Project Summary – FastAPI ML Deployment

In this project, I built and deployed a **Decision Tree Classifier** using FastAPI to predict Iris flower species based on sepal and petal measurements. I learned how to integrate machine learning models into APIs efficiently, validate input data using Pydantic, and containerize the application with Docker for production readiness. This assignment strengthened my skills in model deployment, API development, and presenting technical concepts clearly.

Clear, concise explanation script for each code file:

File 1: train_model.py

“Let’s start with train_model.py.

In this file, I imported scikit-learn’s load_iris and DecisionTreeClassifier.

I loaded the Iris dataset, which contains flower measurements and species labels.

Then, I trained a Decision Tree Classifier, which splits data based on features to classify flowers efficiently. Finally, I saved the trained model using joblib as iris_model.pkl, so it can be loaded later for predictions in the API.”

File 2: main.py

“Next is main.py, the core of my FastAPI application.

Here, I imported FastAPI, Pydantic’s BaseModel, and joblib to load the saved model.

I created a FastAPI app instance and defined an input schema class IrisRequest with sepal and petal measurements as float fields.

I loaded the saved Decision Tree model and defined a POST endpoint /predict. This endpoint receives input data, converts it into a list format, predicts the species using the model, and returns the predicted species as a JSON response."

File 3: requirements.txt

"This file, requirements.txt, lists all dependencies needed to run my project.

It includes:

fastapi

uvicorn (for running the server)

scikit-learn

joblib

When deploying or containerizing, running pip install -r requirements.txt installs these libraries to ensure the application works correctly across environments."

File 4: Dockerfile

"Finally, the Dockerfile enables containerization of my application.

It uses the Python 3.9 base image, sets the working directory to /app, copies the requirements file, and installs dependencies. Then, it copies all project files into the container, exposes port 8000, and runs the FastAPI app with uvicorn on host 0.0.0.0 and port 8000.

This ensures the app can be deployed consistently on any server or cloud platform."