**A Report on**

**A Report on**

Crop Recommendation System

*Submitted for partial fulfillment of award of degree*

**BACHELOR OF TECHNOLOGY**

In

Computer Science & Engineering (Data Science)

By

Akshay Singh

(2000301540006)

Aditi Jain

(2000300100012)

Vaibhav Chauhan

(2000301530063)

Dr. Tripti Sharma

SUPERVISOR



**INDERPRASTHA ENGINEERING COLLEGE, GHAZIABAD,**

**Dr. A P J ABDUL KALAM TECHNICAL UNIVERSITY**

**LUCKNOW**

**May 2024**

**CERTIFICATE**

This is to certify that the Project Report entitled, **"Crop Recommendation System"** and submitted by **Akshay Singh (2000300100012), Aditi Jain (2000300100012), Vaibhav Chauhan (2000301530063)** in partial fulfillment of the requirement for the award of Bachelor of Technology degree in **Computer Science and Engineering (Data Science)** at **Inderprastha Engineering College, Ghaziabad** is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University/Institute for the award of any Degree or Diploma.

**Dr. Tripti Sharma**

HoD CSE (Data Science)

Inderprastha Engineering College

Ghaziabad

Date:18/05/2024

# Acknowledgement

We take this opportunity to thank our teachers and friends who helped us throughout the project.

First and foremost, we would like to thank **Dr. Tripti Sharma (HoD, Computer Science and Engineering (Data Science**)) for her constant support and guidance during the development of the project.

Akshay Singh Aditi Jain

2000301540006 2000300100012

Signature: Signature:

Vaibhav Chauhan

2000301530063

Signature:

# Declaration

*We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.*

*Signature: Signature:*

*Name: Akshay Singh Name: Aditi Jain*

*Roll No.: Roll No.:*

*Signature:*

*Name: Vaibhav Chauhan*

*Roll No.:*

*Date:18/05/2024*

# ABSTRACT

Farming, encompassing soil cultivation, crop production, and livestock management, plays a pivotal role in a nation's economic progress. With approximately 58 percent of the population relying on agriculture as their primary livelihood source, its significance cannot be overstated. Historically, farmers have predominantly employed conventional farming methods, which, while familiar, have often proven imprecise and time-consuming, hampering productivity. Precision farming, on the other hand, offers a solution by meticulously determining optimal practices for each stage of cultivation. This approach involves forecasting weather patterns, analyzing soil characteristics, recommending suitable crops, and determining precise quantities of fertilizers and pesticides. Leveraging technologies such as IoT, data mining, analytics, and machine learning, precision farming streamlines data collection, system training, and result predictions, thereby reducing manual labor and enhancing productivity. Contemporary farmers grapple with numerous challenges, including crop failures due to erratic rainfall and soil infertility, exacerbated by environmental changes. The proposed research aims to address these challenges by facilitating crop selection through Machine Learning and Deep Learning models. The proposed system leverages an ensemble of classifiers, including Naive Bayes and Random Forest, to provide comprehensive and accurate crop recommendations based on farm-specific parameters.

**LIST OF TABLES**

**TABLE NO. TITLE PAGE NO.**

Table 1. Crop\_recommendation\_table 24

Table 2. Accuracies of different models used 34

**LIST OF FIGURES**

**FIGURE NO. TITLE PAGE NO.**

Fig 3.1.1 Use-case diagram 17

Fig 4.1.1. Sequence Diagram 20

Fig 4.2.1. Activity Diagram 20

Fig 4.3.1. Data-flow diagram 21

Fig 4.4.1. E-R Diagram 22

Fig 4.5.1. Gantt Chart 23

Fig 6.1.1. User-Interface for entering values 30

Fig 6.1.2. User-Interface for getting predicted crops 31

### TABLE of CONTENTS

### CHAPTER NO. TITLE PAGE NO.

### ABSTRACT v

### LIST OF TABLES vi

### LIST OF FIGURES vii

1. **INTRODUCTION 10** 
   1. Problem Definition 10
   2. Background about the project 10
   3. Objectives of proposed system 11
   4. Feasibility Study, need and significance 11
   5. Hardware and Software required 12
2. **LITERATURE REVIEW 13**
   1. Existing System - Theoretical and methodological contributions to a particular topic of scholarly papers/ articles /books 13
   2. Proposed System 15
3. **SOFTWARE REQUIREMENT ANALYSIS 16**
   1. Functional Requirements 16
      1. Use Case diagram with proper factoring
      2. Use case descriptions
   2. Nonfunctional Requirements 18
   3. Major Modules and their functionalities 18
4. **SYSTEM ANALYSIS & DESIGN 19**
   1. Sequence diagrams 19
   2. Activity Diagrams 20
   3. DFDs of the project 21
   4. Database Design 22
      1. E-R Diagrams
   5. Gantt Chart 23
5. **IMPLEMENTATION/CORE MODULE 24**
   1. Tables – explaining all fields and their data types for data base used in project. 24
   2. Used Algorithms/Approaches for Research projects. 24
   3. Implementation of Modules/Algorithms. 28
6. **RESULTS / OUTPUTS & TESTING 29** 
   1. All user interfaces and output screens 29
   2. Design and Test Steps / Criteria 30
7. **CONCLUSIONS / RECOMMENDATIOS 34**
8. **REFERENCES 35**
9. **PLAGIARISM REPORT 36**
10. **RESEARCH PAPER PUBLICATION (PUBLISHED PAPER/PAPER STATUS WITH JOURNAL /CONFERENCE NAME) 39**
11. **APPENDICES 40**
    1. Details of software/simulator if any 40
    2. Steps to execute/run/implement the project 40
    3. Coding if any 40

**CHAPTER:1**

**INTRODUCTION**

### Problem Definition

Our project aims to develop an advanced Crop Recommendation System leveraging machine learning techniques to optimize crop yield in Indian agriculture. Agriculture, a vital sector in India, faces challenges due to farmers' difficulty in selecting the most suitable crops based on their specific soil requirements. Precision farming is the key approach, and our system seeks to enhance it through precise crop recommendations.

The system will utilize a comprehensive dataset, including soil-specific attributes from a reputed soil testing lab and general crop data. Parameters such as pH, soil colour, Nitrogen, Phosphorous, Potassium and average rainfall will be considered to create a robust foundation for accurate recommendations. The ensemble learning technique, Majority Voting, will be implemented, employing base learners like Support Vector Machine, K- Nearest Neighbors (KNN) and Random Forest.

### Background about Project

Agriculture in India plays a pivotal role in sustaining the economy and providing livelihoods for a significant portion of the population. However, the sector grapples with challenges, especially for small-scale farmers who often lack access to modern technologies and face difficulties in optimizing crop selection based on soil characteristics. The majority of Indian farms are small, with limited irrigation capabilities, contributing to lower productivity levels compared to global standards.

Precision agriculture has emerged as a transformative approach, emphasizing the precise application of inputs like fertilizers and water to maximize crop yields. In this context, the Crop Recommendation System becomes crucial, addressing the prevalent issue of suboptimal crop choices that hinder productivity.

The proposed system builds on the foundation of precision agriculture, utilizing machine learning techniques to provide accurate and efficient crop recommendations. By integrating soil-specific attributes and leveraging ensemble learning methods, the system aims to empower farmers with actionable insights for better decision-making. The initiative aligns with the broader national goal of enhancing agricultural productivity, ensuring food security, and promoting sustainable farming practices. As India seeks to modernize its agricultural sector, the Crop Recommendation System represents a technological leap forward, offering tailored solutions to small and marginal farmers and contributing to the overall growth and resilience of the agriculture industry.

### Objectives

* + - Data set is collected from Kaggle Website. (https://[www.kaggle.com/code/niteshhalai/crop-recommendation-dataset)](http://www.kaggle.com/code/niteshhalai/crop-recommendation-dataset))
    - Data cleaning and pre-processing techniques are applied to convert the raw data into processing data.
    - The data collected is subject to machine learning system and deep learning system that makes an efficient crop recommendation system.
    - Usage of Ensemble of classifiers makes the model more robust and efficient.
    - Ranking technique used in the project helps us to make efficient decisions.
    - Creating a web application for crop prediction.
    - The main objective is to obtain a better variety of crops that can be grown over the season. The proposed system would help to minimize the difficulties faced by farmers in choosing a crop and maximize the yield.
    - The model predicts the crop yield by studying factors such as rainfall, temperature, humidity etc.

### 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

###### 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.

###### TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

###### SOCIAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### Hardware and Software required:

###### HARDWARE REQUIREMENTS:

* + - System: Pentium Dual Core.
    - Hard Disk: 40 GB.
    - Monitor: 15’ LED
    - Ram: 4 GB

###### SOFTWARE REQUIREMENTS:

* Operating system: Windows 7/10.
* Coding Language: Python version >= 3.8
* Database: SQLite.

## CHAPTER:2

## LITERATURE SURVEY

### Existing System

India is a country where a majority rely on agriculture as their occupation. The repeated cultivation of the same crops without considering soil conditions and the use of same fertilizers lead to reduced crop yields, soil acidification, and top-layer damage. To address these issues, we need to use automated means of predicting the best crops for cultivation for a particular area and region. By utilizing machine learning algorithms, we can help farmers in making informed decisions.

In addition to suggesting the most suitable crop for a specific piece of land, the paper provided information on the required content and quantity of fertilizers, as well as the necessary seeds for cultivation. Machine learning algorithms such as SVM were used for rainfall prediction and Decision Tree for crop prediction. The system further recommends the most suitable crop, provides information on required nutrients and seeds, and displays the market price and approximate yield for the recommended crop [1].

Precise Farming uses advanced technologies such as IOT, Data Mining, Data Analytics, Machine Learning to collect the data, train the systems and predict the results. With the help of technologies Precise farming helps to reduce manual labor and increase productivity. It guides an individual for smart farming.

Around 10 different classification algorithms to find the best model for future prediction. Hyperparameter tuning was applied to these algorithms provide more accuracy. The Random Forest model hyper tuned with Randomized CV was selected as the best model [2].

For most developing countries, agriculture is their primary source of revenue. Modern agriculture is a constantly growing approach for agricultural advances and farming techniques. It becomes challenging for the farmers to satisfy our planet's evolving requirements and the expectations of merchants, customers, etc. [3]. It compares various supervised learning algorithms like KNN, Decision Tree, and Random Forest. It uses Decision Tree and Random Forest Classifier and evaluates the model's performance under two criterions- Entropy and Gini Index. It concluded with Random Forest having the highest accuracy among the three.

Modern agriculture relies heavily on advanced technologies such as sensors, robots, and GPS, revolutionizing farming practices for increased profitability, efficiency, and sustainability. These technologies, including remote sensors and cameras, gather continuous data on soil health, plant conditions, and environmental factors, aiding farmers in making informed decisions. Integration of Arduino microcontrollers and machine learning algorithms like Naïve Bayes and Support Vector Machine further enhances data processing and decision-making accuracy, with a success rate exceeding 95%. This technology holds promise for both rural and urban areas, offering cost-effective and autonomous solutions for agricultural management in Sri Lanka [4].

The agricultural industry is a huge source of wealth for the country. However, when compared to other agricultural products, the yield per hectare is disappointing. There are numerous reasons why marginal farmers in India have a higher suicide rate. The work [5] connects Farmers through a smartphone app. GPS technology aid in user identification and location. The user specifies the area and type of soil in which they want to work, and machine learning algorithms enable the selection of the most profitable user-selected crop yield prediction or crop list. Support Vector Machine (SVM), Artificial Neural Network (ANN), Random Forest (RF), Multivariate Linear Network (MLN), and a combination of regression and KNN are used to estimate crop yields. The Random Forest produced the best results of the three. The rate of accuracy is 95%. Aside from that, the system recommends the best options available.

India is characterized by small farms. Over 75% of total land capitals within the country are less than 5 acres. Most crops are rain nourished, with just about 45% of the land irrigated. As per some estimations, about 55% of total population of India depends on farming. In the US, because of heavy mechanization of agriculture, it is about 5%. India is one of the biggest producers of agricultural products and still has very less farm productivity. Productivity needs to be increased so that farmers can get more pay from the same piece of land with less labor. Precision agriculture provides a way to do it. Ensembling is one such approach. The ensemble technique employed is Majority Voting, with base learners including Support Vector Machine, Naïve Bayes, Multi-layer Perceptron, and Random Forest [6].

#### DISADVANTAGES OF EXISTING SYSTEM

* Efficiency is low.
* The existing system which recommends crop yield is either hardware-based being costly to maintain, or not easily accessible.
* Despite many solutions that have been recently proposed, there are still open challenges in creating a user-friendly application with respect to crop recommendation.
* More number of repeated works.

### Proposed System

In proposed system, the data analysis technology is used to update the crop yield rate change. The concept of this paper is to implement the crop selection method so that this method helps in solving many agriculture and farmers problems. The main thing that sets the paper apart from the existing work is that it does not take pH into account for making the predictions. Upon applying data visualization techniques, it was found that the pH parameter has a very less correlation. Removing pH from the training dataset has no effect on the accuracy achieved by including it. Hence, the proposed system does not take the pH value from the user. The proposed work also takes the humidity and the temperature parameter automatically with the help of the user’s current location. It uses the JavaScript geolocation API to first get the user’s latitude and longitude and then uses the Open Weather Map API to find the temperature and humidity based on that. We have used 10 Machine learning and 4 Deep learning algorithms. The usage of ensemble of classifiers paves a path way to make a better decision on predictions due to the usage of multiple classifiers. Further, a ranking process is applied for decision making in order to select the classifiers results. This project uses Ensemble of classifiers such as Naïve Bayes and Random Forest classifier.

###### ADVANTAGES OF PROPOSED SYSTEM

* + - Useful to people far away from towns/cities.
    - Better time efficiency.
    - Reduction of repeated work.

## CHAPTER:3

## SOFTWARE REQUIREMENTS ANALYSIS

### 3.1 Functional Requirements

Functional requirements define the specific behaviors and functionalities that the

crop recommendation system must exhibit to fulfill its intended purpose effectively.

These requirements outline the system's capabilities and interactions with users and

external components. The functional requirements of the crop recommendation system

encompass various aspects, including user interactions, data processing,

recommendation generation, and system maintenance.

**USE CASE DIAGRAM**

Use case diagrams overview the usage requirement for system. They are useful for presentations to management and/or project stakeholders, but for actual development you will find that use cases provide significantly more value because they describe “the meant” of the actual requirements. A use case describes a sequence of actions that provides something of measurable value to an action and is drawn as a horizontal ellipse.

The Use Case Diagram illustrates the primary interactions between users (Farmers) and the Crop Recommendation System. Here's a detailed description of the key components:

###### Actors:

* + 1. **Farmer:** The primary actor representing users who interact with the Crop

Recommendation System. Farmers play a central role in providing input data and receiving recommendations.

###### Use Cases:

1. **Input Soil Data:**
   * + **Description***:* The Farmer can input essential soil-related attributes such as pH, NPK values, humidity, temperature and average rainfall in his area into the system.
     + **Purpose***:* This allows the system to gather crucial information about the soil, forming the basis for accurate crop recommendations.

###### View Recommended Crop:

* + - **Description***:* The system processes the provided soil data using machine learning models and algorithms to recommend suitable crops.
    - **Purpose***:* Farmers can view the recommended crops tailored to their specific soil conditions, enhancing informed decision-making.

###### Provide Feedback:

* + - * **Description***:* Farmers have the option to provide feedback on the recommended crops, sharing insights that can contribute to system improvement.
      * **Purpose***:* Continuous feedback helps refine the system's accuracy and ensures it adapts to the dynamic nature of agriculture.

###### Relationships:

* **Uses Relationship (Association):** Farmers (actor) are associated with each use case, indicating their direct involvement in inputting data, receiving recommendations, providing feedback, and accessing historical data.
* **Association between Use Cases and System:** Each use case is connected to the Crop Recommendation System, illustrating the functionalities facilitated by the system.

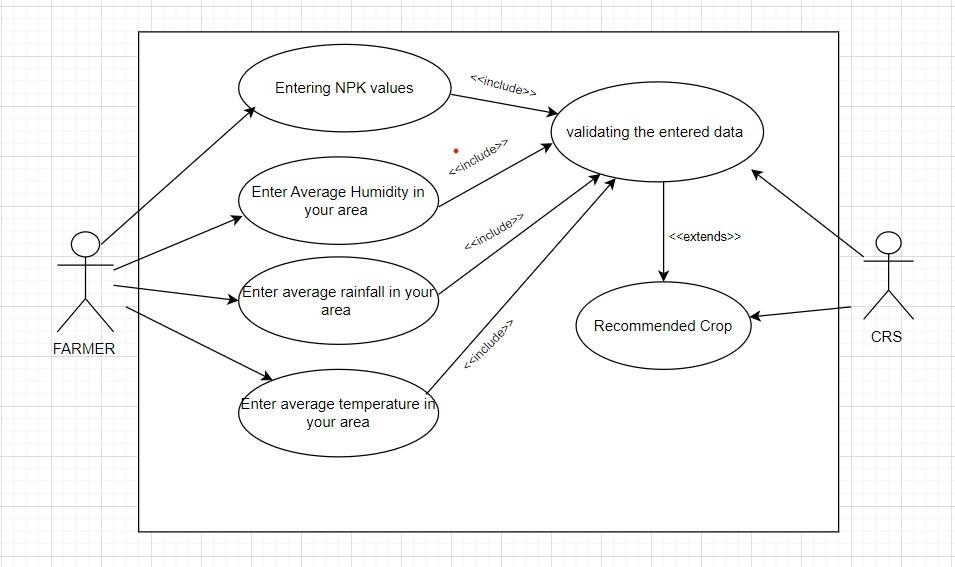


Fig 3.1.1: Use Case Diagram

### 3.2 Non-Functional Requirements

### Non-functional requirements are essential aspects that govern the performance, usability, security, and scalability of the crop recommendation system. Performance requirements dictate that the system should provide timely recommendations, with response times not exceeding a specified threshold, even during peak usage periods. Usability requirements ensure that the user interface is intuitive and accessible, catering to users with varying levels of technical proficiency. Security requirements mandate that user data be encrypted during transmission and storage to prevent unauthorized access or data breaches. Additionally, scalability requirements dictate that the system should be able to handle a growing user base and increasing data volumes without significant degradation in performance. These non-functional requirements are crucial for ensuring the effectiveness, reliability, and user satisfaction of the crop recommendation system.

### 3.3 Major Modules and their functionalities

* Data Collection and Preprocessing Module
* Crop Recommendation Engine
* User Interface Module

MODULES DESCRIPTION:

**Data Collection and Preprocessing Module:**

Responsible for gathering relevant data such as soil quality, climate conditions, and historical crop yields from external sources. Preprocesses the collected data to ensure accuracy and consistency, including cleaning, normalization, and transformation as necessary.

**Crop Recommendation Engine:**

Utilizes advanced algorithms and machine learning techniques to analyze input data and generate personalized crop recommendations. Considers factors such as soil type, climate, precipitation, and historical yield data to provide tailored recommendations to users.

**User Interface Module:**

Provides an intuitive interface through which users interact with the system. Allows

users to input location-specific data, view recommended crops, provide feedback,

and adjust preferences.

## 

## CHAPTER :4

## SYSTEM ANALYSIS AND DESIGN

### 4.1 SEQUENCE DIAGRAM:

A sequence diagram is a type of interaction diagram that illustrates how objects interact in a particular scenario of a system. It depicts the sequence of messages exchanged between objects over time to achieve a specific task or behavior.

In a sequence diagram:

* + 1. **Objects or Actors:** The entities involved in the interaction are represented as vertical lines (also known as lifelines) at the top of the diagram. These could be objects of a software system, actors, or components.
    2. **Messages:** Horizontal arrows between lifelines represent messages exchanged between objects. These messages denote communication or interaction between objects. Messages can be synchronous, asynchronous, or self-messages, depending on the timing and nature of the interaction.
    3. **Activation Bars:** Activation bars, depicted as boxes along the lifelines, represent the period during which an object is active and processing a message. They show the duration of time an object is busy processing a particular message or operation.
    4. **Optional Elements:** Sequence diagrams may also include various optional elements such as loops, conditions, parallel executions, and exceptions, depending on the complexity of the interaction being modeled. Sequence diagrams are particularly useful for visualizing the dynamic behavior of a system, especially in scenarios where multiple objects collaborate to accomplish a task. They help stakeholders understand the flow of control and data between different components of the system, aiding in system design, analysis, and communication among team members.

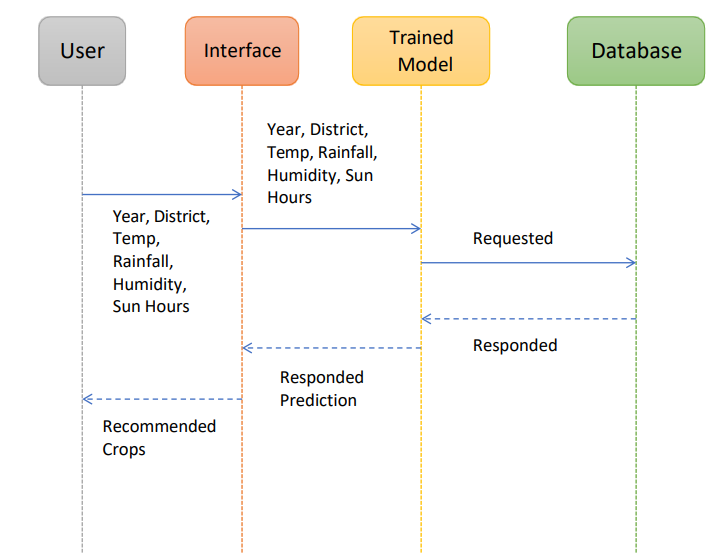


Fig 4.1.1 : Sequence Diagram

### 4.2 ACTIVITY DIAGRAM:

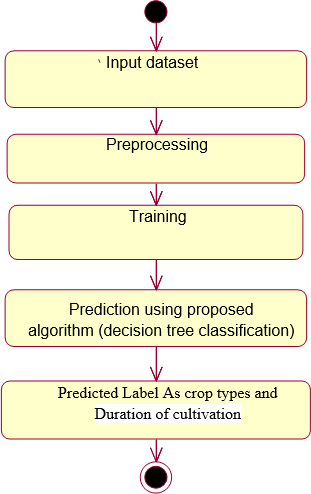
Activity diagram are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. The activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. Activity diagram consist of Initial node, activity final node and activities in between.

Fig 4.2.1: Activity Diagram

### 4.3 DATA FLOW DIAGRAM:

* + 1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system,
       1. various processing carried out on this data, and the output data is generated by this system.
    2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the

system and the information flows in the system.

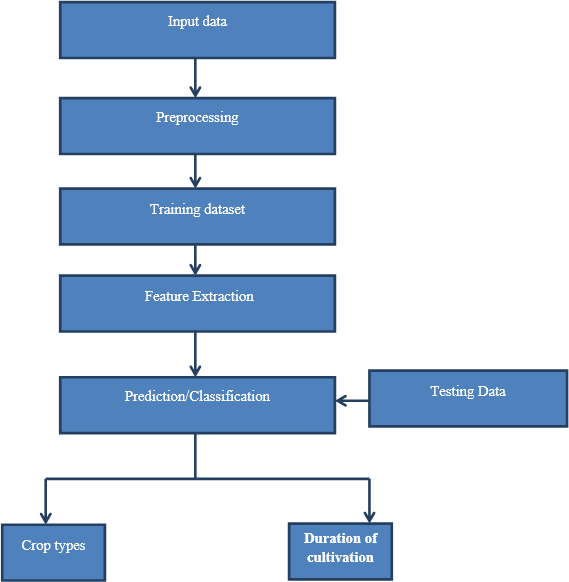
* + 1. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
    2. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

Fig 4.3.1 Data flow Diagram

### 4.4 DATABASE DESIGN:

**4.4.1 E-R DIAGRAM:**

An Entity-Relationship (ER) diagram is ma graphical representation used to model the data structure of a database. It visually depicts the relationships between data entities, helping in the design and understanding of database architecture.

**Key components of E-R Diagram:**

* + 1. **Entities:** Represented by rectangles, entities are objects or concepts that store data. Examples include "User," "Crop," "Soil," and "Climate."
    2. **Attributes:** Represented by ovals connected to their respective entities, attributes are properties or characteristics of entities.
    3. **Relationships:** Represented by diamonds, relationships illustrate how entities interact with each other.
    4. **Primary Keys:** Underlined attributes within entities that uniquely identify each entity instance.
    5. **Foreign Keys:** Attributes that create a link between entities, usually depicted within the entity but referencing a primary key from another entity.

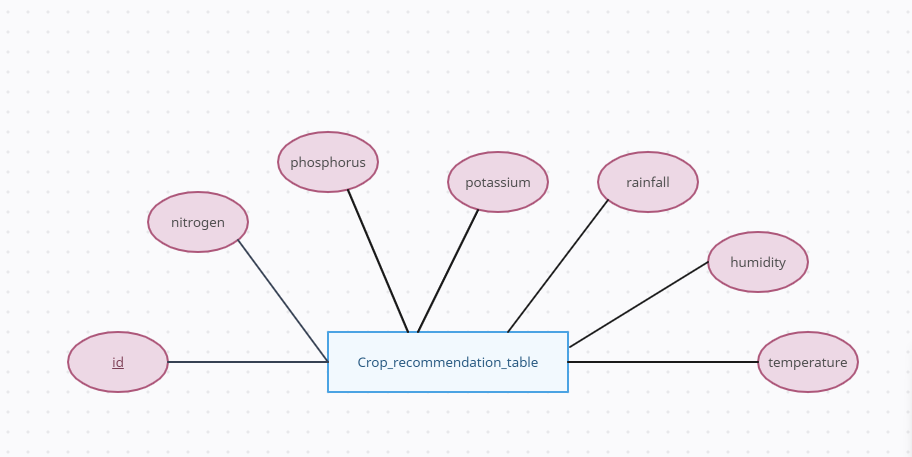


Fig 4.4.1: E-R Diagram

### 4.5 GANTT CHART :

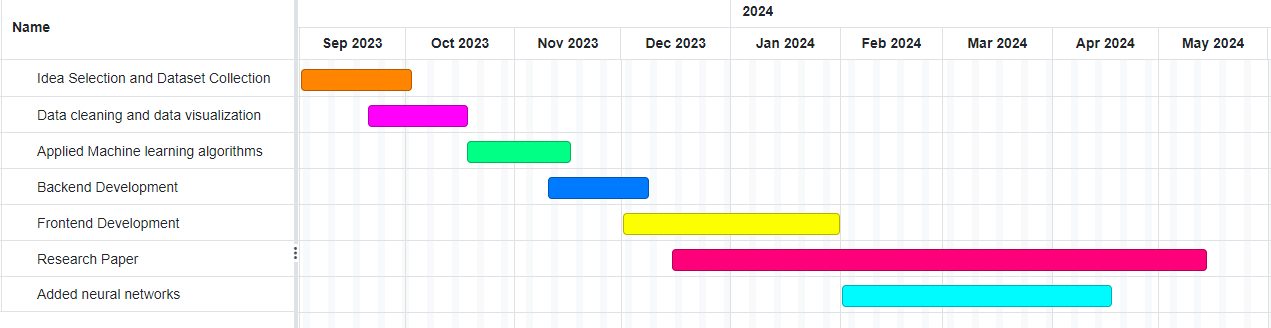


Fig 4.5.1 : Gantt Chart

**CHAPTER 5**

**IMPLEMENTATION / CORE MODULE**

### 5.1 TABLES:

For database, we are using SQLite. We are storing the data collected from the user along with

the crop predicted by the model for further analysis.

|  |  |  |
| --- | --- | --- |
| S.NO | FIELD | DATA TYPE |
| 1. | temperature | INT |
| 2. | humidity | INT |
| 3. | rainfall | INT |
| 4. | nitrogen | INT |
| 5. | phosphorus | INT |
| 6. | potassium | INT |
| 7. | predicted\_crop | STRING |

### TABLE 1: Crop\_recommendation table

### 5.2 Used Algorithms/Approaches for projects

###### RANDOM FOREST ALGORITHM:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning**,** which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, ***"***Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."

Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points.

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

###### DECISION TREE:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes 18 represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node**.

Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

The complete process can be better understood using the below algorithm:

**Step-1:** Begin the tree with the root node, says S, which contains the complete dataset.

**Step-2:** Find the best attribute in the dataset using Attribute Selection Measure (ASM).

**Step-3:** Divide the S into subsets that contains possible values for the best attributes.

**Step-4:** Generate the decision tree node, which contains the best attribute.

**Step-5:** Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

###### NAIVE BAYES ALGORITHM

Naive Bayes is a probabilistic machine learning algorithm based on Bayes' theorem, which calculates the probability of a hypothesis given the observed evidence. Despite its simplicity, Naive Bayes is widely used for classification tasks, particularly in natural

language processing and spam filtering. The algorithm operates on the assumption of independence between features, hence the "naive" descriptor. This simplifies calculations, making it computationally efficient.

The basic steps of Naive Bayes involve:

* + - 1. **Data Preprocessing:** Gather and prepare the training dataset, ensuring it is representative of the problem. Clean and preprocess the data, handling missing values and irrelevant features.
      2. **Feature Extraction:** Identify relevant features that contribute to the classification task. Calculate the likelihood of each feature given the class labels.
      3. **Training:** Calculate the prior probabilities of each class. Use the Bayes' theorem to compute the posterior probabilities based on the observed features.
      4. **Prediction:** Apply the trained model to new data by calculating the probability of each class given the observed features. Assign the class with the highest probability as the predicted class.

Naive Bayes is efficient for large datasets and is particularly effective when the independence assumption holds true. However, its simplicity may result in suboptimal performance for more complex relationships within the data. Despite this, Naive Bayes remains a popular choice due to its ease of implementation and computational efficiency.

Despite its simplicity and efficiency, Naive Bayes may struggle when faced with highly correlated features or when the independence assumption is significantly violated.

However, its ease of implementation and effectiveness in various contexts have solidified its place in the machine learning landscape as a reliable and efficient algorithm for classification tasks.

###### VOTING CLASSIFIER

A voting classifier in machine learning is an ensemble method that combines multiple individual classifiers to make predictions. This technique leverages the wisdom of the crowd, where each classifier acts as a "voter" and contributes to the final decision. The idea is that by aggregating the opinions of multiple classifiers, the overall performance can be more robust and accurate than that of any individual classifier.

There are two main types of voting classifiers: hard voting and soft voting. In hard voting, each classifier votes for a class, and the class that receives the majority of votes is chosen as the final prediction. Soft voting, on the other hand, involves combining the predicted probabilities of each classifier and selecting the class with the highest average probability.

Voting classifiers are versatile and can be constructed using a variety of base classifiers, such as decision trees, support vector machines, or k-nearest neighbors. They are particularly useful when

dealing with diverse and complementary models. This ensemble approach helps mitigate the impact of individual model weaknesses and enhances overall predictive performance. Voting classifiers are employed in various machine learning tasks, including classification and regression, contributing to the robustness and accuracy of predictive models in diverse applications.

* + 1. **DEEP LEARNING**

Deep learning is a subset of machine learning that utilizes neural networks with multiple layers to learn complex patterns in data. It's characterized by its ability to automatically learn hierarchical feature representations from raw data, eliminating the need for manual feature engineering. Deep learning has achieved remarkable success in various domains, including image recognition, natural language processing, and speech recognition.

* + - 1. **Feed Forward Neural Networks (FNN):** Feed Forward Neural Networks, also known as Multi-Layer Perceptrons (MLP), are the simplest form of neural networks. They consist of an input layer, one or more hidden layers, and an output layer. Each neuron in a layer is connected to all neurons in the subsequent layer. FNNs process input data sequentially through the layers without any feedback loops, making them suitable for tasks like classification and regression.
      2. **Recurrent Neural Networks (RNN):** Recurrent Neural Networks are designed to handle sequential data by introducing feedback loops that allow information to persist over time. Each neuron in an RNN receives input not only from the current time step but also from the previous time step. This recurrent connection enables RNNs to capture temporal dependencies in data, making them suitable for tasks like time series prediction, machine translation, and speech recognition.
      3. **Gated Recurrent Unit (GRU):** Gated Recurrent Unit is a variant of RNNs designed to address the vanishing gradient problem, which can occur during training of traditional RNNs. GRUs use gating mechanisms to selectively update and reset their hidden state based on the current input and the previous hidden state. This allows GRUs to capture long-range dependencies more effectively compared to standard RNNs while being computationally efficient.
      4. **Convolutional Neural Networks (CNN):** Convolutional Neural Networks are primarily used for processing grid-like data, such as images and videos. CNNs leverage convolutional layers, which apply filters (kernels) to input data to extract local features. Pooling layers are then used to reduce spatial dimensions and extract the most relevant features. CNNs are widely used in computer vision tasks, including image classification, object detection, and image segmentation, due to their ability to capture spatial hierarchies of features.

### 5.3 Implementation of Modules/Algorithms

* Frontend Module - Created Frontend using HTML, CSS and Javascript.
* Backend Module - Created Backend using Python and Django.
* Crop Recommendation Module – Used python libraries like Scikit-learn, Matplotlib, Pandas, Numpy and Tensorflow etc.

### CHAPTER: 6

### RESULTS / OUTPUTS & TESTING

### 6.1 ALL USER INTERFACE AND OUTPUT SCREEN

In this project, various classifiers are used to obtain the recommendation. Further, a ranking process is applied for decision making in order to select the best classifiers results.

### Screenshot 1:

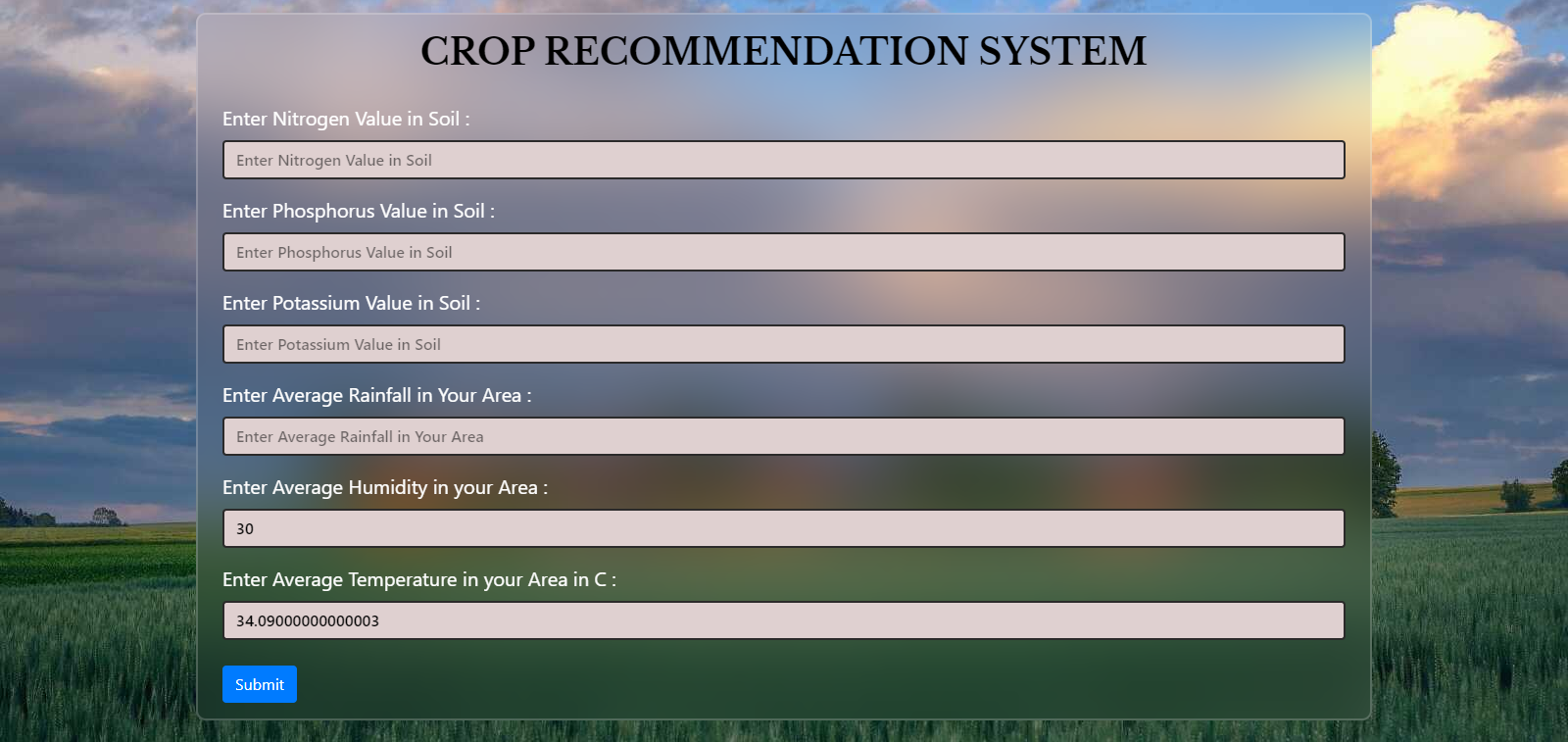
****

Fig 6.1.1: User Interface for entering values

**Screenshot 2:**

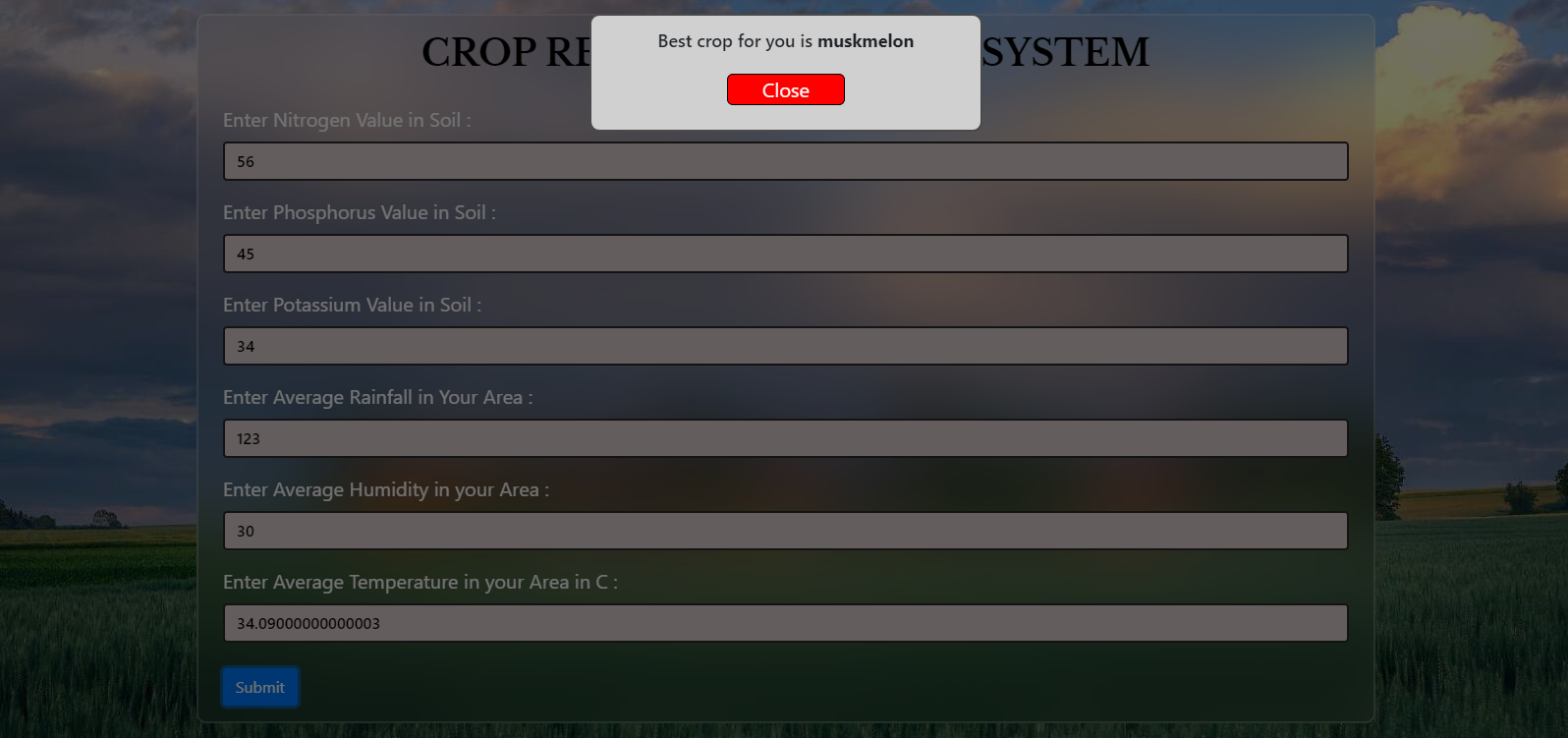
****

Fig 6.1.2: User Interface for getting predicted crop

## 6.2 Design and Test Steps / Criteria

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, subassemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

### TYPES OF TESTS:

**Unit testing:**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration.

This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific

business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### Integration testing:

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### Functional testing:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

### System Testing:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### White Box Testing:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at

least its purpose. It is purpose. It is used to test areas that cannot be reached froma black box level.

### Black Box Testing:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

###### Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

###### Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

###### Test objectives:

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

###### Features to be tested:

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page

### Integration Testing:

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

### Acceptance Testing:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**CHAPTER :7**

**CONCLUSION**

##### This project proposes a novel intelligent system for agricultural crop prediction by using less parameters. For this, the project leverages a bunch of machine learning and deep learning algorithms. Machine learning algorithms – Random Forest and Decision Tree produce the highest accuracy. The key idea is to use ensemble of classifiers for prediction. The usage of ensemble of classifiers paves a path way to make a better decision on predictions due to the usage of multiple classifiers. Further, a ranking process is applied for decision making in order to select the classifiers results.

|  |  |  |
| --- | --- | --- |
| S. No. | Algorithm Used | Accuracy |
| 1. | Logistic Regression | 94% |
| 2. | K-Nearest-Neighbours | 97% |
| 3. | Decision Tree | 98% |
| 4. | Random Forest | 99% |
| 5. | Naïve Bayes | 99% |
| 6. | Support Vector Machine | 95% |
| 7. | Bagging | 98% |
| 8. | Ada Boost | 14% |
| 9. | Gradient Boosting | 98% |
| 10. | Cat Boost | 98% |
| 11. | CNN | 95% |
| 12. | Feed Forward Network | 95% |
| 13. | RNN | 95% |
| 14. | Gated Recurrent Network | 94% |

Table 2. Accuracies of different algorithms used

The solution will benefit farmers to maximize productivity in agriculture, reduce soil degradation in cultivated fields, and reduce fertilizer use in crop production by recommending the right crop by considering various attributes.

This would provide a comprehensive prediction on the basis of geographical, environmental and economic aspects.

**REFERENCES**

##### Assistant Prof Nischitha K, Dhanush Vishwakarma, Mahendra N, Ashwini, Manjuraju M.R “Crop Prediction using Machine Learning Approaches”

1. Senthil Kumar Swami Durai, Mary Divya Shamili “Smart Farming using Machine Learning and Deep Learning Techniques.”

##### Madhuri Shripathi Rao, Arushi Singh, N.V Subba Reddy, Dinesh U Acharya “Crop Prediction using Machine Learning.”

1. Prof. Pradeepa Bandara, Assistant Prof. Thilini Weerasooriya, Ruchirawya T.H., W.J.M. Nanayakkara, Dimantha M.A.C, Pabasara M.G.P “Crop Recommendation System”

##### Rohini Jadhav, Dr. Pawan Bhaladhare “A Machine Learning Based Crop Recommendation System: A Survey.”

1. Rohit Kumar Rajak, Ankit Pawar, Mitalee Pendke, Pooja Shinde, Suresh Rathod, Avinash Devare “Crop Recommendation System to Maximize Crop Yield using Machine Learning Technology”

**PLAGIARISM REPORT**



Plagiarism Checker X Originality Report

**Similarity Found: 3%**

Date: Saturday, May 18, 2024 Statistics: 25 words Plagiarized / 785 Total words

Remarks: Low Plagiarism Detected - Your Document needs Optional Improvement.

C RECM SYST Adi in,sy ing Vav haun B. Deparme oCoerSncand nging Iprha nging lleg ahiba zbad,Iia Dr.KuKu Pfer Dr.iptSma Pfer Deparme oCoerSncand nging Iprha nging lleg ahiba zbad,Iia Absact - Fing, ncasing oil cuivaion, op talivesoc managemela a paion's economprsWitapprimaly perceof popuion lon agricu tir pry liveource, itssnificae nnot e tated. Hticlly, ve omnt empcntl ing et, wh,wfam ve n ove imprece nd im - cumhamper producty.

recion ing, tot haoffersa olutbmeiculy determ ngoptl act eah tage of lttT oacinvolves forectwtpansaly s ing oil charert, rommnduab cr, and termprece ntiesof fertersapesicesLeing tecies sh ining, anayt, macleprecion farmslinesdata ollectstem traareslt icion, hereb reducmaalab u r nd nhaing producty. onteorarfarmersgrapp wh er lle esincing op fa to ic infaas inferilitexaed y nvirnta chaesTprosresh imsto addres e hangesbfactcr sctTprosstle n ensle cs, incing a Ba nd ndFt, o ovide comprheive nd e recommet ed farm - sific paramet.

Keywor – Pion icuCrop recommet St(CRS , achine Lear S or achine ( , D ion ee, ndFt, itrn - Pphorus - Pasiu (NKK - Nt nehb(KN , Convolutl eura Nor CN), Nl Nor ( RN Fore,t optof precion farmtehniquesfailitatbtCRS holdsprisfor oming usinab agricul tesBeling rgeted inputsareducwtage, erscan mizeonmeaimpacw maxiztput, cibuing t - t erm agricul iliee nd sit T Crop ommndatStfor agricust s cropsfor iases bason y parame, lud Temperature, all, umitNoge PsaPphorusva. ting c lasifier maclear alear ait, he eceml desned o ovide op omndat bason leparersThis tecy - drn icuasion enha ' op ld y recommeing sitab sao tir lacidergehic nd limaic facorsThe proseemb ml proveetin uggesing iate cropswh osive licat guing agricul oductainform farm ab mart rat cnges emphasesimpnta slect mettaddres lturaclleng, contring o ia'secic owh y optizcrop ld os iffere laondions II.LITERE URVEY Ind cntrwe a jory ly agricuastir cupat he eated cuivaion tscropswhout

consing oil it nd he e sfertersletrcryie, saidion, t - ladamTo addres e s eswneed o e automatnsicttt cr for ltion a icuarea nd region.Buting clear ait, e hefarm ma informis.

In ittsttmt ule crop ora pec iecof nd, tpaper provide d informion the ired onte and aitof iliz, east necsy eedsfor lttMhine learaorhmssasSMwus for infapredtandDisTree crop icion. he ysfurt recomme tosule cres informion requnutrntsas, and isla he ket ice nd approxyiefor omnded cr Around 10 different classification algorithms to find the best model for future prediction. Hyperparameter tuning was applied to these algorithms provide more accuracy. rob, ndGSrevolutingfarm practesfor easprofitaby, efficnc nd usinaby.

hes teciesincing msns camerasgacinuousdaon oil heah, lacit, nd nvirnta facorsaingfarmersin kinginformed decionsIntegratof ino mocollersamaclear ait aïByesa upport Vector acfurtencesdata procesing nddecion - maaccuray, wh s ces excing T tecy isfor otruraa urban easofferct - eta autonom olut agrltura managemein Si La The icuaindusy huge ourcof wh tcntrHe ver, he comptotagrlturapr, he yieper e isointTe numer onswminafar Indve a higuide rahe wk [ 5 Aroving he eter om he datashe macning mlsuc Logt Rs Nve Ba,S Vector Mhinesere imp F K eras la uch , axP, Ftteee, LS etc w imp Around ten machine learning algorithms were evaluated for crop recommendation, including Logistic Regression, K-Nearest Neighbours, Decision Tree, Random Forest, Naive Bayes, Support Vector Machine, Bagging, AdaBoost, Gradient Boosting, and various Deep learning models like Feed Forward Networks, CNN, RNN, and Gated Recurrent Networks (GRN) were also used. Comparison b/w different algorithms used S. No. Algorithm Used Accuracy 1.

Logistic Regression 94% 2. K-Nearest- Neighbours 97% 3. Decision Tree 98% 4. Random

Forest 99% 5. Naïve Bayes 99% 6. Support Vector Machine 95% 7. Bagging 98% 8. Ada

Boost 14% 9. Gradient Boosting 98% 10. Cat Boost 98% 11. CNN 95% 12. Feed Forward

Network 95% 13. RNN 95% 14. Gated Recurrent Network 94% Naïve Bayes and Random Forest achieved an accuracy of higher than 99%. So, they were selected as the base learners for the proposed ensemble model. V. RES The proswprentsa eb applicaion hat ir t hnica knowtoperatstfarmersca easy e tget he t ule op y enterluesP and rall. She stem omaiclly hesthumit and m perature met ed t us hicalocatleing Jcrt olocatAPa pen WeatM I.

Wn he er lic the ubm on, hegtbes recomme op the luese aslert he w e. The proposed work uses an ensemble of Naïve Bayes and Random Forest by employing the voting classifier strategy. The proposed work achieved an accuracy of 99.54%. VI I .REFER Crop redion ing ahine ning Approac, TERNIOAL ORN O GEERIN EARCH& TECHOG(IJ Volume09, s 08 (Augus [2] Shil Kwi Di, My D Sili, mart ing ing ac LearandDLning hniqu,

Dion AlicsJna me et J. Phys Conf. Ser. Rajak, Rohit Kumar, Ankit Pawar, Mitalee Pendke, Pooja Shinde, Suresh Rathod, and Avinash Devare. "Crop recommendation system to maximize crop yield using machine learning technique." International Research Journal of Engineering and Technology 4, no.

12 (2017): 950-953. INTERNET SOURCES:

2% -

https:[//w](http://www.researchgate.net/publication/352375978_Crop_Recommendation_System_u)ww[.researchgate.net/publication/352375978\_Crop\_Recommendation\_System\_u](http://www.researchgate.net/publication/352375978_Crop_Recommendation_System_u) sing\_Machine\_Learning

4% -

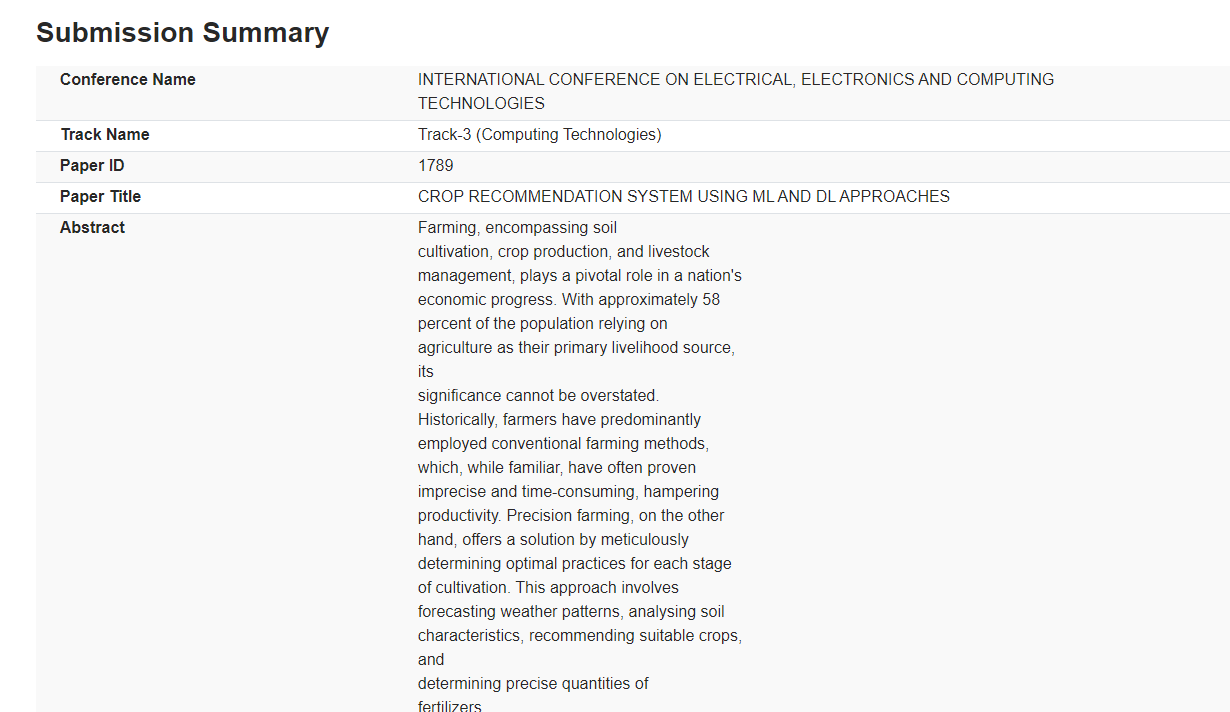
https:[//w](http://www.academia.edu/49636443/Crop_Recommendation_System_using_Machine_L)ww[.academia.edu/49636443/Crop\_Recommendation\_System\_using\_Machine\_L](http://www.academia.edu/49636443/Crop_Recommendation_System_using_Machine_L) earning

**RESEARCH PAPER PUBLICATION**

We have submitted a review paper (A review on Crop Recommendation System) to the IPEC JST and the research paper (Crop Recommendation System using ML and DL approaches) to the ICEECT-2024 conference. Our submission is under review.

* **IPEC JOURNAL:** The goal of the Journal is to provide a platform for researchers and practitioners to share their knowledge and research in the field of current science and technological applications and their advancement.

# INTERNATIONAL CONFERENCE ON ELECTRICAL, ELECTRONICS AND COMPUTING TECHNOLOGIES (ICEECT-2024): The ICEECT-2024 organized by Department of Electrical Electronics and Communication Engineering, Sharda University, India. The conference is technically sponsored by IEEE Uttar Pradesh Section, India. Uttar Pradesh Section is located in Region 10 and is represented at the India Council. IEEE UP Section interfaces with industries and academia through various technical and humanitarian activities. Conference Proceedings will be abstracted and indexed by IEEE Xplore.



**APPENDICES**

### Details of Software/Simulator:

**VISUAL STUDIO**

Visual Studio Code is a lightweight but powerful source code editor which runs on your desktop and is available for Windows, macOS and Linux. It comes with built-in support for JavaScript, TypeScript and Node.js and has a rich ecosystem of extensions for other languages and runtimes (such as C++, C#, Java, Python, PHP, Go, .NET).

**Steps to execute/run/implement:**

Step 1: Create a new virtual env by using virtualenv module.

Step 2: Install all the required modules by using the command pip install -r “requirements.txt”.

Step 3: Go to CRS\_frontend folder and start the live server.

### Coding:

**MACHINE LEARNING**

# import libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

import warnings warnings.filterwarnings('ignore')

# load dataset and take overview

dataset = pd.read\_csv("../Datasets/Crop\_Recommendation\_Dataset.csv") dataset.head()

# View basic information related to dataset

dataset.info()

# find duplicate rows

series = dataset.duplicated() series[series == True]

# View duplicate rows

dataset.loc[[105,176,274,874,1390]]

# Remove duplicate rows

dataset.drop([105,176,274,874,1390],axis=0,inplace=True)

# count of duplicate rows to cross check that all duplicates has been removed

dataset.duplicated().sum()

# count of null values in every column

col = dataset.columns col = list(col)

for j in col:

print(f"{j} => {dataset[j].isnull().sum()}")

# Drop all rows where any value is null, As number of rows are not much so we can drop them

dataset.dropna(inplace=True)

for j in col:

print(f"{j} => {dataset[j].isnull().sum()}# View columns present

list(dataset.columns)

# Find crops that are present as label

unique\_crops = dataset['label'].unique() unique\_crops = list(unique\_crops)

# Find count of datapoints for every crop

dataset['label'].value\_counts()

# plot counts of crop

plt.figure(figsize=(25,8))

sns.countplot(data=dataset,x='label')

# effect of Nitrogen on crop

plt.figure(figsize=(15,8))

plt.axes(xlabel="Nitrogen", ylabel="Crop")

sns.scatterplot(data=dataset, x='N', y='label', hue='label')

# effect of Phosphorus on crop

plt.figure(figsize=(15,12))

plt.axes(xlabel="Phosphorus", ylabel="Crop") sns.scatterplot(data=dataset, x='P', y='label', hue='label')

# effect of Potassium on crop

plt.figure(figsize=(15,12))

plt.axes(xlabel="Potassium", ylabel="Crop")

sns.scatterplot(data=dataset, x='K', y='label', hue='label')

# effect of temperature on crop

plt.figure(figsize=(20,12)) plt.axes(xlabel="Temperature", ylabel="Crop")

sns.scatterplot(data=dataset, x='temperature', y='label', hue='label')

# effect of humidity on crop

plt.figure(figsize=(22,12)) plt.axes(xlabel="humidity", ylabel="Crop")

sns.scatterplot(data=dataset, x='humidity', y='label', hue='label')

# effect of ph on crop

plt.figure(figsize=(20,10)) plt.axes(xlabel="ph", ylabel="Crop")

sns.scatterplot(data=dataset, x='ph', y='label', hue='label')

# effect of rainfall on crop

plt.figure(figsize=(20,9)) plt.axes(xlabel="Rainfall", ylabel="Crop")

sns.scatterplot(data=dataset, x='rainfall', y='label', hue='label')

# crop to digit dict

crop\_to\_digit = { 'rice' : 1,

'maize' : 2,

'chickpea' : 3,

'kidneybeans' : 4,

'pigeonpeas' : 5,

'mothbeans' : 6,

'mungbean' : 7,

'blackgram' : 8,

'lentil' : 9,

'pomegranate' : 10,

'banana' : 11,

'mango' : 12,

'grapes' : 13,

'watermelon' : 14,

'muskmelon' : 15,

'apple' : 16,

'orange' : 17,

'papaya' : 18,

'coconut' : 19,

'cotton' : 20,

'jute' : 21,

'coffee' : 22

}

# digit to crop dict

digit\_to\_crop = { 1 : 'rice',

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

}

# Transform label from string to integer

# Fit and transform the 'Category' column dataset['label'] = dataset['label'].map(crop\_to\_digit)

# View basic analysis of columns

dataset.describe()

# find correlation matrix

corr\_matrix = dataset.corr() corr\_matrix

# Seperating target variable from features

X = dataset.drop('label',axis=1) Y = dataset['label']

# Splitting Dataset

from sklearn.model\_selection import train\_test\_split

# Assuming 'X' is your feature matrix and 'y' is your target variable

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.15, random\_state=42) print("Types of crop in train data = ", len(Y\_train.value\_counts()))

print("Types of crop in test data = ", len(Y\_test.value\_counts()))

from sklearn.linear\_model import LogisticRegression

from sklearn.neighbors import KNeighborsClassifier from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import RandomForestClassifier from sklearn.naive\_bayes import GaussianNB

from sklearn.svm import SVC

from sklearn.ensemble import BaggingClassifier

from sklearn.ensemble import GradientBoostingClassifier from sklearn.ensemble import AdaBoostClassifier

from sklearn.metrics import accuracy\_score from sklearn.metrics import confusion\_matrix

models = {

'Logistic Regression': LogisticRegression(),

'K-Nearest Neighbors': KNeighborsClassifier(), 'Decision Tree': DecisionTreeClassifier(),

'Random Forest': RandomForestClassifier(), 'Naive Bayes': GaussianNB(),

'Support Vector Machine': SVC(), 'Bagging': BaggingClassifier(),

'AdaBoost': AdaBoostClassifier(),

'Gradient Boosting': GradientBoostingClassifier()

}

for model\_name, model in models.items(): # train model

model.fit(X\_train, Y\_train) # predict values for test data

Y\_predict = model.predict(X\_test)

print(f"Accuracy of {model\_name} : {accuracy\_score(Y\_test,Y\_predict)\*100}")

Final Results :

Accuracy of Logistic Regression : 95.74468085106383 Accuracy of K-Nearest Neighbors : 96.96048632218846 Accuracy of Decision Tree : 98.48024316109422 Accuracy of Random Forest : 98.78419452887537 Accuracy of Naive Bayes : 99.3920972644377

Accuracy of Support Vector Machine : 96.96048632218846 Accuracy of Bagging : 99.08814589665653

Accuracy of AdaBoost : 17.325227963525837 Accuracy of Gradient Boosting : 97.87234042553192

**DEEP LEARNING**

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import MinMaxScaler

from keras.models import Sequential

from keras.layers import Dense, LSTM, GRU, Conv1D, MaxPooling1D, Flatten

from sklearn.preprocessing import LabelEncoder

# Load the dataset

data = pd.read\_csv("../../Datasets/Crop\_Recommendation\_Dataset.csv")

data.drop('ph',axis=1,inplace=True)

label\_encoder = LabelEncoder()

data['label'] = label\_encoder.fit\_transform(data['label'])

# Convert string columns to numeric type

numeric\_columns = ['N', 'P', 'K', 'temperature', 'humidity', 'rainfall']

data[numeric\_columns] = data[numeric\_columns].apply(pd.to\_numeric, errors='coerce')

# Drop rows with NaN values

data.dropna(inplace=True)

# Separate features and labels

X = data.drop('label', axis=1)

Y = data['label']

# Normalize the features using Min-Max scaling

scaler = MinMaxScaler()

X\_normalized = scaler.fit\_transform(X)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X\_normalized, Y, test\_size=0.15, random\_state=42)

print(“Feed forward network”)

# Reshape features for input

X\_train\_rnn = X\_train.reshape(-1, 1, X\_train.shape[1])

X\_test\_rnn = X\_test.reshape(-1, 1, X\_test.shape[1])

model = Sequential([

Dense(128, activation='relu6'), # Dense layer with 32 units and ReLU activation

Dense(64, activation='relu6'), # Dense layer with 32 units and ReLU activation

Dense(32, activation='elu'), # Dense layer with 32 units and ReLU activation

Dense(len(Y.unique()), activation='softmax') # Output layer with softmax activation for multi-class classification

])

# Compile the model

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

# Train the model

model.fit(X\_train\_rnn, Y\_train, epochs=45, batch\_size=32, validation\_split=0.2)

# Evaluate the model on test data

test\_loss, test\_accuracy = model.evaluate(X\_test\_rnn, Y\_test)

print("Test Loss:", test\_loss)

print("Test Accuracy:", test\_accuracy)

print(“Gated Recurrent Unit”)

X\_train\_gru = X\_train.reshape(-1, 1, X\_train.shape[1])

X\_test\_gru = X\_test.reshape(-1, 1, X\_test.shape[1])

# Define the GRU model

model = Sequential([

GRU(64, input\_shape=(1, X\_train.shape[1])), # GRU layer with 64 units

Dense(32, activation='relu'), # Dense layer with 32 units and ReLU activation

Dense(len(Y.unique()), activation='softmax') # Output layer with softmax activation for multi-class classification

])

# Compile the model

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

# Train the model

model.fit(X\_train\_rnn, Y\_train, epochs=45, batch\_size=32, validation\_split=0.2)

# Evaluate the model on test data

test\_loss, test\_accuracy = model.evaluate(X\_test\_rnn, Y\_test)

print("Test Loss:", test\_loss)

print("Test Accuracy:", test\_accuracy)

print(“Recurrent Neural Network”)

X\_train\_rnn = X\_train.reshape(-1, 1, X\_train.shape[1])

X\_test\_rnn = X\_test.reshape(-1, 1, X\_test.shape[1])

model = Sequential([

LSTM(80, input\_shape=(1, X\_train.shape[1])), # LSTM layer with 64 units

Dense(64, activation='relu6'), # Dense layer with 32 units and ReLU activation

Dense(32, activation='elu'), # Dense layer with 32 units and ReLU activation

Dense(len(Y.unique()), activation='softmax') # Output layer with softmax activation for multi-class classification

])

# Compile the model

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

# Train the model

model.fit(X\_train\_rnn, Y\_train, epochs=45, batch\_size=32, validation\_split=0.2)

# Evaluate the model on test data

test\_loss, test\_accuracy = model.evaluate(X\_test\_rnn, Y\_test)

print("Test Loss:", test\_loss)

print("Test Accuracy:", test\_accuracy)

print(“Convolutional Neural Network”)

# Define the CNN model

model = Sequential([

Conv1D(64, 4, activation='relu6', input\_shape=(X\_train\_cnn.shape[1], 1)), # Convolutional layer with 64 filters and kernel size 3

Conv1D(128, 2, activation='relu6'), # Convolutional layer with 128 filters and kernel size 3

MaxPooling1D(2), # Max pooling layer

Flatten(), # Flatten layer

# Dense(512, activation='relu6'), # Dense layer with 512 units and ReLU activation

Dense(256, activation='relu6'), # Dense layer with 512 units and ReLU activation

Dense(128, activation='relu6'), # Dense layer with 512 units and ReLU activation

Dense(64, activation='relu6'), # Dense layer with 512 units and ReLU activation

Dense(32, activation='relu6'), # Dense layer with 512 units and ReLU activation

Dense(len(Y.unique()), activation='softmax') # Output layer with softmax activation for multi-class classification

])

# Compile the model

model.compile(optimizer='adam', loss='sparse\_categorical\_crossentropy', metrics=['accuracy'])

# Train the model

model.fit(X\_train\_cnn, Y\_train, epochs=45, batch\_size=32, validation\_split=0.2)

# Evaluate the model on test data

test\_loss, test\_accuracy = model.evaluate(X\_test\_cnn, Y\_test)

print("Test Loss:", test\_loss)

print("Test Accuracy:", test\_acccuracy)