FTL Machine Learning Project: Deployment Submission

Deployment

1. Overview

Deployment phase consists in making the machine learning model, which is an ensemble of Random Forest and XGBoost to predict crop yields in Ethiopia, available in a real-world or production setting. This step will involve serializing the trained model, serving mechanism configuration using Flask API, basic security, and monitoring/logging to assure reliability. The procedures were developed to emulate a deployable framework that can be scaled to a cloud platform to make it practical by the agricultural community.

2. Model Serialization

The trained ensemble model, consisting of a Random Forest Regressor and an XGBoost Regressor, which

- ➢ Was serialized using the joblib library to save the model objects to disk.
- ➢ The models were saved as .pkl files (random_forest_model.pkl and xgboost_model.pkl) in a models directory.

This format was chosen for its efficiency in storing Python objects and compatibility with the Flask API for loading during serving. The serialization process was logged to track success, ensuring no data loss during saving.

3. Model Serving

The serialized models are served using a Flask-based web application, which provides a lightweight and flexible platform for hosting the prediction API. The application runs locally on ***http://0.0.0.0:5000*** in debug mode, making it accessible for testing. The Flask framework was chosen because of its simplicity for small-scale deployment and ease of integration with Python-based machine learning models. For a production environment, the Flask app would be deployed on a cloud service.

4. API Integration

The machine learning model is integrated into an API with a single endpoint, /predict, accessible via POST requests. The API accepts JSON input containing a features object with the top 10 selected features ***(Prev_Year_Yield, Crop_Sugar cane, Years_Since_2000, Precipitation, Water_Availability, NDVI_Solar_Interaction, Temperature, NDVI, SolarRadiation, Prev_Year_NDVI),*** normalized to the 0–1 range used during training. The response is a JSON object with a prediction field (yield in kg/ha) and a unit field. Example input and output are as follows:

- INPUT

```json
{
  "features": {
    "Prev_Year_Yield": 0.5,
    "Crop_Sugar cane": 1,
    "Years_Since_2000": 20,
    "Precipitation": 0.3,
    "Water_Availability": 0.1,
    "NDVI_Solar_Interaction": 0.15,
    "Temperature": 0.4,
    "NDVI": 0.45,
    "SolarRadiation": 0.35,
    "Prev_Year_NDVI": 0.4
  }
}
```

- OUTPUT

```json
{
  "prediction": 38134.0537049378,
  "unit": "kg/ha"
}
```

- The API loads the serialized models, computes the weighted ensemble prediction (60% Random Forest, 40% XGBoost), and returns the result, ensuring consistency with the training pipeline.

5. Security Considerations

Basic security measures were implemented to protect the API. Authentication is handled using **flask-httpauth** with HTTP Basic Authentication, requiring a username (admin) and password (password123) to access the /predict endpoint. This prevents unauthorized access during testing. However, this is a placeholder for production, where more robust methods like OAuth2 or JSON Web Tokens (JWT) would be used, along with HTTPS encryption to secure data in transit. Input validation is also performed to ensure the features object contains all required fields, returning a 400 error if invalid. In a real-world setting, additional measures would be added to make it more secure.

6. Monitoring and Logging

The deployed model's performance is monitored and logged to a file named model_deployment.log. Logging is configured with the logging module to record timestamps, log levels, and messages, capturing key events such as model serialization, prediction requests, and errors. Performance metrics (MAE, MSE, and R^2) are logged on the test set to provide an understanding for ongoing evaluation. An alerting mechanism is not implemented in this simulation but could involve sending notifications. Future enhancements would include real-time monitoring dashboards to track prediction accuracy and other important things.