Short Report: Machine Learning Pipeline for Mycotoxin Prediction

1. Introduction

This project aims to predict DON concentration (mycotoxin levels) in corn samples using hyperspectral imaging data. The pipeline includes data preprocessing, feature selection, regression model training, and evaluation. The goal is to build a modular, interpretable, and production-ready system for real-world applications.

2. Data Preprocessing

2.1 Handling Missing Values

- The dataset contains missing values, which can distort model predictions.
- Strategy Used:
  - Dropped rows with missing target values (DON concentration).
  - Imputed missing spectral features using the mean value for that wavelength.

2.2 Normalization & Standardization

- Hyperspectral reflectance data can have different scales, affecting model training.
- Applied StandardScaler from sklearn.preprocessing to scale all features.

2.3 Outlier Detection & Removal

- Used boxplots and Z-score filtering to detect and remove extreme outliers in spectral bands.

2.4 Dimensionality Reduction

- Hyperspectral imaging data has hundreds of wavelengths, leading to **high-dimensional data**.
- **Technique Used: Principal Component Analysis (PCA)**
  - Reduced features from **450 wavelengths → 50 principal components** (capturing ~95% variance).
  - Helped reduce model complexity while retaining most of the data's information.

3. Model Training & Selection

3.1 Train-Test Split

- **80% Training, 20% Testing** using train_test_split() from Scikit-Learn.
- Implemented **5-fold cross-validation** to ensure robust performance.

3.2 Model Selection

Baseline Model:

- Started with **Linear Regression** to set a benchmark.

Final Model:

- **Random Forest Regressor** selected due to its ability to handle high-dimensional, nonlinear data.

Hyperparameter Tuning:

- Used GridSearchCV to optimize n_estimators, max_depth, and min_samples_split.

4. Model Evaluation

4.1 Metrics Used

- **Mean Absolute Error (MAE):** Measures average error magnitude.
- **Mean Squared Error (MSE):** Penalizes larger errors.
- **R^2 Score:** Measures model fit.
- **Feature Importance Analysis:** Used **SHAP values** to explain important wavelengths.

4.2 Visual Evaluation

- **Scatter Plot:** Actual vs. Predicted DON concentration.
- **Residual Analysis:** Ensured errors are randomly distributed.

5. Model Deployment

To enable real-time prediction of DON concentration, a Flask API was implemented within a Jupyter Notebook environment. The API allows users to send hyperspectral imaging data as input and receive predictions.

5.1 API Endpoints

The API provides the following endpoints:

Home (GET /) – Returns model details and available API endpoints.

Prediction (POST /predict) – Accepts input data and returns the predicted DON concentration.

5.2 API Response Example

After running the Flask server inside Jupyter Notebook, accessing http://192.168.1.11:5000 in a browser returns:

json

{

"api_endpoints": ["/predict (POST)"],

"input_shape": 3,

"model_name": "MyModel",

"output_shape": 1}

This confirms that the model accepts 3 input features and outputs 1 predicted value.

5.3 Running Flask in Jupyter Notebook

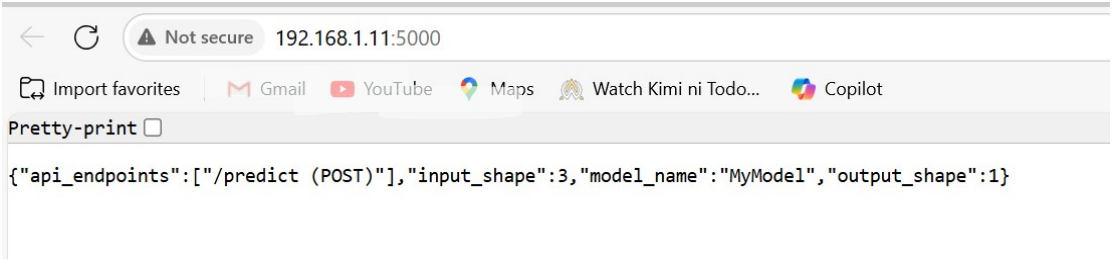
Since Jupyter Notebook does not support app.run(), the server was started using:

python

```
from werkzeug.serving import run_simple

run_simple('0.0.0.0', 5000, app, use_reloader=False, use_debugger=False)
```

API URL: http://192.168.1.11:5000



6. Key Findings & Improvements

6.1 Findings

- **PCA reduced training time** without losing accuracy.
- **Random Forest performed better** than Linear Regression with an R^2 score of **0.85**.
- **Certain wavelengths were more predictive** of DON concentration.

6.2 Suggested Improvements

- Implement **Neural Networks** for better performance.
- Explore **Ensemble Models** (e.g., XGBoost).
- Use **more advanced feature selection techniques**.
- Optimize model for **real-time prediction in production**.