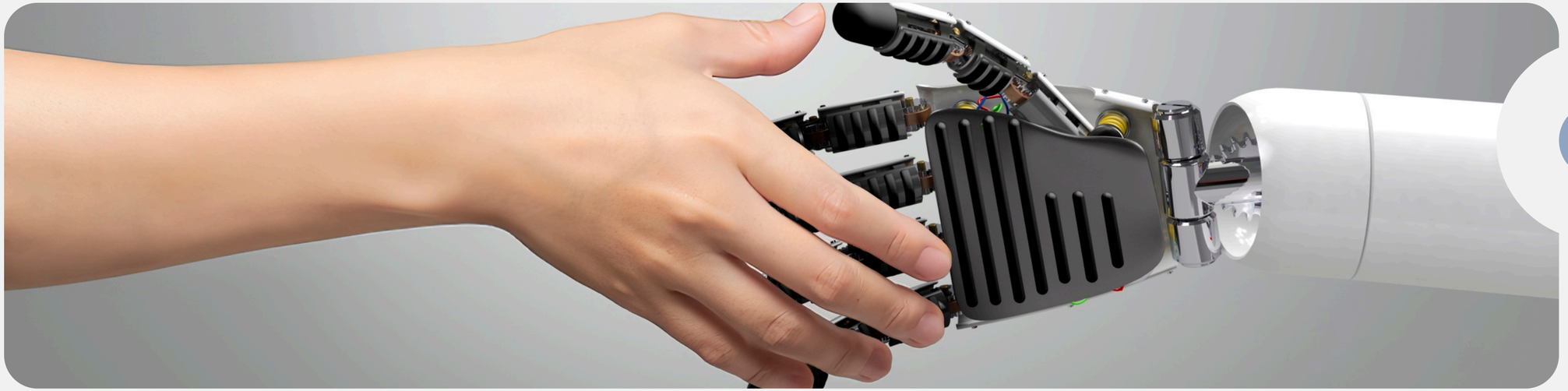
- FEATURE MATRIX CONSTRUCTION
- MISSING VALUE HANDLING
- FIXED FEATURE SCHEMA FOR INFERENCE CONSISTENCY
- REFERENCE DATASET FOR MONITORING BASELINE

Training and validation

- MODEL TRAINING WITH HYPERPARAMETER TUNING
- HOLDOUT EVALUATION USING STANDARD METRICS
- DECISION POINT: MODEL READY FOR REGISTRATION

Outputs

- TRAINED MODEL
- VALIDATION METRICS
- REFERENCE DATASET



Model registration with MLflow

Model Registration and Version Control

What is logged

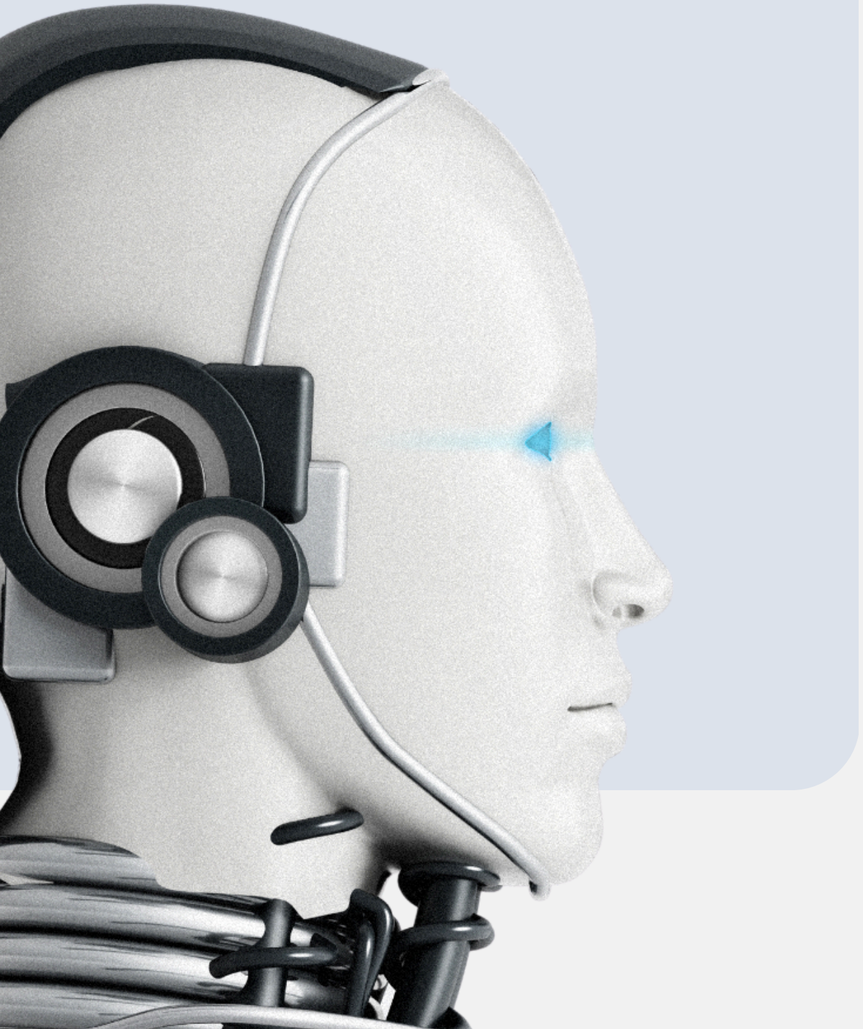
- Training parameters
- Validation metrics
- Model artifacts and environment metadata
- Lifecycle tags (dataset, task)

Why this matters

- Full experiment reproducibility
- Model versioning and auditability

Result

- Registered, reproducible model version



Deployment and Inference

Inference layer (FastAPI)

- /predict - single record inference
- /predict_batch - batch processing
- Strict input feature schema validation

Production data logging

- Input features
- Model prediction / probability
- Inference timestamp

Purpose

- Decoupling training from inference
- Creating a continuous production data stream for monitoring

Data Monitoring and Drift Detection

Baseline vs current

- Reference data - stable baseline dataset
- Current data - production inference logs

Evidently report includes

- Data summary (types, missing values, statistics)
- Data drift detection across features
- Identification of shifted distributions

Value

- Early detection of data distribution changes
- Preventing silent model degradation

Closing the Lifecycle Loop

Results:

- Data drift alert - data pipeline investigation
- Sustained drift - model retraining planning
- Degraded model performance - version rollback

What the demo demonstrates:

- A complete, connected ML lifecycle
- Separation of training, inference, and monitoring
- Practical ModelOps workflow with real artifacts

Next steps:

- Scheduled monitoring and alerts
- Quality monitoring with delayed labels
- Centralized MLflow backend for team usage