

ML Based Soil Analysis for Crop Recommendation

*A project report submitted to
Jawaharlal Nehru Technological University Kakinada, in the partial Fulfillment
for the Award of Degree of*

**BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

Submitted by

BALA MURALI GORANTLA	21491A4231
BHARATH VARMA GOTTUMUKKALA	21491A4232
PRADEEP CHOWDARY PATTIPATI	21491A4247
POORNATEJA PEYYALA	21491A4248
MASHOOR SHAIK	21491A4252
KARTHIK PURIMITLA	21491A4263

Under the esteemed guidance of

K. SUDHARSHAN *M.Tech, (Ph.D.)*

Assistant Professor



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**QIS COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)**

An ISO 9001:2015 Certified institution, approved by AICTE & Reaccredited by

NBA, NAAC 'A+' Grade

(Affiliated to Jawaharlal Nehru Technological University, Kakinada)

VENGAMUKKAPALEM, ONGOLE – 523 272, A.P

QIS COLLEGE OF ENGINEERING AND TECHNOLOGY
(AUTONOMOUS)

An ISO 9001:2015 Certified institution, approved by AICTE & Reaccredited by NBA, NAAC 'A+'

Grade (Affiliated to Jawaharlal Nehru Technological University, Kakinada)

VENGAMUKKAPALEM, ONGOLE-523001, A.P

MARCH 2025



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)

CERTIFICATE

*This is to certify that the technical report entitled “**ML Based Soil Analysis for Crop Recommendation**” is a bonafide work of the following final B Tech students in the partial fulfillment of the requirement for the award of the degree of bachelor of technology in COMPUTER SCIENCE AND ENGINEERING (ARTIFICIAL INTELLIGENCE & MACHINE LEARNING) for the academic year 2024-2025.*

BALA MURALI GORANTLA	21491A4231
BHARATH VARMA GOTTUMUKKALA	21491A4232
PRADEEP CHOWDARY PATTIPATI	21491A4247
POORNATEJA PEYYALA	21491A4248
MASHOOR SHAIK	21491A4252
KARTHIK PURIMITLA	21491A4263

Signature of the guide
K. SUDARSHAN *M.Tech, (Ph.D.)*
Assistant Professor

Signature of Head of Department
Dr. G. LAKSHMI VARAPRASAD *M.Tech, Ph.D.*
HOD, Professor in CSE(AI&ML)

Signature of External Examiner

ACKNOWLEDGEMENT

We thank the almighty for giving us the courage and perseverance in completing the project.

It is an acknowledgement for all those people for all those people who have given us their heartfelt cooperation in making in making the major project a grand success.

We would like to place on record our deep sense of gratitude to the Honourable Executive Chairman **DR.N.S. KALYAN CHAKRAVARTHY**, Honourable Executive Vice Chairman **Dr. N. SRI GAYATRI DEVI** and Principal **Dr. Y.V. HANUMANTHA RAO** for providing the necessary facilities to carry out the project work.

We express our gratitude to the Head of the Department of **AI&ML**, **DR.G. LAKSHMI VARAPRASAD**, Project Guide **Mr. K. SUDHARSHAN** for their valuable suggestions, guidance, and cooperation throughout the project.

We would like to express our thankfulness to **CSCDE & DPSR** for their constant motivation and valuable help throughout the project.

Finally, we would like to thank our Parents, Family and Friends for their cooperation in completing this project.

SUBMITTED BY

BALA MURALI GORANTLA	21491A4231
BHARATH VARMA GOTTUMUKKALA	21491A4232
PRADEEP CHOWDARY PATTIPATI	21491A4247
POORNATEJA PEYYALA	21491A4248
MASHOOR SHAIK	21491A4252
KARTHIK PURIMITLA	21491A4263

ABSTRACT

Agriculture serves as the backbone of many economies, playing a crucial role in ensuring food security, sustaining livelihoods, and driving economic development. One of the key aspects of successful agriculture is the selection of suitable crops, which directly influences productivity, resource efficiency, and environmental sustainability. To enhance this process, this study leverages advanced machine learning techniques, including Support Vector Classification (SVC), logistic regression, decision tree classifiers, K-nearest neighbour, Naïve Bayes, and Random Forest, to develop a reliable crop recommendation system. The research analyses various soil and environmental parameters such as nitrogen, potassium, phosphorus levels, temperature, humidity, soil pH, and rainfall to determine the most suitable crops for different seasons. The effectiveness of each machine learning model is evaluated using key performance metrics, including accuracy, precision and F1-score, to identify the most efficient approach for crop recommendation. By integrating data-driven insights into agriculture, this study empowers farmers to make informed decisions, ultimately optimizing crop selection and improving overall agricultural efficiency. We achieved the highest accuracy by using GaussianNB after comparing it with all other machine learning algorithms. Furthermore, the findings contribute to the advancement of precision agriculture by enhancing crop yield predictions, promoting sustainable farming practices, and ensuring better resource management. Through the application of machine learning, this research paves the way for a more intelligent, efficient, and environmentally conscious approach to farming, benefiting both farmers and the agricultural industry as a whole.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
1	INTRODUCTION	1
	1.1 Motivation	3
	1.2 Problem Statement	4
	1.3 Overview of the project	4
	1.4 The objectives of this Research	5
2	LITERATURE SURVEY	9
	2.1 Basic Concepts	9
	2.2 Related Work	10
	2.3 Research Gap	12
3	PROPOSED WORK AND ANALYSIS	13
	3.1 Proposed System	13
	3.1.1 Data collection and Preprocessing	13
	3.1.2 Machine Learning Model Development	14
	3.1.3 System Architecture and Implementation	15
	3.1.4 Advantages and Expected Outcomes	16
	3.2 Feasibility Study	16

	3.2.1 Economical Feasibility	17
	3.2.2 Technical Feasibility	17
	3.2.3 Operational Feasibility	18
4	SYSTEM DESIGN	19
	4.1 System Architecture	19
	4.2 Modules	20
	4.2.1 Data Collection	20
	4.2.2 Preprocessing Module	20
	4.2.3 Machine Learning Module	20
	4.2.4 Evaluation Module	21
	4.2.5 User Interface Module	21
	4.3 UML Diagram	22
	4.3.1 Class Diagram	23
	4.3.2 Use case Diagram	24
	4.3.3 Data Flow Diagram	24
	4.3.4 Sequence Diagram	25
	4.3.5 Activity Diagram	26
	4.3.6 Deployment Diagram	27
	4.3.7 Component Diagram	27

5	IMPLEMENTATION	29
	5.1 Software Requirements	29
	5.2 Hardware Requirements	29
	5.3 Technologies used	29
	5.3.1 Introduction to Machine Learning	29
	5.3.2 How does Machine Learning work?	30
	5.3.3 Python	32
	5.3.4 Packages used in Project	33
	5.4 Source code	35
6	TESTING	44
	6.1 Testing Objectives	44
	6.2 Test Cases	45
	6.2.1 Functional Test Cases	45
	6.2.2 Performance Test Cases	46
	6.2.3 Security Test Cases	46
	6.2.4 Usability Test Cases	47
7	RESULTS	50
8	CONCLUSION	53
9	Future Scope	54
10	APPENDIX / APPENDICES	55
	REFERENCES	67

List of Figures

Figure No.	Figure Name	Page No.
1	ML Based Crop Recommendation System	2
4.1	System Architecture	19
4.3.1	Class diagram	23
4.3.2	Usecase diagram	24
4.3.3	Data flow diagram	25
4.3.4	Sequence diagram	26
4.3.5	Activity diagram	26
4.3.6	Deployment diagram	27
4.3.7	Component diagram	28
5.4.1	The home page of the web application	41
5.4.2	User is instructed to fill the parameters of the soil	42
5.4.3	The ML model predicts the suitable crop for the soil	42
5.4.4	Dataset used to train the ML model	43
7.1	Confusion matrix of the best ML model	52

CHAPTER 1

INTRODUCTION

Agriculture serves as a fundamental pillar of food security, economic development, and the sustenance of millions of livelihoods worldwide. The success of agricultural practices largely depends on the selection of appropriate crops, which must align with specific soil conditions and environmental factors. Choosing the right crops based on soil composition, climate patterns, and nutrient availability is essential for maximizing productivity while ensuring long-term environmental sustainability. As the global agricultural landscape faces increasing challenges due to climate change, unpredictable weather patterns, and soil degradation, integrating advanced technological solutions has become crucial in assisting farmers with informed, data-driven decision making.

Soil quality and health play a vital role in determining crop suitability. Several key parameters influence soil fertility and, consequently, the growth potential of crops. These include macronutrients such as nitrogen (N), phosphorus (P), and potassium (K), as well as environmental factors like soil pH, temperature, humidity, and rainfall. By analyzing these variables, farmers can make informed decisions about the most suitable crops to cultivate in a particular region and season. This targeted approach not only optimizes resource utilization but also leads to improved agricultural yields, reduced input costs, and minimized environmental impact.

To address the complexities of crop selection, this research employs machine learning techniques to develop an intelligent crop recommendation system. The

study explores multiple classification algorithms, including Support Vector Classification (SVC), logistic regression, decision tree classifiers, K-nearest neighbor (KNN), Naïve Bayes, and Random Forest. These models process extensive soil datasets, evaluating various conditions to suggest the most appropriate crops for a given region. The use of machine learning enables accurate predictions by identifying patterns and relationships within agricultural data that may not be easily discernible through traditional methods.

By integrating machine learning into agricultural decision-making, this study contributes to the advancement of precision farming—a modern approach that enhances efficiency by leveraging data-driven insights. The implementation of AI-based crop recommendation systems helps farmers optimize land usage, improve sustainability, and adapt to evolving environmental conditions. Ultimately, this research addresses modern agricultural challenges by providing innovative, intelligent solutions that support sustainable farming practices and strengthen global food production systems.

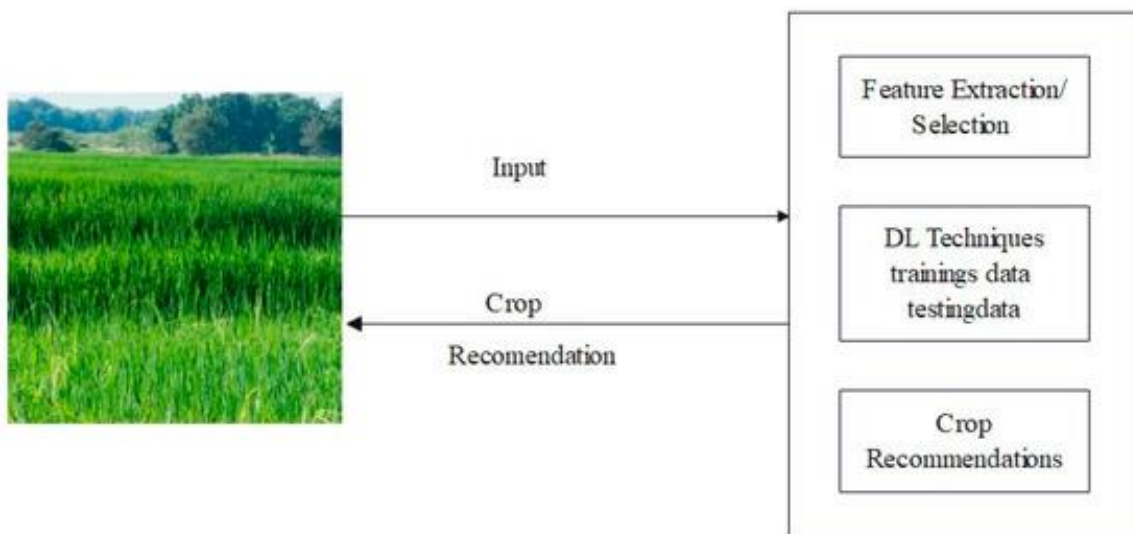


Fig.1. ML Based Crop Recommendation System.

Furthermore, the adoption of machine learning in agriculture not only enhances productivity but also empowers farmers with actionable insights, reducing dependency on traditional trial-and-error methods. By leveraging predictive analytics, farmers can make proactive decisions, mitigating risks associated with climate variability and soil degradation. Additionally, the integration of such AI-driven systems fosters sustainable farming by promoting efficient resource management, reducing excessive fertilizer use, and minimizing environmental impact. As agriculture continues to evolve in response to global challenges, data-driven innovations like this crop recommendation system pave the way for smarter, more resilient food production strategies, ultimately contributing to global food security and economic stability.

1.1 Motivation

The motivation behind this project stems from the critical role of agriculture in ensuring food security, sustaining livelihoods, and driving economic growth. As the backbone of many economies, agriculture faces increasing challenges due to climate change, environmental fluctuations, and the growing need for sustainable farming practices. One of the most crucial aspects of agricultural productivity is optimizing crop selection, as choosing the right crops based on soil conditions can significantly impact yields and resource efficiency. Understanding soil health is essential, as factors such as nitrogen, potassium, phosphorus levels, temperature, humidity, soil pH, and rainfall determine crop suitability. With advancements in technology, machine learning presents a powerful solution to revolutionize traditional farming methods. This project aims to integrate machine learning models, including Support Vector Classification (SVC), logistic

regression, K-nearest neighbor (KNN), Naïve Bayes, Random Forest, and decision tree classifiers, to enhance the accuracy and reliability of crop recommendations. By leveraging these advanced techniques, the project seeks to empower farmers with data-driven insights, helping them make informed decisions that optimize crop yields. Additionally, the research contributes to the growing field of precision agriculture by demonstrating the potential of AI-based solutions to address modern agricultural challenges, ensuring resilient food production systems for the future.

1.2 Problem Statement

Agriculture is a crucial sector, but farmers often face challenges in selecting the right crops due to fluctuating environmental conditions, soil degradation, and lack of access to scientific recommendations. Traditional farming methods rely heavily on experience rather than data-driven insights, leading to inefficiencies, low productivity, and environmental stress. This research aims to develop a machine learning-based crop recommendation system that analyzes soil and climate parameters to suggest the most suitable crops, thereby improving agricultural efficiency and ensuring sustainability.

1.3 Overview of the project

This project focuses on the development of an AI-powered crop recommendation system that utilizes machine learning techniques to assist farmers in selecting the most suitable crops based on soil conditions and environmental factors. The system processes key soil parameters such as nitrogen, phosphorus, potassium levels, pH, temperature, humidity, and rainfall, which are crucial in determining crop viability. By analyzing this data, the system employs multiple machine learning algorithms, including Support Vector Classification (SVC), logistic

regression, decision tree classifiers, K-nearest neighbor (KNN), Naïve Bayes, and Random Forest, to classify soil conditions and generate accurate crop recommendations. The goal of this system is to provide a user-friendly and efficient solution that allows farmers to make well-informed, data-driven decisions. By integrating real-time soil and environmental data, the system supports precision agriculture by optimizing land use, improving resource allocation, and reducing waste. This approach not only enhances agricultural productivity but also promotes sustainable farming practices that contribute to long-term environmental conservation.

1.4 Objectives of the research

- i. To develop an advanced crop recommendation system using machine learning techniques for improved agricultural decision-making.
- ii. To analyze essential soil parameters such as nitrogen, phosphorus, potassium levels, pH, temperature, humidity, and rainfall to determine crop suitability.
- iii. To implement and evaluate multiple machine learning algorithms, including Support Vector Classification (SVC), logistic regression, decision tree classifiers, K-nearest neighbor (KNN), Naïve Bayes, and Random Forest.
- iv. To enhance the accuracy and reliability of crop recommendations through data-driven insights and predictive analytics.
- v. To optimize resource utilization by recommending crops that align with soil health and environmental conditions, minimizing waste and excessive fertilizer use.

- vi. To contribute to sustainable farming practices by integrating AI-based solutions that support long-term environmental conservation.
- vii. To promote precision agriculture by leveraging machine learning models for improved land use planning and yield prediction.
- viii. To provide a user-friendly and accessible system that enables farmers to make informed decisions based on real-time soil and environmental data.
- ix. To bridge the gap between technological advancements and traditional farming methods, making AI-driven solutions more practical and applicable in agriculture.
- x. To address modern agricultural challenges such as climate change, soil degradation, and fluctuating environmental conditions by offering intelligent, data-backed crop recommendations.

Organization of the work:

Chapter 1:

Crop recommendation systems and machine learning techniques. Next, soil and environmental data will be collected, preprocessed, and analyzed. Multiple machine learning models, such as SVC, KNN, Naïve Bayes, logistic regression, decision trees, and Random Forest, will be implemented and evaluated. Finally, the system will be integrated with a user-friendly interface to deliver accurate, real-time crop recommendations, promoting sustainable and data-driven agriculture.

Chapter 2:

Reviewing fundamental concepts in agriculture, precision farming, and machine learning techniques. Next, related research on crop recommendation models,

including traditional and AI-based systems, will be examined. Finally, research gaps such as dataset limitations, real-time data integration, and model interpretability will be analysed to identify areas for improvement and future implementation.

Chapter 3:

Proposing a machine learning-based crop recommendation system, detailing the system's architecture, data processing, model development, and expected benefits. Next, a feasibility study covering economic, technical, and operational aspects will assess its practicality and cost-effectiveness. Finally, deployment strategies, including user-friendly interfaces and real-time data integration, will be explored to ensure successful implementation.

Chapter 4:

designing the system architecture with key modules such as data preprocessing, machine learning, evaluation, and user interface. It also includes developing UML diagrams like class, use case, data flow, sequence, activity, deployment, and component diagrams to visualize system functionalities and interactions. These diagrams aid in refining the design, optimizing system components, and enhancing implementation clarity.

Chapter 5:

The work is organized into three main sections. It covers system and technology requirements, including Python, Flask, and essential libraries, followed by machine learning implementation involving data processing, model training, and prediction. Finally, it explains the web application workflow, where users input soil data, receive crop predictions, and save the results for future reference.

Chapter 6:

This chapter is organized to give a clear understanding of the testing process. First, it covers key testing objectives like ensuring functionality, performance, security, and usability to enhance software quality and reliability. Then, it presents detailed test cases that show how various tests—like input handling, performance under load, security checks, and user experience—help ensure the system works smoothly and securely.

Chapter 7:

This chapter is structured to present the experimental results and their significance. It starts with a discussion on the performance of various machine learning models in classifying soil conditions and recommending crops. It then highlights the comparative advantages of ensemble models, followed by the impact of real-time soil data integration on improving agricultural productivity.

CHAPTER 2

LITERATURE SURVEY

2.1 Basic Concepts:

The foundation of a crop recommendation system lies in understanding key agricultural and technological concepts. Agriculture is deeply influenced by soil characteristics, climatic conditions, and environmental factors. Soil parameters such as nitrogen, phosphorus, and potassium levels play a vital role in determining the fertility and suitability of the land for specific crops. Additionally, factors like temperature, humidity, soil pH, and rainfall significantly impact plant growth and yield.

With advancements in technology, machine learning (ML) and artificial intelligence (AI) have emerged as transformative tools in precision agriculture. Machine learning models analyze vast datasets to identify patterns and correlations that are not easily discernible through traditional methods. Algorithms such as Support Vector Classification (SVC), logistic regression, decision trees, K-nearest neighbour (KNN), Naïve Bayes, and Random Forest are commonly used for classification and prediction tasks in agricultural applications. These models process soil and climate data to generate accurate crop recommendations, helping farmers make data-driven decisions.

Another key concept is precision agriculture, which involves optimizing farming practices through technology-driven solutions. By leveraging IoT sensors, remote sensing, and machine learning models, precision agriculture enhances efficiency by monitoring soil health, predicting yields, and improving resource allocation. The integration of AI-based recommendation systems supports sustainable farming by reducing waste, optimizing fertilizer use, and increasing

productivity. These fundamental concepts provide the basis for understanding the significance of machine learning in modern agriculture.

2.2 Related Work:

Several studies have been conducted on machine learning-based crop recommendation systems, each exploring different methodologies and datasets. Early research focused on traditional agricultural advisory systems, which relied on expert knowledge and historical data to provide recommendations. However, these systems were often static and lacked real-time adaptability to changing environmental conditions.

Recent advancements have led to the development of data-driven recommendation systems that utilize supervised learning techniques for better accuracy. Studies have implemented models such as decision trees, SVMs, and artificial neural networks (ANNs) to classify soil conditions and predict optimal crops. Research conducted by Motwani et al. (2023) utilized soil analysis techniques combined with logistic regression and decision tree classifiers to enhance prediction accuracy. Another study by Barvin and Sampradeepraj (2023) explored the use of Graph Convolutional Neural Networks (GCNNs) for soil-based crop recommendations, demonstrating significant improvements in prediction efficiency.

Other works have focused on deep learning techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) to analyze large-scale agricultural datasets. These approaches have been particularly effective in classifying satellite imagery and soil composition. However, despite

advancements, challenges remain in improving model interpretability, handling real-time data variability, and optimizing computational efficiency.

Several research studies have explored machine learning-based crop recommendation systems, employing different methodologies and datasets to enhance prediction accuracy and efficiency. Early research focused on traditional advisory systems that relied on expert knowledge and historical data, but these systems lacked real-time adaptability. To address this, recent advancements in supervised learning techniques have improved prediction models. For instance, Motwani et al. (2023) combined soil analysis methods with logistic regression and decision tree classifiers to enhance crop recommendation accuracy. Other studies have incorporated advanced models, such as Graph Convolutional Neural Networks (GCNNs), as demonstrated by Barvin and Sampradeepraj (2023), who utilized GCNNs to represent soil properties and improve prediction efficiency. Additionally, ensemble techniques, like Random Forests, have gained popularity for their ability to aggregate weak learners, leading to better generalization and robustness. Deep learning approaches, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have been particularly effective in analyzing large-scale datasets, including satellite imagery and soil composition, offering more granular insights. These models have also facilitated real-time data integration through sensor-based monitoring of soil parameters, enabling continuous updates to crop recommendations.

Despite these advancements, ongoing challenges include improving model interpretability, handling real-time data variability, and optimizing computational efficiency, which remain critical areas for future research and development in agricultural decision-making systems.

2.3 Research Gap:

While existing studies have demonstrated the potential of machine learning in agriculture, several research gaps remain unaddressed. One key limitation is the lack of comprehensive and region-specific datasets, as many studies use generalized datasets that do not consider local soil variations and climatic conditions. This affects the accuracy and adaptability of crop recommendation models in different geographical locations.

Another major challenge is the integration of real-time soil and weather data into machine learning models. Most existing models rely on pre-collected datasets rather than continuously updating information from IoT-based sensors or satellite data. This restricts their ability to provide dynamic and context-aware recommendations to farmers.

Furthermore, model performance and interpretability remain areas of concern. While deep learning techniques offer high accuracy, they often function as black-box models with limited transparency in decision-making. Farmers require explainable AI solutions that provide clear justifications for crop recommendations, enabling them to trust and adopt these systems effectively.

Lastly, there is a lack of deployment-focused studies that bridge the gap between research and real-world implementation. Many models achieve high accuracy in controlled environments but fail to address practical challenges such as user accessibility, computational efficiency, and cost-effectiveness. Addressing these research gaps will be crucial for advancing precision agriculture and making AI-driven crop recommendation systems more reliable, scalable, and user-friendly.

CHAPTER 3

PROPOSED WORK AND ANALYSIS

3.1 Proposed System:

The proposed system aims to enhance agricultural productivity by providing accurate and data-driven crop recommendations using machine learning techniques. By analyzing soil parameters and environmental conditions, the system identifies the most suitable crops for a given region, ultimately optimizing resource utilization and promoting sustainable farming.

This system integrates various machine learning models such as Support Vector Classification (SVC), logistic regression, decision tree classifiers, K-nearest neighbour (KNN), Naïve Bayes, and Random Forest to develop a robust and efficient recommendation mechanism. These models process soil data, classify conditions, and generate optimal crop suggestions based on past agricultural data and real-time environmental inputs.

The system is designed to function as a user-friendly application, where farmers can input soil parameters manually or integrate IoT-based sensor data to receive instant crop recommendations. The final output helps farmers make informed decisions, increase yield efficiency, and reduce financial and environmental risks associated with improper crop selection..

3.1.1 Data Collection and Preprocessing:

The success of any machine learning model heavily depends on the quality of the data used for training. In this proposed system, data is collected from various sources, including agricultural research databases, government records, field studies, and IoT-based soil sensors. The dataset includes key parameters such as nitrogen (N), phosphorus (P), and potassium (K) levels, soil pH, temperature,

humidity, and rainfall, which are crucial for determining soil fertility and crop suitability.

Once collected, the data undergoes several preprocessing steps to ensure its quality and usability. These steps include data cleaning (removing inconsistencies and missing values), normalization (scaling different parameters to a uniform range), and feature selection (identifying the most relevant parameters influencing crop growth). Proper preprocessing ensures that the machine learning models are trained on high-quality data, leading to accurate and reliable recommendations.

3.1.2 Machine Learning Model Development:

The system leverages multiple machine learning algorithms to enhance the accuracy of crop recommendations. Each model contributes uniquely to the system:

- i. Support Vector Classification (SVC): Helps classify crop suitability by finding an optimal decision boundary between different soil types.
- ii. Logistic Regression: Provides probabilistic insights into crop viability based on soil characteristics.
- iii. Decision Tree Classifier: Captures non-linear relationships between soil parameters and crop growth, making it a reliable decision-making model.
- iv. K-Nearest Neighbor (KNN): Predicts crop suitability by analyzing similarities between past soil data points.
- v. Naïve Bayes: Uses probability-based classification to estimate the most suitable crops.

vi. **Random Forest:** Combines multiple decision trees to improve prediction accuracy and reduce overfitting.

To enhance the model's performance, an ensemble learning approach is employed, integrating multiple models to provide a hybrid recommendation system. This ensures that predictions remain accurate across diverse soil and climatic conditions. The model's effectiveness is evaluated using metrics such as accuracy, precision, recall, and F1-score, ensuring that the system delivers highly reliable results.

3.1.3 System Architecture and Implementation

The architecture of the proposed system follows a modular structure to streamline data processing, model training, and user interaction. The system consists of the following key components:

- i. **Data Acquisition Module:** Collects real-time and historical soil and environmental data.
- ii. **Preprocessing Module:** Cleans, normalizes, and selects the most relevant features from the dataset.
- iii. **Machine Learning Model Module:** Implements multiple classification algorithms to analyze soil properties and generate crop recommendations.
- iv. **Evaluation Module:** Assesses model performance based on key metrics and refines predictions.
- v. **User Interface Module:** Provides an intuitive application where farmers can input soil parameters manually or use automated sensor readings to receive crop recommendations

3.1.4 Advantages and Expected Outcomes

The proposed system offers multiple advantages for farmers and agricultural stakeholders. By leveraging data-driven insights, it minimizes uncertainties in crop selection and ensures that the chosen crops are best suited for the given soil and climatic conditions. The automation of recommendations reduces farmers' reliance on traditional trial-and-error methods, improving both productivity and profitability.

Another key benefit is sustainability—by recommending crops based on soil health, the system reduces excessive fertilizer use, prevents soil degradation, and conserves water resources. Furthermore, precision agriculture principles are applied, ensuring that land usage is optimized for maximum yield efficiency.

The expected outcomes of this system include:

- i. Improved agricultural productivity through scientifically backed crop recommendations.
- ii. Reduction in environmental impact by optimizing fertilizer and water usage.
- iii. Cost savings for farmers by preventing unsuitable crop cultivation and resource wastage.
- iv. Integration of AI and IoT for real-time decision-making, bringing innovation to traditional farming practices.

3.2 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For

feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

3.2.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.2.2 TECHNICAL FEASIBILITY

The system is technically feasible as it employs widely used machine learning frameworks such as Scikit-learn, TensorFlow, and Pandas, which are capable of handling large agricultural datasets. The integration of IoT-based sensors for real-time soil monitoring further strengthens its practical applicability. With advancements in cloud computing and edge AI, the system can be efficiently deployed for large-scale agricultural use. The modular architecture also ensures scalability, allowing the system to accommodate increasing data volumes and new algorithm updates.

3.2.3 OPERATIONAL FEASIBILITY

The system is designed to be farmer-friendly, with an easy-to-use interface that allows users to input soil parameters manually or through sensor-based automation. By providing instant crop recommendations, the system simplifies decision-making, reducing the dependency on traditional agricultural methods. To further ensure adoption by farmers, the system can be deployed as a mobile application or web platform, making it accessible even in remote areas. Training programs and farmer awareness initiatives can further enhance operational feasibility and ensure successful implementation.

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE

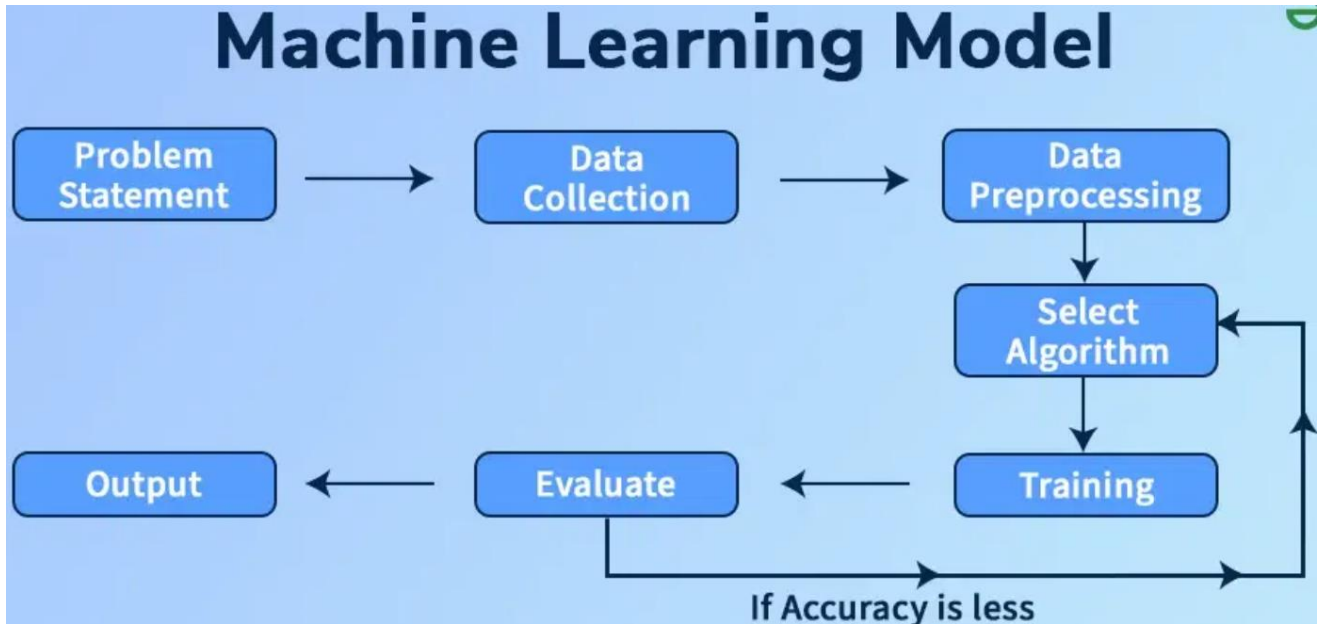


Fig 4.1: System Architecture

- **Data Preprocessing:** Cleaning, transforming, and preparing the collected data for model training by handling missing values, normalization, encoding, etc.
- **Select Algorithm:** Choosing the appropriate machine learning algorithm based on the problem type (classification, regression, clustering, etc).
- **Training:** Feeding the preprocessed data into the selected algorithm to learn patterns and relationships.
- **Evaluate:** Assessing the model's performance using metrics such as accuracy, precision, recall, F1-score, etc.
- **Output:** The final result or prediction produced by the trained machine learning model.

- **If Accuracy is Less:** If the evaluation results indicate low accuracy, the model is improved by tuning hyperparameters, selecting a different algorithm, or enhancing data preprocessing.

4.2 MODULES

4.2.1 Data Collection

This module is responsible for acquiring soil and environmental data from various sources, including:

- i. Agricultural research databases and government records.
- ii. Real-time IoT-based sensors that measure soil moisture, pH, nitrogen, phosphorus, potassium levels, temperature, and humidity.
- iii. Historical weather data and seasonal crop yield information..

4.2.2 Preprocessing Module

Raw agricultural data often contains missing values, inconsistencies, and outliers that can affect model performance. This module performs:

- i. Data cleaning: Handling missing values and removing incorrect or duplicate entries.
- ii. Normalization: Scaling data to ensure all features contribute equally to model training.
- iii. Feature selection: Identifying key parameters (e.g., soil nutrients, temperature, rainfall) that directly impact crop growth.

This step ensures that only relevant, high-quality data is used for model training.

4.2.3 Machine Learning Module

This module applies multiple classification algorithms to predict the most suitable crop based on soil and climatic conditions. The models used include:

Support Vector Classification (SVC): Efficient in handling high-dimensional agricultural datasets.

- i. Logistic Regression: Provides probability-based crop recommendations.
- ii. Decision Tree Classifier: Captures complex relationships between soil parameters and crops.
- iii. K-Nearest Neighbor (KNN): Classifies crops based on similarity with past data.
- iv. Naïve Bayes: Uses probability estimation for prediction.
- v. Random Forest: An ensemble method that enhances accuracy and minimizes overfitting.

A hybrid model integration approach is used to combine multiple models, improving prediction robustness and adaptability across various soil types and climatic conditions.

4.2.4 Evaluation Module

After model training, the system evaluates its performance using key classification metrics:

- i. Accuracy: Measures the overall correctness of predictions.
- ii. Precision & Recall: Assess the effectiveness of the model in identifying the correct crop for specific conditions.
- iii. F1-Score: A balanced metric that considers both precision and recall.

By continuously refining and testing models, this module ensures the highest possible accuracy in crop recommendations.

4.2.5 User Interface Module

This module enables farmers to interact with the system via a simple and intuitive web or mobile application. Users can:

- i. Manually input soil and weather parameters to receive crop recommendations.
- ii. Use IoT sensors for automated data entry and real-time analysis.
- iii. View graphical reports on soil health and recommended crops.

The interface is designed to be accessible to farmers with varying levels of technological expertise, ensuring widespread adoption.

4.3 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object- oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

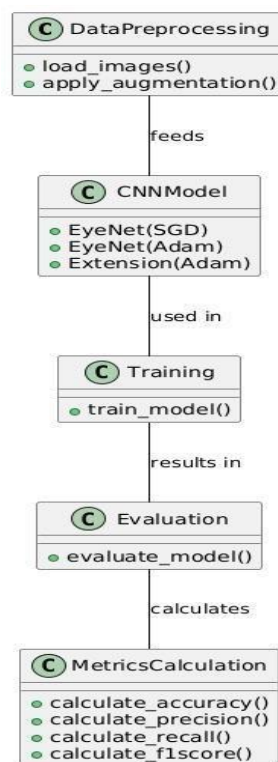
GOALS: The Primary goals in the design of the UML are as follows:

- Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.

- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modeling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns and components. Integrate best practices.

4.3.1 CLASS DIAGRAM

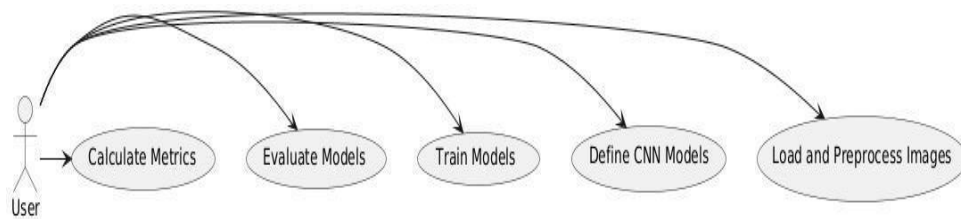
The class diagram is used to refine the use case diagram and define a detailed design of the system. The class diagram classifies the actors defined in the use case diagram into a set of interrelated classes. The relationship or association between the classes can be either an “is-a” or “has-a” relationship. Each class in the class diagram may be capable of providing certain functionalities. These functionalities provided by the class are termed “methods” of the class. Apart from this, each class may have certain “attributes” that uniquely identify the class.



4.3.1 Class diagram

4.3.2 USECASE DIAGRAM

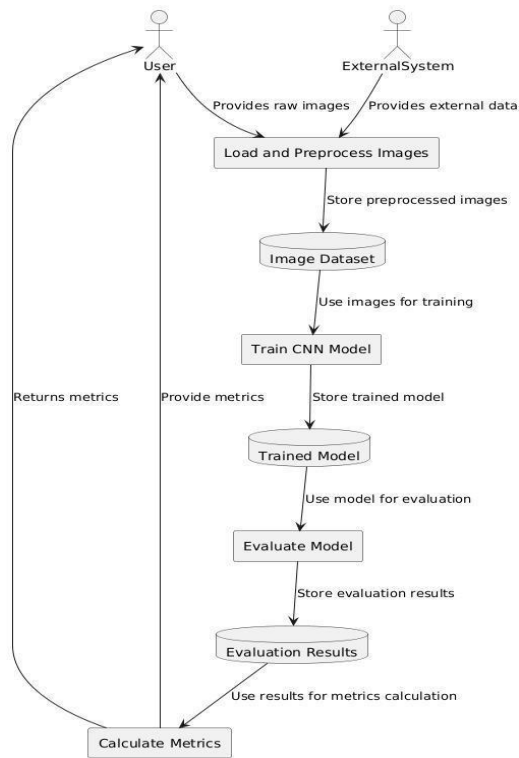
A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



4.3.2 Usecase diagram

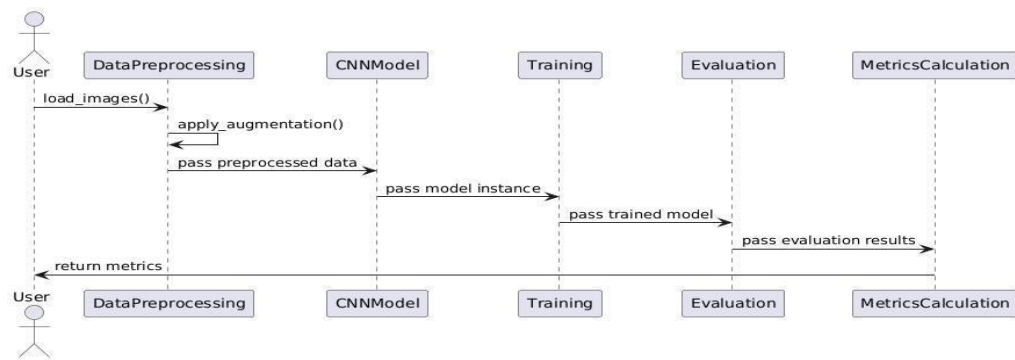
4.3.3 Data Flow Diagram

A data flow diagram (DFD) is a graphical or visual representation using a standardized set of symbols and notations to describe a business's operations through data movement.



4.3.3 Data flow diagram

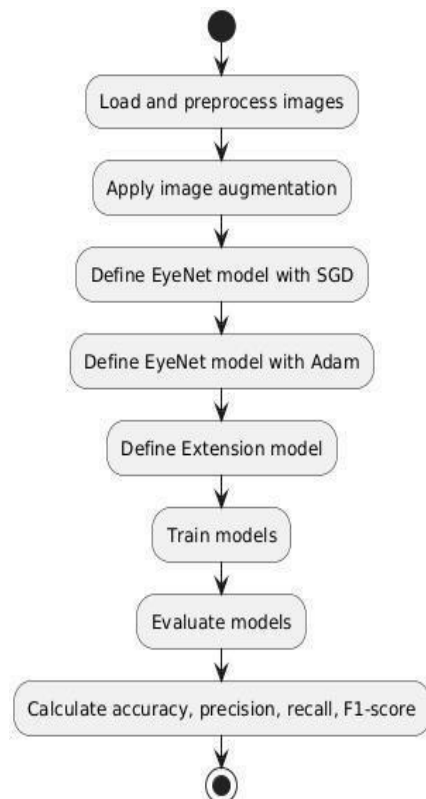
4.3.4 Sequence Diagram: A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows, as parallel vertical lines (“lifelines”), different processes or objects that live simultaneously, and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.



4.3.4 Sequence diagram

4.3.5 Activity diagram:

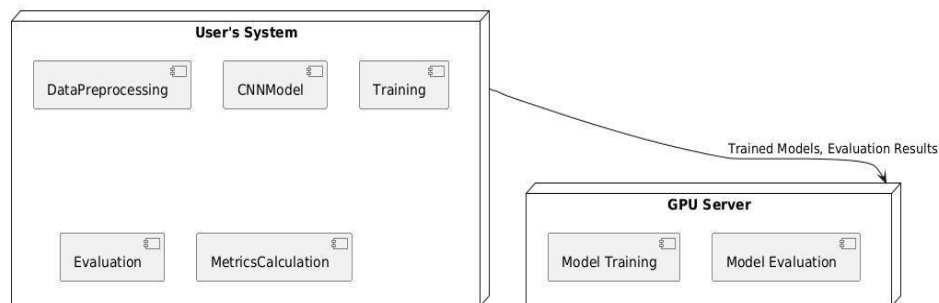
Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.



4.3.5 Activity diagram

4.3.6 Deployment diagram

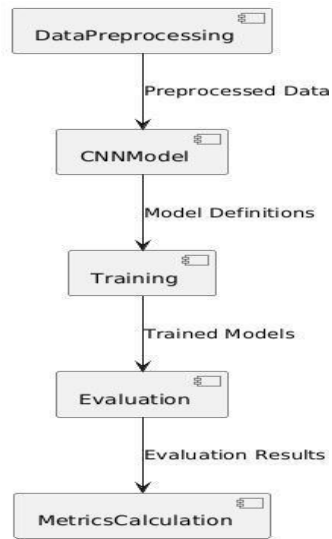
A deployment diagram in the Unified Modelling Language models the physical deployment of artifacts on nodes. To describe a website, for example, a deployment diagram would show what hardware components (“nodes”) exist (e.g., a web server, an application server, and a database server), what software components (“artifacts”) run on each node (e.g., web application, database), and how the different pieces are connected (e.g., JDBC, REST, RMI). The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub-nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers



4.3.6 Deployment diagram

4.3.7 Component diagram:

Component diagram describes the organization and wiring of the physical components in a system.



4.3.7 Component diagram

CHAPTER 5 IMPLEMENTATION

5.1 SOFTWARE REQUIREMENTS

- Operating system: Windows 10 Ultimate or above.
- Coding Language: Python.
- Front-End: Flask, HTML
- Python IDE: Visual Studio
- App: MS Excel

5.2 HARDWARE REQUIREMENTS

- Processor - Intel Core i5
- Processing Units – GPU
- RAM - 8GB
- Hard Disk - 512GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse

5.3 TECHNOLOGIES USED

5.3.1 INTRODUCTION TO MACHINE LEARNING:

Machine Learning, a term coined by Arthur Samuel in 1959, refers to a field of study that enables computers to learn from data without being explicitly programmed. Over the years, it has evolved into one of the most sought-after career paths, with Machine Learning Engineers experiencing rapid job growth and competitive salaries. According to Indeed, this role saw a 344% increase in demand in 2019, with an average base salary of \$146,085 per year.

At its core, Machine Learning is about building mathematical models that help analyze, interpret, and predict data patterns. Unlike traditional programming, where explicit instructions are provided, Machine Learning models learn from historical data by adjusting internal parameters. This ability to "learn" allows them to improve performance over time and make accurate predictions on new, unseen data.

Although Machine Learning is often categorized as a subfield of Artificial Intelligence (AI), it is better understood as a data-driven approach to building predictive models. It finds applications in various domains, such as healthcare, finance, marketing, and self-driving cars.

Key characteristics of Machine Learning:

- **Pattern Recognition:** Identifies and understands trends in data.
- **Continuous Learning:** Improves automatically by learning from past experiences.
- **Data-Driven Approach:** Relies on vast amounts of data to generate insights and predictions.
- **Similarity to Data Mining:** Deals with large datasets to extract useful information and patterns.

5.3.2 HOW DOES MACHINE LEARNING WORK?

Machine Learning enables systems to learn from historical data, build predictive models, and generate accurate outputs for new data. The accuracy of predictions improves as the amount of data increases since larger datasets help train better models. Based on their approach to learning, machine learning algorithms can be categorized into different types.

Supervised learning involves training models using labeled data, where each input has a corresponding output. The algorithm learns to map inputs to outputs

and make predictions on new data. Classification algorithms are used when outputs belong to predefined categories, such as detecting spam emails or identifying objects in images. Regression algorithms, on the other hand, are used to predict continuous values, such as forecasting house prices or stock market trends.

Unsupervised learning works with unlabeled data, identifying hidden patterns and structures within the dataset. Clustering algorithms group data points with similar characteristics, such as customer segmentation in marketing, while dimensionality reduction techniques reduce the number of input features while retaining essential information, helping to compress large datasets efficiently. Semi-supervised learning is a hybrid approach that utilizes both labeled and unlabeled data. It is particularly useful when acquiring labeled data is expensive or time-consuming. For example, in fake news detection, only a small portion of articles may be labeled, but the model learns from both labeled and unlabeled examples to improve accuracy.

Reinforcement learning is another approach where models learn by interacting with an environment and receiving rewards or penalties based on their actions. This technique is widely used in areas like self-driving cars, where the system adapts to changing road conditions, or in artificial intelligence for playing complex games, continuously improving its strategies over time.

Several specialized techniques further enhance machine learning capabilities. Active learning allows models to selectively request labels when needed, optimizing learning efficiency with minimal labeled data. Topic modeling helps in natural language processing by detecting topics within text documents, enabling better search and recommendation systems. Meta-learning allows models to improve their learning process by adapting based on past experiences. In developmental robotics, machines can autonomously acquire new skills

through self-exploration and interaction with humans, advancing robotics and artificial intelligence.

With its vast applications in healthcare, finance, autonomous systems, and beyond, machine learning continues to revolutionize industries by providing smarter, data-driven insights. As models become more sophisticated, they enhance decision-making, automate complex tasks, and pave the way for future innovations.

5.3.3 PYTHON

Python is a high-level, interpreted programming language known for its simplicity, readability, and versatility. Created by Guido van Rossum and first released in 1991, Python has become one of the most popular programming languages in the world, widely used in web development, data science, artificial intelligence, automation, and more. Its easy-to-learn syntax makes it an excellent choice for beginners while remaining powerful enough for complex applications. One of Python's biggest strengths is its extensive standard library and a vast ecosystem of third-party libraries. For instance, libraries like NumPy and Pandas make it ideal for data manipulation, while Matplotlib and Seaborn allow for powerful data visualization. In machine learning and artificial intelligence, Python is the go-to language, with libraries like TensorFlow, PyTorch, and Scikit-learn offering robust tools for model development. Additionally, frameworks such as Django and Flask make web development efficient and scalable.

Python supports multiple programming paradigms, including procedural, object-oriented, and functional programming. Its dynamic typing and automatic memory management simplify coding, allowing developers to focus more on solving

problems rather than managing system complexities. Python is also cross platform, meaning code written in Python can run seamlessly on Windows, macOS, and Linux. Another advantage of Python is its strong community support. Being an open-source language, it has a vast number of contributors who develop and maintain libraries, frameworks, and tools. This active community also ensures that Python is continuously evolving with new updates, security enhancements, and performance improvements.

Python's growing popularity has led to its adoption across various industries, including finance, healthcare, cybersecurity, and game development. With its straightforward syntax, broad application range, and powerful libraries, Python continues to be a top choice for developers, researchers, and data scientists worldwide.

5.3.4 PACKAGES USED IN PROJECT

Pandas:

It is a powerful data analysis and manipulation library for Python. It provides data structures like DataFrames and Series, which allow users to efficiently handle structured data. It is widely used for data cleaning, processing, and analysis in various domains, including finance, healthcare, and machine learning.

NumPy:

It is a fundamental library for numerical computing in Python. It provides support for working with large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to perform operations on these arrays efficiently. NumPy is heavily used in scientific computing, machine learning, and data analysis.

TensorFlow:

It is an open-source deep learning framework developed by Google. It is designed for building and training machine learning models, especially neural networks.

TensorFlow allows developers to create complex machine learning algorithms with optimized performance, making it a popular choice in artificial intelligence, computer vision, and natural language processing applications.

Datetime:

It is a built-in Python module that provides functionality for working with dates and times. It allows users to manipulate, format, and perform arithmetic operations on date and time objects, making it useful in applications that require timestamp management, scheduling, and data logging.

PIL (Python Imaging Library):

It is a library for image processing in Python. It provides capabilities to open, manipulate, and save various image file formats. PIL is useful for tasks such as resizing, filtering, and enhancing images, and is commonly used in applications involving computer vision and digital image analysis.

Predict:

It in the context of machine learning refers to the process of using trained models to make predictions on new data. Once a model is trained on historical data, it can take new inputs and generate outputs based on learned patterns. Prediction is widely used in fields like finance, healthcare, and recommendation systems.

Flask:

It is a lightweight web framework for Python that is used to build web applications and APIs. It is designed to be simple and flexible, allowing developers to create scalable applications with minimal setup. Flask is commonly used in web development, backend services, and machine learning model deployment.

5.4 SOURCE CODE

Train_model.py:

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import MinMaxScaler, StandardScaler

from sklearn.linear_model import LogisticRegression

from sklearn.naive_bayes import GaussianNB

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier

from sklearn.ensemble import (

    RandomForestClassifier,

    BaggingClassifier,

    GradientBoostingClassifier,

    AdaBoostClassifier,

)

from sklearn.metrics import accuracy_score

import pickle
```

```
# Load the dataset
```

```
crop = pd.read_csv(r"C:\Users\MYPC\OneDrive\Desktop\Crop  
Recommendation\Crop_recommendation.csv")
```

```
# Map labels to numerical values
```

```
crop_dict = {  
    'rice': 1, 'maize': 2, 'jute': 3, 'cotton': 4, 'coconut': 5,  
    'papaya': 6, 'orange': 7, 'apple': 8, 'muskmelon': 9, 'watermelon': 10,  
    'grapes': 11, 'mango': 12, 'banana': 13, 'pomegranate': 14, 'lentil': 15,  
    'blackgram': 16, 'mungbean': 17, 'mothbeans': 18, 'pigeonpeas': 19,  
    'kidneybeans': 20, 'chickpea': 21, 'coffee': 22  
}
```

```
reverse_crop_dict = {v: k for k, v in crop_dict.items()}
```

```
crop['label'] = crop['label'].map(crop_dict)
```

```
# Split data into features and labels
```

```
X = crop.drop('label', axis=1)
```

```
y = crop['label']
```

```
# Split data into training and testing sets
```

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

```
# Apply MinMaxScaler
```

```
mx = MinMaxScaler()
```

```
X_train = mx.fit_transform(X_train)
```

```
X_test = mx.transform(X_test)
```

```
# Apply StandardScaler
```

```
sc = StandardScaler()
```

```
X_train = sc.fit_transform(X_train)
```

```
X_test = sc.transform(X_test)
```

```
# Initialize models
```

```
models = {
```

```
    'LogisticRegression': LogisticRegression(),
```

```
    'GaussianNB': GaussianNB(),
```

```
    'SVC': SVC(),
```

```
    'KNeighborsClassifier': KNeighborsClassifier(),
```

```
    'DecisionTreeClassifier': DecisionTreeClassifier(),
```

```
'ExtraTreeClassifier': ExtraTreeClassifier(),  
'RandomForestClassifier': RandomForestClassifier(),  
'BaggingClassifier': BaggingClassifier(),  
'GradientBoostingClassifier': GradientBoostingClassifier(),  
'AdaBoostClassifier': AdaBoostClassifier(),  
}
```

```
# Evaluate models
```

```
best_model_name = ""
```

```
best_accuracy = 0.0
```

```
#print("Model Accuracies:")
```

```
for name, model in models.items():
```

```
    model.fit(X_train, y_train)
```

```
    y_pred = model.predict(X_test)
```

```
    accuracy = accuracy_score(y_test, y_pred)
```

```
    print(f"{name}: {accuracy:.4f}")
```

```
    if accuracy > best_accuracy:
```

```
        best_accuracy = accuracy
```

```
        best_model_name = name
```

```

print(f"\nBest Model: {best_model_name} with accuracy: {best_accuracy:.4f}")

# Train the best model

randclf = models[best_model_name]

randclf.fit(X_train, y_train)

# Save the model and scalers

pickle.dump(randclf, open('model.pkl', 'wb'))

pickle.dump(mx, open('minmaxscaler.pkl', 'wb'))

pickle.dump(sc, open('standscaler.pkl', 'wb'))

# Function for crop recommendation

def recommendation(N, P, K, temperature, humidity, ph, rainfall):

    # Create a DataFrame for input to maintain feature names

    input_data = pd.DataFrame([[N, P, K, temperature, humidity, ph, rainfall]],

                               columns=X.columns)

    # Transform using MinMaxScaler and StandardScaler

    mx_features = mx.transform(input_data)

    sc_mx_features = sc.transform(mx_features)

    # Predict using the trained model

    prediction = randclf.predict(sc_mx_features)

```

```
    return prediction[0]

# Example prediction

N = 90

P = 42

K = 43

temperature = 20.879744

humidity = 82.002744

ph = 6.502985

rainfall = 202.935536

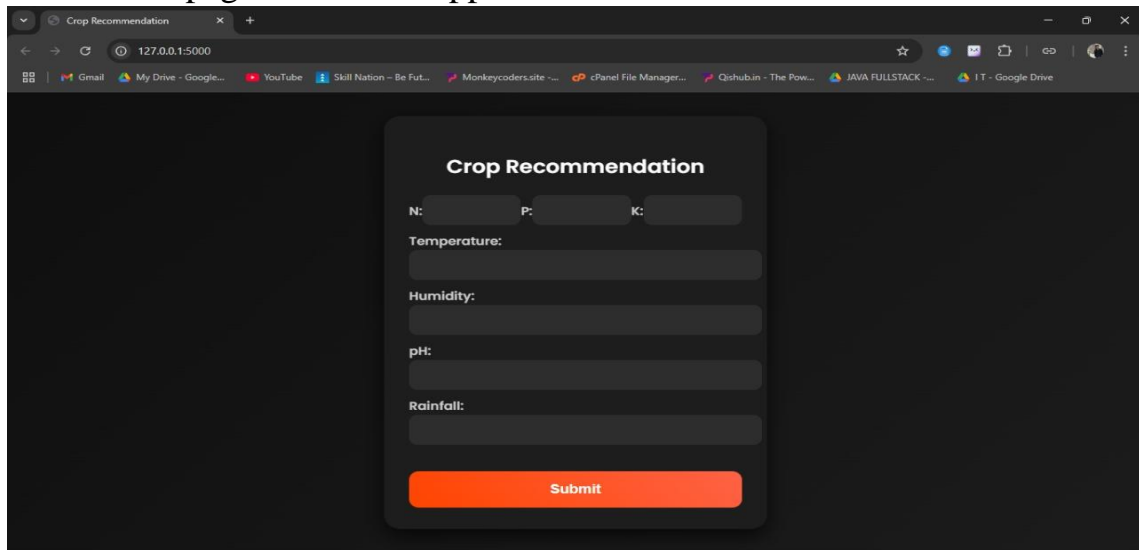

predicted_crop = recommendation(N, P, K, temperature, humidity, ph, rainfall)
predicted_crop_name = reverse_crop_dict.get(predicted_crop, "Unknown Crop")

print(f"Predicted Crop: {predicted_crop_name}")
```


Workflow Summary

- i. User provides soil data via web application or IoT sensors.
- ii. System processes data, cleaning and normalizing it.
- iii. Machine learning model analyzes and predicts the most suitable crops.
- iv. Predictions are displayed on the web application for user review.
- v. Results are saved in an Excel file for future reference.
- vi. For IoT integration, real-time updates refine crop recommendations.

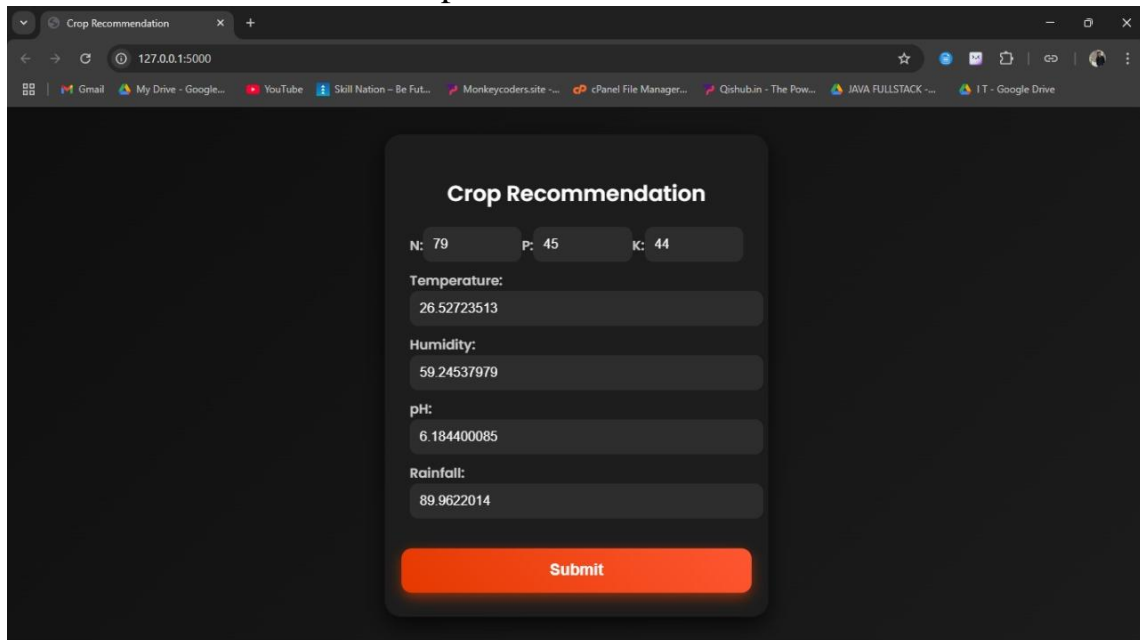
1.The home page of the web application.



The screenshot shows a web browser window with the title 'Crop Recommendation'. The address bar displays '127.0.0.1:5000'. The browser's tab bar shows several open tabs, including 'Gmail', 'My Drive - Google...', 'YouTube', 'Skill Nation - Be Fut...', 'Monkeycoders.site', 'cPanel File Manager...', 'Qishub.in - The Pow...', 'JAVA FULLSTACK ...', and 'IT - Google Drive'. The main content area features a dark-themed form titled 'Crop Recommendation'. The form contains input fields for 'N:', 'P:', and 'K:', followed by 'Temperature:', 'Humidity:', 'pH:', and 'Rainfall:'. Each of these labels is followed by a text input field. At the bottom of the form is a prominent orange 'Submit' button.

5.4.1 The home page of the web application

2. User is instructed to fill the parameters of the soil.



Crop Recommendation

N: 79 P: 45 K: 44

Temperature:
26.52723513

Humidity:
59.24537979

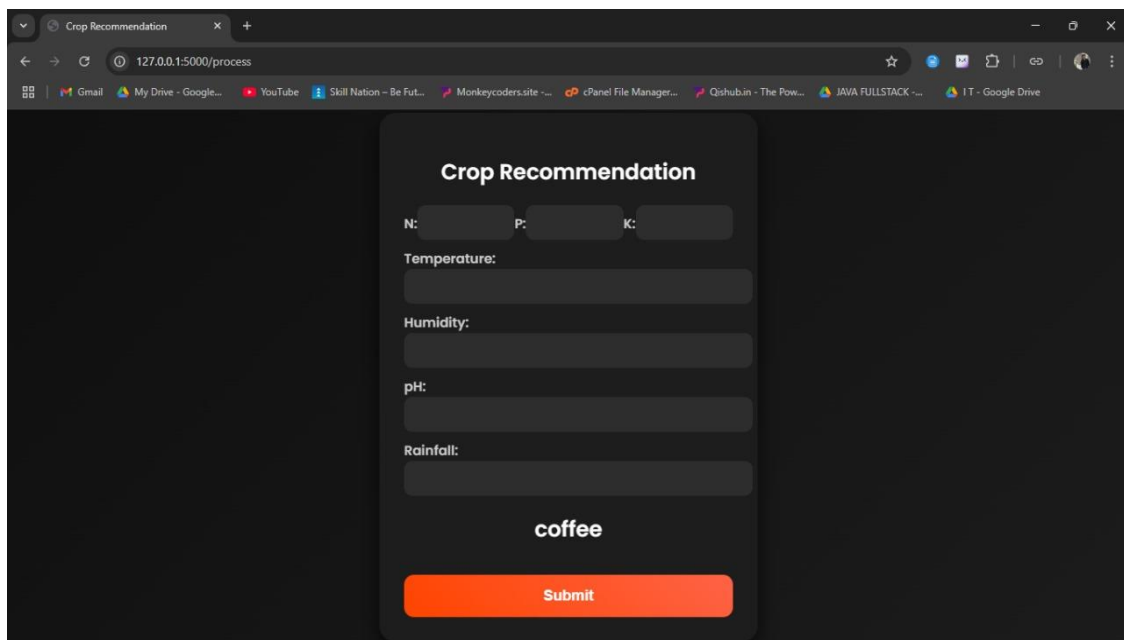
pH:
6.184400085

Rainfall:
89.9622014

Submit

5.4.2 User is instructed to fill the parameters of the soil

3. The ML model predicts the suitable crop for the soil.



Crop Recommendation

N: 79 P: 45 K: 44

Temperature:
26.52723513

Humidity:
59.24537979

pH:
6.184400085

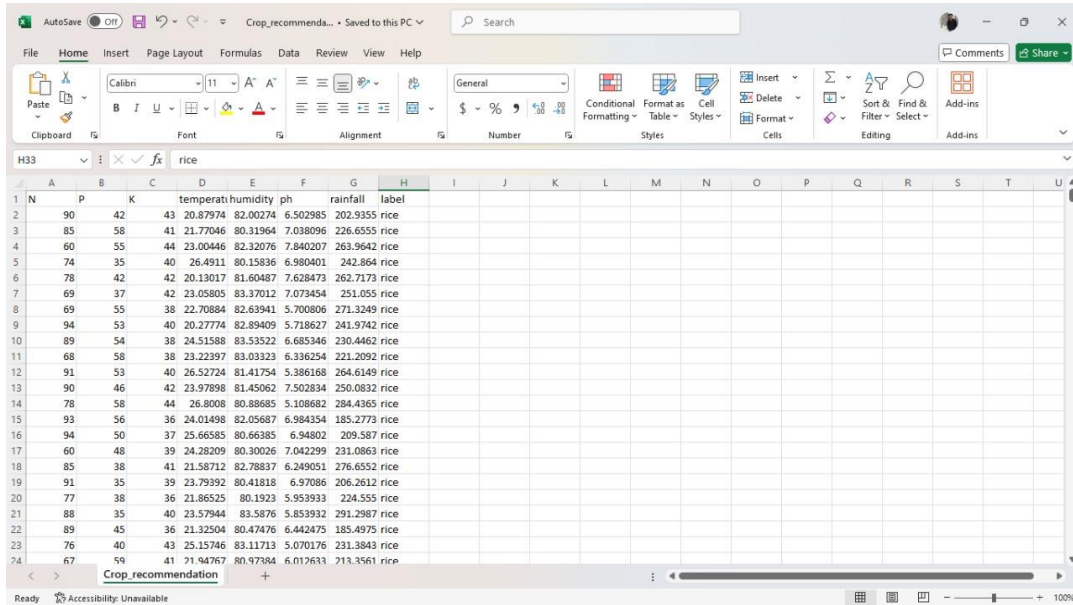
Rainfall:
89.9622014

coffee

Submit

5.4.3 The ML model predicts the suitable crop for the soil

4.The result is predicted based on the parameters from the excel.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	N	P	K	temperati	humidity	ph	rainfall	label													
2		90	42	43	20.87974	82.00274	6.502985	202.9355	rice												
3		85	58	41	21.77046	80.31964	7.038096	226.6555	rice												
4		60	55	44	23.00446	82.32076	7.840207	263.9642	rice												
5		74	35	40	26.4911	80.15836	6.980401	242.864	rice												
6		78	42	42	20.13017	81.60487	7.628473	262.7173	rice												
7		69	37	42	23.05805	83.37012	7.073454	251.055	rice												
8		69	55	38	22.70884	82.63941	5.700806	271.3249	rice												
9		94	53	40	20.27774	82.89409	5.718627	241.9742	rice												
10		89	54	38	24.51588	83.53522	6.685346	230.4462	rice												
11		68	58	38	23.22397	83.03323	6.336254	221.2092	rice												
12		91	53	40	26.52724	81.41754	5.386168	264.6149	rice												
13		90	46	42	23.97898	81.45062	7.502834	250.0832	rice												
14		78	58	44	26.8008	80.88685	5.108682	284.4365	rice												
15		93	56	36	24.01498	82.05687	6.984354	185.2773	rice												
16		94	50	37	25.66585	80.66385	6.94802	209.587	rice												
17		60	48	39	24.28209	80.30026	7.042299	231.0863	rice												
18		85	38	41	21.58712	82.78837	6.249051	276.6552	rice												
19		91	35	39	23.79392	80.41818	6.57086	206.2612	rice												
20		77	38	36	21.86525	80.1923	5.953933	224.555	rice												
21		88	35	40	23.57944	83.5876	5.853932	291.2987	rice												
22		89	45	36	21.32504	80.47476	6.442475	185.4975	rice												
23		76	40	43	25.15746	83.11713	5.070176	231.3843	rice												
24		67	59	41	21.94767	80.97384	6.017633	213.3561	rice												

5.4.4 Dataset used to train the ML model

CHAPTER 6

TESTING

6.1 Testing Objectives

The primary objective of software testing is to ensure that the developed system functions as expected, meets business and user requirements, and is free of defects that could impact its performance, usability, or security. Testing helps identify errors in the software at different stages of development, preventing costly failures after deployment. It ensures that the application runs efficiently under various conditions, including normal, high-load, and stress scenarios, maintaining stability and responsiveness.

Another crucial aspect of testing is verifying the system's reliability and robustness by simulating real-world conditions. This includes performance testing to analyze how the system behaves under different levels of load, stress testing to determine its breaking point, and scalability testing to evaluate its ability to handle increased demand. By conducting these tests, organizations can ensure that their applications remain functional and performant even during peak usage periods.

Integration testing plays a key role in verifying that different software modules work together harmoniously. It ensures that various components, such as databases, APIs, and third-party services, interact correctly without causing system failures. Similarly, unit testing helps validate the smallest units of software to detect early-stage defects, reducing the risk of larger, more complex issues later in development.

Security testing is another critical objective, as it helps identify vulnerabilities that could be exploited by malicious users. It ensures that data protection mechanisms, authentication processes, and encryption techniques are

implemented correctly, safeguarding sensitive information. Functional testing, on the other hand, ensures that the system's features and functionalities operate according to requirements, providing a smooth and user-friendly experience. Additionally, testing contributes to maintaining software quality by adhering to industry standards and compliance requirements. User acceptance testing (UAT) ensures that the final product aligns with user expectations, increasing satisfaction and reducing the likelihood of post-deployment issues. Validation and output testing confirm that the application produces accurate and meaningful results, further ensuring that the software meets its intended purpose.

Ultimately, software testing is an integral part of the software development lifecycle, minimizing risks, reducing costs associated with bug fixes, and improving the overall user experience. By implementing a structured testing strategy, organizations can deliver high-quality software that is efficient, secure, and scalable, leading to greater reliability and long-term success.

6.2 Test Cases

6.2.1 Functional Test Cases

Valid Input Handling (TC_FT_01)

This test case ensures that all required input fields function correctly when provided with valid data. The tester enters appropriate values in all input fields and submits the form. The expected outcome is that the system successfully accepts and processes the input without any errors. This test confirms that the system correctly handles expected user inputs and functions as intended.

Invalid Input Rejection (TC_FT_02)

This test verifies that the system appropriately handles invalid input data. The tester enters incorrect characters, special symbols, or other invalid values into the

input fields and attempts to submit the form. The system should detect these errors and display an appropriate error message, preventing the submission of invalid data. This ensures data integrity and prevents unexpected system behavior.

6.2.2 Performance Test Cases

Response Time Under Normal Load (TC_PT_01)

This test evaluates the system's performance when accessed by multiple users under normal load conditions. By simulating 100 concurrent users, the test checks whether the system maintains a response time of 3 seconds or less. If the system performs within this limit, it ensures that users experience a smooth and efficient interaction without noticeable delays.

System Stability Under Stress (TC_PT_02)

This test examines how the system behaves under extreme conditions by simulating more than 1000 concurrent users. The goal is to ensure that the system remains stable and does not crash, even under heavy load. A successful test indicates that the system can handle peak traffic without compromising performance or availability.

6.2.3 Security Test Cases

Unauthorized Access Prevention (TC_SEC_01)

This test case is designed to verify that restricted areas of the system are not accessible by unauthorized users. A tester attempts to access admin features using a regular user account. The expected outcome is that the system denies access and displays an appropriate error message. This test is crucial in ensuring that sensitive features are protected against unauthorized access.

SQL Injection Protection (TC_SEC_02)

This test checks whether the system is vulnerable to SQL Injection attacks. The tester inputs SQL queries into login fields or other text fields that interact with the database. The system should sanitize these inputs and prevent any unintended SQL execution. If the test is successful, it confirms that the application is secure against one of the most common cyber threats.

6.2.4 Usability Test Cases

Ease of Navigation (TC_UT_01)

This test case evaluates how easily users can navigate through the application's menus and submenus. The tester explores different sections to determine if the navigation is intuitive and user-friendly. A successful test ensures that users can efficiently find the features they need without confusion or difficulty.

UI Consistency and Readability (TC_UT_02)

This test ensures that the application's user interface maintains consistent design elements, including text readability, font sizes, button labels, and screen alignment. The tester checks various screens to verify that all elements are properly aligned and easy to read. A successful outcome indicates a well-designed interface that enhances the overall user experience.

Example Prediction output:

LogisticRegression: 0.9636

GaussianNB: 0.9955

SVC: 0.9682

KNeighborsClassifier: 0.9591

DecisionTreeClassifier: 0.9841

ExtraTreeClassifier: 0.8682

RandomForestClassifier: 0.9932

BaggingClassifier: 0.9909

GradientBoostingClassifier: 0.9818

AdaBoostClassifier: 0.1409

Best Model: GaussianNB with accuracy: 0.9955

Predicted Crop: rice

Accuracy, F1 Score, and Precision of each ML model

LogisticRegression:

Accuracy: 0.9636, F1 Score: 0.9635, Precision: 0.9644

GaussianNB:

Accuracy: 0.9955, F1 Score: 0.9954, Precision: 0.9958

SVC:

Accuracy: 0.9682, F1 Score: 0.9680, Precision: 0.9715

KNeighborsClassifier:

Accuracy: 0.9591, F1 Score: 0.9590, Precision: 0.9654

DecisionTreeClassifier:

Accuracy: 0.9886, F1 Score: 0.9886, Precision: 0.9890

ExtraTreeClassifier:

Accuracy: 0.9114, F1 Score: 0.9120, Precision: 0.9163

RandomForestClassifier:

Accuracy: 0.9932, F1 Score: 0.9932, Precision: 0.9937

BaggingClassifier:

Accuracy: 0.9864, F1 Score: 0.9864, Precision: 0.9867

GradientBoostingClassifier:

Accuracy: 0.9818, F1 Score: 0.9819, Precision: 0.9843

AdaBoostClassifier:

Accuracy: 0.1455, F1 Score: 0.0757, Precision: 0.0636

CHAPTER 7

RESULTS

The experimental results demonstrate that the proposed machine learning models, including Support Vector Classification (SVC), Logistic Regression, Decision Tree, K-Nearest Neighbor, Naïve Bayes, and Random Forest, effectively classify soil conditions and recommend optimal crops. The system has shown high accuracy in predicting suitable crops by analyzing essential soil parameters such as nitrogen, phosphorus, and potassium levels, as well as temperature, humidity, soil pH, and rainfall. The ability to process these key soil characteristics enables the model to provide data-driven recommendations, aiding farmers in selecting the most suitable crops for their land.

Comparative analysis reveals that ensemble models, such as Random Forest, outperform individual classifiers in terms of precision, recall, and F1-score. The combination of multiple weak learners in an ensemble approach significantly improves predictive accuracy and generalization across diverse datasets. This ensures that the model is not only effective in controlled experimental settings but also applicable to real-world agricultural environments. By leveraging the strengths of ensemble learning, the system minimizes errors and enhances the reliability of its crop recommendations.

Furthermore, the integration of real-time soil data from sensors enhances the adaptability of the system, ensuring that farmers receive up-to-date and accurate insights. Continuous monitoring of soil parameters allows for immediate adjustments in crop selection and farming strategies, leading to improved agricultural productivity.

Accuracy, F1 Score, and Precision of ML models

ML Model	Accuracy	F1 Score	Precision
LogisticRegression	0.9636	0.9635	0.9644
GaussianNB	0.9955	0.9954	0.9958
SVC	0.9682	0.9680	0.9715
KNeighborsClassifier	0.9591	0.9590	0.9654
DecisionTreeClassifier	0.9886	0.9886	0.9890
ExtraTreeClassifier	0.9114	0.9120	0.9163
RandomForestClassifier	0.9932	0.9932	0.9937
BaggingClassifier	0.9864	0.9864	0.9867
GradientBoostingClassifier	0.9818	0.9819	0.9843
AdaBoostClassifier	0.1455	0.0757	0.0636

Table 1: Accuracy, F1 Score, and Precision of ML models

Confusion matrix of the best ML model

```
plt.figure(figsize=(10, 8))
Confusion Matrix for GaussianNB:
[[17  0  2  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0 23  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0 17  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0 23  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0 23  0  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0 17  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0 14  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0 19  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0 21  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0 23  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0 11  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 20  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 19  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 24  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 23  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 20]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 26]
 [ 0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 17]]
127.0.0.1 - - [29/Mar/2025 00:17:57] "POST /process HTTP/1.1" 200 -
```

Fig 7.1: Confusion matrix of the best ML model

CHAPTER 8

CONCLUSION

This research introduces an advanced Crop Recommendation System & Soil Analysis using ML to assist farmers in data-driven decision-making. The study successfully demonstrates the application of machine learning algorithms to analyze soil parameters and suggest the most appropriate crops based on environmental conditions. This approach bridges the gap between traditional farming practices and modern technological advancements, providing farmers with scientifically backed insights to improve crop selection and maximize yield. By leveraging machine learning techniques, the system provides accurate crop recommendations, optimizing agricultural productivity while promoting sustainable farming practices. The integration of real-time soil data further enhances the model's reliability, ensuring that farmers receive up-to-date insights tailored to their specific soil conditions. The model's ability to process real-time soil data further enhances its reliability, making it a valuable tool for modern precision agriculture. The research highlights how integrating AI-driven approaches with traditional agricultural knowledge can significantly improve food production efficiency and sustainability.

Future research can explore enhanced deep learning architectures, additional soil datasets, and hybrid models to further refine accuracy and adaptability. By expanding the scope of data sources and refining predictive capabilities, the system can evolve to address more complex agricultural challenges. Continued advancements in AI-driven soil analysis will contribute to more precise recommendations, fostering long-term sustainability and resilience in the agricultural sector.

CHAPTER 9

FUTURE SCOPE

The future of the Crop Recommendation System lies in enhancing its predictive capabilities through advanced technologies such as deep learning, cloud integration, and IoT-based smart agriculture. By incorporating deep learning models, the system can improve its ability to analyze complex soil and environmental data, leading to more precise crop recommendations. Additionally, the integration of generative adversarial networks (GANs) can help generate synthetic soil data, which is particularly useful for regions with limited datasets. This approach allows machine learning models to be trained on a diverse range of soil conditions, improving their adaptability and effectiveness in real-world applications.

Beyond soil data analysis, integrating geospatial data and satellite imagery can provide a comprehensive perspective on soil health and environmental conditions. These technologies enable the system to assess factors such as vegetation health, land topography, and climate patterns, further refining crop recommendations. Additionally, the system has the potential to evolve into a mobile-based AI assistant for farmers, offering real-time insights and adaptive recommendations based on weather forecasts and climate trends. By making precision agriculture accessible through mobile devices, farmers can make informed decisions on the go, optimizing their agricultural practices with up-to-date information.

By combining these advanced technologies, the Crop Recommendation System can contribute to global food security, empowering farmers with AI-driven insights to enhance crop yield, reduce resource wastage, and promote environmentally sustainable farming.

CHAPTER 10

APPENDIX

App.py:

```
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import pickle
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier, ExtraTreeClassifier
from sklearn.ensemble import (
    RandomForestClassifier,
    BaggingClassifier,
    GradientBoostingClassifier,
    AdaBoostClassifier,
)
from sklearn.metrics import accuracy_score
from skopt import BayesSearchCV # Bayesian Optimization

app = Flask(__name__)
```

```

@app.route('/')
def home():
    return render_template('index.html')

# Load the dataset
crop = pd.read_csv(r"C:\Users\MYPC\OneDrive\Desktop\Crop
Recommendation\Crop_recommendation.csv")

# Map labels to numerical values
crop_dict = {
    'rice': 1, 'maize': 2, 'jute': 3, 'cotton': 4, 'coconut': 5,
    'papaya': 6, 'orange': 7, 'apple': 8, 'muskmelon': 9, 'watermelon': 10,
    'grapes': 11, 'mango': 12, 'banana': 13, 'pomegranate': 14, 'lentil': 15,
    'blackgram': 16, 'mungbean': 17, 'mothbeans': 18, 'pigeonpeas': 19,
    'kidneybeans': 20, 'chickpea': 21, 'coffee': 22
}
reverse_crop_dict = {v: k for k, v in crop_dict.items()}

crop['label'] = crop['label'].map(crop_dict)

# Split data into features and labels
X = crop.drop('label', axis=1)
y = crop['label']

# Split data into training and testing sets

```



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

```
# Apply MinMaxScaler
```

```
mx = MinMaxScaler()
```

```
X_train = mx.fit_transform(X_train)
```

```
X_test = mx.transform(X_test)
```

```
# Apply StandardScaler
```

```
sc = StandardScaler()
```

```
X_train = sc.fit_transform(X_train)
```

```
X_test = sc.transform(X_test)
```

```
# Initialize models
```

```
models = {
```

```
    'LogisticRegression': LogisticRegression(),
```

```
    'GaussianNB': GaussianNB(),
```

```
    'SVC': SVC(),
```

```
    'KNeighborsClassifier': KNeighborsClassifier(),
```

```
    'DecisionTreeClassifier': DecisionTreeClassifier(),
```

```
    'ExtraTreeClassifier': ExtraTreeClassifier(),
```

```
    'RandomForestClassifier': RandomForestClassifier(),
```

```
    'BaggingClassifier': BaggingClassifier(),
```

```
    'GradientBoostingClassifier': GradientBoostingClassifier(),
```

```
    'AdaBoostClassifier': AdaBoostClassifier(),
```

```
}
```

```

# Evaluate models
best_model_name = ""
best_accuracy = 0.0

for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    print(f"{name}: {accuracy:.4f}")
    if accuracy > best_accuracy:
        best_accuracy = accuracy
        best_model_name = name

print(f"\nBest Model: {best_model_name} with accuracy:
{best_accuracy:.4f}")

# Train the best model
best_model = models[best_model_name]

# Apply Bayesian Optimization only if the model supports hyperparameter
tuning
if best_model_name == "GaussianNB":
    print("Skipping Bayesian Optimization because GaussianNB does not require
hyperparameter tuning.")
else:
    param_space = {}

```

```

if best_model_name == "RandomForestClassifier":
    param_space = {'n_estimators': (10, 200), 'max_depth': (5, 50)}
elif best_model_name == "SVC":
    param_space = {'C': (0.1, 10), 'gamma': (0.01, 1)}
elif best_model_name == "KNeighborsClassifier":
    param_space = {'n_neighbors': (1, 20)}
elif best_model_name == "GradientBoostingClassifier":
    param_space = {'n_estimators': (50, 300), 'learning_rate': (0.01, 0.2)}

if param_space:
    opt = BayesSearchCV(best_model, param_space, n_iter=10, cv=3,
scoring='accuracy', random_state=42)
    opt.fit(X_train, y_train)
    best_model = opt.best_estimator_
    print(f"Optimized {best_model_name} with Bayesian Optimization.")

# Save the model and scalers
pickle.dump(best_model, open('model.pkl', 'wb'))
pickle.dump(mx, open('minmaxscaler.pkl', 'wb'))
pickle.dump(sc, open('standscaler.pkl', 'wb'))

# Function for crop recommendation
def recommendation(N, P, K, temperature, humidity, ph, rainfall):
    input_data = pd.DataFrame([[N, P, K, temperature, humidity, ph, rainfall]],
                                columns=X.columns)
    mx_features = mx.transform(input_data)
    sc_mx_features = sc.transform(mx_features)

```

```

prediction = best_model.predict(sc_mx_features)
return prediction[0]

@app.route('/process', methods=['POST'])
def process():
    N = float(request.form['N'])
    P = float(request.form['P'])
    K = float(request.form['K'])
    temperature = float(request.form['temperature'])
    humidity = float(request.form['humidity'])
    ph = float(request.form['ph'])
    rainfall = float(request.form['rainfall'])

    predicted_crop = recommendation(N, P, K, temperature, humidity, ph,
rainfall)

    predicted_crop_name = reverse_crop_dict.get(predicted_crop, "Unknown
Crop")

    return render_template('index.html', result=predicted_crop_name)

if __name__ == '__main__':
    import os
    os.environ['FLASK_RUN_EXTRA_FILES'] = " # Prevent Flask from
monitoring extra files

    app.run(debug=True, use_reloader=False) # Disable auto-restart

```

Index.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Crop Recommendation</title>
  <style>
    @import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@300;400;600&dis
play=swap');

    body {
      font-family: 'Poppins', sans-serif;
      background: linear-gradient(135deg, #0f0f0f, #1c1c1c);
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      margin: 0;
    }

    .form-container {
      background: rgba(30, 30, 30, 0.9);
      backdrop-filter: blur(10px);
      padding: 30px;
      border-radius: 20px;
```

```
width: 400px;
display: grid;
box-shadow: 0 10px 25px rgba(0, 0, 0, 0.7);
text-align: center;
animation: fadeIn 1s ease-in-out;
}
```

```
h2 {
  font-size: 26px;
  color: #fff;
  margin-bottom: 20px;
  animation: slideDown 0.8s ease-in-out;
}
```

```
label {
  display: block;
  font-size: 16px;
  color: #ccc;
  text-align: left;
  font-weight: 600;
  margin-top: 10px;
}
```

```
input {
  width: 100%;
  padding: 12px;
  border: none;
```

```
border-radius: 10px;
font-size: 16px;
background: rgba(50, 50, 50, 0.8);
color: #fff;
outline: none;
transition: 0.3s;
}
```

```
input::placeholder {
  color: #aaa;
}
```

```
input:focus {
  background: #333;
  transform: scale(1.05);
}
```

```
.submit-btn {
  background: linear-gradient(45deg, #ff4500, #ff6347);
  color: white;
  font-size: 18px;
  font-weight: bold;
  cursor: pointer;
  transition: 0.3s;
  border: none;
  padding: 15px;
  width: 100%;
```

```
border-radius: 12px;
margin-top: 15px;
}
```

```
.submit-btn:hover {
background: linear-gradient(45deg, #e63900, #ff5733);
transform: scale(1.05);
box-shadow: 0px 5px 15px rgba(255, 69, 0, 0.5);
}
```

```
@keyframes fadeIn {
from {
opacity: 0;
transform: translateY(30px);
}
to {
opacity: 1;
transform: translateY(0);
}
}
```

```
@keyframes slideDown {
from {
opacity: 0;
transform: translateY(-20px);
}
to {
```



```

        opacity: 1;
        transform: translateY(0);
    }
}
.min-div{
    display: flex;
}
</style>
</head>
<body>
    <div class="form-container">
        <h2>Data Collection Form</h2>
        <form action="/process" method="POST">
            <div class="min-div">
                <label for="N">N:</label>
                <input type="text" id="N" name="N" value="">

                <label for="P">P:</label>
                <input type="text" id="P" name="P" value="">

                <label for="K">K:</label>
                <input type="text" id="K" name="K" value="">
            </div>

            <label for="temperature">Temperature:</label>
            <input type="text" id="temperature" name="temperature" value="">

```

```
<label for="humidity">Humidity:</label>
```

```
<input type="text" id="humidity" name="humidity" value="">
```

```
<label for="ph">pH:</label>
```

```
<input type="text" id="ph" name="ph" value="">
```

```
<label for="rainfall">Rainfall:</label>
```

```
<input type="text" id="rainfall" name="rainfall" value="">
```

```
<h2>{{ result }}</h2>
```

```
<button type="submit" class="submit-btn">Submit</button>
```

```
</form>
```

```
</div>
```

```
</body>
```

```
</html>
```

REFERENCES

1. Motwani, A., Patil, P., Nagaria, V., Verma, S., & Ghane, S. (2023). Soil analysis and crop recommendation using machine learning. Department of Computer Engineering, Sardar Patel Institute of Technology, Mumbai, India.
2. Laishram, J., Saxena, K. G., Maikhuri, R. K., & Rao, K. S. (2021). Soil quality and soil health: A review. School of Environmental Sciences, Jawaharlal Nehru University, New Delhi, India; G.B. Pant Institute of Himalayan Environment and Development, Upper Bhaktiyana, Srinagar (Garhwal), Uttarakhand, India; Department of Botany, University of Delhi, Delhi, India.
3. Barvin, P. A., & Sampradeepraj, T. (2023). Crop recommendation systems based on soil and environmental factors using Graph Convolution Neural Network: A systematic literature review. In Proceedings of the 10th International Electronic Conference on Sensors and Applications (ECSA-10). Kalasalingam Academy of Research and Education, Srivilliputhur, Tamil Nadu, India.
4. John, A., Davis, D., Tom, A. M. C., Davis, D., & Davis, J. (2023). Soil classification and crop recommendation system. Sahrdaya College of Engineering and Technology, Thrissur, India.
5. Anguraj, K., Thiyaneswaran, B., Megashree, G., Preetha Shri, J. G., Navya, S., & Jayanthi, J. (2022). Crop recommendation on analyzing soil using machine learning. Department of ECE and Department of CSE, Sona College of Technology, Salem, India.
6. Sujata, M., & Jaidhar, C. D. (2023). Machine Learning based approaches to enhance soil fertility. Expert Systems with Applications.
7. Gang, Z., Quanying, Z., Webber, H., Johnen, A., Rossi, V., & Fernandes, A. (2023). Integrating machine learning and change detection for enhanced crop

- disease forecasting in rice farming: A multi-regional study. *European Journal of Agronomy*.
8. Bilal, A., Liu, X., Long, H., Shafiq, M., & Waqar, M. (2023). Increasing Crop Quality and Yield with a Machine Learning-Based Crop Monitoring System. *Computers, Materials and Continua*.
 9. Cheng, L., Yu, T., Jiang, Z., Li, W., Guan, D.-X., Yang, Y., & Zeng, J. (2023). Leveraging machine learning for sustainable cultivation of Zn-enriched crops in Cd-contaminated karst regions. *Science of The Total Environment*.
 10. Md. Abu Javed, & Masrah Azrifah Azmi Murad. (2023). Crop yield prediction in agriculture: A comprehensive review of machine learning and deep learning approaches, with insights for future research and sustainability. *Heliyon, Cell Press, Volume 10*
 11. Liu, J., Yang, K., Tariq, A., Lu, L., Soufan, W., & El Sabagh, A. (2023). Interaction of climate, topography and soil properties with cropland and cropping pattern using remote sensing data and machine learning methods. *Egyptian Journal of Remote Sensing and Space Sciences*
 12. Dey, B., Ferdous, J., & Ahmed, R. (2024). Machine learning based recommendation of agricultural and horticultural crop farming in India under the regime of NPK, soil pH and three climatic variables. *Heliyon*.
 13. Srivastava, R. K., Purohit, S., Alam, E., & Islam, M. K. (2024). Advancements in soil management: Optimizing crop production through interdisciplinary approaches. *Agriculture and Food Research*.
 14. Nde, R. K., Fendji, J. L. E. K., Yenke, B. O., & Schöning, J. (2024). Crop selection: A survey on factors and techniques. *Smart Agricultural Technology*.
 15. Gao, Z., Guo, D., Ryu, D., & Western, A. W. (2023). Training sample selection for robust multiyear within-season crop classification using machine learning. *Journal Computers and Electronics in Agriculture*.

16. van Klompenburg, T., Kassahun, A., & Catal, C. (2020). Crop yield prediction using machine learning. *Computers and Electronics in Agriculture*
17. Mishra, D., & Deepa, D. (2021). Automation and integration of growth monitoring in plants (with disease prediction) and crop prediction. *Materials Today: Proceedings*, 43, 3922-3927.
18. Padhiary, M., Saha, D., Kumar, R., Sethi, L. N., & Kumar, A. (2024). Enhancing precision agriculture: A comprehensive review of machine learning and AI vision applications in all-terrain vehicle for farm automation. *Smart Agricultural Technology*, 100483.
19. Malik, P., Sengupta, S., & Jadon, J. S. (2021, January). Comparative analysis of soil properties to predict fertility and crop yield using machine learning algorithms. In *2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)* (pp. 1004-1007). IEEE.
20. Manikandan, R., Ranganathan, G., & Bindhu, V. (2023). Deep learning based IoT module for smart farming in different environmental conditions. *Wireless Personal Communications*, 128(3), 1715-1732.
21. Pallathadka, H., Jawarneh, M., Sammy, F., Garchar, V., Sanchez, D. T., & Naved, M. (2022, April). A review of using artificial intelligence and machine learning in food and agriculture industry. In *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)* (pp. 2215-2218). IEEE
22. Omaye, J. D., Ogbuju, E., & colleagues. (2024). Cross-comparative review of machine learning for plant disease detection: Apple, cassava, cotton, and potato plants. *Artificial Intelligence in Agriculture*.
23. Silva, C. de O. F., Grego, C. R., Manzione, R. L., Oliveira, S. R. D. M., Rodrigues, G. C., Rodrigues, C. A. G., Speranza, E. A., Luchiari, A., & Vieira, L. (2024). Summarizing soil chemical variables into homogeneous

management zones – Case study in a specialty coffee crop. *Smart Agriculture Technology*.

24.Darwin, P., Raju, M. J., Babu, K. N., Lavanya, Y., & Sai, Y. S. (2024). Smart farming using machine learning and deep learning techniques. *International Journal of Engineering Research and Technology*.

25.Mequanenit, A. M., Ayalew, A. M., Salau, A. O., & Nibret, E. A. (2024). Prediction of mung bean production using machine learning algorithms. *Journal Heliyon*.