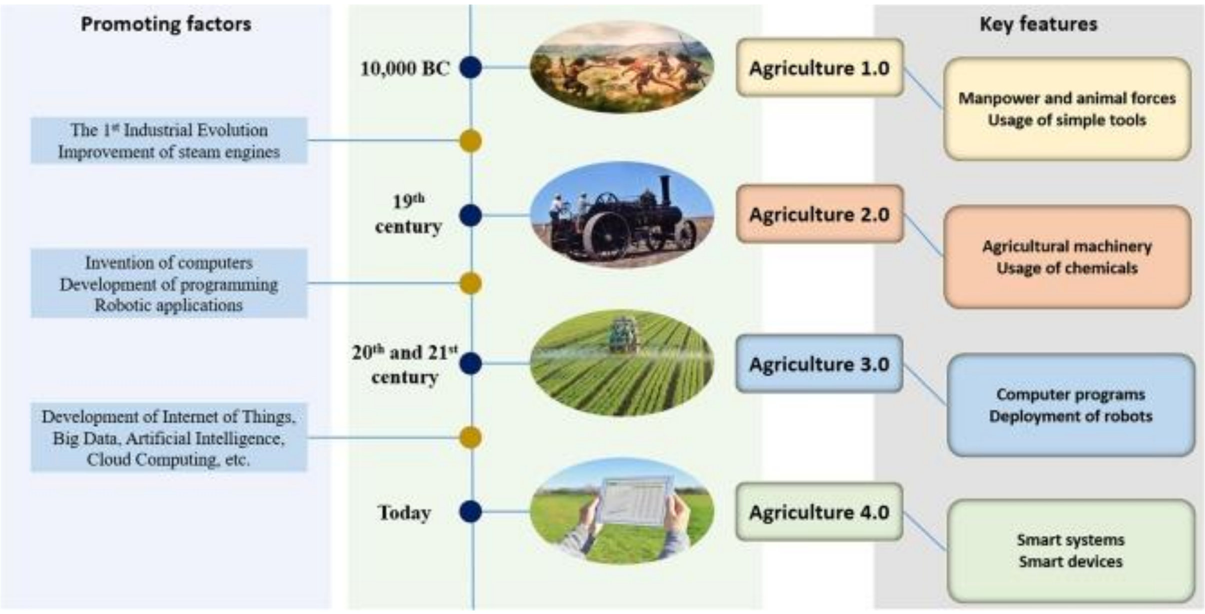
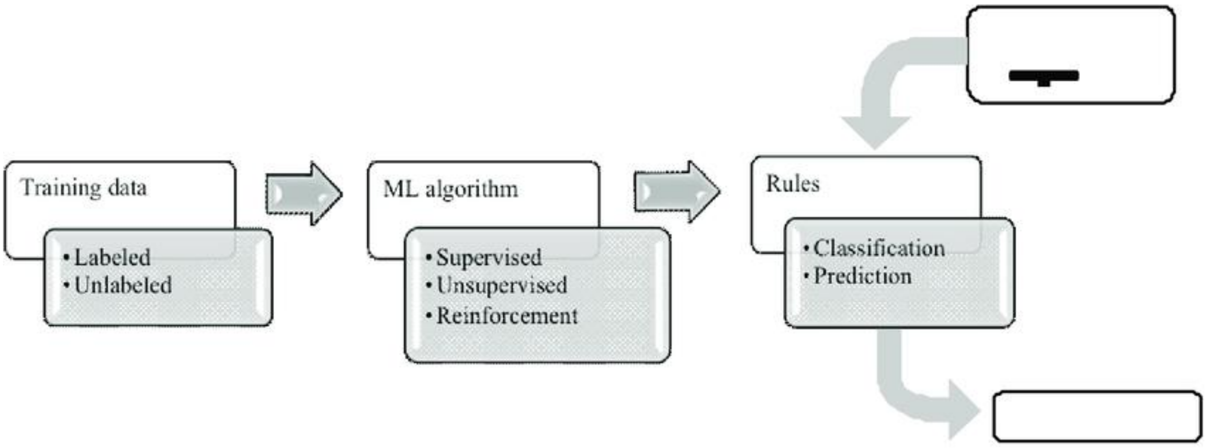
🌽 Project Title

AI-Based Crop Yield Prediction & Decision Support System for Smart Farming





🎯 What Exactly Is Our Project?

This project uses **Machine Learning** to help farmers by:

1️⃣ **Predicting crop yield** based on soil and climate conditions
2️⃣ **Recommending the most suitable crop** for given conditions

It supports:

- **Single input prediction** (form-based)
- **Bulk prediction** (CSV upload with large datasets)

This is a **Predictive + Advanced Analytics** project.

🧠 What Problems Does It Solve?

- Farmers cannot accurately predict yield due to climate variability
- Wrong crop selection causes loss
- Decisions are often intuition-based

👉 Our system provides **data-driven AI recommendations**.

🛠️ Technology Stack (FINAL)

🔹 **Programming Language**

- **Python**

🔹 **Machine Learning**

- **scikit-learn**
- Algorithms:
  - Random Forest Regressor (Yield)
  - Random Forest Classifier (Crop Recommendation)

🔹 **Data Handling & Visualization**

- pandas, numpy
- matplotlib, seaborn

🔹 **Backend (Deployment)**

- **Flask**

🔹 **Frontend (UI)**

- HTML, CSS
- Bootstrap

🔹 **Model Storage**

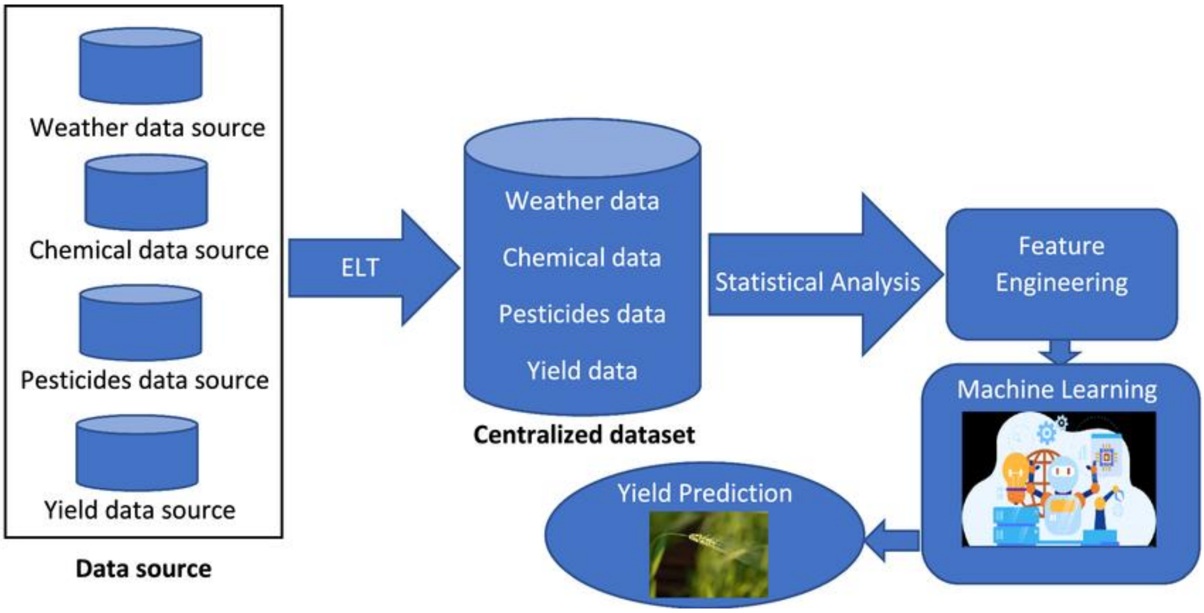
- joblib (.pkl files)

🧩 What ML Models Are We Using?

Task	Model Type	Algorithm
Crop Yield Prediction	Regression	Random Forest Regressor
Crop Recommendation	Classification	Random Forest Classifier

📌 Using **two models** = advanced analytics ✓

🏗 Overall System Architecture



Agriculture Dataset

↓

Data Preprocessing

↓

ML Model Training (Offline)

↓

Save Models (.pkl)

↓

Flask Web Application

↓

User Input / CSV Upload

↓

Prediction & Recommendation

↓

Decision Support Output

🪜 STEP-BY-STEP IMPLEMENTATION PLAN

🟢 STEP 1: Dataset Collection

- Collect agriculture dataset (CSV)
- Includes soil, weather, crop, yield

🟢 STEP 2: Data Preprocessing

- Handle missing values
- Encode categorical features
- Scale numerical features
- Feature selection

📌 **Same preprocessing saved & reused in Flask**

🟢 STEP 3: Exploratory Data Analysis (EDA)

- Yield vs Rainfall
- Yield vs Temperature
- Soil Type vs Crop

📊 Use graphs (important for marks)

🟢 STEP 4: ML Model Training (Offline)

- Train multiple models
- Compare performance
- Select best models

Metrics:

- RMSE, MAE, R^2 (Yield)
- Accuracy (Crop recommendation)

🟢 STEP 5: Save Trained Models

```
joblib.dump(yield_model, "yield_model.pkl")

joblib.dump(crop_model, "crop_model.pkl")
```

🟢 STEP 6: Build Flask Application

- Load trained models
- Create prediction routes
- Handle:
  - Form input
  - CSV upload

🟢 STEP 7: Prediction Logic

- Apply same preprocessing
- Predict:
  - Expected Yield
  - Best Crop

🟢 STEP 8: Output & Decision Support

- Show predicted yield
- Recommend crop
- Give simple explanation

Example:

“Rice is recommended due to high rainfall and suitable soil pH.”

🟢 STEP 9: Bulk CSV Prediction (Advanced)

- User uploads CSV
- Validate columns
- Predict for all rows
- Generate downloadable result CSV

🟢 STEP 10: Testing & Validation

- Test multiple scenarios
- Handle wrong inputs
- Validate predictions