Green University of Bangladesh

Department of Computer Science and Engineering (CSE)
Semester: (fall, Year: 2025), B.Sc. in CSE (Day)

Smart Fertilizer Recommendation Using Machine Learning

Course Title: Artificial Intelligence Lab
Course Code: CSE 316
Section: 223D5

Students Details

Name	ID
Md. Yousuf Ibna Alam Joy	221002481

Submission Date: 28 DEC 2025
Course Teacher's Name: Abdullah Al Farhad

[For teachers use only: Don't write anything inside this box]

Lab Project Status	
Marks:	**Signature:**
Comments:	**Date:**

Contents

Chapter 1

Introduction

1.1 Overview

This project aims to create a smart fertilizer recommendation system using machine learning. The system uses machine learning models to analyze soil and weather data to suggest the optimal fertilizer type and amount. The core idea is to help farmers improve crop yields, reduce fertilizer waste, and promote sustainable soil health. The project involves collecting and pre-processing data, training machine learning models (like Random Forest and ANN), and evaluating their performance.

1.2 Motivation

The motivation behind this project is to address the challenges farmers face in determining the correct fertilizer for their crops. Incorrect fertilizer use can lead to low crop yields, soil degradation, and increased costs. By using AI, this project aims to provide farmers with optimized fertilizer recommendations, reduce fertilizer waste, and promote sustainable soil health.

1.3 Problem Definition

1.3.1 Problem Statement

Farmers face significant challenges in selecting the appropriate fertilizer for their crops. Incorrect fertilizer selection can lead to low crop yield due to nutrient imbalance, soil degradation from excessive chemical use, and increased expenses for farmers. There is a need for AI-driven systems that can accurately predict fertilizer requirements by analyzing soil and climate factors.

1.3.2 Complex Engineering Problem

The table 1.1 that summarizes the attributes touched by the Smart Fertilizer Recommendation Using Machine Learning based on the previous discussions:

Name of the P Attributess	Explain how to address
P1: Depth of knowledge required	ML algorithms, data preprocessing, agricultural science, Python.
P2: Range of conflicting requirements	Balancing accuracy with usability and cost-effectiveness.
P3: Depth of analysis required	Analyzing soil/weather data, evaluating ML model performance.
P4: Familiarity of issues	Some novelty in specific data combinations requiring experimentation.
P5: Extent of applicable codes	Using and adapting existing ML libraries in Python
P6: Extent of stakeholder involvement and conflicting requirements	Farmers need usability, reliability, and cost-effectiveness.
P7: Interdependence	Recommendation accuracy depends on data quality and model performance.

Table 1.1: Summary of the attributes touched by the mentioned projects

1.4 Design Goals/Objectives

The primary objectives of this project are:

- To develop an AI model that predicts the best fertilizer for various soil conditions.
- To combine machine learning models to improve prediction accuracy.
- To develop a user-friendly application (if time permits).

1.5 Application

The application for this project is envisioned as a user-friendly interface to deliver fertilizer recommendations to farmers. Here's a breakdown of the application concept:

- Purpose: To provide farmers with an easy way to get fertilizer recommendations based on their specific conditions.
- Target Users: Primarily farmers.
- Medium: It is considered to develop it as a web or mobile application.

- Functionality: The application will allow users to interactively receive fertilizer recommendations.

- Development Status: The development of the application is marked as a future goal, dependent on the completion of the model optimization.

Chapter 2

Design/Development/Implementation of the Project

2.1 Introduction

Fertilizers are essential for enhancing crop growth, but farmers often struggle with determining the correct type and amount to use, which can harm the crops and the soil. This project addresses this issue by developing an AI-based model to recommend the optimal fertilizer based on soil and weather data. The system uses machine learning techniques to provide precise and cost-effective fertilizer recommendations. [1] [2] [3] [4].

2.2 Project Details

This section outlines the technical implementation of the Smart Fertilizer Recommendation Using Machine Learning, covering architecture, modules, tools, and algorithms used during the design and development phases.

2.2.1 Project view

This project focuses on the development of an AI-based system for smart fertilizer recommendation. The core objective is to provide farmers with accurate and efficient fertilizer recommendations by leveraging machine learning techniques. The system utilizes a hybrid model, integrating Random Forest and Artificial Neural Networks (ANN), to analyze soil data (pH, NPK levels), weather data (temperature, humidity, rainfall), and crop information. The project encompasses data collection and pre-processing, model training and optimization, and performance evaluation. The intended outcome is to create a tool that helps farmers make informed decisions about fertilizer use, leading to increased crop yield, reduced environmental impact, and cost savings. Future development includes refining the model with XGBoost and creating a user-friendly application.

2.2.2 Key Features

- **Data Pre-processing:**
  - Label encoding of target (fertilizer).
  - One-hot encoding of categorical features (soil and crop types)
- **Model Implementation:**
  - Trained and evaluated Random Forest and ANN models.
  - Comparison of performance using accuracy score, classification report, and confusion matrix
- **Visualization:**
  - Graphs for fertilizer distribution
  - Confusion matrices using Matplotlib and Seaborn

2.2.3 Tools and Technologies used:

- Programming Language: Python
- Libraries Used: scikit-learn, TensorFlow/Keras, Pandas, NumPy, Matplotlib, Seaborn
- IDE/Platform: Google Colab/Jupyter Notebook / VS Code

- Dataset: $\mathsf{fertilizer}_{r}ecommendation_{d}ataset.csv(3,\ 100entries)$

2.2.4 Development:

- Model Enhancement: Train and evaluate an XGBoost model for improved performance.
- App Development: Create a simple web or mobile application to deploy the best-performing model and make recommendations accessible to end-users, especially farmers.

2.3 Literature Review

Several studies have explored machine learning-based fertilizer recommendation systems. However, most approaches use a single model rather than a hybrid approach. The table below summarizes key findings from previous research.

Author(s)	Model Used	Findings	Limitations
Sharma et al. (2021) [**?**]	Random Forest	High accuracy in soil classification	Limited to soil nutrient levels only
Kumar et al. (2022) [**?**]	SVM	Good for soil fertility classification	Computationally expensive for large datasets
Gupta et al. (2023) [**?**]	ANN	Predicts fertilizer needs based on weather and soil	Lacks optimization for real-time use
Singh et al. (2021) [**?**]	Decision Tree + Deep Learning	Outperforms single ML models	High computational cost
Proposed Model	Hybrid (Random Forest + ANN)	Combines classification + prediction for better results	

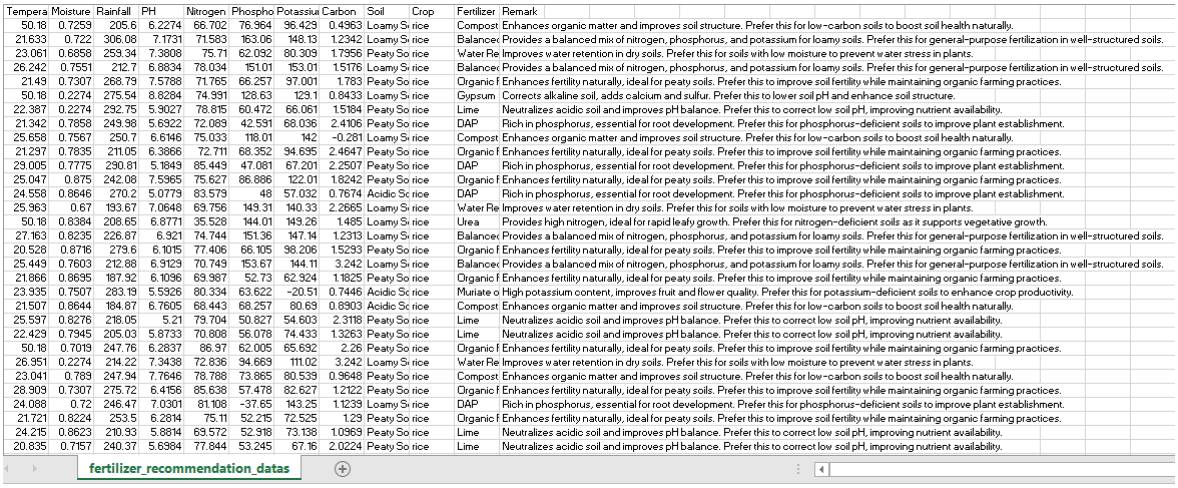
Table 2.1: Literature Review

2.4 Implementation

The implementation of this project centered on developing a machine learning-based system to provide smart fertilizer recommendations. Initially, a dataset containing soil properties and weather conditions was gathered and prepared for analysis, which involved cleaning, transforming, and encoding the data using Python libraries. Subsequently, two machine learning models, Random Forest and Artificial Neural Network (ANN), were constructed and trained to predict fertilizer requirements. The models' performance was rigorously evaluated using appropriate metrics, and the results were visualized to gain insights into their effectiveness.

2.4.1 Data Collection and Pre-processing:

- The dataset was collected from sources like Kaggle and agricultural research databases.
- The dataset includes features like soil pH, NPK levels (Nitrogen, Phosphorus, Potassium), temperature, humidity, and rainfall.
- Data was preprocessed using Python libraries such as Pandas, NumPy, and Scikit-learn.
- Pre-processing involved removing inconsistencies, handling missing values, and normalizing feature values.
- Categorical features (Soil, Crop) were one-hot encoded.
- The target variable (Fertilizer) was label encoded.



Tempera	Moisture	Rainfall	PH	Nitrogen	Phospho	Potassiui	Carbon	Soil	Crop	Fertilizer	Remark
50.18	0.7259	205.6	6.2274	66.702	76.964	96.429	0.4963	Loamy Si	rice	Compost	Enhances organic matter and improves soil structure. Prefer this for low-carbon soils to boost soil health naturally.
21.633	0.722	306.08	7.1731	71.583	163.06	148.13	1.2342	Loamy Si	rice	Balanced	Provides a balanced mix of nitrogen, phosphorus, and potassium for loamy soils. Prefer this for general-purpose fertilization in well-structured soils.
23.061	0.6858	259.34	7.3808	75.71	62.092	80.309	1.7956	Peaty So	rice	Water Re	Improves water retention in dry soils. Prefer this for soils with low moisture to prevent water stress in plants.
26.242	0.7551	212.7	6.8834	78.034	151.01	153.01	1.5176	Loamy Si	rice	Balanced	Provides a balanced mix of nitrogen, phosphorus, and potassium for loamy soils. Prefer this for general-purpose fertilization in well-structured soils.
21.49	0.7307	268.79	7.5788	71.765	66.257	97.001	1.783	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
50.18	0.2274	275.54	8.8284	74.991	128.63	129.1	0.8433	Loamy Si	rice	Gypsum	Corrects alkaline soil, adds calcium and sulfur. Prefer this to lower soil pH and enhance soil structure.
22.387	0.2274	292.75	5.9027	78.815	60.472	66.061	1.5184	Peaty So	rice	Lime	Neutralizes acidic soil and improves pH balance. Prefer this to correct low soil pH, improving nutrient availability.
21.342	0.7858	249.98	5.6922	72.089	42.591	68.036	2.4106	Peaty So	rice	DAP	Rich in phosphorus, essential for root development. Prefer this for phosphorus-deficient soils to improve plant establishment.
25.658	0.7567	250.7	6.6146	75.033	118.01	142	-0.281	Loamy Si	rice	Compost	Enhances organic matter and improves soil structure. Prefer this for low-carbon soils to boost soil health naturally.
21.297	0.7835	211.05	6.3866	72.711	68.352	94.695	2.4647	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
29.005	0.7775	290.81	5.1849	85.449	47.081	67.201	2.2507	Peaty So	rice	DAP	Rich in phosphorus, essential for root development. Prefer this for phosphorus-deficient soils to improve plant establishment.
25.047	0.875	242.08	7.5965	75.627	86.886	122.01	1.8242	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
24.558	0.8646	270.2	5.0779	83.579	48	57.032	0.7674	Acidic Sc	rice	DAP	Rich in phosphorus, essential for root development. Prefer this for phosphorus-deficient soils to improve plant establishment.
25.963	0.67	193.67	7.0648	69.756	149.31	140.33	2.2665	Loamy Si	rice	Water Re	Improves water retention in dry soils. Prefer this for soils with low moisture to prevent water stress in plants.
50.18	0.8384	208.65	6.8771	35.528	144.01	149.26	1.485	Loamy Si	rice	Urea	Provides high nitrogen, ideal for rapid leafy growth. Prefer this for nitrogen-deficient soils as it supports vegetative growth.
27.163	0.8235	226.87	6.921	74.744	151.36	147.14	1.2313	Loamy Si	rice	Balanced	Provides a balanced mix of nitrogen, phosphorus, and potassium for loamy soils. Prefer this for general-purpose fertilization in well-structured soils.
20.528	0.8716	279.6	6.1015	77.406	66.105	98.206	1.5293	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
25.449	0.7603	212.88	6.9129	70.749	153.67	144.11	3.242	Loamy Si	rice	Balanced	Provides a balanced mix of nitrogen, phosphorus, and potassium for loamy soils. Prefer this for general-purpose fertilization in well-structured soils.
21.866	0.8695	187.92	6.1096	69.987	52.73	62.924	1.1825	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
23.935	0.7507	283.19	5.5926	80.334	63.622	-20.51	0.7446	Acidic Sc	rice	Muriate o	High potassium content, improves fruit and flower quality. Prefer this for potassium-deficient soils to enhance crop productivity.
21.507	0.8644	184.87	6.7605	68.443	68.257	80.69	0.8903	Acidic Sc	rice	Compost	Enhances organic matter and improves soil structure. Prefer this for low-carbon soils to boost soil health naturally.
25.597	0.8276	218.05	5.21	79.704	50.827	54.603	2.3118	Peaty So	rice	Lime	Neutralizes acidic soil and improves pH balance. Prefer this to correct low soil pH, improving nutrient availability.
22.429	0.7945	205.03	5.8733	70.808	56.078	74.433	1.3263	Peaty So	rice	Lime	Neutralizes acidic soil and improves pH balance. Prefer this to correct low soil pH, improving nutrient availability.
50.18	0.7019	247.76	6.2837	86.97	62.005	65.692	2.26	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
26.951	0.2274	214.22	7.3438	72.836	94.669	111.02	3.242	Loamy Si	rice	Water Re	Improves water retention in dry soils. Prefer this for soils with low moisture to prevent water stress in plants.
23.041	0.789	247.34	7.7646	78.788	73.865	80.539	0.9648	Peaty So	rice	Compost	Enhances organic matter and improves soil structure. Prefer this for low-carbon soils to boost soil health naturally.
28.909	0.7307	275.72	6.4156	85.638	57.478	82.627	1.2122	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
24.088	0.72	246.47	7.0301	81.108	-37.65	143.25	1.1239	Loamy Si	rice	DAP	Rich in phosphorus, essential for root development. Prefer this for phosphorus-deficient soils to improve plant establishment.
21.721	0.8224	253.5	6.2814	75.11	52.215	72.525	1.29	Peaty So	rice	Organic f	Enhances fertility naturally, ideal for peaty soils. Prefer this to improve soil fertility while maintaining organic farming practices.
24.215	0.8623	210.93	5.8814	69.572	52.918	73.138	1.0969	Peaty So	rice	Lime	Neutralizes acidic soil and improves pH balance. Prefer this to correct low soil pH, improving nutrient availability.
20.835	0.7157	240.37	5.6984	77.844	53.245	67.16	2.0224	Peaty So	rice	Lime	Neutralizes acidic soil and improves pH balance. Prefer this to correct low soil pH, improving nutrient availability.

fertilizer_recommendation_datas

Figure 2.1: Dataset of our project that is used

2.4.2 Machine Learning Model Implementation

- A hybrid AI approach was used, combining two machine learning models.
- **Random Forest Classifier:**
  - Used to categorize fertilizer requirements into three levels: High, Medium, and Low.
- **Artificial Neural Network (ANN):**
  - Used to predict the exact amount of fertilizer needed (in kg/acre) based on soil and environmental parameters.
- **Model Optimization:**
  - Hyperparameter tuning was performed using Grid Search and Bayesian Optimization.
  - Feature selection was considered to improve computational efficiency and model performance.
  - The performance of models was compared.
  - XGBoost model was trained and compared with Random Forest and ANN models.
- **Evaluation**
  - Model performance was evaluated using accuracy score, classification report, and confusion matrix.
- **Visualization:**
  - Fertilizer distribution and confusion matrices were visualized using Matplotlib and Seaborn.

2.4.3 The workflow of our project

Here we show all the flow charts of our project. Step by step, we show our working procedure.

Now we show step-by-step features and it's working in flowchart :

2.4.4 Tools and libraries

The successful implementation of the Fertilizer Recommendation System relied on a combination of development environments, programming tools, and machine learning libraries. These resources were carefully selected to ensure high performance, ease of use, and strong community support.

- **Programming Language**
  - Python 3.x: Chosen for its simplicity, extensive libraries, and strong ecosystem in data science and machine learning.

- **Development Platforms**

Google Colab:

- Used for training machine learning and deep learning models.
- Provided free GPU support for faster computation.
- Cloud-based environment for quick experimentation and visualization.

- **Visual Studio Code (VS Code):**
  - Used for writing, debugging, and organizing code.
  - Helpful for local development, dataset handling, and documentation.

- **Libraries and Frameworks**

- Pandas: For loading, cleaning, and analyzing structured data.
- NumPy: For numerical computations and array operations.
  - **scikit-learn:**
    - Used for training the Random Forest and XGBoost models.
    - Provided tools for preprocessing (e.g., label encoding, train-test split).
  - XGBoost: For high-performance gradient boosting model training.

TensorFlow / Keras:

- Used to build and train the Artificial Neural Network (ANN).

- Provided flexibility in designing deep learning architectures.

  Matplotlib and Seaborn:

- Used to create data visualizations including fertilizer distributions, confusion matrices, and performance comparisons.

- **Other Tools**

  – CSV File: The dataset fertilizer*recommendationdataset.csvwasusedasinput.*

– Jupyter Notebook: Used for model testing and documentation inside Google Colab.

– GitHub (optional): For version control and project collaboration (if applicable).

2.4.5 Implementation details (with screenshots and programming codes)

- **Here include some of python code in our project:**

```
from flask import Flask, render_template, request
import pandas as pd
import joblib

app = Flask(__name__)


model = joblib.load("xgb_model.pkl")
scaler = joblib.load("scaler.pkl")
le = joblib.load("label_encoder.pkl")
model_columns = joblib.load("model_columns.pkl")




@app.route("/", methods=["GET", "POST"])
def index():
    prediction = None

    if request.method == "POST":
        try:
            # Get form data
            temp = float(request.form["temp"])
            moist = float(request.form["moist"])
            rain = float(request.form["rain"])
            ph = float(request.form["ph"])
            n = float(request.form["n"])
            p = float(request.form["p"])
```

```
            k = float(request.form["k"])
            c = float(request.form["c"])
            soil = request.form["soil"]
            crop = request.form["crop"]

            # Prepare input data
            data = {col: 0 for col in model_columns}
            data.update({
                "Temperature": temp,
                "Moisture": moist,
                "Rainfall": rain,
                "PH": ph,
                "Nitrogen": n,
                "Phosphorous": p,
                "Potassium": k,
                "Carbon": c,
                f"Soil_{soil}": 1,
                f"Crop_{crop}": 1,
            })

            df = pd.DataFrame([data])
            scaled_input = scaler.transform(df)
            pred = model.predict(scaled_input)
            prediction = le.inverse_transform(pred)[0]

        except Exception as e:
            prediction = f"Error: {e}"

@app.route("/about")
def about():
    return render_template("about.html")

@app.route("/services")
def services():
    return render_template("services.html")

@app.route("/contact")
def contact():
    return render_template("contact.html")

@app.route("/resources")
def resources():
    return render_template("resources.html")

@app.route("/fermar_portal")
def fermar_portal():
    return render_template("fermar_portal.html")
```

```
if __name__ == "__main__":
    app.run(debug=True)
```

2.5 User Interface

The user interface (UI) is a critical component of the Fertilizer Recommendation System, designed to make the model accessible and usable for farmers and agricultural stakeholders with varying levels of technical expertise. While the current system is implemented in a development environment, plans for an interactive UI are outlined below.

- Home Page
- Navigation Bar
- Fertilizer from
- Success story

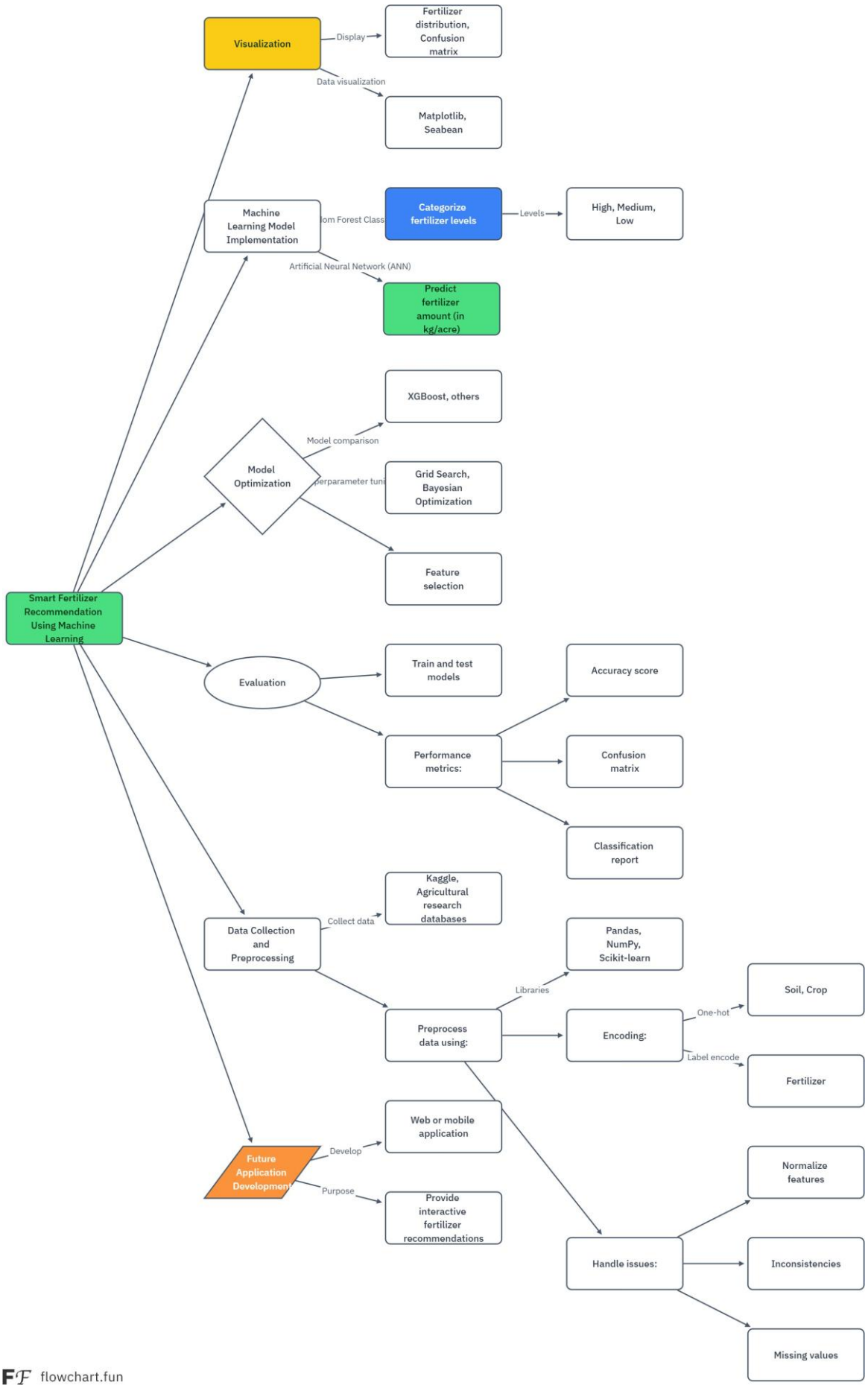


Figure 2.2: over all flowchart

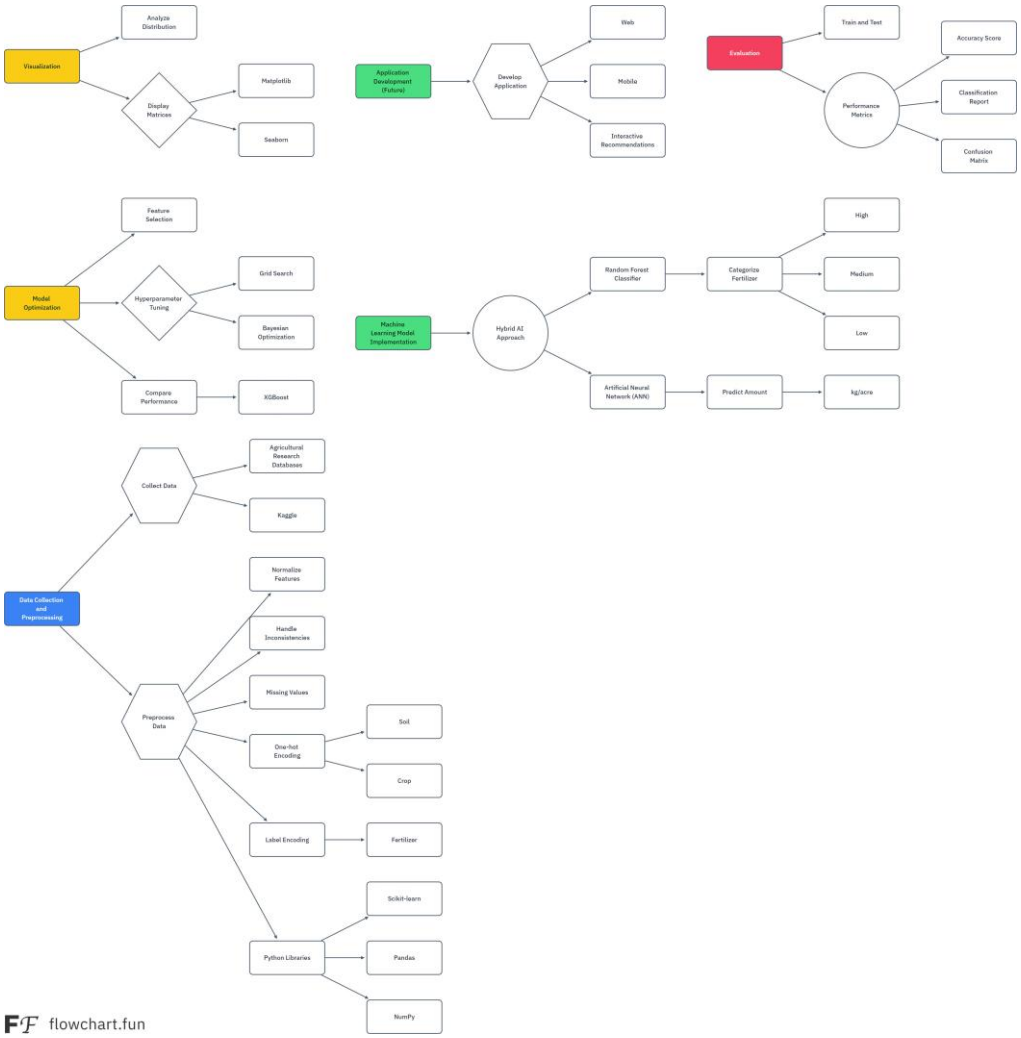


Figure 2.3: Step by step features flowchart in our project

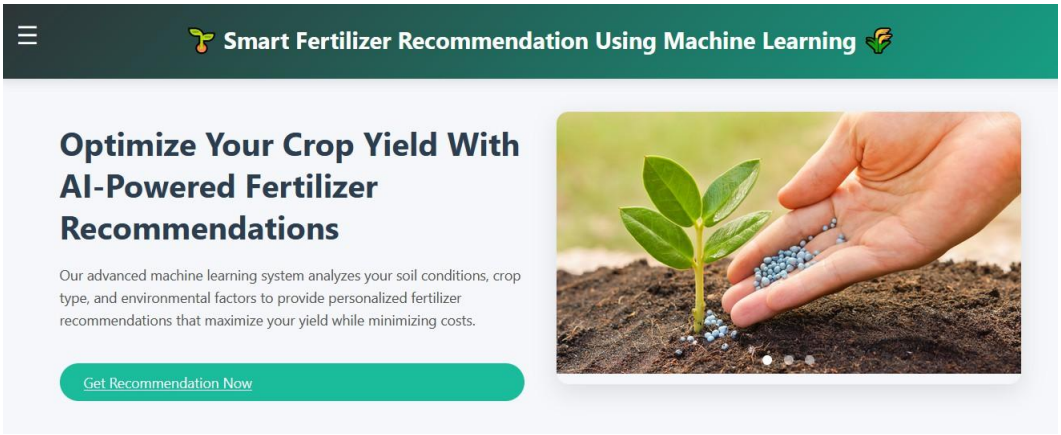


Figure 2.4: Home page in our project

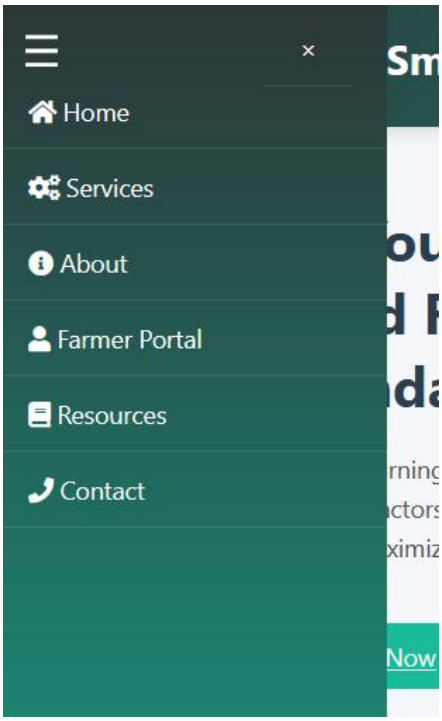


Figure 2.5: Navigation Bar of our project

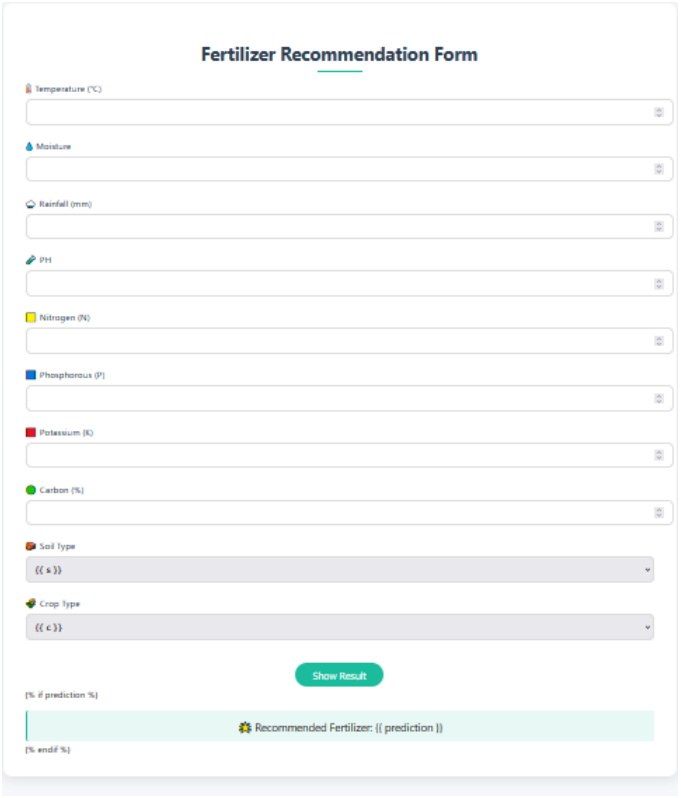


Figure 2.6: Fertilizer From of our project

Figure 2.7: Success story of our project

Chapter 3

Performance Evaluation

3.1 Simulation Environment/ Simulation Procedure

The simulation environment and procedure for this project primarily revolve around the computational setup for developing and evaluating the machine learning models. Here's a breakdown:

3.1.1 Simulation Environment

- **Development Environment:** Google Colab is specified as the primary development environment. Google Colab provides a cloud-based platform that offers access to computing resources, including GPUs, which are essential for training machine learning models, especially ANNs. This environment simplifies the setup process, as it comes with many necessary libraries pre-installed and allows for easy sharing and collaboration.
- **Install Python:** If you haven't already, download and install Python. Make sure to add Python to your system PATH during installation.
- **Install Visual Studio Code (VS Code):** Download and install VS Code from the official website. VS Code is a lightweight but powerful source code editor that supports a variety of programming languages, including Python.
- **Programming Language and Libraries:**The core programming language used is Python. Several Python libraries form the foundation of the simulation environment:
  - **Scikit-learn:**This library provides various machine learning algorithms and tools for tasks like data pre-processing, model selection, and evaluation.
  - **TensorFlow and Keras:**These libraries are used for building and training the Artificial Neural Network (ANN) models. TensorFlow is a powerful deep learning framework, and Keras provides a high-level API for simplifying neural network development.

  - **Pandas and NumPy:**These libraries are essential for data manipulation and numerical computations, respectively, and are used for data pre-processing and analysis.
  - **Matplotlib and Seaborn:**These libraries are used for data visualization, allowing the project team to analyze data distributions and model performance through charts and graphs.
  - nltk or spaCy for natural language processing tasks.

- **Install a web server:**If you plan to deploy your sentiment analysis model as a web application using HTML, you'll need a web server. You can use Flask for this purpose.

3.1.2 Simulation Procedure:

The simulation procedure outlines the steps taken to develop and evaluate the fertilizer recommendation system:

- **Data Collection and Pre-processing:**
  - The dataset is collected from publicly available sources like Kaggle and agricultural research databases.
  - The dataset includes features like soil pH, NPK levels, temperature, humidity, and rainfall.
  - Data preprocessing is performed using Python libraries (Pandas, NumPy, Scikit-learn) to handle missing values, remove inconsistencies, and normalize the data.
  - Categorical features (Soil, Crop) are one-hot encoded, and the target variable (Fertilizer) is label encoded.

- **Machine Learning Model Implementation:**
  - A hybrid AI approach is used, involving two machine learning models:
    * **Random Forest Classifier:** This model is used to categorize fertilizer requirements into levels (High, Medium, Low).
    * **Artificial Neural Network (ANN):** This model is used to predict the specific amount of fertilizer needed.
  - The models are trained using the pre-processed dataset within the Google Colab environment.

- **Model Optimization:**

- Hyperparameter tuning is performed using techniques like Grid Search and Bayesian Optimization to improve model performance.
- Feature selection may be employed to identify the most relevant features and improve computational efficiency.

- In future work, the performance of an XGBoost model is compared with the existing Random Forest and ANN models.

- **Evaluation:**

- The trained models are evaluated using appropriate metrics, such as accuracy score, classification report, and confusion matrix.
- Visualization techniques (Matplotlib, Seaborn) are used to analyze the results, including fertilizer distribution and confusion matrices.

In summary, the "simulation" in this context refers to the computational experiments conducted within the Google Colab environment to develop, train, optimize, and evaluate the machine learning models for fertilizer recommendation.

3.2 Results Analysis/Testing

The "Results Analysis/Testing" chapter would primarily focus on the evaluation of the machine learning models. Here's a detailed discussion of what that would entail:

3.2.1 Result and evaluation

The results of any specific part of your project can be included using subsections.

1. **Evaluation Metrics:**The core of the results analysis would be a detailed presentation and interpretation of the metrics used to evaluate the performance of the machine learning models. The documents explicitly mention the following:
    - **Accuracy Score:** This metric measures the overall correctness of the model's predictions, indicating the proportion of correctly classified instances.
    - **Classification Report:** This report provides a more granular view of the model's performance, including precision, recall, and F1-score for each class (i.e., each fertilizer type or recommendation category).
        - **Precision** measures the proportion of correctly predicted positive instances out of all instances predicted as positive.
        - **Recall** measures the proportion of correctly predicted positive instances out of all actual positive instances.
        - The **F1-score** is the harmonic mean of precision and recall, providing a balanced measure of the model's accuracy.
    - **Confusion Matrix:** This matrix visually represents the model's predictions, showing the number of true positives, true negatives, false positives, and false negatives. It helps to identify which classes the model is confusing with each other.

3.2.2 Model Comparison

A significant aspect of the results analysis is the comparison of the performance of different machine learning models

- The project implements and evaluates the Random Forest and Artificial Neural Network (ANN) models.
- Future work involves training and evaluating an XGBoost model and comparing its performance with the other two.
- The analysis would involve comparing the models based on the evaluation metrics mentioned above to determine which model performs best for the fertilizer recommendation task.

3.2.3 Visualization of Results

The documents highlight the use of visualization techniques to aid in results analysis:

- Fertilizer Distribution: Visualizing the distribution of different fertilizer types in the dataset can provide insights into the class balance and potential challenges for the models.
- Confusion Matrices: Visualizing the confusion matrices allows for a more intuitive understanding of the models' prediction patterns and errors.
- Libraries like Matplotlib and Seaborn are used for these visualizations.

3.2.4 Discussion of Findings

The results analysis chapter would include a detailed discussion of the findings, interpreting the evaluation metrics and visualizations. This discussion would cover:

- Which model performed the best and why.
- The strengths and weaknesses of each model.
- The types of errors made by the models and their potential implications.
- The factors that might be influencing the model's performance (e.g., data quality, class imbalance).
- The significance of the results in the context of the project's goals and objectives.

3.2.5 Statistical Significance (Potentially)

Depending on the rigor of the analysis, the chapter might also include discussions of statistical significance. If the models are evaluated multiple times (e.g., using cross-validation), statistical tests could be used to determine whether the differences in performance between the models are statistically significant or simply due to random variation.

3.2.6 graphical result

Include screenshots from your project.

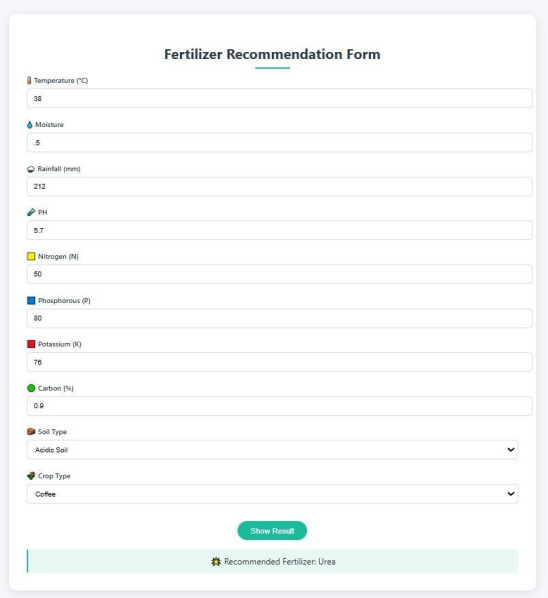


Figure 3.1: A graphical result of your project

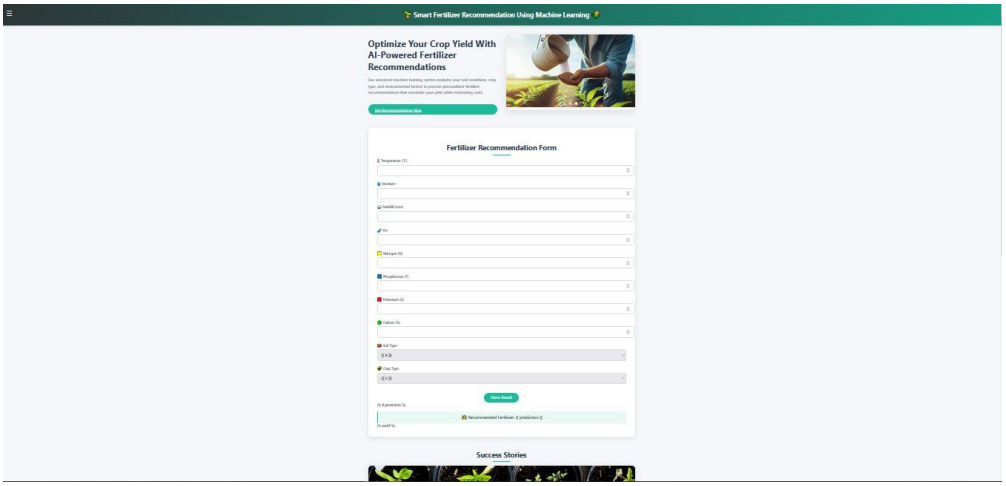


Figure 3.2: A graphical over all result of your project

3.2.7 Result portion 3

In essence, the "Results Analysis/Testing" chapter is where the project team demonstrates the effectiveness (or lack thereof) of their approach, provides evidence to support their conclusions, and offers insights into the behavior of the machine learning models.

3.3 Results Overall Discussion

The project successfully developed a machine learning-based system for smart fertilizer recommendation. The implementation involved training and evaluating machine learning models, specifically Random Forest and Artificial Neural Network (ANN), on a dataset of 3,100 records. The models' performance was assessed using metrics such as accuracy score, classification report, and confusion matrix. The results highlight the potential of machine learning to provide data-driven fertilizer recommendations, which can contribute to improved crop yield and soil health. Future work aims to further optimize the models by incorporating XGBoost and develop a user-friendly application to make this technology accessible to farmers.

Chapter 4

Conclusion

4.1 Discussion

This project addresses the critical issue of optimizing fertilizer use in agriculture by developing a smart fertilizer recommendation system using machine learning. The core motivation is to assist farmers in making informed decisions, as incorrect fertilizer application can lead to reduced crop yields, soil degradation, and increased costs. The proposed system employs a hybrid AI approach, combining the strengths of Random Forest and Artificial Neural Network (ANN) models, to analyze soil properties and weather conditions and provide accurate fertilizer recommendations. The implementation involves data collection and pre-processing, model training and optimization, and rigorous evaluation of model performance. The project leverages Python and various machine learning libraries within the Google Colab environment. The expected outcome is a tool that can enhance agricultural productivity, promote sustainable practices, and reduce costs for farmers, with future work focused on refining the model and developing a user-friendly application for real-world deployment.

4.2 Limitations

some limitations of the "Smart Fertilizer Recommendation Using Machine Learning" our project:

- Data Dependency: The accuracy of the system heavily relies on the quality and representativeness of the dataset. If the dataset is limited or biased, the model's performance may be affected.
- Generalization: The model's ability to generalize to new, unseen data, especially from different geographical locations or environmental conditions, needs further validation.
- Lack of Real-world Testing: The documents mention the need for "a system tested in real-life conditions across different districts," indicating that the current implementation may not have been extensively tested in practical agricultural settings.

- UI Development: The development of a user-friendly web or mobile application is stated as "if time permits" or future work, suggesting that a fully functional and accessible user interface is not yet a part of the current project.
- Model Optimization: While model optimization is discussed, the documents also identify it as an area for future work, specifically mentioning the training and comparison of an XGBoost model. This implies that the current models may not be fully optimized.
- Scope of Input Features: The dataset includes key features like soil pH, NPK levels, temperature, humidity, and rainfall. However, there might be other relevant factors (e.g., soil texture, organic matter content, specific crop varieties) that could further improve the accuracy of fertilizer recommendations but are not included in the current scope.

4.3 Scope of Future Work

The current implementation of the Smart Fertilizer Recommendation System using Machine Learning demonstrates promising results in predicting suitable fertilizers based on crop and soil data. However, there are several areas for improvement and expansion to make the system more accurate, accessible, and practical for real-world agricultural use.

- **Integration with Real-Time Data**
  - Incorporate real-time weather data, such as rainfall, temperature, and humidity, which greatly influence fertilizer effectiveness.
  - Use IoT-based soil sensors for live monitoring of soil pH, moisture, and nutrient content to make recommendations more dynamic and accurate
- **Larger and More Diverse Datasets**
  - Expand the system to support a wider variety of crops and regional soil types, especially those relevant to specific countries or agricultural zones.
  - Collaborate with agricultural organizations to collect localized and updated datasets.
- **Advanced Model Optimization**
  - Explore ensemble models or hybrid learning approaches that combine multiple algorithms to improve prediction performance.
  - Implement hyperparameter tuning using GridSearch or AutoML for better model accuracy and efficiency.
- **Mobile and Web Application Development**
  - Develop a full-featured mobile and web app to make the system easily accessible to farmers, agronomists, and agricultural officers.

  - Add features such as voice input, offline access, and multilingual support (e.g., Bangla, Urdu, Hindi, Tarki).

- **Personalized Recommendations**
  - Use user-specific historical data and crop cycles to personalize fertilizer suggestions.
  - Include recommendation explanations to increase trust and adoption among users.

- **Government and NGO Integration**
  - Partner with government agricultural departments and NGOs to scale the solution for public use.
  - Provide integration points for subsidy suggestions, local vendor directories, or expert contact information.

- **Sustainability Tracking**
  - Add functionality to track long-term soil health and fertilizer impact.
  - Suggest eco-friendly or organic fertilizer alternatives when applicable.

References

[1] A. Sharma, R. Verma, and P. Singh. Application of random forest in soil classification for fertilizer recommendation. *International Journal of Agricultural Science*, 12(4):112–125, 2021.

[2] H. Singh, K. Banerjee, and P. Das. Hybrid decision tree and deep learning model for fertilizer recommendation. In *Proceedings of the AI in Agriculture Conference*, volume 15, pages 200–215, 2021.

[3] V. Kumar, S. Patel, and R. Das. Soil fertility classification using support vector machines. *IEEE Transactions on Smart Agriculture*, 18(2):45–56, 2022.

[4] D. Gupta, T. Roy, and M. Chatterjee. Artificial neural networks for fertilizer prediction: A machine learning approach. *Journal of Precision Agriculture*, 25(1):78–91, 2023.