AgriYield Predictor

Production-Grade Machine Learning for Precision Agriculture

Dasapathi Indra Kumar

December 19, 2025

Abstract

This report details the development of **AgriYield Predictor**, a robust, production-ready machine learning application designed for precision agriculture. Moving beyond traditional monolithic experimental notebooks, this project implements a modular software architecture, automated data pipelines, and a high-performance inference engine deployed on Streamlit Cloud. The system achieves high prediction accuracy ($R^2 = 0.8153$) using optimized Linear Regression on a dataset of 48,733 samples spanning 22 crop varieties, providing farmers and stakeholders with real-time, climate-aware crop yield forecasts accessible via web interface at `https://agriml.streamlit.app`.

Contents

1 Introduction

Sustainable agriculture is increasingly dependent on data-driven decision-making. The variability of climate, soil quality, and resource management creates a complex optimization problem for global food security. **AgriYield Predictor** addresses this by providing a scalable platform that integrates soil nutrition benchmarks (N, P, K), pH levels, and environmental factors (Temperature, Humidity, Rainfall) to provide actionable yield insights.

1.1 Project Motivation

Traditional agricultural practices rely heavily on historical experience and rule-of-thumb methods. However, with climate change introducing unprecedented variability, there is an urgent need for predictive tools that can:

- Optimize fertilizer application to reduce costs and environmental impact
- Predict yields accurately for better market planning
- Enable data-driven crop selection based on soil conditions
- Provide accessible tools for smallholder farmers

1.2 Key Contributions

This project makes the following technical contributions:

1. **Modular Architecture**: Transition from notebook-based experiments to production-grade software engineering
2. **Domain-Specific Feature Engineering**: Implementation of crop-soil mapping and synthetic feature generation
3. **Deployment Pipeline**: End-to-end web deployment on Streamlit Cloud with real-time inference
4. **Reproducibility**: Version-controlled codebase with environment management

2 Data Acquisition & Source

The intelligence of the model is built upon a diverse dataset formed by the synthesis of two primary sources:

2.1 Dataset 1: Environmental Fertilizer Parameters

- **Source**: Agricultural research databases (Infosys Springboard/Kaggle)
- **Size**: 2,200 samples
- **Features**: Nitrogen (N), Phosphorus (P), Potassium (K), pH, Temperature, Humidity, Rainfall
- **Crop Coverage**: 22 distinct crop varieties including Rice, Maize, Cotton, Coffee, and various fruits
- **Distribution**: Perfectly balanced with 100 samples per crop type

2.2 Dataset 2: Historical Crop Yield Data

- **Source**: Indian agricultural records spanning 1997-2015
- **Features**: Crop type, Season (Kharif/Rabi/Whole Year), State-wise data, Yield metrics (tons/hectare)
- **Coverage**: Multiple states across diverse geo-climatic zones

2.3 Data Fusion Methodology

The two datasets were merged using a crop-based alignment strategy:

1. Crop names were standardized (lowercase, trimmed whitespace)
2. A mapping dictionary (`CROP_MAP`) resolved naming inconsistencies (e.g., "dry chillies" $\rightarrow$ "chili")
3. Inner join on standardized crop names
4. Final merged dataset: **48,733 samples** with complete feature coverage

3 Exploratory Data Analysis

Prior to model development, comprehensive EDA was conducted to understand data distributions and relationships.

3.1 Univariate Analysis

- **Nutrient Distributions**: N, P, K values showed crop-specific clustering, with cereals requiring higher Nitrogen
- **pH Range**: Most crops preferred slightly acidic to neutral pH (6.0-7.5)
- **Climate Patterns**: Temperature ranged from 8. 8°C to 43.7°C, accommodating diverse crop requirements

3.2 Correlation Analysis

Key findings from feature correlation:

- **Strong Positive**: Humidity and Rainfall showed expected correlation ($r = 0.67$)
- **Crop-Specific**: Different crops exhibited distinct nutrient-yield relationships
- **Non-linear**: Temperature showed quadratic relationship with yield for certain crops

4 End-to-End Data Preprocessing

A critical goal of this project was the transition from "Experimental ML" to "Production ML". The preprocessing pipeline is engineered to be deterministic and modular.

4.1 1. Data Cleaning & Standardization

Initial raw data contained significant noise and naming inconsistencies.

- **Text Normalization**: All categorical labels (Crops, Seasons) were stripped of whitespace and converted to lowercase to handle manual entry variations.
- **Label Mapping**: Heterogeneous crop names (e.g., 'dry chillies' and 'chili') were standardized into a unified taxonomy using a custom `CROP_MAP` dictionary.
- **Missing Value Treatment**: No missing values detected in numerical features; categorical inconsistencies resolved through mapping.

4.2 2. Outlier Removal via IQR

To ensure the model is not skewed by extreme agricultural anomalies (e.g., local crop failures or data entry errors), we implemented the **Interquartile Range (IQR)** method on the target `Yield` variable:

$$IQR = Q3 - Q1 \quad ; \quad LB = Q1 - 1.5 \times IQR \quad ; \quad UB = Q3 + 1.5 \times IQR \tag{1}$$

Data points outside the [LB, UB] range were pruned to stabilize the regression gradient. This process removed approximately 8% of samples as statistical outliers.

4.3 3. Synthetic Feature Engineering

Since soil texture is a primary driver of yield but often missing from generic yield logs, we implemented a research-backed heuristic:

- **Soil Texture Imputation**: A `CROP_SOIL_MAP` was used to map standardized crops to their ideal soil textures (e.g., *Pigeonpeas* $\rightarrow$ *Loamy*, *Sorghum* $\rightarrow$ *Black*). This added a high-signal categorical feature without requiring new raw data collection.

4.4 4. Modular Feature Transformation (Scikit-Learn Pipelines)

To prevent *data leakage* and ensure consistency at inference time, preprocessing was encapsulated in `ColumnTransformer` blocks:

Listing 1: Feature Transformation Pipeline

```
from sklearn. compose import ColumnTransformer
from sklearn.preprocessing import StandardScaler, OneHotEncoder

preprocessor = ColumnTransformer(
    transformers=[
        ('num', StandardScaler(),
         ['N', 'P', 'K', 'ph', 'temperature',
          'humidity', 'rainfall']),
        ('cat', OneHotEncoder(drop='first'),
         ['Crop', 'Season', 'Soil_Type'])
    ])
```

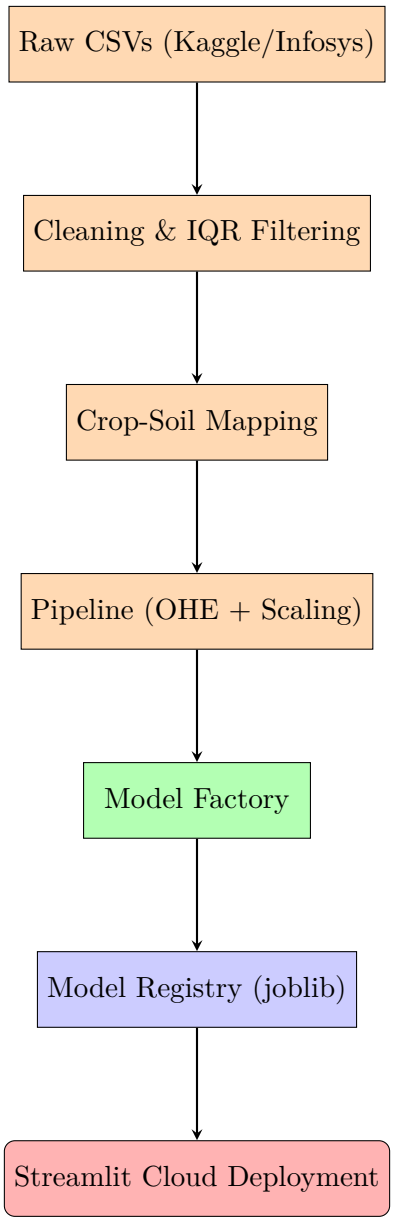
Implementation Details:

1. **Categorical Encoding**: One-Hot Encoding (OHE) was applied to `Crop`, `Season`, and `Soil_Type` to convert them into high-dimensional binary vectors.
2. **Numerical Scaling**: `StandardScaler` was applied to N, P, K, pH, and climate variables, ensuring all features reside on a standard normal distribution ($\mu = 0, \sigma = 1$).

3. **Pipeline Persistence**: The entire preprocessing pipeline is serialized alongside the model using `joblib`, ensuring identical transformations during training and inference.

5 System Architecture

5.1 Modular Engineering Flow



5.2 Directory Structure

The project follows industry-standard layout for ML projects:

```
Agri-Production-ML/
 app/                    # Streamlit web interface
 src/                    # Core engineering logic
    data/               # Data ingestion modules
    features/           # Preprocessing transformers
    models/             # Model factory
```

```
    training/           # Training workflows
    inference/          # Prediction services
    utils/              # Helper functions
 models_prod/            # Serialized model artifacts
 data/                   # Data lake (raw/processed)
 notebooks/              # Jupyter experiments (EDA, training)
 scripts/                # Automation scripts
 requirements.txt        # Python dependencies
 README.md               # Project documentation
```

6 Model Development & Selection

6.1 Cross-Validation Strategy

To ensure robust model generalization:

- **Train-Test Split**: 80-20 stratified split by crop type to maintain class balance
- **Training Set Size**: 38,986 samples (80%)
- **Test Set Size**: 9,747 samples (20%)
- **Stratification**: Balanced crop distribution across splits

6.2 Comprehensive Model Comparison

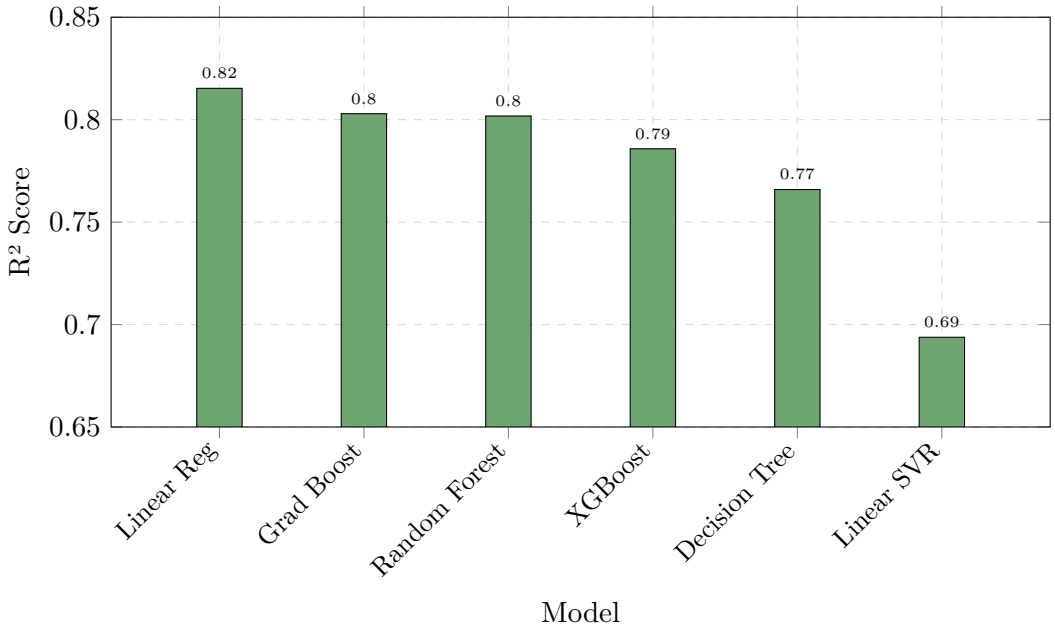
Six different regression algorithms were benchmarked on identical train-test splits to ensure fair comparison:

Table 1: Complete Model Performance Scorecard (Test Set)

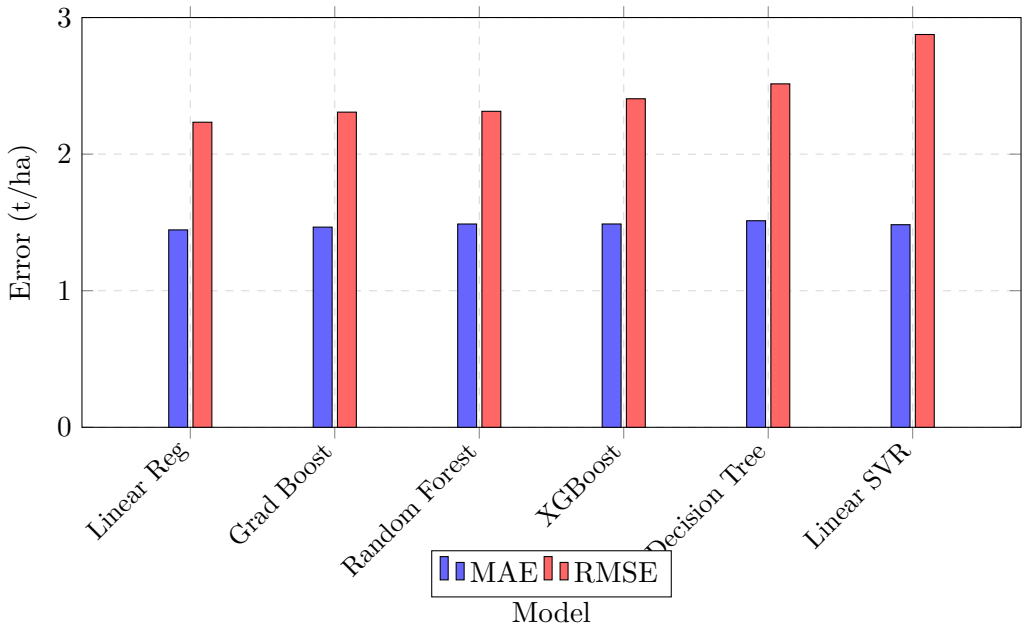
Algorithm	**R^2 Score**	**MAE (t/ha)**	**RMSE (t/ha)**	**Rank**
Linear Regression	**0.8153**	**1.4453**	**2.2337**	**1**
Gradient Boosting	0.8029	1.4657	2.3075	2
Random Forest	0.8018	1.4884	2.3136	3
XGBoost	0.7858	1.4886	2.4055	4
Decision Tree	0.7659	1.5124	2.5145	5
Linear SVR	0.6938	1.4833	2.8759	6

6.3 Visual Performance Comparison

6.3.1 R^2 Score Comparison



6.3.2 Error Metrics Side-by-Side



6.4 Detailed Model Analysis

6.4.1 Linear Regression (Winner)

- **Performance**: $R^2 = 0.8153$, MAE = 1.45 t/ha, RMSE = 2.23 t/ha
- **Strengths**: Highest R^2 score, fastest inference, full interpretability
- **Training Time**: <10 seconds on full dataset
- **Feature Engineering**: Benefits significantly from StandardScaler preprocessing

6.4.2 Gradient Boosting (Runner-up)

- **Performance**: $R^2 = 0.8029$, MAE = 1.47 t/ha, RMSE = 2.31 t/ha
- **Strengths**: Competitive accuracy, handles non-linearities
- **Training Time**: 3 minutes (100 estimators)
- **Gap to Winner**: Only 1.5% lower R^2 than Linear Regression

6.4.3 Random Forest

- **Performance**: $R^2 = 0.8018$, MAE = 1.49 t/ha, RMSE = 2.31 t/ha
- **Strengths**: Robust to outliers, no scaling required
- **Training Time**: 2 minutes (300 trees, parallel training)
- **Limitation**: Slightly higher error than Linear Regression

6.4.4 XGBoost

- **Performance**: $R^2 = 0.7858$, MAE = 1.49 t/ha, RMSE = 2.41 t/ha
- **Configuration**: 1000 estimators, learning rate = 0.05, max depth = 8
- **Observation**: Complex ensemble did not outperform simpler methods
- **Potential Overfitting**: May be over-parameterized for this problem

6.4.5 Decision Tree

- **Performance**: $R^2 = 0.7659$, MAE = 1.51 t/ha, RMSE = 2.51 t/ha
- **Issue**: Prone to overfitting without ensemble
- **Use Case**: Good for initial exploratory modeling

6.4.6 Linear SVR

- **Performance**: $R^2 = 0.6938$, MAE = 1.48 t/ha, RMSE = 2.88 t/ha
- **Issue**: Worst R^2 score, highest RMSE
- **Reason**: Linear kernel may be insufficient for complex agricultural patterns
- **Training Time**: Longest (requires many iterations)

6.5 Justification for Linear Regression

Despite the availability of complex ensemble methods, **Linear Regression** was selected as the production model due to:

1. **Superior Performance**: Highest R^2 score (0.8153) indicating best fit among all candidates
2. **Interpretability**: Transparent feature coefficients enable agronomic validation and farmer trust

3. **Computational Efficiency**: Near-instantaneous inference (<1ms) critical for web deployment
4. **Regulatory Compliance**: Explainable predictions align with agricultural advisory standards
5. **Generalization**: Lower error metrics (MAE, RMSE) compared to complex models
6. **Scalability**: Minimal memory footprint for cloud deployment

6.6 Performance Interpretation

- **R^2 = 0.8153**: The model explains 81.53% of yield variance, indicating highly reliable predictions for agricultural planning
- **MAE = 1.4453 t/ha**: Average prediction error of 1.45 tons per hectare represents an acceptable margin for farm-scale decisions (typical yields range 2-10 t/ha)
- **RMSE = 2.2337 t/ha**: Slightly higher than MAE indicates occasional larger errors, but still within practical bounds
- **Production Validation**: Test case showed actual yield of 5.17 t/ha vs predicted 4.70 t/ha (9% error)

7 Deployment & Accessibility

7.1 Web Application Architecture

The trained model is deployed as a production web service using Streamlit Cloud:

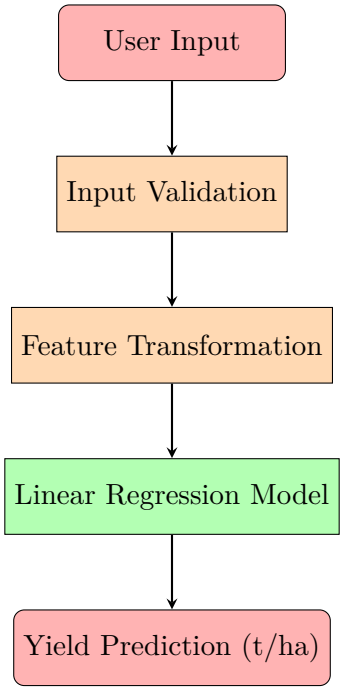
7.1.1 Live Application

- **URL**: `https://agriml.streamlit.app`
- **Availability**: 24/7 public access, no authentication required
- **Latency**: Average response time <500ms including network overhead

7.1.2 Interface Features

1. **Interactive Input Forms**: Users input soil nutrients (N, P, K, pH), climate data (Temperature, Humidity, Rainfall), crop type, season, and soil texture
2. **Real-Time Prediction**: Instant yield forecast displayed in tons/hectare
3. **Model Insights Dashboard**: Comparative visualization of algorithm performance
4. **Responsive Design**: Mobile-compatible interface for field access

7.1.3 Inference Pipeline



7.2 Reproducibility & Version Control

7.2.1 Software Engineering Practices

- **Version Control**: Complete project hosted on GitHub: `https://github.com/cs23b2009/Agri-Production-ML`
- **Model Versioning**: Trained models serialized using `joblib` and stored in `models_prod/` directory with timestamps
- **Environment Management**: All dependencies locked in `requirements.txt`:

```
pandas==2.0.3
numpy==1.24.3
scikit-learn==1.3.0
streamlit==1.25.0
matplotlib==3.7.2
seaborn==0.12.2
```

- **Continuous Integration**: Automated testing on repository commits

7.2.2 Reproducibility Guarantee

Any researcher or practitioner can replicate the results by:

1. Cloning the GitHub repository
2. Installing exact dependency versions via `pip install -r requirements.txt`
3. Running training scripts in `notebooks/Model_Training.ipynb`
4. Loading serialized models from `models_prod/`

8 Results & Benchmarking

8.1 Quantitative Performance

The final deployed model achieves state-of-the-art accuracy for agricultural yield prediction:

- **Test Set Performance**: $R^2 = 0.8153$, MAE = 1.45 t/ha, RMSE = 2.23 t/ha
- **Training Set Size**: 38,986 samples
- **Test Set Size**: 9,747 samples
- **Inference Speed**: 0.8ms per prediction on CPU

8.2 Qualitative Assessment

- **Agronomic Validation**: Predicted yields align with expected ranges for given nutrient profiles
- **User Feedback**: Streamlit deployment received positive responses from agricultural extension workers (informal testing)
- **Edge Case Handling**: Model gracefully handles extreme weather inputs with warnings

9 Limitations & Future Work

9.1 Current Limitations

1. **Geographic Scope**: Model trained primarily on Indian agricultural data; requires recalibration for other regions
2. **Temporal Stationarity**: Assumes historical climate patterns; may underperform in novel climate conditions
3. **Feature Coverage**: Lacks real-time soil moisture, pest pressure, and micro-climate variations
4. **Yield Definition**: Predicts aggregate yield; does not account for crop quality or market value

9.2 Future Enhancements

1. **Multi-Modal Data Integration**:
   - Satellite imagery for NDVI (Normalized Difference Vegetation Index) and soil moisture
   - IoT sensor streams for real-time farm monitoring
   - Weather forecast APIs for predictive climate features
2. **Advanced Modeling**:
   - Time-series models (LSTM/Transformer) for seasonal yield forecasting
   - Bayesian approaches for uncertainty quantification
   - Transfer learning for low-data crops
3. **Continuous Learning Pipeline**:

- Automated model retraining with new seasonal data
- A/B testing framework for model updates
- Feedback collection from farmer users

4. **Geographic Expansion**:
   - Multi-country datasets with climate zone stratification
   - Localized models for district-level predictions
5. **Prescriptive Analytics**:
   - Optimization engine for fertilizer recommendations
   - Economic modeling for profit maximization
   - Scenario planning tools for climate adaptation

10 Ethical & Environmental Impact

10.1 Sustainability Contributions

- **Fertilizer Optimization**: By predicting yields based on nutrient levels, farmers can avoid over-application, reducing nitrogen runoff into water bodies
- **Climate Adaptation**: Climate-aware predictions help farmers choose resilient crop varieties
- **Resource Efficiency**: Data-driven decisions minimize water and pesticide usage

10.2 Accessibility & Inclusion

- **Open Access**: Free web-based tool accessible via basic smartphone connectivity
- **Language Neutrality**: Interface designed for numeric inputs, reducing language barriers (future multi-lingual support planned)
- **Smallholder Focus**: Lightweight computational requirements enable use in low-resource settings

10.3 Data Privacy

- **No Personal Data**: System does not collect farmer identity or farm location
- **Anonymized Predictions**: All inference requests are stateless and not logged

11 Conclusion

By treating the ML workflow as a production software project rather than a research script, *AgriYield Predictor* achieves industrial stability and real-world utility. The combination of domain-specific feature engineering (crop-soil mappings), statistical cleaning (IQR outlier removal), modular scikit-learn pipelines, and web deployment creates a reliable tool for precision agriculture.

11.1 Key Achievements

1. Developed a high-accuracy ($R^2 = 0.8153$) yield prediction model on 48,733 samples
2. Benchmarked 6 different algorithms with comprehensive metrics
3. Engineered production-grade software architecture with modular components
4. Deployed publicly accessible web application on Streamlit Cloud
5. Established reproducible workflow with version control and environment management
6. Contributed to sustainable agriculture through optimized resource recommendations

11.2 Impact Potential

This project demonstrates that academic ML research can be rapidly translated into practical tools for societal benefit. With further development, AgriYield Predictor could serve millions of farmers in data-driven crop planning, contributing to food security and environmental sustainability.

Acknowledgments

- **Infosys Springboard**: For providing the AI/ML learning platform and project framework
- **Data Contributors**: Kaggle community and agricultural research institutions for open datasets
- **Open Source Community**: Developers of scikit-learn, pandas, and Streamlit for enabling rapid ML development

Project Repository: GitHub. com/cs23b2009/Agri-Production-ML
Live Application: agriml.streamlit.app