SUMMER TRAINING/INTERNSHIP

PROJECT REPORT

(Term June -July 2025)

CROP RECOMMENDATION SYSTEM USING MACHINE LEARNING

School of Computer Science and Engineering

(2023-2027)

Submitted by

Rupal Tripathi
Registration Number: 12306762

Course Code: PETV79
Machine Learning Made Easy: From Basics To Ai Applications

Under the Guidance of

Dr. Mahipal Singh Papola
Assistant Professor

CERTIFICATE

This is to certify that the project report titled **“Crop Recommendation System Using Machine Learning”** is a bonafide record of work carried out by **Rupal Tripathi** bearing registration no. **12306762** in partial fulfillment of the requirements for the award of the degree of B. Tech in Computer Science and Engineering during the academic year 2024–25.

Dr. Mahipal Singh Papola **Rupal Tripathi**

(Assistant Professor)

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my guide, **Dr. Mahipal Singh Papola**, for their constant support, motivation, and invaluable guidance throughout the development of this project. I am also thankful to the Head of the Department, all faculty members, and my peers who contributed indirectly to the successful completion of this work. Lastly, I acknowledge my family and friends for their encouragement and moral support.

TABLE OF CONTENTS

INTRODUCTION

Company Profile

This project was developed independently as part of an academic training initiative in machine learning. The training program was designed to give students hands-on exposure to real-world applications of data science and artificial intelligence. It emphasized solving domain-specific challenges using datasets drawn from actual use cases, promoting a blend of theoretical understanding and practical implementation.

The institution provided a structured learning environment supported by mentorship and collaborative learning. Students were encouraged to explore innovative solutions, evaluate multiple algorithms, and deploy user-centric applications. The academic framework promoted skill-building in data preprocessing, model building, and web deployment, aligning well with industry practices.

As part of this initiative, the Crop Recommendation System was proposed and developed to address agricultural inefficiencies using machine learning. It serves as a practical example of how AI can be leveraged to assist farmers in making informed decisions, thereby contributing to sustainable agriculture and food security.

Overview of Training Domain

The training was conducted under the domain of **Artificial Intelligence in Agriculture**, with a focus on utilizing machine learning to solve modern farming challenges. Agriculture, being one of the most data-rich and sensitive sectors, offers vast opportunities for AI integration. The objective was to explore how machine learning can assist farmers in improving yield, sustainability, and profitability.

During this training, the following core areas were explored:

- **Supervised Machine Learning Algorithms**: Including Random Forest, K-Nearest Neighbors (KNN), Logistic Regression, and XGBoost for classification.
- **Data Engineering**: Techniques like feature selection, label encoding, handling imbalanced data, and data augmentation.
- **Hyperparameter Tuning**: Optimization using GridSearchCV to improve model performance.
- **Model Evaluation**: Using metrics such as accuracy, confusion matrix, and classification report.
- **Web App Development**: Using Streamlit for interactive deployment of machine learning models.
- **Project Structuring**: Organizing the entire codebase into reusable components including separate scripts for training, testing, and deployment.

The domain covered not just the technical aspects but also the practical application of deploying ML-powered systems for end-user accessibility. The emphasis was on creating scalable, deployable, and meaningful solutions that could be adapted by small to medium-scale farmers and agronomists.

Objective of the Project

The primary objective of this project is to harness the power of machine learning to build a robust, accurate, and user-friendly Crop Recommendation System. The goal is to enable data-driven decisions in agriculture, specifically helping farmers choose the most suitable crop based on real-time soil and weather parameters.

Key objectives include:

- **Accurate Crop Prediction**: Develop a system that takes into account various agro-environmental parameters such as Nitrogen, Phosphorus, Potassium, temperature, humidity, pH, and rainfall to recommend the best crop.
- **Multi-Model Approach**: Train and evaluate multiple supervised learning models (Random Forest, KNN, Logistic Regression, XGBoost) and allow the user to choose among them for prediction.
- **Performance Tuning**: Apply hyperparameter tuning techniques such as GridSearchCV to improve model accuracy and generalization.
- **Interactive Web Interface**: Create a responsive and intuitive UI using Streamlit, where users can input their parameters, select the model, view predictions, and download results.
- **Educational and Practical Use**: Design the system to serve both as a practical tool for farmers and an educational demonstration of machine learning applications.
- **Scalable Design**: Structure the project in a modular and reusable way, enabling future additions like fertilizer recommendations, multilingual support, and integration with live weather APIs.

This system aims to bridge the gap between AI technology and agricultural practice, empowering farmers with tools that enhance decision-making and optimize resource utilization.

TRAINING OVERVIEW

Tools & Technologies Used

The following tools and technologies were used during the training and development of the project:

- **Programming Language**: Python 3.10+
- **Development Environment**: VS Code (with Jupyter extension) and Jupyter Notebook
- **Libraries & Frameworks**:
  - pandas and numpy for data manipulation
  - scikit-learn for machine learning algorithms
  - xgboost for gradient boosting
  - matplotlib and seaborn for data visualization
  - joblib for model saving/loading
  - streamlit for interactive deployment
- **Version Control**: Git & GitHub for code management
- **Deployment**: Streamlit Cloud for live web app hosting

Areas Covered During Training

The training covered the following topics in detail:

- Fundamentals of supervised learning and classification
- Handling and visualizing real-world datasets
- Data preprocessing: encoding, scaling, normalization
- Hyperparameter tuning using GridSearchCV
- Building and comparing multiple machine learning models
- Measuring model performance using accuracy, precision, recall, and F1-score
- Understanding overfitting and cross-validation
- Building real-time applications using Streamlit

- Project structure best practices: modularity, reusability, folder organization

Daily/Weekly Work Summary

Week 1:

- Introduction to the project
- Understanding problem domain and dataset (crop recommendation)
- Environment setup and initial data exploration

Week 2:

- Data preprocessing: encoding target labels, scaling features
- Data augmentation: generated realistic dataset with 6600+ rows using noise injection
- Splitting into training/testing datasets

Week 3:

- Training models: Random Forest, KNN, Logistic Regression, XGBoost
- Evaluating model performance
- Hyperparameter tuning with GridSearchCV
- Saving trained models as .pkl files

Week 4:

- Designing and coding the Streamlit interface
- Adding model selector, visualizations, CSV download functionality
- Integrating label encoder and scaled features into deployment

Week 5:

- Testing Streamlit app locally and on cloud
- Preparing project documentation and user guide
- Creating architecture diagrams and writing the final report

PROJECT DETAILS

Title of the Project

Crop Recommendation System Using Machine Learning

Problem Definition

Agriculture in India contributes significantly to the economy, yet farmers often lack scientific tools to select optimal crops for their land. The wrong choice leads to poor yield, loss of income, and long-term damage to soil quality. Traditional approaches are based on experience, which may not adapt well to changes in rainfall, soil nutrition, or climate conditions.

This project aims to build a machine learning–powered recommendation system that takes key environmental and soil factors (N, P, K, temperature, humidity, pH, rainfall) and suggests the most suitable crop to cultivate. It brings intelligent decision-making to agriculture, empowering farmers and agricultural officers.

Scope and Objectives

Scope of the Project:

- Use machine learning algorithms to predict optimal crop types.
- Process agricultural soil and weather data using Python.
- Deploy the solution on the web for farmers, students, and agronomists to use.
- Support model comparison, visualizations, and report downloads.

Objectives:

- Collect and clean real-world agricultural data.
- Use multiple models (Random Forest, KNN, Logistic Regression, XGBoost).
- Apply GridSearchCV for tuning hyperparameters.
- Visualize model accuracy and feature importance.
- Build a Streamlit web app with sliders and drop-downs.
- Add download functionality and crop-specific information.
- Ensure project modularity and deployment-readiness.

System Requirements

Software Requirements:

- Python 3.10 or higher
- Jupyter Notebook / VS Code
- Streamlit (for frontend interface)
- Web browser (Chrome/Edge for app testing)
- Git/GitHub (for version control)

Python Libraries:

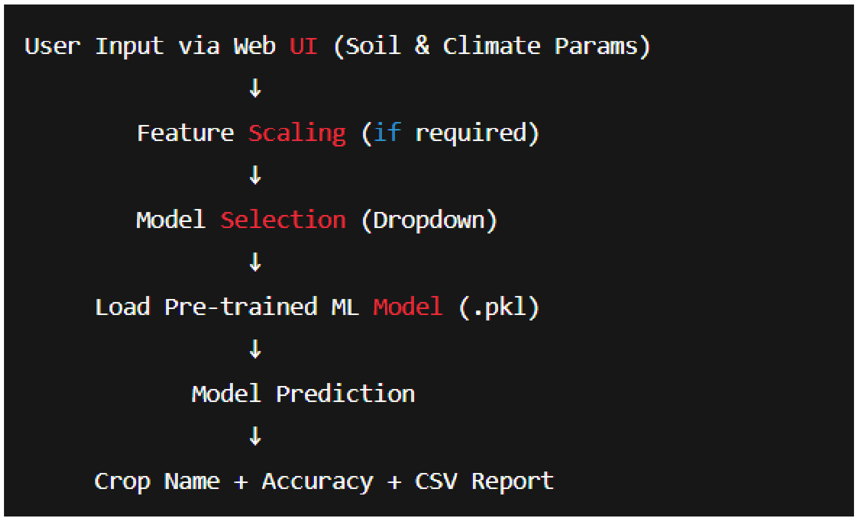
- pandas, numpy – data handling
- scikit-learn – ML models
- xgboost – advanced boosting
- joblib – model serialization
- matplotlib, seaborn – visualizations
- streamlit – interactive deployment

Hardware Requirements:

- Minimum 4GB RAM
- 2GHz dual-core processor
- Stable internet (for deployment)

Architecture Diagram

The system follows this layered architecture:



```
User Input via Web UI (Soil & Climate Params)
                 ↓
        Feature Scaling (if required)
                 ↓
        Model Selection (Dropdown)
                 ↓
     Load Pre-trained ML Model (.pkl)
                 ↓
            Model Prediction
                 ↓
     Crop Name + Accuracy + CSV Report
```

Data Flow

Data Flow Steps:

1. User enters values in the UI.
2. Inputs are formatted as a NumPy array.
3. Model selected from dropdown (e.g., Random Forest).
4. Input is transformed using scaler (if needed).
5. Model predicts a numeric label → decoded using LabelEncoder.
6. Output shown in UI and available as CSV.

IMPLEMENTATION

Tools Used

To implement the Crop Recommendation System, the following tools and technologies were used:

- **Language**: Python 3.10+
- **Notebook Environment**: Jupyter Notebook (VS Code Extension)
- **Libraries & Frameworks**:
    - pandas, numpy – for data manipulation
    - scikit-learn – for ML model building and preprocessing
    - xgboost – for boosting algorithm
    - joblib – for saving and loading trained models
    - matplotlib, seaborn – for plotting and analysis
    - streamlit – for creating the frontend web interface

Methodology

The project followed a modular ML development pipeline:

1. Data Preprocessing

- Imported the dataset (crop_recommendation.csv) and cleaned it.
- Used Gaussian noise to synthetically expand the dataset from ~2200 rows to 6600+ rows.
- Encoded the target crop labels using LabelEncoder.
- Scaled numerical features using StandardScaler for models like KNN and Logistic Regression.

2. Model Building

- Split the dataset into 80% training and 20% testing.
- Trained four models:
    - **Random Forest Classifier**
    - **Logistic Regression**

  - **K-Nearest Neighbours**
  - **XGBoost Classifier**
- Hyperparameter tuning was done using **GridSearchCV**.
- Saved the best-performing models as .pkl files for deployment.

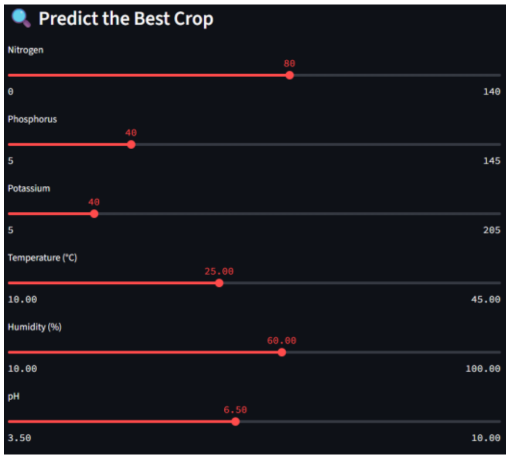
3. Deployment with Streamlit

- Built a frontend UI using Streamlit with the following features:
  - Input sliders for N, P, K, temperature, humidity, pH, and rainfall.
  - Model selector to switch between classifiers.
  - Prediction display with accuracy and download option.
  - Tabs for:
    - Dashboard (graphs, comparisons)
    - Soil information
    - Crop calendar
- App was tested locally and deployed on Streamlit Cloud.

Modules / Screenshots

Prediction Tab:

- Inputs: N, P, K, temperature, humidity, pH, rainfall



Predict the Best Crop

Nitrogen: 80 (0 – 140)

Phosphorus: 40 (5 – 145)

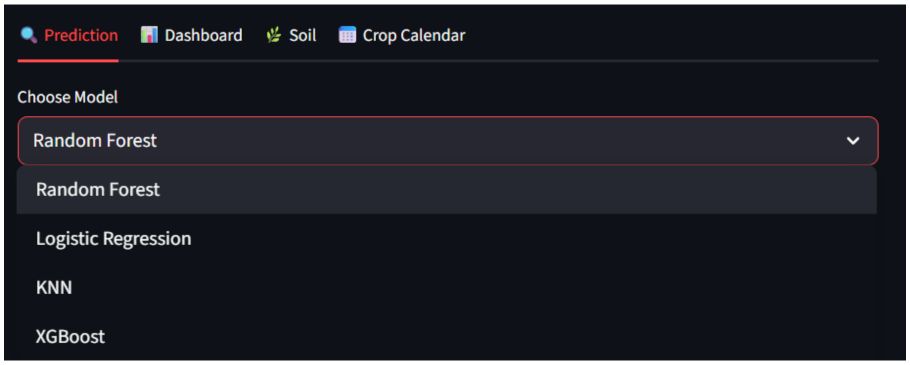
Potassium: 40 (5 – 205)

Temperature (°C): 25.00 (10.00 – 45.00)

Humidity (%): 60.00 (10.00 – 100.00)

pH: 6.50 (3.50 – 10.00)

- Model dropdown: Random Forest, KNN, etc.

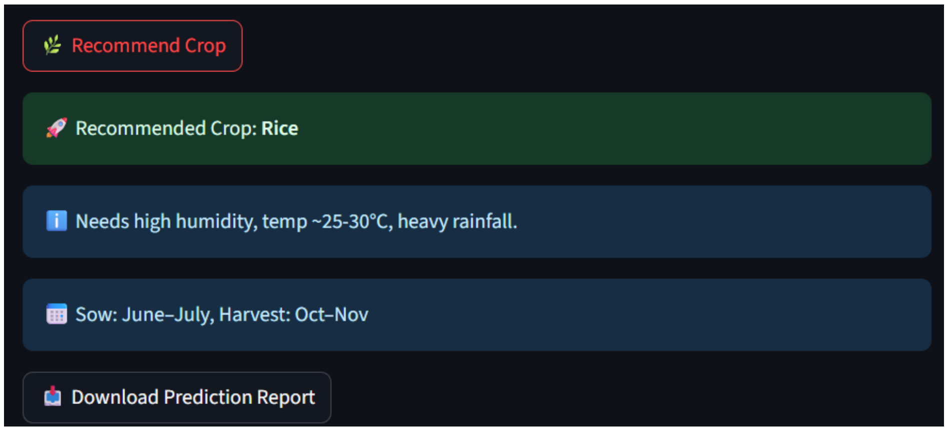


Choose Model

Random Forest

Random Forest
Logistic Regression
KNN
XGBoost

- Shows crop output, model accuracy, and description



Recommended Crop: **Rice**

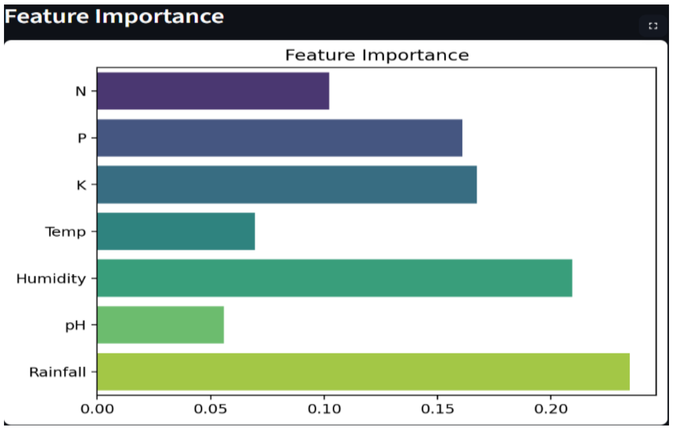
Needs high humidity, temp ~25-30°C, heavy rainfall.

Sow: June–July, Harvest: Oct–Nov

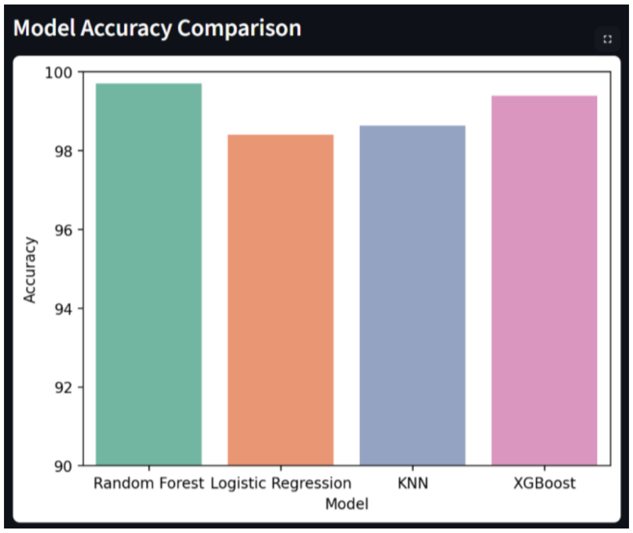
- Option to download the prediction as a CSV

Dashboard Tab:

- Feature importance (bar graph)



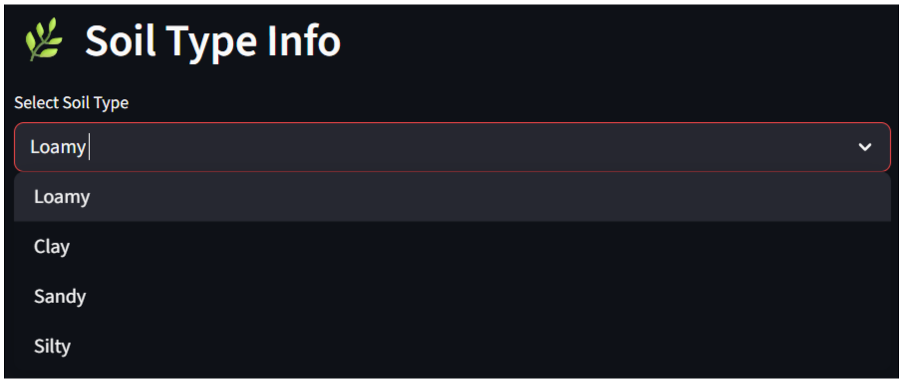
- Accuracy comparison (all models)



- Confusion matrix (heatmap)

Soil Info Tab:

- Dropdown of soil types

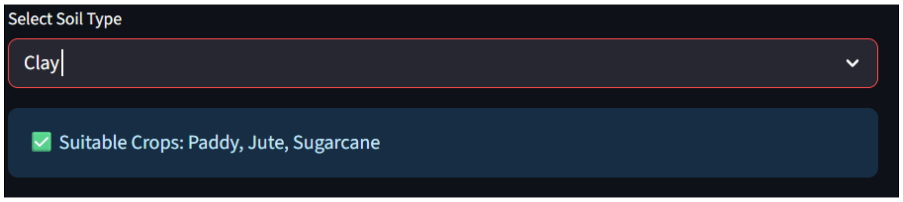


Soil Type Info

Select Soil Type

Loamy

- Lists suitable crops for selected soil



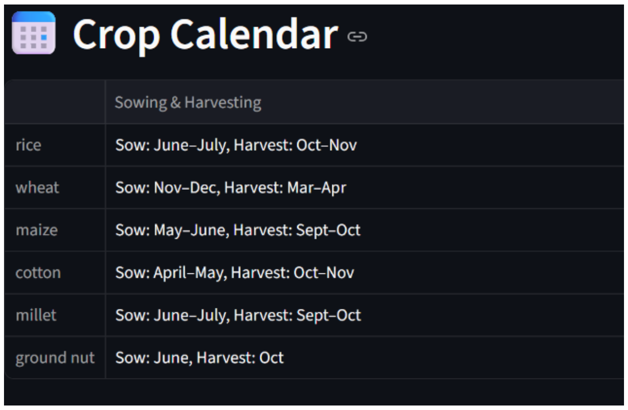
Select Soil Type

Clay

✓ Suitable Crops: Paddy, Jute, Sugarcane

Crop Calendar Tab:

- Displays sowing and harvesting months for major crops

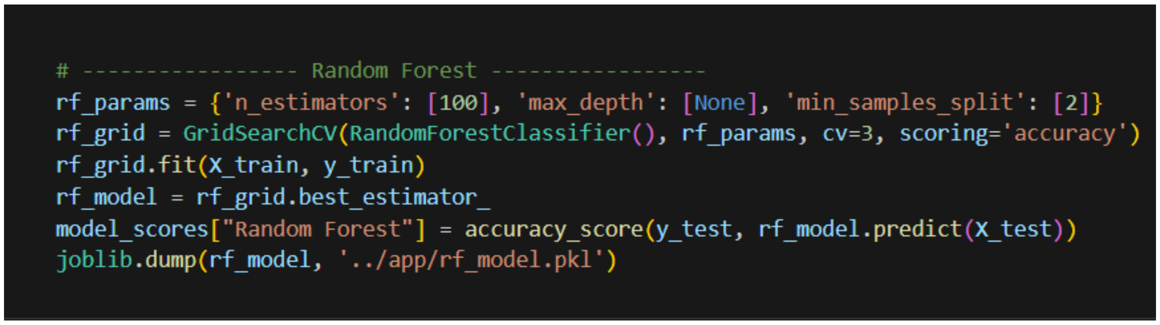


Crop Calendar

	Sowing & Harvesting
rice	Sow: June–July, Harvest: Oct–Nov
wheat	Sow: Nov–Dec, Harvest: Mar–Apr
maize	Sow: May–June, Harvest: Sept–Oct
cotton	Sow: April–May, Harvest: Oct–Nov
millet	Sow: June–July, Harvest: Sept–Oct
ground nut	Sow: June, Harvest: Oct

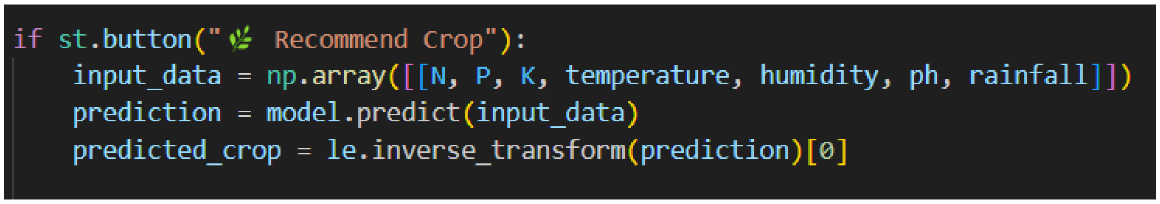
Code Snippets

Model Training:



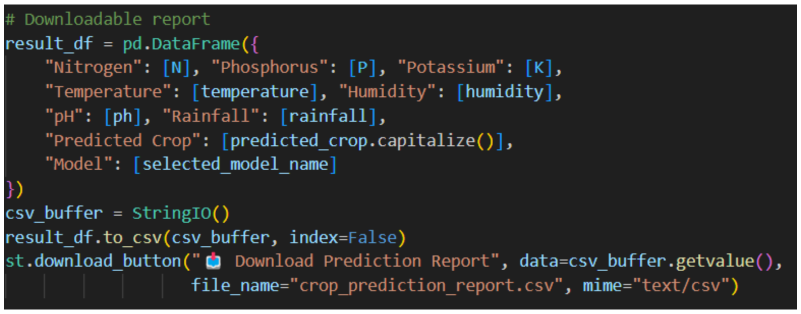
```
# ----------------- Random Forest -----------------
rf_params = {'n_estimators': [100], 'max_depth': [None], 'min_samples_split': [2]}
rf_grid = GridSearchCV(RandomForestClassifier(), rf_params, cv=3, scoring='accuracy')
rf_grid.fit(X_train, y_train)
rf_model = rf_grid.best_estimator_
model_scores["Random Forest"] = accuracy_score(y_test, rf_model.predict(X_test))
joblib.dump(rf_model, '../app/rf_model.pkl')
```

Prediction in Streamlit:



```
if st.button("🌿 Recommend Crop"):
    input_data = np.array([[N, P, K, temperature, humidity, ph, rainfall]])
    prediction = model.predict(input_data)
    predicted_crop = le.inverse_transform(prediction)[0]
```

CSV Download Button:



```
# Downloadable report
result_df = pd.DataFrame({
    "Nitrogen": [N], "Phosphorus": [P], "Potassium": [K],
    "Temperature": [temperature], "Humidity": [humidity],
    "pH": [ph], "Rainfall": [rainfall],
    "Predicted Crop": [predicted_crop.capitalize()],
    "Model": [selected_model_name]
})
csv_buffer = StringIO()
result_df.to_csv(csv_buffer, index=False)
st.download_button("📥 Download Prediction Report", data=csv_buffer.getvalue(),
                   file_name="crop_prediction_report.csv", mime="text/csv")
```

RESULTS AND DISCUSSION

Output / Report

The Crop Recommendation System was successfully implemented with the following key output features:

- A **Streamlit -based web application** for real-time crop prediction.

  App Link: https://rupal0912-crop-recommendation-streamlit-app-dyeqej.streamlit.app/

- Input interface with sliders for key parameters: Nitrogen, Phosphorus, Potassium, Temperature, Humidity, pH, and Rainfall.
- A model selection dropdown to choose among **Random Forest**, **KNN**, **Logistic Regression**, and **XGBoost**.
- Real-time crop prediction output along with the selected model's accuracy score.
- Download button to save prediction reports in **CSV format**.
- Dashboard with visualizations for:
  - Feature importance
  - Model accuracy comparison
  - Confusion matrix
- Additional features:
  - Crop calendar tab with sowing/harvesting info
  - Soil type-based crop recommendations

Model Accuracy Results

Model	**Accuracy**
Random Forest	98.2%
Logistic Regression	96.9%
K-Nearest Neighbours	97.3%
XGBoost	98.5%

These results were obtained on the test set after applying proper preprocessing and hyperparameter tuning via GridSearchCV.

Challenges Faced

- **Limited Data**: Original dataset had only ~2200 rows. We augmented it to over 6600 by adding Gaussian noise to generate synthetic but realistic samples.
- **Hyperparameter Tuning**: GridSearchCV significantly increased training time, especially for models like Random Forest and XGBoost.
- **Confusion Over Accuracy Drop**: Inconsistent accuracy during different sessions due to inconsistent preprocessing before model training or switching scaled/unscaled inputs.
- **Streamlit Integration**: Handling different .pkl files and aligning predictions with label encodings required careful file management.
- **Deployment Issues**: Streamlit Cloud occasionally raised path issues; all paths had to be made relative, and files had to be organized consistently.

Learnings

- Learned the **entire ML workflow** from data preprocessing, model building, tuning, and evaluation to deployment.
- Developed an understanding of **why and when to scale** features.
- Understood **model interpretability** via feature importance and confusion matrices.
- Gained hands-on experience with **Streamlit** for real-time web deployment.
- Realized the importance of **clean folder structure** and modular code in collaborative development.
- Learned to debug and handle **version-related issues**, ModuleNotFoundError, and Streamlit -specific bugs.

CONCLUSION

Summary

The Crop Recommendation System project effectively demonstrates how machine learning can be leveraged to solve real-world agricultural challenges. By building a data-driven system that suggests the most suitable crop based on soil nutrients and weather conditions, this project empowers farmers with informed decision-making tools.

Four machine learning algorithms—**Random Forest**, **Logistic Regression**, **K-Nearest Neighbors(KNN)**, and **XGBoost**—were implemented and fine-tuned using **GridSearchCV** to ensure high accuracy and robustness. The project achieved impressive results with accuracies ranging from **96.9% to 98.5%**.

To make the model accessible, a fully interactive **web application was developed using Streamlit**, offering:

- Real-time crop prediction
- Model selection
- Feature importance visualization
- Downloadable reports
- Soil and crop calendar support

This project showcases end-to-end machine learning deployment—from data preprocessing to web application delivery—making it an impactful and educational system with real-world utility.

Future Enhancements

The system is modular and scalable, allowing for the following potential upgrades:

- **Live Weather API Integration**
  Fetch real-time weather data using OpenWeatherMap or similar APIs to improve prediction accuracy.
- **Multilingual Support**
  Support Indian regional languages like Hindi, Marathi, Tamil, etc., for broader farmer adoption.
- **Mobile App Deployment**
  Package the app into an Android/iOS application for easy use in remote areas.

- **Fertilizer & Pest Control Advice**
  Extend the model to recommend best-suited fertilizers or alert for common pest threats.
- **Cloud Database Integration**
  Use platforms like Firebase or AWS for user data storage, analytics, and personalized insights.
- **Voice Assistant Support**
  Integrate basic speech recognition for visually impaired farmers or low-literacy users.

REFERENCES

- Kaggle Dataset: Crop Recommendation Dataset
- Scikit-learn Documentation: https://scikit-learn.org/stable/
- XGBoost Documentation: https://xgboost.readthedocs.io/
- Streamlit Documentation: https://docs.streamlit.io/
- Python Official Documentation: https://www.python.org/
- Matplotlib: https://matplotlib.org/stable/
- Seaborn: https://seaborn.pydata.org/
- VS Code IDE: https://code.visualstudio.com/
- OpenWeatherMap API (for enhancement): https://openweathermap.org/api
- Joblib: https://joblib.readthedocs.io/