SUMMER TRAINING

PROJECT REPORT

(Term June-July 2025)

(CROP RECOMMENDATION SYSTEM)

Submitted by

Name: Aayush Sharma
Registration Number: 12317299

Name: Aditya Choudhary
Registration Number: 12321604

Name: Nitesh Kumar
Registration Number: 12323263

Name: Himanshu Yadav
Registration Number: 12322606

Course Code : PETV79

Under the Guidance of

Mahipal Singh Papola

School of Computer Science and Engineering

CONTENTS OF THE REPORT

BONAFIDE CERTIFICATE

Certified that this project report “**CROP RECOMMENDATION SYSTEM**” is the bona fide work of “**Aayush Sharma, Aditya Choudhary, Nitesh Kumar, Himanshu Yadav**” who carried out the project work under my supervision.

SIGNATURE
<<Name of the Supervisor>>

Aayush Sharma

Aditya Choudhary

Nitesh Kumar

Himanshu Yadav

SIGNATURE

<<Signature of the Head of the Department>>
SIGNATURE
<<Name>>
HEAD OF THE DEPARTMENT

<<Signature of the Supervisor>>

1. INTRODUCTION

1.1 OVERVIEW OF TRAINING DOMAIN:

The training focused on applying Python and ML to a real-world problem. It started with an overview of basic concepts like supervised and unsupervised learning. During the training, participants learned how to build test and deploy models leveraging Pandas, Scikit-learn, and Streamlit.

As part of their capstone project, students were motivated to build complete machine learning applications. The training emphasized the following:

- Data cleaning and preprocessing
- Exploratory Data Analysis (EDA)
- Feature selection
- Model training and testing
- Evaluation metrics like accuracy, precision, recall, F1-score
- Building user-friendly web interfaces with Streamlit

We selected this domain because machine learning is increasingly being utilized in industries such as agriculture, healthcare, and finance for informed decision-making.

1.2 OBJECTIVE OF THE PROJECT:

The primary goal of this project is to create a smart Crop Recommendation System that assists farmers or agricultural officials in selecting the best crop based on specific soil and environmental conditions. The system utilizes a machine learning model developed using a publicly accessible dataset that includes parameters such as:

- Nitrogen (N)
- Phosphorus (P)
- Potassium (K)
- Heat level
- Moisture
- Precipitation

By inputting these parameters into an easy-to-use web interface, the system forecasts the optimal crop to grow, striving to enhance yield and boost farming efficiency.

The main objectives of the project are:

- To implement machine learning methods in a practical agricultural context.
- To develop and enhance a classification model for maximum accuracy.

- To launch a web application with Streamlit that is straightforward, engaging, and accessible to non-technical users.
- To showcase how AI can improve conventional farming methods and aid in the development of smart agriculture.

This initiative not only demonstrates the use of machine learning but also encourages sustainable farming by advising suitable crop choices according to soil and climate factors.

2. TRAINING AND OVERVIEW

2.1 TOOLS AND TECHNOLOGIES USED:

Throughout the training, we examined multiple tools, technologies, and libraries crucial for Machine Learning and deploying models. These resources assisted us in comprehending the complete process of creating a data-focused application from the ground up. The main technologies employed in the project consist of:

- Python: The main programming language utilized during the training for data handling and model creation.
- Pandas and NumPy: For handling data and performing mathematical calculations.
- Matplotlib and Seaborn: For visualizing data and conducting EDA (Exploratory Data Analysis).
- Scikit-learn: For training and assessing machine learning models like Random Forest, Decision Tree, and Logistic Regression.
- Streamlit: A minimalistic Python framework utilized for developing an interactive web app for the crop recommendation model.
- Jupyter Notebook / VS Code: To develop and validate Python code in a structured and clear manner.
- PIL (Python Imaging Library): Used to open, resize, and display images of the predicted crops inside the application in a standardized format (360x360 pixels).
- OS Module: Used to check and access the file paths of image assets and ensure compatibility during runtime.
- JSON Module: Utilized to read crop-related additional information such as ideal climate and tips from a structured crop_info.json file

2.2 AREAS COVERED DURING TRAINING:

The training was both thorough and industry-related. It began at the basics and gradually proceeded to practical development of the project. Some of the key areas are the following:

- Machine learning basic introduction:
  Learning about ML types: supervised, unsupervised and reinforcement learning.
- Data Preprocessing:
  Missing value treatment, encoding categorical data, normalization and train-test split.
- Model Building:
  Application of algorithms (e.g. Random Forest, Decision Tree), optimization of hyper parameters, and metrics of evaluation model.
- Streamlit deployment:
  Generating an easy-to-use UI with Streamlit to be able to expose the model to end-users.
- Project Planning & Documentation:
  The end-to-end pipeline design, including testing on the real data, writing the technical document of the project.

2.3 DAILY/WEEKLY WORK SUMMARY:

Table 1: Weekly Work Summary

WEEK 1	Introduction to Python for ML, working with Pandas, NumPy, and understanding ML lifecycle
WEEK 2	Introduction to scikit library and Supervised learning techniques, data preprocessing, train-test split
WEEK 3	Model building using Random Forest, evaluation using accuracy & confusion matrix
WEEK 4	Exploring other models like decision tree and application of models
WEEK 5	Testing, bug fixing, adding the UI and deploying using streamlit using VScode
Final Days	Preparing the project report, documentation, dataset organization, final deployment

3. PROJECT DETAILS

3.1 TITLE OF THE PROJECT:

“Machine-learning based Smart Crop Recommendation System”

This project will be to come up with an intelligent crop suggestion tool that should guide a farmer or even an agricultural officer to select the best crop depending on the soil and environmental variables through the application of machine learning.

3.2 PROJECT DETAILS:

The choice of crop to be grown in modern agriculture is very important to rejuvenate productivity and sustainability of the crops by using climatic and soil conditions as the basis of the choice. Most farmers normally depend on traditional knowledge or guesswork that usually results in poor crop output, depleted soil or even farm incurring a loss.

A modern need is the presence of a smart data-driven technology that virtually helps a farmer take an insightful crop decision based on quantifiable measurements such as the amount of nitrogen, phosphorus, potassium availability, temperature, humidity, pH, and the amount of rainfall.

The given project addresses it by training a machine learning model to suggest the best crop that should be planted in a given set of conditions and deploying it through a user-friendly web interface.

3.3 SCOPE AND OBJECTIVES:

Scope

The project will aim at assisting:

- Agricultural professional and farmers make sound decisions.
- Facilitate (promote) sustainable and data driven agriculture.
- Be an educative means of learning in terms of actual ML implementation.

Objectives

- To construct real agricultural data machine learning classification model.
- In order to forecast the choice of the most effective crop regarding the soil and weather characteristics.
- To make an interactive response based web application in crop prediction.
- To offer visible information, crops, and advice to the users on the basis of the forecast.

3.4 SYSTEM REQUIREMENTS:

Software Requirements

- Python 3.9 or above
- Streamlit
- Pandas
- Scikit-learn
- Pillow (PIL)
- JSON & OS modules
- streamlit-toggle-switch (for theme toggle)
- Web Browser (for UI)

Hardware Requirements

- Minimum 4 GB RAM
- A modern CPU (i3 or higher)
- Internet connection (for deployment or image loading if needed)

3.5 ARCHITECTURE DESIGN:

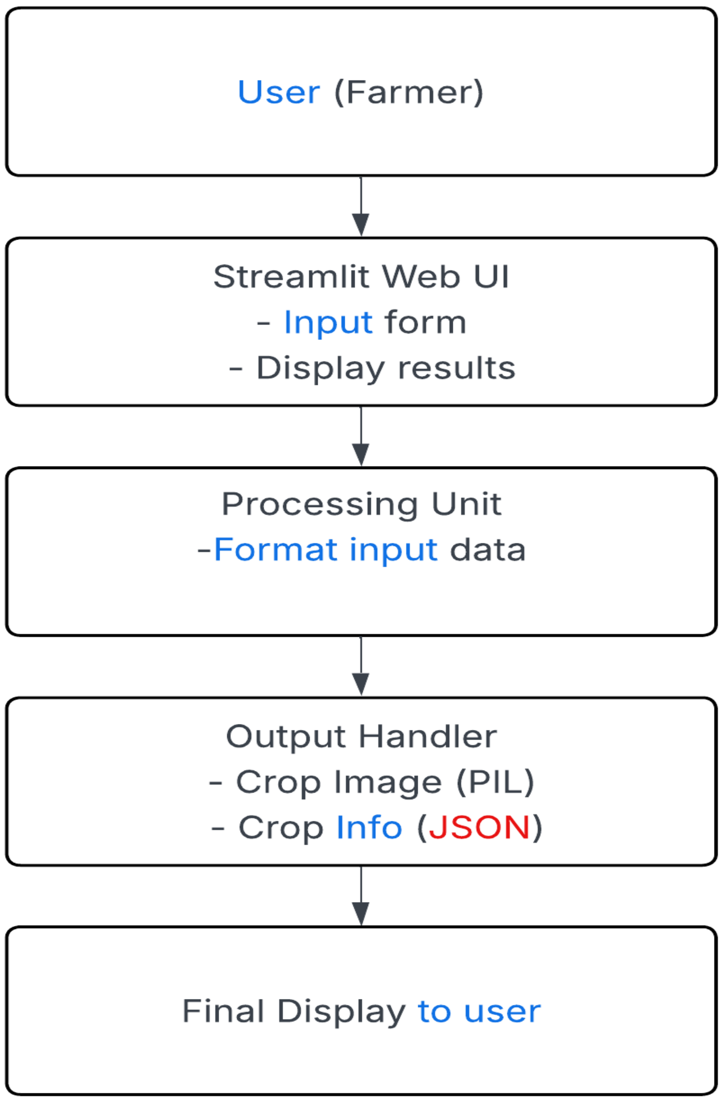


Figure 1: Architecture Design

3.6 DATA FLOW/UML DIAGRAM:

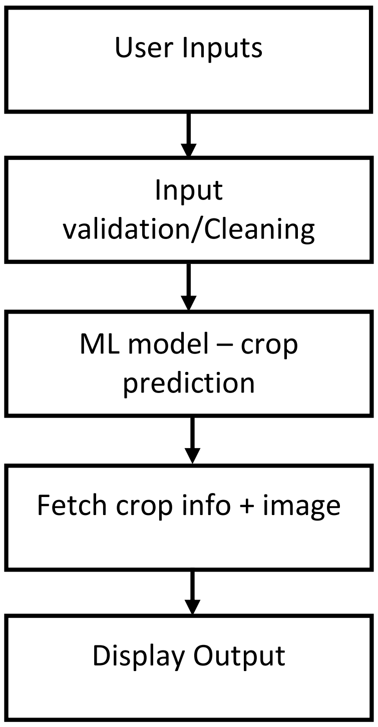
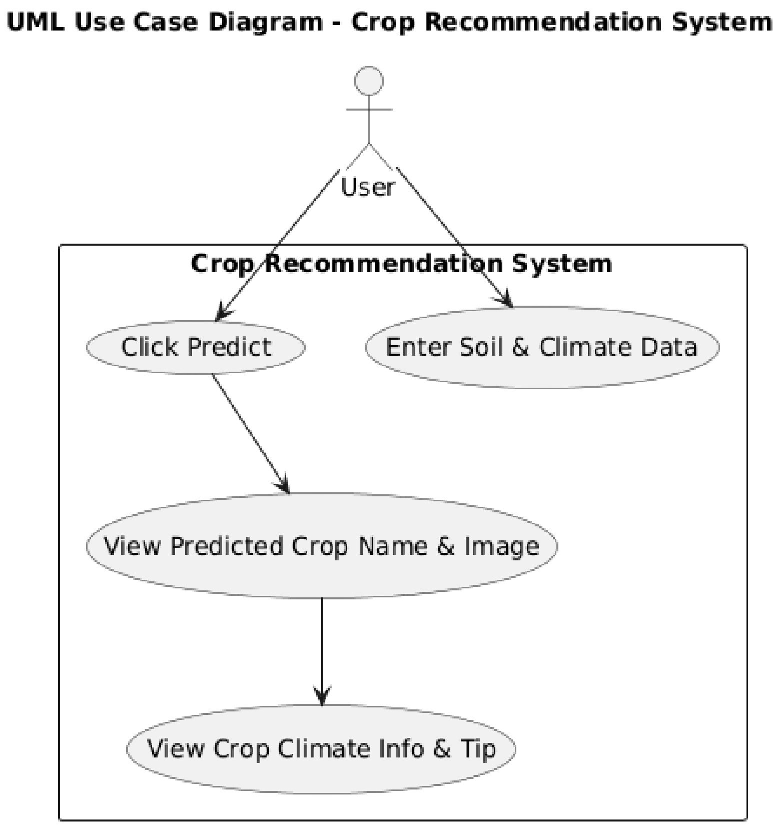


Figure 2: Data Flow Diagram



UML Use Case Diagram - Crop Recommendation System
User
Crop Recommendation System
Click Predict
Enter Soil & Climate Data
View Predicted Crop Name & Image
View Crop Climate Info & Tip

Figure 3: UML Diagram

4. IMPLEMENTATION

4.1 TOOLS USED:

Table 2: Tools Used

Tool/Library	**Purpose**
Python	Core language for coding the ML model and app
Pandas	Data preprocessing and loading the dataset
Scikit-learn	ML model building, training, evaluation
Pickle	Saving and loading the trained model
Streamlit	Building the user interface for the app
PIL(image/library)	Loading and resizing crop images
OS and JSON	File handling and dynamic crop info display

4.2 METHODOLOGY:

1. Data Collection & Exploration:

- Dataset: Crop_recommendation.csv from Kaggle:

  https://www.kaggle.com/datasets/atharvaingle/crop-recommendation-dataset

- Contains features: N, P, K, temperature, humidity, pH, rainfall
- Target variable: Recommended crop

2. Data Preprocessing:

- Checked for missing values and outliers
- Normalization not needed as tree-based models are scale-invariant

3. Train-Test Split:

- Split the dataset into 80% training and 20% testing using train_test_split.

4. Model Selection & Training:

- Tried multiple classifiers: Random Forest, Decision Tree, Logistic Regression
- Final Model: RandomForestClassifier (best accuracy)

5. Model Evaluation:

- Evaluated using accuracy, confusion matrix, and manual testing
- Final accuracy achieved: ~99.3%

6. Model Saving:

- Saved the trained model using pickle for reuse in the app.

7. Web Application with Streamlit:

- Built an interactive UI using Streamlit
- Users input soil/weather values → model predicts crop → UI displays crop name, image, climate, and tips

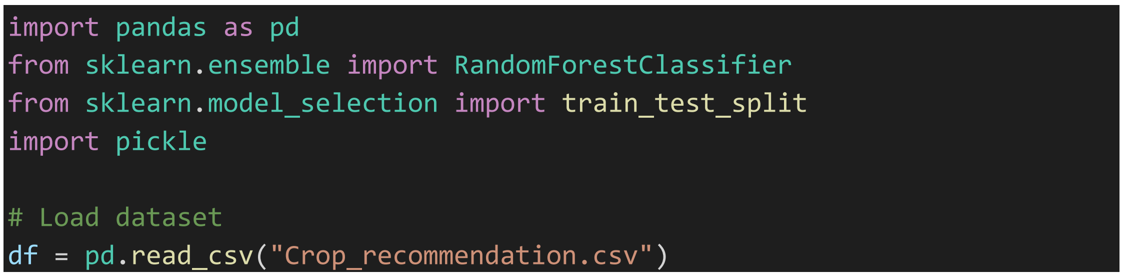
8. UI Enhancements:

- Dark/Light mode toggle
- Centered crop image and sections
- Resized crop images (360x360)
- Used .json file to show crop tips and ideal climate info

4.3 MODULES / SCREENSHOTS:

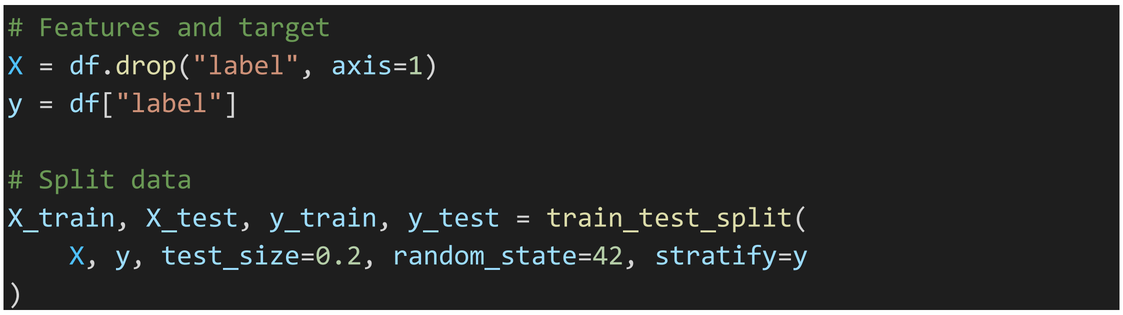
Module 1: ML Model Training (train_model.py):

- Reads the CSV file
- Trains RandomForestClassifier
- Saves model as .pkl



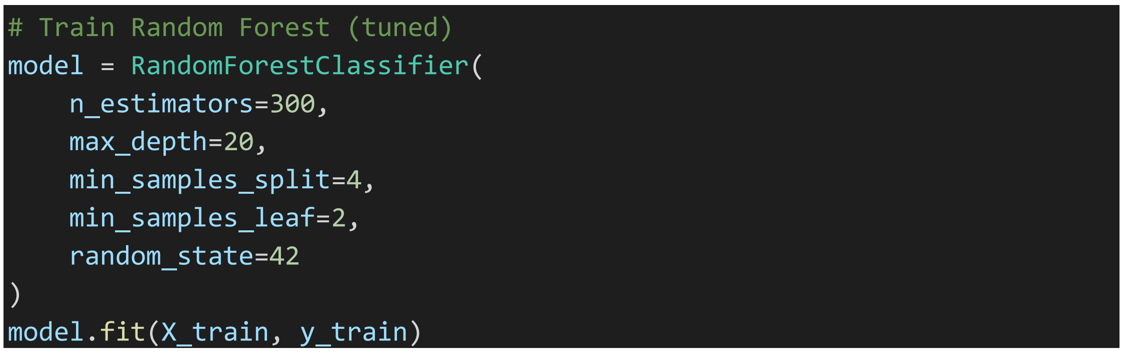
```python
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import pickle

# Load dataset
df = pd.read_csv("Crop_recommendation.csv")
```

Figure 4: Loading Dataset



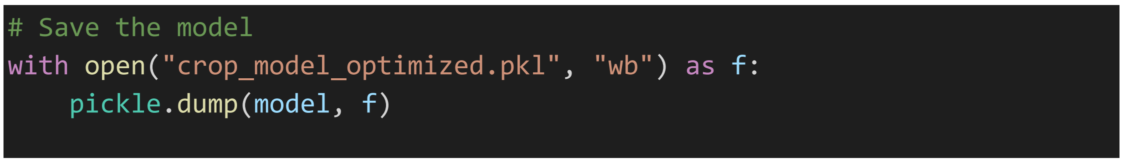
```python
# Features and target
X = df.drop("label", axis=1)
y = df["label"]

# Split data
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)
```

Figure 5: Train_test split



```python
# Train Random Forest (tuned)
model = RandomForestClassifier(
    n_estimators=300,
    max_depth=20,
    min_samples_split=4,
    min_samples_leaf=2,
    random_state=42
)
model.fit(X_train, y_train)
```

Figure 6: Training on Random Forest



```python
# Save the model
with open("crop_model_optimized.pkl", "wb") as f:
    pickle.dump(model, f)
```

Figure 7: Saving the model

Module 2: Web App UI (app.py):

- Uses Streamlit to collect inputs
- Loads model and predicts crop
- Displays image, tip, and info

```python
# Input Fields
with left_col:
    st.subheader("📥 Input Parameters")
    N = st.number_input("Nitrogen content (N)", 0.0, 150.0, step=1.0)
    P = st.number_input("Phosphorus content (P)", 0.0, 150.0, step=1.0)
    K = st.number_input("Potassium content (K)", 0.0, 200.0, step=1.0)
    temperature = st.number_input("Temperature (°C)", 0.0, 50.0,
step=0.1)
    humidity = st.number_input("Humidity (%)", 0.0, 100.0, step=0.1)
    ph = st.number_input("pH value", 0.0, 14.0, step=0.1)
    rainfall = st.number_input("Rainfall (mm)", 0.0, 300.0, step=1.0)

    if st.button("🔍 Predict Crop"):
        input_data = [[N, P, K, temperature, humidity, ph, rainfall]]
        prediction = model.predict(input_data)[0]

        # Save result in session
        st.session_state.prediction = prediction
```

Figure 8: Collecting Input

```python
# Load model
with open("crop_model_optimized.pkl", "rb") as file:
    model = pickle.load(file)
```

Figure 9: Loading Dataset

```python
if "prediction" in st.session_state:
    prediction = st.session_state.prediction
    with right_col:
        st.subheader("🌟 Recommended Crop")
        st.success(f"**{prediction.title()}**")

        # Crop Image Card
        image_path = f"images/{prediction}.png"
        if os.path.exists(image_path):
            img = Image.open(image_path)
            img = img.resize((360, 360))
            st.markdown("<div style='text-align: center;'>",
unsafe_allow_html=True)
            st.image(img, caption=prediction.title(),
use_container_width=False)
            st.markdown("</div>", unsafe_allow_html=True)
        else:
            st.info("📷 Crop image not available.")

        # Extra info if available
        if prediction in crop_info:
            st.markdown("### 🌾 Crop Details")
            st.markdown(f"**🌦️ Climate:**
{crop_info[prediction]['climate']}")
            st.markdown(f"**📌 Tip:** {crop_info[prediction]['tip']}")
        else:
            st.info("ℹ️ No additional info available for this crop.")
```

Figure 10: Prediction-image-info

Module 3: Crop Info Storage (crop_info.json)

- Stores crop-wise climate and tip
- Accessed dynamically after prediction

```
{
  "rice": {
    "climate": "Hot and humid with standing water",
    "tip": "Grows well in clay soils and high rainfall areas"
  },
  "maize": {
    "climate": "Warm with moderate rainfall",
    "tip": "Prefers well-drained fertile soils"
  },
  "chickpea": {
    "climate": "Cool and dry climate",
    "tip": "Best grown in sandy loam soil with low moisture"
  },
  "kidneybeans": {
    "climate": "Mild warm climate",
    "tip": "Requires well-drained loamy soil and moderate watering"
  }
```

Figure 11: Snippet of info

Module 4: Image Display

- Image loaded from /images/ folder based on predicted crop
- Uses PIL to resize and show images with center alignment

```
image_path = f"images/{prediction}.png"
        if os.path.exists(image_path):
            img = Image.open(image_path)
            img = img.resize((360, 360))
```

Figure 12: Loading and crop image

4.4 CODE SNIPPETS:

Code to predict crop:

```
input_data = [[N, P, K, temperature, humidity, ph, rainfall]]
prediction = model.predict(input_data)[0]
st.success(f"Recommended Crop: {prediction}")
```

Code to load crop image:

```
img = Image.open(f"images/{prediction}.png")
img = img.resize((360, 360))
st.image(img, caption=prediction.title())
```

Code to display crop info:

```
if prediction in crop_info:
    st.markdown(f"**Climate:** {crop_info[prediction]['climate']}")
    st.markdown(f"**Tip:** {crop_info[prediction]['tip']}")
```

5. RESULT

5.1 OUTPUT:

The ultimate deliverable of the work are formulated to be a working Crop Recommendation System in Box (Web-based) that takes different soil, environmental aspects like:

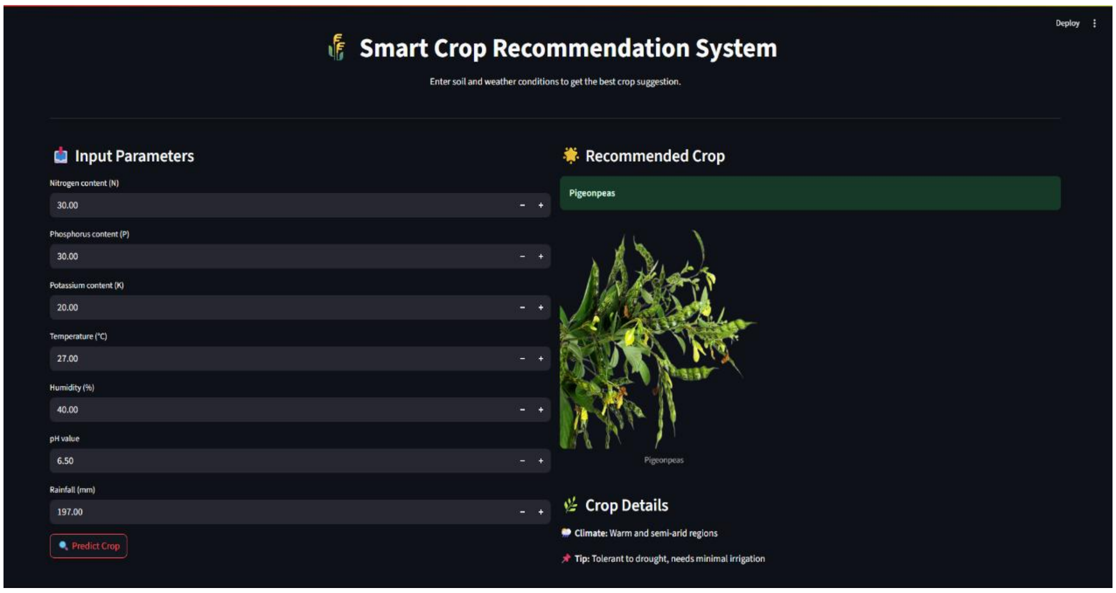
- Phosphorus (P): Nitrogen (N), Potassium (K)
- Temperature, humidity, pH, Rainfall

When entering those numbers the system will tell you what crop is the best choice with the help of trained machine-based model (Random Forest Classifier) of very high accuracy at about 99.3%.

User interface:

- Well, free interactive and simple on Streamlit
- Has dynamic display of crop images
- Displays climatic suggestions and agricultural advice

Farmers, agricultural consultants or students will be free to test and see what the correct crop to take is within the specified soil, weather conditions



Smart Crop Recommendation System

Enter soil and weather conditions to get the best crop suggestion.

Input Parameters

Nitrogen content (N): 30.00

Phosphorus content (P): 30.00

Potassium content (K): 20.00

Temperature (°C): 27.00

Humidity (%): 40.00

pH value: 6.50

Rainfall (mm): 197.00

Recommended Crop

Pigeonpeas

Crop Details

Climate: Warm and semi-arid regions

Tip: Tolerant to drought, needs minimal irrigation

Figure 13: Input and Output

5.2 CHALLENGES FACED:

Learning about Datasets and Data Cleaning

- The scholar was using the tabular data containing floating numbers with precision, which cast doubt on the accuracy of the real user entries (e.g. rainfall:102.76 mm but typed by the user 103).
- It was solved by changing the step sizes in input fields and the possibility of a range in general use.

Selection and Accuracy of Model

- In the first step, several algorithms were tried: Logistic Regression, Decision Tree, etc.
- Random Forest was the last choice because it proved to be best in terms of tabular, imbalanced data.

Integration and UI of Web Apps

- There were 22 crops and coordinating the images and tips, together with crop information was handled with caution.
- Consistency in the size of images (360x360) without deformation was achieved with the help of PIL and Streamlit hacks.

5.3 LEARNINGS:

The given project allowed practically experiencing many aspects of both Machine Learning and web-deployment:

Full Pipeline in ML

- Linear interpolation to save and revisit the model through pickle, model training, preprocessing, and data loading

Model Evaluation

- Discovered how to evaluate the performance of a model based on the accuracy score and confusion matrix as well as inputs on the real world test instances

Development of Streamlit App

- They acquired experience in constructing a web application without frontend code (HTML/CSS)
- Handling of input and output, layout management, images inclusion and on-the-fly renderings.

JSON & Files Handling

- Stored auxiliary files such as tips and climate info in used .json files
- Crammed and synchronized with forecasts.

Partner Work and Reporting

- Understood the documentation of the system in an orderly manner and according to project submission specifications.

6. CONCLUSION

6.1 SUMMARY:

The Crop Recommendation System designed in the project has managed to show how machine learning can be used to address real-life agricultural issues. The system can also learn regularities between nutrients that are present in the soil and the climatic conditions to be able to propose the most appropriate crop that will be grown.

This project was a mixture of:

- Model training and prediction through Data Science
- Implementation in python programming
- Streamlit in construction of a user-friendly web interface

In the final product, the user is able to enter values such as Nitrogen (N), Phosphorus (P), Potassium (K), temperature, humidity, pH and rainfall and the user will get a precise forecast and the image of the crop and climatic demands and the expert advices on farming. The model had a higher rate of accuracy as compared to 99 %, which makes it reliable in application of the model.

During the project, we came across key theory topics relating to data preprocessing, model selection, file management and web deployment. The lessons of this project have reinforced our knowledge of the whole machine learning process - data to deployment.

Although this system is a learning project, it can be possible to apply it in real life to promote smart farming and sustainable agriculture in India and other regions.