End-to-End MLOps Pipeline on the Iris Dataset

(Data Preprocessing • Model Training • Experiment Tracking • Deployment • Automation)

This project was developed to demonstrate how a machine learning model can be taken from **raw data to production** using modern MLOps practices.

As a sample ML problem, the **Iris dataset** is used to build and deploy a classification model.

🎯 Project Objective

The main purpose of this project is to show **how a standard machine learning problem is transformed into a fully automated and reproducible MLOps workflow**.

To achieve this, the classic **Iris flower classification dataset** is used.

The goal is to:

- load and preprocess the Iris dataset
- train a machine learning classification model
- track experiments and compare model performance
- version data and model artifacts
- deploy the trained model through an API
- automate the pipeline using CI/CD principles

This project simulates what a real ML team does when preparing a model for production.

📘 What We Did in This Project

** 1️⃣ Data Loading & Preprocessing**

- Loaded the Iris dataset (150 samples, 4 numerical features, 3 classes)

- Cleaned and standardized numerical features

- Prepared the dataset for model training

- Saved processed data with DVC for reproducibility

** 2️⃣ Model Training**

Trained classification models such as:

- Logistic Regression

- Random Forest Classifier

- Support Vector Machine (SVM)

Key performance metrics (accuracy, precision, recall) were logged with MLflow.

** 3️⃣ Experiment Tracking**

With **MLflow**, we tracked:

- model parameters

- training metrics

- model artifacts

- comparison between multiple model runs

** 4️⃣ Model Serving (FastAPI)**

After selecting the best model:

- exported the trained model

- created an API using FastAPI

- implemented a `/predict` endpoint

- validated input data

- served predictions as JSON

** 5️⃣ Containerization (Docker)**

To make deployment portable:

- created a Dockerfile

- containerized the FastAPI application

- ensured the system runs identically everywhere

** 6️⃣ CI Automation (GitHub Actions)**

Created a CI pipeline that:

- automatically tests code

- checks formatting

- rebuilds the Docker image

- runs workflows on each push

📊 Dataset Information (Iris Dataset)

| Feature | Description |

|---------|-------------|

| sepal_length | Length of the sepal (cm) |

| sepal_width | Width of the sepal (cm) |

| petal_length | Length of the petal (cm) |

| petal_width | Width of the petal (cm) |

| target | Iris-setosa / Iris-versicolor / Iris-virginica |

📬 Contact

Busenur Durak

GitHub: https://github.com/busenur-durak

LinkedIn: https://www.linkedin.com/in/busenur-durak

Iris Veri Seti ile Uçtan Uca MLOps Pipeline

(Veri İşleme • Model Eğitimi • Deney Takibi • Model Servis Etme • Otomasyon)

Bu proje, makine öğrenimi modellerinin **ham veriden üretime** nasıl taşındığını göstermek amacıyla geliştirilmiştir.

Örnek çalışma olarak klasik **Iris çiçek sınıflandırma veri seti** kullanılmıştır.

🎯 Projenin Amacı

Bu projenin temel amacı, basit bir makine öğrenimi probleminden başlayarak **tamamen otomatik, tekrarlanabilir ve üretime hazır bir MLOps yapısı kurmaktır**.

Bu kapsamda Iris veri seti kullanılarak:

- veriyi yükleme ve ön işleme
- sınıflandırma modeli eğitme
- deneyleri MLflow ile takip etme
- veriyi ve modeli DVC ile versiyonlama
- modeli FastAPI üzerinden servis etme

- Docker ile konteynerleştirme

- CI pipeline oluşturma

adımları uygulanmıştır.

📘 Bu Projede Yaptıklarımız

** 1️⃣ Veri Yükleme ve Ön İşleme**

- Iris veri seti yüklendi

- Sayısal özellikler normalize edildi

- Eğitim/veri bölünmesi yapıldı

- İşlenmiş veriler DVC ile versiyonlandı

** 2️⃣ Model Eğitimi**

Aşağıdaki sınıflandırma modelleri denendi:

- Lojistik Regresyon

- Random Forest

- SVM

MLflow ile:

- parametreler

- metrikler

- model karşılaştırmaları

kayıt altına alındı.

** 3️⃣ Deney Takibi (MLflow)**

Her model için:

- accuracy
- precision
- recall
- kullanılan parametreler

MLflow üzerinden karşılaştırıldı.

** 4️⃣ Modeli API Olarak Sunma (FastAPI)**

En iyi model seçildikten sonra:

- model dosyası export edildi
- FastAPI ile `/predict` endpoint'i oluşturuldu
- giriş doğrulaması yapıldı
- sonuçlar JSON olarak döndürüldü

** 5️⃣ Docker ile Dağıtım**

Modelin her ortamda aynı şekilde çalışması için:

- Dockerfile oluşturuldu
- API konteynerize edildi

** 6️⃣ CI Otomasyonu (GitHub Actions)**

Pipeline;

- otomatik test
- kod kalite kontrolü
- Docker build

adımlarını içermektedir.

📊 Iris Veri Seti Özellikleri

Özellik	Açıklama
sepal_length	Çanak yaprak uzunluğu
sepal_width	Çanak yaprak genişliği
petal_length	Taç yaprak uzunluğu
petal_width	Taç yaprak genişliği
target	3 sınıf: setosa, versicolor, virginica

📬 İletişim

Busenur Durak

GitHub: https://github.com/busenur-durak

LinkedIn: https://www.linkedin.com/in/busenur-durak