VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI

Project Report on

"Predicting the Total Crop Production in Area and Advising Best Alternative crop"

Submitted in the partial fulfillment for the requirements of the degree of

BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING

Submitted By

Vedashrutha D S	**1BY19CS178**
Sandesh D Gowda	**1BY19CS157**
Vishal Jain H K	**1BY19CS183**
Hrishikesh Kamath	**1BY19CS025**

Under the guidance of

Dr.Nagabhushan S V
Associate Professor
Department of CSE,
BMSIT&M

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT
YELAHANKA, BENGALURU - 560064.

2022-2023

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI

BMS INSTITUTE OF TECHNOLOGY AND MANAGEMENT
YELAHANKA, BENGALURU – 560064

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work entitled **"Predicting The Total Crop Production In Area And Advising Best Alternative Crop"** is a bonafide work carried out by **Vedashrutha D S (1BY19CS178), Sandesh D Gowda(1BY19CS157),Vishal Jain H K (1BY19CS183), Hrishikesh Kamath (1BY19CS025),** in partial fulfillment for the award of **Bachelor of Engineering Degree in Computer Science and Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2022-2023. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report. The project report has been approved as it satisfies the academic requirements in respect of project work for B.E. Degree.

Signature of the Guide	**Signature of the HOD**	**Signature of Principal**
Dr Nagabhushan S V Associate Professor Dept. of CSE, BMSIT&M	Dr. Thippeswamy G Professor & HOD, Dept. of CSE, BMSIT&M	Dr. Mohan Babu G N Principal, BMSIT&M

External VIVA-VOCE

Name of the Examiners	Signature with Date
1.	
2	

ACKNOWLEDGEMENT

We are happy to present this project after completing it successfully. This project would not have been possible without the guidance, assistance and suggestions of many individuals. We would like to express our deep sense of gratitude and indebtedness to each and every one who has helped us make this project a success.

We heartily thank our Principal**, Dr. MOHAN BABU G N, BMS Institute of Technology and Management,** for his constant encouragement and inspiration in taking up this project.

We heartily thank our Head of the Department**, Dr. Thippeswamy G, Department of Computer Science and Engineering, BMS Institute of Technology and Management,** for his constant encouragement and inspiration in taking up this project.

We gracefully thank our Project Guide, **Dr Nagabhushan S V**, **Associate Professor, Department of Computer Science and Engineering,** for his guidance, support, and advice.

Special thanks to all the teaching and non-teaching staff members of Computer Science and Engineering Department for their help and kind co-operation.

Lastly we thank our parents and friends for the support and encouragement given to us in completing this precious work successfully.

Vedashrutha D S (1BY19CS178)
Sandesh D Gowda (1BY19CS157)
Vishal Jain H K (1BY19CS183)
Hrishikesh Kamath (1BY19CS025)

ABSTRACT

This project aims at creating a machine learning-based system that can forecast the overall agricultural production in a given area and suggest substitute crops. Planning and resource allocation for agriculture must be accurate when predicting crop yield. The system builds prediction models using historical agricultural data, such as weather patterns, soil properties, crop-specific variables, and past production statistics. By taking into account the interactions between many variables, a variety of machine learning algorithms, including random forest, support vector machines, and gradient boosting, are used to create precise prediction models. A recommendation engine is also incorporated into the system, which analyses projected crop production to identify substitute crops based on resource availability, market need, crop rotation benefits, and climatic compatibility. The system promises to help farmers choose the best crops, reduce risks, increase yields, and adapt to shifting agricultural dynamics. Through comprehensive experimentation and validation using actual agricultural data, the project's effectiveness is assessed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1
INTRODUCTION

CHAPTER-1

INTRODUCTION

1.1 Background

For agricultural nations like India, crop prediction and suggestion are crucial. Numerous advantages that directly assist the agriculture industry and the overall development of the nation are provided by these initiatives.The increase in production and yield is a notable benefit. Farmers may choose crops, allocate resources, and modify their farming practises with confidence by using reliable crop prediction models. They are better able to predict the best planting period, estimated yield, and potential crop losses as a result. Farmers may thereby increase their yield and productivity, which boosts agricultural output. The models shed light on crop diseases as well.

Projects for crop prediction and advice can help with risk reduction. Risks to agriculture include erratic weather patterns, pest outbreaks, and changes in the market. Farmers are able to evaluate and successfully manage these risks thanks to accurate crop predictions. They can choose wisely when it comes to crop diversification, insurance protection, and putting mitigation measures in place. This increases their resilience while lowering the susceptibility of their farming activities.

History

Machine learning has developed techniques for recommending crops over time. To make simple recommendations at first, rule-based systems based on expert knowledge were established. Machine learning algorithms were used to examine past data for better recommendations as data availability expanded. More precise and site-specific recommendations are now possible because to the development of precision agriculture and sensor technologies. Real-time data collecting and processing were made possible by the introduction of IoT devices and big data. Techniques for collaborative filtering have developed, utilising shared knowledge and experiences to provide individualised recommendations. Deep learning and ensemble techniques continue to be used in ongoing developments to improve accuracy, scalability, and usability. Overall, these technologies give farmers the ability to take into account certain circumstances and environmental elements while making informed decisions, optimising resources, and increasing crop yields.

Key responsibilities

Data Collection and Preparation: Assembling pertinent data from a range of sources, including weather, soil, and previous crop yield statistics. The data would be prepared for analysis by being cleaned, pre-processed, and transformed. Handling missing values, scaling features, and encoding categorical variables are some examples of activities that may be involved. Activities that might be included include handling missing values, scaling features, and encoding categorical variables.

Model Selection and Training: To find the best models for the crop recommendation problem, we have compared several machine learning strategies such as decision trees, KNN, support vector machines. With the prepared dataset, the models would be trained. Their performance are be evaluated using cross-validation, and hyperparameters would be adjusted for the best outcomes. The trained models would be used with the prepared dataset. Cross-validation would be used to assess their performance, and hyperparameters would be tuned for the best results.

Evaluation and Validation: Utilize relevant evaluation criteria, such as accuracy, precision, recall, or F1 score, to rate the effectiveness of the trained models. This would include either using methods like k-fold cross-validation or dividing the dataset into training and testing sets. To make sure the models can accurately select crops, they would be tested against real-world conditions. The models would undergo testing under actual environmental conditions to ensure that they can choose crops appropriately.

Deployment and Integration: We create a Web UI using Streamlit that enables users to enter their parameters along with an API that allows users to communicate with the server and serve the user with customized crop recommendations in real-time.

1.2 Literature Survey

Year	Author	Purpose	Methods mentioned	Inference
2018	Konstantinos G. Liakos ,Patrizia Busato , Dimitrios Moshou , Simon Pearson ID and Dionysis Bochtis.	Machine Learningin Agriculture ”Institute for Bio- Economy and Agri Technology”.	SVM and ANN	SVM is used here for binary classifier andANN is used for pattern recognition
2018	Arun Kumar, Naveen Kumar, Vishal Vats.	Efficient Crop Yield Prediction Using Machine Learning Algorithms.	Support Vector Machine, and Least Squared Support Vector Machine.	It shows that SVM is better here compared to the complexity.
2019	Mohsen Shahhosseini , Rafael A Martinez-Feria , Guiping HU and Sotirios V Archontoulis.	Maize yield andnitrate loss Prediction with machine learning algorithms.	LASSO Regression, Extreme Gradient Boosting, Ridge Regression, randomforests.	Pre-growing season prediction of crop production of outputssuch as grain yields and nitrogen losses can provide best suggestion of crops to farmers.

2019	S.R.Rajeswari , ParthKhunteta, Subham Kumar,Amrit Raj Singh,Vaibhav Pandey	Smart Farming Prediction Using Machine Learning.	Bayesian networkand ANN.	Bayesian network is used to form the Statistical analysis of the given dataset. ANN is used to compares the patterns which has thenonlinear effect and concept.
2020	M.Kalimuthu, P.Vaishnavi, M.Kishore.	Crop Prediction Using Machine Learning	Naïve Bayes	In the Naïve Bayes method the accuracy of the model is 97%.
2021	Pavan Patil, Virendra Panpatil,Prof. Shrikant Kokate.	Crop Prediction System using Machine Learning Algorithms	Decision tree andNaïve Bayes.	The combination classification algorithmof naïve bayes anddecision tree classifier are better performing than use of single classifier model.
2022	Ankita Sharma;Anushtha Tamrakar;Sourajita Dewasi;Nenavath Srinivas Naik	Early Prediction of Crop Yield in India using MachineLearni ng	Knn,Random Forest	This is the combination of Decision tree and Random Forest Regressor for yield prediction in wich DT gave highest accuracy

1.3 Motivation

The expanding importance and promise of artificial intelligence (AI) and its applications in numerous fields are the driving forces behind this machine learning effort. A kind of artificial intelligence called machine learning has proven to be very adept at processing and analysing vast amounts of data, drawing insightful conclusions, and making precise predictions. This project intends to use machine learning to address a particular issue or take advantage of an opening in our target domain. We aim to enhance decision-making, automate processes, increase efficiency, or optimise results by utilising cutting-edge algorithms and approaches.

Due to their potential to optimise crop output, and decision-making, crop yield prediction and recommendation systems utilising machine learning have drawn substantial attention in the agricultural area. These programmes use machine learning algorithms to examine both past and current data, giving farmers insights and suggestions on how to increase their output and profitability. The following section provides more insight about this

Yield Prediction:

The goal of crop yield prediction is to forecast the potential yield of a particular crop during a specific growing season. Machine learning models can discover patterns and correlations to create precise yield estimates by studying historical data, including weather patterns, soil characteristics, and agronomic practises. These forecasts can assist farmers in anticipating crop productivity, allocating resources, and streamlining their processes.

Crop Recommendation:

The main goal of crop recommendation systems is to make recommendations for the best crop(s) for a given area or location in light of a variety of parameters. In order to deliver individualised recommendations, machine learning models analyse both historical and current data, including market demands, soil fertility, climate conditions, and water availability. These suggestions support farmers in selecting crops while taking into account elements like profitability, sustainability, and risk management.

Machine Learning Techniques:

Crop yield forecasting and recommendation systems use a variety of machine learning methods. These include ensemble methods, decision trees, random forests, support vector machines, and neural networks. In order to create accurate and trustworthy models, feature engineering, data pre-processing, and model evaluation procedures are essential.

Benefits:

Machine learning-based methods for crop yield prediction and recommendation have various advantages. They aid farmers in increasing total crop productivity, minimising environmental effect, maximising resource allocation, and lowering production costs. These technologies facilitate the use of precision agriculture techniques, improve decision-making, and aid farmers in adjusting to shifting environmental factors and market needs.

1.4 Problem Statement

To create a reliable and precise system that can anticipate crop yields for various Indian locations(Karnataka,Telangana,Tamilnadu,Kerala,Maharashtra..etc)and then give farmers suggestions for replacement crops that would be more appropriate given particular conditions based on Temperature, Humidity, Nitrogen, Phosphorous, Potassium values and to improve decision-making, optimise resource allocation, and raise crop productivity in India, the project will make use of machine learning algorithms and local agricultural data(Kaggle dataset).

Important aspects of problem statement:

Geographical Variations: India has a wide range of geographical and climatic circumstances, temperature ranges(-2℃ to 50℃), and Humidity ranges(40% to 100%). These geographic variances should be taken into consideration by the system, which should then offer region-specific(District-State-City) crop production and suggestions for alternative crops.

Crop Diversity: India is renowned for its diverse agricultural production, with a vast variety of crops being grown(Rice,Maize,Chickpea,KidneyBeans,mothbeans etc) in various states

and geographical areas. The system should be able to manage several crops and offer precise forecasts and suggestions for different cash crops, horticulture crops, and staple crops grown in India.

1.5 Aim and Objective

Our Project aims at creating a reliable and accurate system that can forecast crop yields for various fields or regions and give farmers suggestions for alternative crops that might be more suitable under particular conditions. The initiative intends to improve decision-making, optimise resource allocation, and boost crop output by utilising machine learning algorithms and agricultural data.

Objectives:

1.5.1 Develop Crop Yield Prediction Model: Create a machine learning model that can precisely estimate crop yields based on past and present information, such as weather patterns, soil features, agronomic practises, and crop-specific variables. In order to accurately anticipate yield, the goal is to develop a model that can effectively capture the relationships and patterns in the data.

1.5.2 Identify Relevant Factors: Find the important elements, such as soil characteristics, planting dates, and management techniques, that have a substantial impact on crop production, such as meteorological variables (temperature, precipitation). The goal is to pinpoint the components that are most crucial to the prediction model's accuracy and interpretability.

1.5.3 Evaluate Model Performance: Using appropriate assessment metrics, such as mean absolute error (MAE) or root mean square error (RMSE), evaluate the performance of the crop yield forecast model. Making ensuring the model achieves high accuracy and can successfully generalise to new data is the goal.

1.5.4 Recommend Alternative Crops: Create a system for recommending alternative crops depending on particular requirements or restrictions, such as crop rotation, market demand, water availability, and climate suitability. Providing farmers with feasible solutions that might be more adaptable, profitable, or sustainable given their conditions is the goal.

1.5.5 Validate Recommendations: Compare the recommended alternative crops to actual field trials or professional expertise to verify their accuracy and efficiency. The goal is to make sure that the recommendations fit the needs of farmers and are feasible.

1.5.6 User-Friendly Interface: Make a user-friendly interface such as streamlit that enables farmers to enter their environmental factors, and crop preferences, such as an online or mobile application. The goal is to give farmers a simple platform so they can simply get yield projections and suggestions for alternative crops.

1.6 Challenges

There are a number of difficulties that may occur when designing a crop yield forecast and best alternative crop suggestion model. Listed below are some typical problems and possible solutions.

1.6.1 Data Availability: Our project should take into account the accessibility and availability of Indian-specific agricultural data, such as historical weather data, soil data, crop-specific data, and yield records.Kaggle website provides us a large repository of dataset for crop prediction and crop yield prediction.For accurate predictions and suggestions, efforts should be taken to gather, clean, and pre-process pertinent data.
Crop prediction parameters include nitrogen ,phosphorous,potassium,temperature,humidity, hainfall,ph values.

1. Nitrogen Range(0-100mg/kg)
2. Phosphorous Range(0-100mg/kg)
3. Potassium Range(0-50mg/kg)
4. Temperature Range(15-50°C)
5. Humidity Range(0-100gm/m^3)
6. Rainfall Range(30-300mm)
7. PH values(5-8)

1.6.2 Data Quality and Preprocessing: Agricultural data may have omissions, anomalies, errors, or inconsistent values,missing values. The model's performance may be affected by several problems. Effective data cleaning and pre-processing is essential, and this includes handling missing values, finding outliers, and normalising the data. Advanced algorithms can be used to detect and handle outliers, while imputation techniques can be used to fill in missing numbers.

1.6.3 Geographical Variations: India has wide regional variances, including varied soil types and climatic patterns in various places. To ensure accurate forecasts, the model must take into account region-specific characteristics. One method is to partition the data into smaller regions and create unique models for each region. Using geographic features as input variables or clustering techniques to place comparable regions together are alternative strategies.

1.6.4 Crop Diversity: The variety of crops grown in India poses a difficulty because each crop has its own growth habits, nutrient needs, and reactions to the environment. The model can be created to handle numerous crops and incorporate crop-specific features and concerns in order to manage this. The model's performance can also be enhanced by ensuring a varied and inclusive dataset that includes a range of crops.

1.6.5 Local Constraints and Challenges: India's agriculture suffers unique difficulties due to socio-economic issues, resource access restrictions, Climatic changes, soil Changes. In order to make recommendations that are workable and realistic within the Indian agricultural setting, the system should take into account these regional restrictions and difficulties.
Historical Agricultural Datasets are hidden based on the several privacy issues based on Indian Agricultural policies and private data centres.

A government does not necessarily required to possess a dataset. Datasets that are owned by the government in one nation may be held by the private sector in another. For instance, there are various perspectives on who owns national address registers around the world. Some are controlled by the federal government, some by local organisations, and still others by private companies. Regardless of who owns the data, policymakers should be aware of it and consider ways to make it more accessible. Kaggle dataset provides us the solution for the large repository for public accessed dataset.

1.6.6 User Adoption: The model's acceptance and use by farmers can be difficult. User adoption can be boosted by making sure the interface is user-friendly, clearly articulating the benefits of the model, and incorporating user feedback into the development process. Information dissemination and trust-building can also be facilitated by working in partnership with regional agricultural specialists, extension personnel, and farmer communities.

The crop yield prediction and alternative crop suggestion model may be designed to offer precise and helpful results for farmers in India by tackling these issues with the right approaches and taking into account the unique requirements and context of the project.

1.7 Organization of the thesis

Chapter 1: Introduction Provides introduction to the project also deals with literature survey and literature review has been carried out to collect data (i.e., relevant background information) to find associated problem(s), cause(s) and understanding the same

Chapter 2: Overview: Provides overview of the work carried out and the defines the expected output

Chapter 3: Requirement Specification Describes the problem statement derived from the literature review and identified research gaps through the critical review of relevant papers (as discussed in the chapter 1)

Chapter4: Detailed Design Provides the detailed design phases followed during the process of project execution.

Chapter5: Implementation Provides details on the algorithm implementation on Arduino and face recognition module.

Chapter 6: Testing Provides the details on tests and validation conducted on the implemented algorithm.

Chapter 7: Experimental Results Provides the summary of the results for the respective inputs.

Chapter 8: Conclusion Deals with the discussion of conclusions and future directions of the current dissertation work based on interpretations of the results presented in the previous chapters.

CHAPTER-2
OVERVIEW

CHAPTER-2

OVERVIEW

This Project consists of four major components

1.Crop Recommendation

An important issue in agriculture that can be resolved with machine learning approaches is crop recommendation. Predicting the crop that is most suited for a specific set of environmental factors, such as soil type, climate, and terrain. A well-liked machine learning algorithm for crop recommendation is support vector machine (SVM). A supervised learning method known as the SVM can be applied to classification and regression issues. The SVM classifier is trained on a dataset for crop recommendation that contains details about the environmental conditions and the crops that are most appropriate for those settings. This training data is used by the SVM algorithm to determine the hyperplane that best separates the various crop classes.

2.Yield Prediction

Using historical data, yield prediction in agriculture determines crop output based on a few different variables[2]. With the help of machine learning methods like decision tree regressors, crop yield estimates can be made with a high degree of accuracy.

A decision tree regressor is a supervised learning method that builds a tree-like model of decisions and their results. It can be especially useful when the required variable—in this case, crop yield—is continuous and numerical. The decision tree regressor creates a tree with nodes that each represent a decision rule based on a feature value by segmenting the dataset into subsets depending on the values of several features. The tree's leaves stand in for the expected values.

3.Profit Predictor

The k-Nearest Neighbours (k-NN) classifier for profit prediction in machine learning uses past data to create a model that predicts future earnings based on certain input variables. A supervised learning technique that can be applied to classification or regression applications is the k-NN algorithm. The first stage in using the k-NN algorithm to predict profits is to gather and prepare a dataset of previous earnings and pertinent input variables. These input

variables could consist of elements like consumer demographics, sales information, marketing costs, and other pertinent data that may have an impact on earnings.

4. Disease Encyclopaedia

There are numerous situations where farmers unable to earn good profit because of the effect of crop plant diagnosed with various disease, hence to provide a natural view about various disease that could effect the plant, a module is built to know the disease analysis, prevention and control strategies.

Overall, firms trying to make knowledgeable choices regarding upcoming investments or marketing tactics may find utilising a k-NN classifier for profit prediction to be an effective tool. Businesses can get important insights into the elements that affect their earnings by analysing historical data and spotting patterns in consumer behaviour. They can then utilise this knowledge to optimise their processes and raise profitability.

CHAPTER-3
REQUIREMENT SPECIFICATION

CHAPTER-3

REQUIREMENT SPECIFICATION

3.1 FUNCTIONAL REQUIREMENTS

- The system should accept soil information, climatic data, type of crop to be grown, season type from the user.
- The user can select from a list of menu to navigate to crop suggestion form or to yield prediction form
- The system should display the predicted yield for the given data.
- The system should suggest the best alternative crop if the crop choosen by the farmer results in a loss

3.2 NONFUNCTIONAL REQUIREMENTS

- The system should be able to handle a dataset of atleast 10000 data values
- The system should be generate the output in a reasonable amount of time
- The system should be user friendly to use with clear instructions and simple interface
- The system should provide accuracy for the generated output
- The system should be reliable and have minimal downtime

3.3 SOFTWARE REQUIREMENT SPECIFICATION

- Python
- Anaconda(PyCharm)
- StreamLit
- Chrome Web Browser

Software Libraries Required

- Numpy
- Pandas
- Sklearn

- Base64

PYTHON

Python is an interpreted, high-level, general-purpose programming language. Code readability is a key component of Python's design philosophy, which is achieved by the use of extensive indentation. Its object-oriented technique and language characteristics are designed to help programmers write clear, understandable code for both small and large projects.

Python has dynamic typing and garbage collection. Numerous programming paradigms are supported, particularly procedural, object-oriented, and structured programming.Python is commonly referred to as a "batteries included" language because of its vast standard library.

Figure 3.1: Python Language

Anaconda

An extensive selection of scientific libraries, packages, and tools are included with the Python programming language distribution known as Anaconda. It streamlines the installation and maintenance of Python packages and is intended for data science and machine learning operations. Conda, the package manager included with Anaconda, makes it simple to construct and maintain isolated Python environments.

Figure 3.2 Anaconda

Key features of Anaconda

Pre-installed scientific libraries: Anaconda comes with well-known libraries for data analysis, machine learning, and scientific computing, including NumPy, Pandas, Matplotlib, SciPy, and scikit-learn.

Conda package manager: Conda makes it easier to manage various environments and install, update, and remove Python packages.

Environment management: You can establish and manage isolated Python environments with Anaconda, which makes it simple to work on several projects with various dependencies.

An application launcher and graphical user interface (GUI) for managing environments and packages are both provided by Anaconda Navigator.

Cross-platform compatibility: Linux, macOS, and Windows operating systems are all supported by Anaconda.

Key features of PyCharm:

Intelligent code editor: PyCharm makes it simpler to write and comprehend Python code by offering enhanced code completion, code navigation, and syntax highlighting.

Debugger: The robust debugger that comes with PyCharm enables you to step through your code, analyse variables, and find problems.

Integrated testing framework: PyCharm enables you to develop and run tests inside the IDE by supporting a number of testing frameworks, including pytest, unittest, and doctest.

Version control system integration: PyCharm has functionality for code versioning, branching, and merging and smoothly interacts with version control systems like Git, Mercurial, and Subversion.

Working on projects that are on distant servers or virtual machines is possible with PyCharm's support for remote development.

Support for web development: PyCharm offers Python web development tools, including the Django and Flask frameworks and HTML.

Figure 3.3 PyCharm

Streamlit

An open-source Python library called Streamlit is used to create web apps for data science and machine learning projects. It enables programmers to design interactive dashboards, data visualizations, and user interfaces quickly and easily.

The engineers that developed this Python-based library had machine learning in mind. Data scientists and machine learning engineers are not web developers, thus they have no interest in spending weeks learning how to use these frameworks to build online apps. If a tool can present data and gather modeling-related aspects, they prefer one that is easier to understand and use. Streamlit lets you to create applications that look fantastic with just a few lines of code. Simple and logical Both inexperienced and seasoned Python developers can utilise Streamlit because of its clear and user-friendly API.

Features of StreamLit

Rapid prototyping: You can easily convert your Python scripts into interactive web applications with Streamlit. It allows for quick development and iteration, letting you see the changes as you make changes to your code in real time.

Data-focused: Streamlit was created especially for applications involving data science and machine learning. Common data manipulation and visualization tools like Pandas, Matplotlib, and Plotly are supported natively by this program.

Automatic widget generation: Streamlit automatically creates interactive widgets like sliders, dropdowns, and checkboxes from your Python variables. This makes it simple to provide interactive controls for looking through and modifying data.

Streamlined development workflow: It is made possible by live code reloading, which automatically executes any changes you make to your code. Without having to restart the application, the findings are available right away

3.4 HARDWARE REQUIREMENT SPECIFICATION

The hardware requirements for machine learning (ML) vary depending on the complexity of the models and the size of the datasets involved. However, here are some general guidelines for hardware requirements for ML:

- **CPU**: A powerful CPU is essential for machine learning tasks. A CPU with at least 4 cores is recommended for most ML tasks. Intel Core i7 or i9 processors or their AMD Ryzen equivalents are a good choice.
- **GPU**: A Graphics Processing Unit (GPU) is important for machine learning tasks that involve deep learning or neural networks. A high-end GPU with at least 8GB of VRAM is recommended. NVIDIA GPUs are the most commonly used for machine learning tasks.
- **RAM**: Machine learning tasks require a large amount of RAM to store and manipulate large datasets. At least 16GB of RAM is recommended for most ML tasks, and 32GB or more is recommended for larger tasks.
- **Storage**: A fast and large storage device is important for storing large datasets. A Solid State Drive (SSD) is recommended for fast read and write speeds.

CHAPTER-4
DETAILED DESIGN

CHAPTER-4

DETAILED DESIGN

4.1 Use case diagram

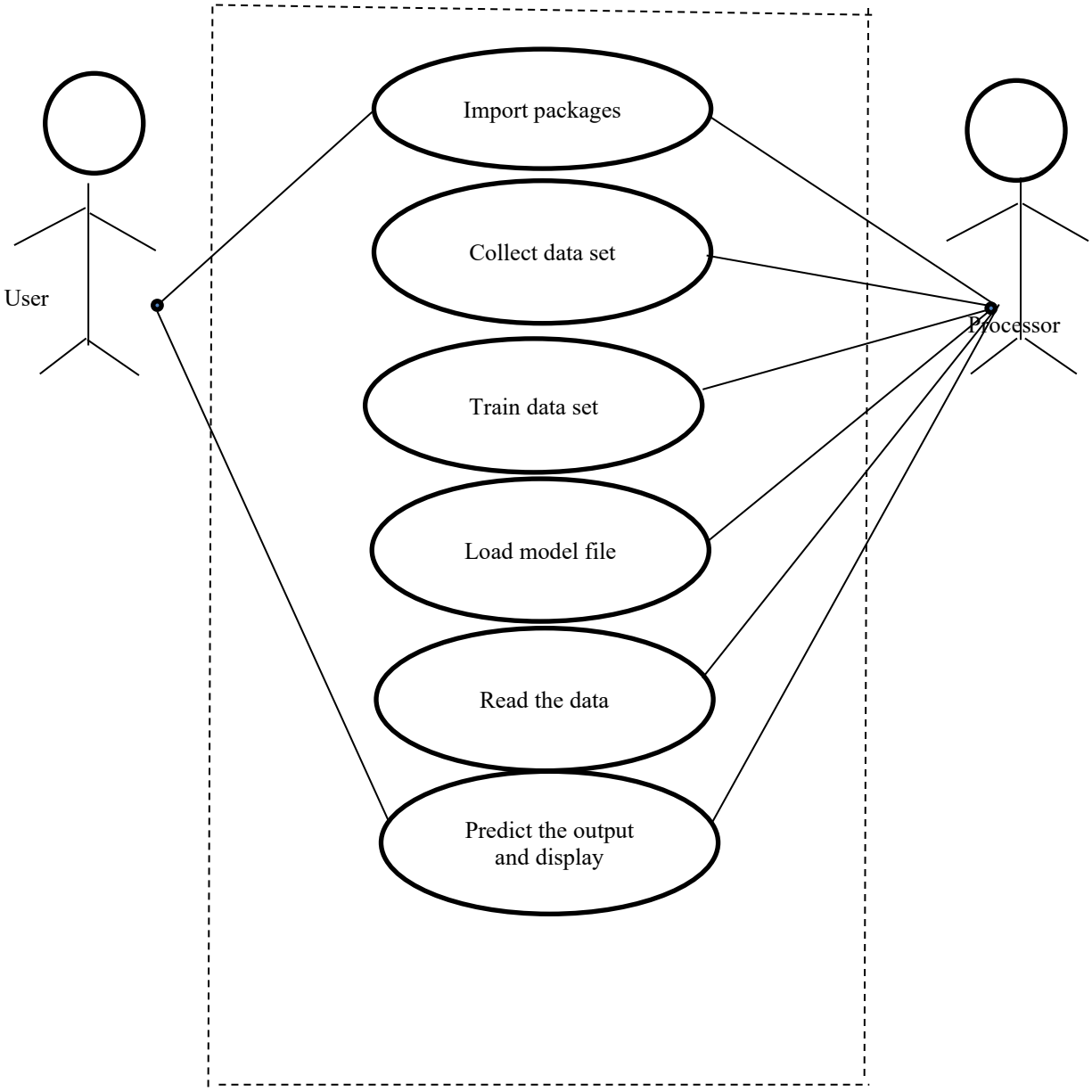


Figure 4.1: Diagram of the use cases for the suggested model

User and processor make up a use case in which user provides input to the system and processor processes the data and produces output. The flow is depicted in the diagram above (Fig. 4.1).The dataset affects both the input to the system and the system's output. The system and code are run by the first user, and model and library package are imported and loaded. The user loads the train dataset and trains the system before building a model.

Following a code run, output is shown in accordance with the given data input.

4.2 SEQUENCE DIAGRAM

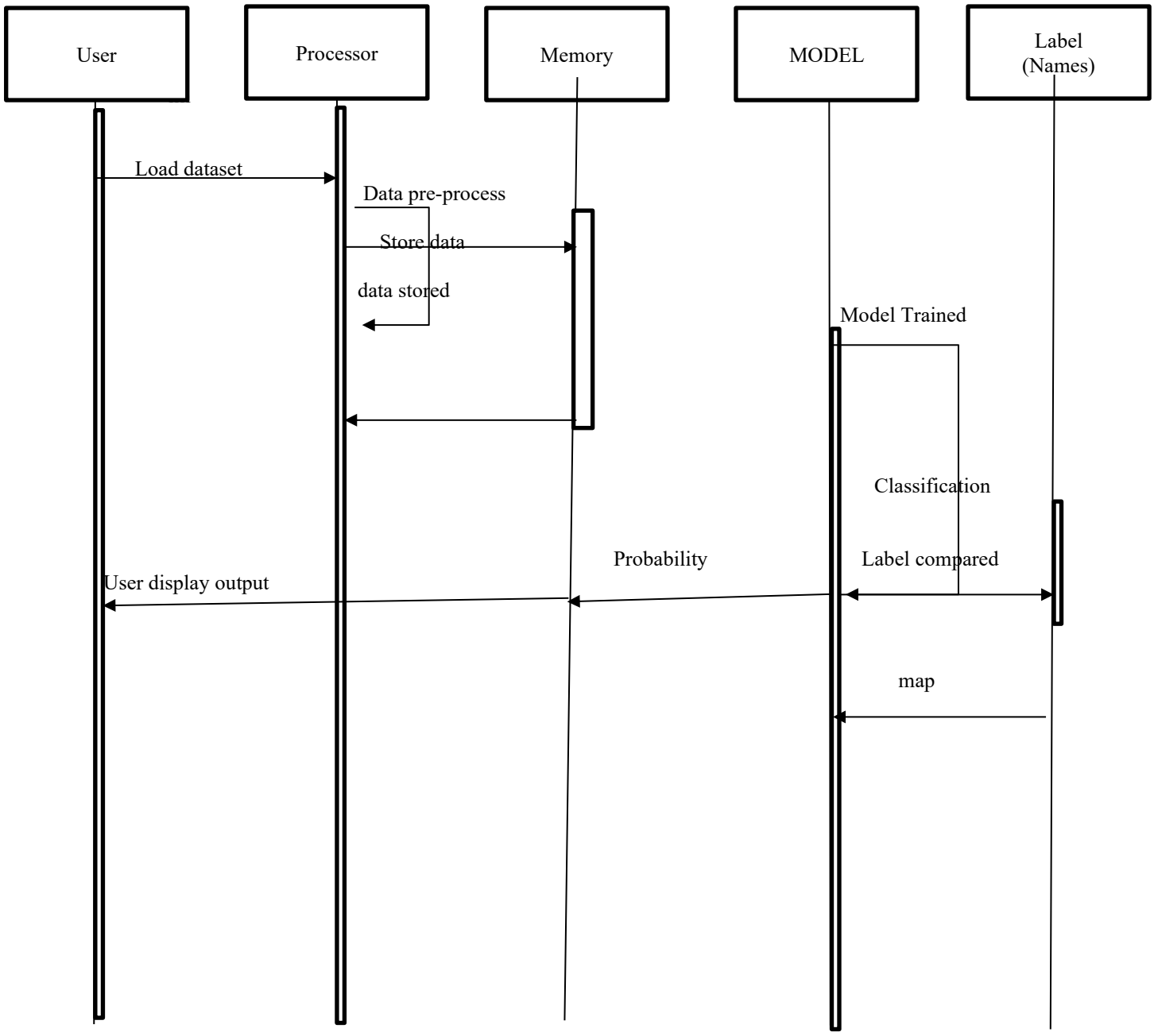


Figure 4.2: Sequence Diagram for the proposed model

Sequence diagram consists of 5 different blocks namely user, processor, memory, Model and labels as shown in the above figure Fig 4.2

User will provide the input data through the csv files already saved in the system where preprocessing of data is done which is differentiator parameters and after that those are stored

in the memory unit. After preprocessing and storing of csv is done, trained model file is loaded where the featured of the file is extracted for classifying the output. After classifying the output, Recommendation values are displayed.

4.3 DATA FLOW DIAGRAM

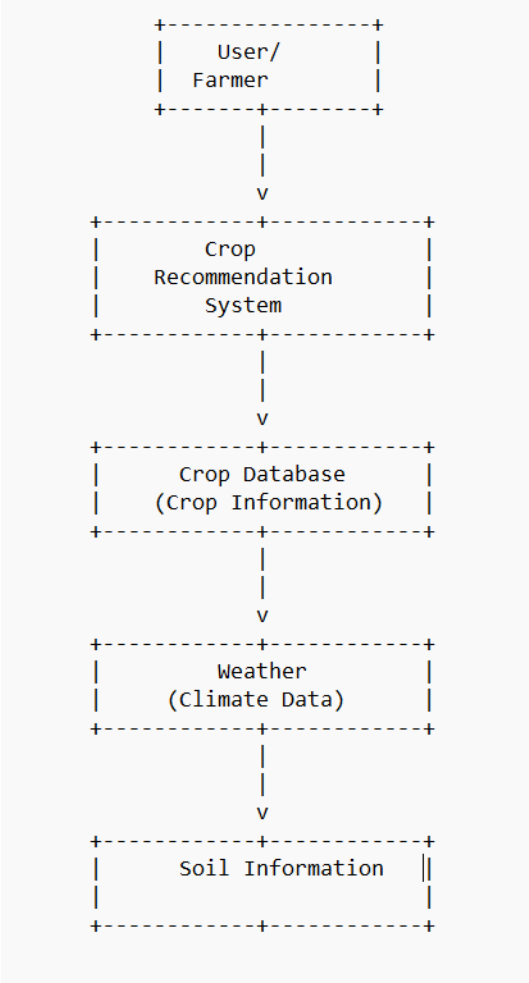


Figure 4.3.1: Data Flow Diagram level-0

The figure 4.3 shows the DFD at level-0 The User/Farmer represents the user who interacts with the crop recommendation system, giving input and receiving recommendations,

The main element that takes user input and uses a variety of data sources to produce crop recommendations is the crop recommendation system. It might have decision-making algorithms, models, or rule-based systems. The dataset includes details on many crops, including their traits, growing requirements, production potential, and compatibility for particular environments. This information acts as the recommendation system's knowledge base. Temperature, precipitation, humidity, and other pertinent meteorological variables are all

included in the weather data. This information aids in determining if crops are suitable given the meteorological conditions at hand or in the near future.For various regions or areas, the Soil details section provides details on the characteristics of the soil, including its pH level, nutrient makeup, and texture. This information aids in figuring out which crops grow well in which types of soil.

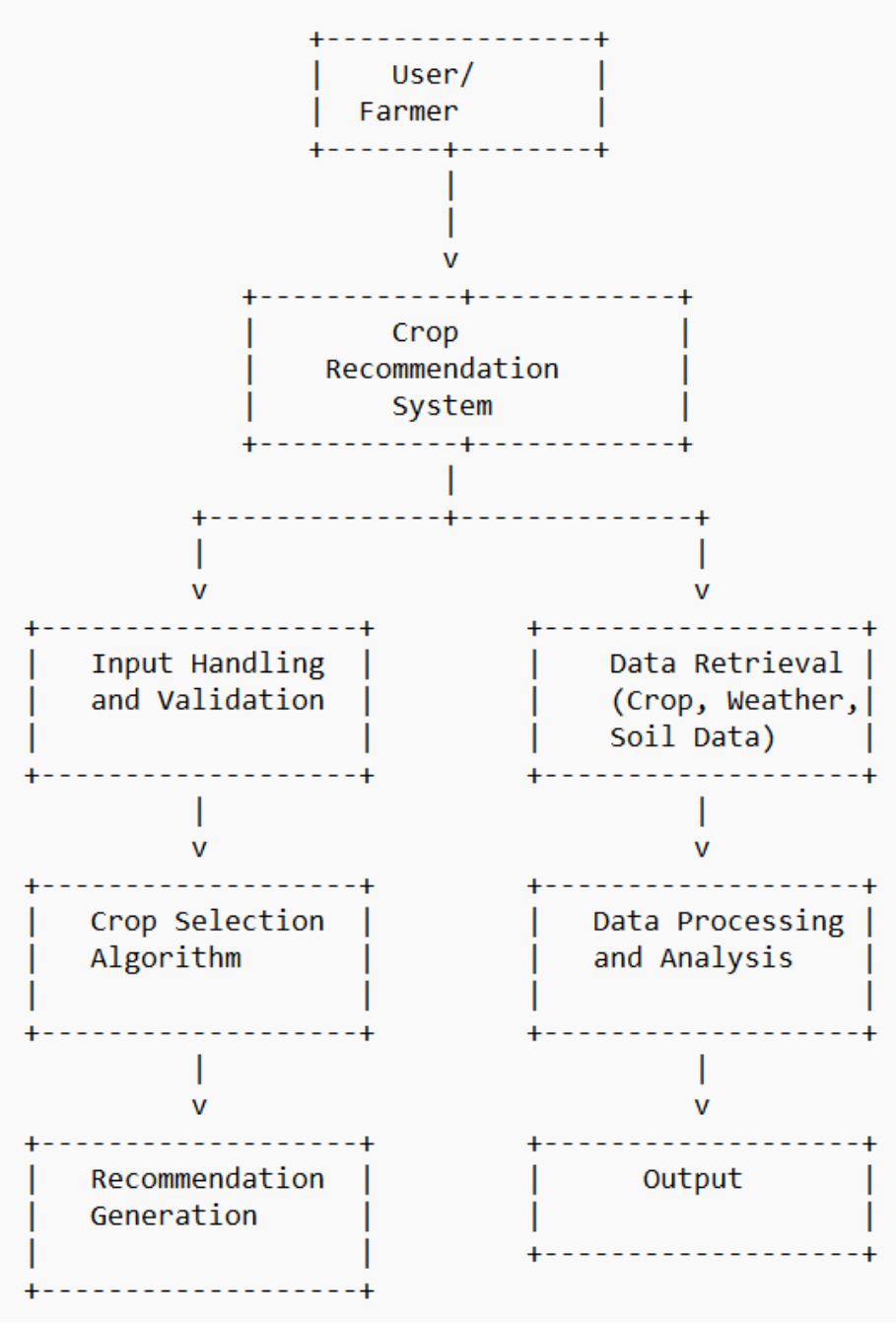


Figure 4.3.2: Data Flow Diagram level-1

In Figure 4.4 we can see that the User/Farmer serves as the crop recommendation system's primary source of input, supplying details like location, preferences, and limits.

The task of validating and processing user input falls under the purview of the input handling and validation procedure. Prior to further processing, it makes sure the information is accurate and comprehensive.The data retrieval process pulls pertinent information from a variety of databases, including those for crops, the environment, and soil. These databases each provide data on crops, weather patterns, and soil characteristics.

Utilising the input data, the crop selection algorithm process employs an algorithm or model to choose appropriate crops depending on user preferences, weather conditions, and soil qualities.

4.4 ACTIVITY DIAGRAM

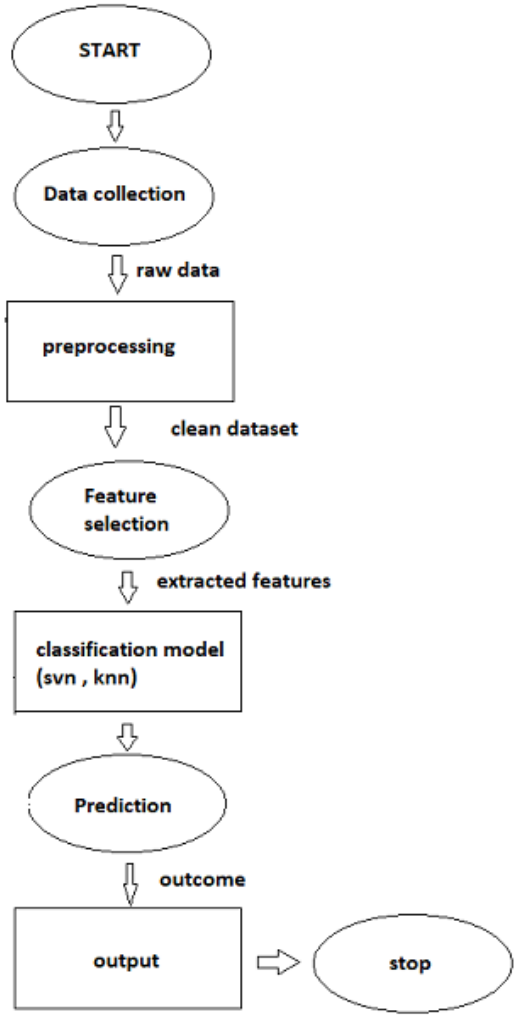


Figure 4.4: Activity Diagram for the proposed model

The first step in the process is acquiring pertinent information, such as past records, weather patterns, soil parameters, crop characteristics, and management techniques. The data is then

cleaned and transformed during preprocessing in order to ensure consistency and get rid of noise or outliers.The preprocessed data is used to train the prediction model, which identifies patterns and connections between input variables and crop yields.

To help the trained model make more precise predictions, real-time data is gathered, such as the current weather.Current circumstances and pertinent factors serve as the prediction model's input data.Based on the input data, the prediction model uses the discovered patterns to create predictions about crop yields.The outcomes are examined to assess the anticipated crop yields and spot any potential problems.

CHAPTER-5
IMPLEMENTATION

CHAPTER-5
IMPLEMENTATION

Using machine learning, we were able to identify how to distinguish between the data. Even though the model may not seem very impressive, the decisions made will determine the model. The following stages describe how to run the code and make it operate, though this criterion might change in the future as machine learning and AI in general advance:

- Data collection
- Data preparation
- Model selection
- Training
- Evaluation
- Hyper parameter tweaking
- Prediction

5.1 Gathering the data

Data collection is the process of compiling and examining data from various sources. Data collecting allows one to maintain track of prior events so that data analysis can be used to find repeating patterns. The Kaggle website is where the 'Crop Recommendation' dataset is gathered. The dataset uses 7 characteristics and distinct crops as class labels. The soil's ratios of

(i) Nitrogen
(ii) Phosphorus,
(iii) Potassium,
(iv) Humidity,
(v) Temperature (expressed in degrees Celsius),
(vi) Ph value, and
(vii) Rainfall (expressed in millimetres) are all taken into consideration.

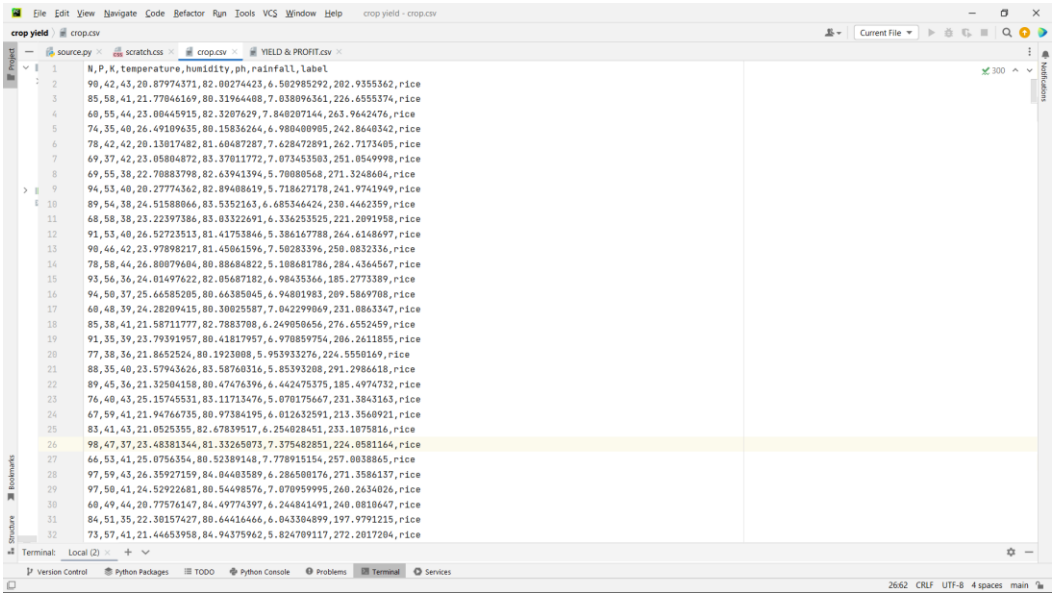


Figure 5.1:Sample Dataset(in csv format)

5.2 Data Preprocessing

The process of modifying raw data into a form that analysts and data scientists can use in machine learning algorithms to find insights or forecast outcomes is called Data preprocessing. In this project, the data processing method is to find missing values. Getting every data point for every record in a dataset is tough. Empty cells, values like null or a specific character, such as a question mark, mightall indicate that data is missing. The dataset used in the project didn't have any missing values.

- Identify the missing values
- Identifythe rows with missing values in a specific column:
  missing_value = diff [columns_name].isnull()
  diff[missing_ value]
- Calculate the number of missing values in each column:
  diff.isnull().sum()

- Removing missing values
  Drop rows with any missing values:
  diff.dropna();

- Drop specific columns:
  diff.drop(columns_to_be_dropped, axis=1)

- Drop columns with less than a certain number of non-null values:
  diff.dropna(threshold_value = min_nonnull, axis=1)

5.3 Train and Test Split

Using the train_test_split() method of the scikit learn module, the dataset is divided into a training dataset and a testing dataset. The dataset, which contains 2200 data, has been split into two datasets: a training dataset, which contains 1760 data, and a testing dataset, which contains 440 data.

5.4 Fitting the Model

Fitting is the process of altering the model's parameters to improve accuracy. On data for which the target variable is known, an algorithm is run to build a machine learning model. By contrasting the model's outputs with the target variable's actual, observed values, the accuracy of the model is assessed. A machine learning model's capacity to generalise data similar to that with which it was trained is known as model fitting. A model that accurately approximates the output while the inputs are uncertain is referred to as having a good model fit.

5.5 Score of training dataset

Using a trained machine learning model, scoring, also known as prediction, is the process of generating values from fresh incoming data. Each model's score over a training dataset is calculated using the model.score() method, which reveals how well the model has learned.

5.6 Model Prediction

"Prediction" refers to the outcome of an algorithm after it has been trained on a prior dataset and applied to new data when predicting the likelihood of a particular result. using the test feature dataset and the predict() method to predict the model. The output was provided as an array of forecasted numbers.

5.7 Confusion matrix and Classification Report

The metrics module in the scikit learn package was used to import the confusion matrix and classification report algorithms, which use test datasets' actual labels and predicted values to produce their results.

Confusion matrix : The frequency of true negatives, false negatives, true positives, and false positives is given by the confusion matrix.

Classification report : A classification algorithm's predictions are measured using the Classification Report metric. Precision, recall, and the model's f1 score are provided.

Precision is the capacity of a classifier to count the number of positive predictions that are generally accurate. The ratio of true positives to the total of both true and false positives for each class is used to compute it.

$$Precision = \frac{TP}{TP + FP}$$

When Precision, Positive Prediction Accuracy, True Positive, and False Positive

A classifier's **recall** is its capacity to extract all positive examples from the confusion matrix. The ratio of true positives to the total of true positives and false negatives for each class is used to compute it.

$$Recall = \frac{TP}{TP + FN}$$

Where FN-False Negative; Recall- The proportion of correctly detected positives;

The **F1** score, with 0.0 being the poorest and 1.0 being the greatest, is a weighted harmonic mean of recall and precision

F1 scores are frequently lower than accuracy assessments since precision and recall are taken into account during the computation.

$$F1\ score = \frac{2 * PR}{(P + R)}$$

Where P-Precision; R-Recall

5.8 Accuracy

Model accuracy is defined as the proportion of correct forecasts to all predictions. the metrics module's precision.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP-True Positive; FP-False Positive; TN-True Negative; FN-False Negative

5.8 Deployment Process

Create the Prediction Model: Create and train the prediction model in Python using historical crop data, meteorological data, soil properties, and other pertinent information. Preprocess the data using the necessary statistical or machine learning algorithms.

Create the Streamlit Application: Create the user interface using the Streamlit Python module for creating interactive web applications. Create the application's layout, add input fields for user information (such as weather data and soil details), and provide the output section's expected crop yields or suggested practices.

Establishing the Development Environment: Make sure Python is set up on your local computer. To isolate dependencies and keep the environment clean, use a virtual environment. Using a package manager like pip or another, install the necessary Python packages, including Streamlit.

Program Testing: Test the Program On your local computer, thoroughly test the Streamlit program. Check that the user interface performs as it should, evaluate the forecasts and suggestions made by the model, and fix any faults or problems found during testing.

Launch the Streamlit Program: Go to the project directory in your command-line interface and execute the Streamlit program by using the following command:
Copy the code and execute the app.py
The filename of your Streamlit application script should be substituted for app.py.

Access the program: Following the command's execution, the program will launch and provide a local web address, such as http://localhost:8501. To access the installed crop prediction system on your local host, launch a web browser and type the given address.

Continuous Improvement: Track performance, keep an eye on user input, and make the required adjustments to the application and prediction model. If there are any improvements or bugs, update the code and restart Streamlit.

5.9 OUTCOMES:

Increased Yield and Productivity: Farmers can choose the right crops, plant them at the right times, and use the best cultivation techniques by properly forecasting crop yields. This may result in better crop management, better resource allocation, and eventually higher productivity and yields.

Risk Mitigation: Crop prediction systems assist farmers in reducing the risks brought on by pests, and diseases. Farmers can reduce possible losses and safeguard their harvests by getting early suggestions on crop varieties, pest management techniques, and disease control methods.

Resource Optimization: Crop recommendation systems utilize resources like water, fertilizer, and pesticides in the most efficient way possible. Farmers may minimize resource waste and lessen environmental effect by receiving exact suggestions based on crop requirements and environmental considerations.

Cost reduction: Effective crop advice can assist farmers in lowering expenses. Farmers can save costs while preserving the best possible crop growth and health by avoiding overusing or underusing inputs.

Increased Sustainability: Crop prediction and advice systems encourage environmentally friendly agriculture methods. These methods promote water conservation, and biodiversity preservation by recommending crop rotations, cover crops, and conservation practises.

Market Planning: Predictions of agricultural yields that are accurate might help farmers plan their marketing tactics. Farmers may decide on pricing, supply chain management, and market timing to maximize profitability by being aware of the anticipated yields in advance.

Decision Support: Data-driven decision help is offered to farmers by crop recommendation systems. These systems help farmers make the best decisions possible throughout the agricultural production cycle by taking into account many variables such as weather patterns, soil conditions, and market demands.

Efficiency Gains: Crop prediction and recommendation procedures are more operationally efficient thanks to automation and digitization. In order to save time and effort, farmers can access real-time data, get immediate recommendations, and streamline their operations.

Development of Sustainable Agriculture: The results of crop prediction and recommendation systems taken as a whole contribute to the overall development of sustainable agriculture. These systems make it easier to optimize resources, reduce risks, and make data-driven decisions, resulting in more resilient and ecologically friendly agricultural practises.

5.10 Implementation Screenshots

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression,LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.metrics import f1_score,classification_report,r2_score,mean_squared_error
from sklearn.metrics import accuracy_score,precision_score,recall_score,mean_absolute_error
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import streamlit as st
from PIL import Image
from streamlit_option_menu import option_menu
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
import base64
from sklearn.tree import DecisionTreeRegressor
```

Figure 5.2: Necessary Modules For the Application

Pandas: It is an open source library used to import the crop dataset and crop yield dataset required for the crop yield prediction. It is used to open,read and close .csv file

Numpy: It is used to arrange the data which is in different DataFrame and matching it with appropriate modules.The Data is stored in a .csv file which is stored as dataframe,we need this module to manipulate the dataframe

Sklearn: It is simple efficient tool for predictive data analysis and for importing various inbuilt algorithms and displaying the accuracy metrics such as accuracy_score, precision_score, recall_score, Classification report.

Streamlit : An open-source Python library called Streamlit is used to create web apps for data science and machine learning projects. It enables programmers to design interactive dashboards, data visualizations, and user interfaces quickly and easily.
The application by default runs on port 8501.

Base64 : Encoding binary to ASCII.This module is used to convert the binary image data to Ascii for processing

```
if option == 'CROP RECOMMENDER':
    dataset1 = pd.read_csv('crop.csv')
    dataset1['label'] = dataset1['label'].map({'rice': '1',
                                               'maize': '2',
                                               'chickpea': '3',
                                               'kidneybeans': '4',
                                               'pigeonpeas': '5',
                                               'mothbeans': '6'})
    x1 = dataset1.iloc[:, :-1].values
    y1 = dataset1.iloc[:, -1].values
    x1_train, x1_test, y1_train, y1_test = train_test_split(x1, y1, test_size=0.25, random_state=0)
    lr1 = SVC()
    lr2 = KNeighborsClassifier()
    lr3= GaussianNB()
    lr1.fit(x1_train, y1_train)
    lr2.fit(x1_train, y1_train)
    lr3.fit(x1_train, y1_train)
    lr1_pred = lr1.predict(x1_test)
    lr2_pred = lr2.predict(x1_test)
    lr3_pred = lr3.predict(x1_test)
    col1, col2, col3 = st.columns(3)
    with col1:
        n = st.number_input('Nitrogen in mg/Kg', step=1,max_value=100,min_value=10)
    with col2:
        p = st.number_input('Phosphurus  in mg/Kg', step=1,max_value=100,min_value=10)
    with col3:
        k = st.number_input('Potassium in mg/Kg', step=1,max_value=50,min_value=0)
    with col1:
        temp = st.number_input('Temperature IN Deg(C)', min_value=5,max_value=45)
    with col2:
        ph = st.number_input('ENTER PH VALUE')
        humidity = st.number_input('ENTER HUMIDITY IN g/m^3 ')
    with col3:
        rainfall = st.number_input('ENTER RAINFALL IN mm')
    with col2:
        sub1 = st.button('RECOMMEND THE BEST CROP FOR ME')
    new1 = [[n, p, k, temp, ph, humidity, rainfall]]
    if sub1:
        dict = {'1': 'White Rice : Oryza sativa', '2': 'Maize : Zea mays', '3': 'Chick peas : Cicer arietinum',
                '4': 'Kidney beans : Phaseolus vulgaris', '5': 'Pigeon peas : Cajanus cajan',
                '6': 'Moth beans: Vigna aconitifolia'}
        lr1_res = lr1.predict(new1)
        lr1_res1 = lr2.predict(new1)
        lr1_res2 = lr3.predict(new1)
        st.header('THE BEST CROP RECOMMENDED FOR YOU BY THE MODEL IS(svm) : ')
        st.success(dict[lr1_res[0]])
        st.header('THE BEST CROP RECOMMENDED FOR YOU BY THE MODEL IS(knn) : ')
        st.success(dict[lr1_res1[0]])
        st.header('THE BEST CROP RECOMMENDED FOR YOU BY THE MODEL IS(nvb) : ')
```

Figure 5.3: Crop recommendation interface

Figure 5.2 is the snippet which creates an interface where crop.csv file is read and is labelled with the crop names(rice,maize,chickpea etc).Further this data is classified with three different classifiers namely 1)SVC, 2)KNN, 3)Naïve Bayes.

These classifiers get their input through the streamlit interface. The corresponding temperature,humidity,ph values are obtained from the input form and are fed to the classifiers

Classifiers provides the result accordingly,and based on their output we can choose the best crop.

```
elif option == 'YIELD PREDICTOR':
    dataset2 = pd.read_csv("YIELD & PROFIT.csv")
    temp = pd.read_csv("YIELD & PROFIT.csv")
    print(dataset2['Crop'].unique())

    a1 = dataset2['State_Name'].unique().tolist()
    b1 = np.arange(1, len(a1) + 1, 1)
    c1 = {i: j for i, j in zip(a1, b1)}
    dataset2['State_Name'] = dataset2['State_Name'].map(c1)
    print(c1.keys())

    a2 = dataset2['District_Name'].unique().tolist()
    b2 = np.arange(1, len(a2) + 1, 1)
    c2 = {i: j for i, j in zip(a2, b2)}
    dataset2['District_Name'] = dataset2['District_Name'].map(c2)
    print(c2.values())

    a3 = dataset2['Season'].unique().tolist()
    b3 = np.arange(1, len(a3) + 1, 1)
    c3 = {i: j for i, j in zip(a3, b3)}
    dataset2['Season'] = dataset2['Season'].map(c3)

    a4 = dataset2['Crop'].unique().tolist()
    b4 = np.arange(1, len(a4) + 1, 1)
    c4 = {i: j for i, j in zip(a4, b4)}
    dataset2['Crop'] = dataset2['Crop'].map(c4)

    dataset2 = dataset2.dropna()

    col1, col2, col3 = st.columns(3)
    with col1:
        state1 = st.selectbox('ENTER STATE NAME : ', options=c1)
        cs = state1
        state1 = c1[state1]
    with col2:
        cd = temp[temp["State_Name"] == cs]
        c22 = cd["District_Name"].unique()
        district1 = st.selectbox('ENTER DISTRICT NAME : ', options=sorted(c22))
        district1 = c2[district1]
    with col3:
        crop_year1 = st.number_input('ENTER CROP YEAR : ', min_value=2015, step=1)
        c32 = c3
        season1 = st.selectbox('ENTER SEASON : ', options=sorted(c32))
    with col1:
        season1 = c3[season1]
        area1 = st.number_input('ENTER CULTIVATING AREA in acres : ', min_value=1, step=1)
    with col2:
        c42 = c4
        crop1 = st.selectbox('ENTER CROP : ', options=sorted(c42))
```

Figure 5.4: Yield Prediction

Figure 5.3 shows the code for Yield prediction which maps the state name with its corresponding districts name.

The function returns the best crop for the particular area based on the parameters such as state name,district name.

```
    with col1:
        season1 = c3[season1]
        area1 = st.number_input('ENTER CULTIVATING AREA in acres : ', min_value=1, step=1)
    with col2:
        c42 = c4
        crop1 = st.selectbox('ENTER CROP : ', options=sorted(c42))
        crop1 = c4[crop1]
        sub2 = st.button('PREDICT PRODUCTION OF CROP')

    new2 = [[state1, district1, crop_year1, season1, crop1, area1]]

    if sub2:
        x2 = dataset2.iloc[:, :6].values
        y2 = dataset2.iloc[:, 6].values
        x2_train, x2_test, y2_train, y2_test = train_test_split(x2, y2, test_size=0.1, random_state=11000)
        lr = DecisionTreeRegressor()
        lr.fit(x2_train, y2_train)
        lr_pred = lr.predict(x2_test)
        lr_res = lr.predict(new2)

        st.header('PREDICTED GROSS yeild in tons: ')
        st.success((int(lr_res[0])))

        print("MAE : {0}".format(round(mean_absolute_error(y2_test, lr_pred), 2)))
        print("MSE : {0}".format(round(mean_squared_error(y2_test, lr_pred), 4)))
        print("RMSE  : {0}".format(round(np.sqrt(mean_squared_error(y2_test, lr_pred)), 2)))
        print("R2 score  : {0}".format(r2_score(y2_test, lr_pred) * 100))
        st.header('REPORT')
        report1 = dataset2.describe()
        report1_df = pd.DataFrame(report1).transpose()
        st.dataframe(report1_df, height=212, width=750)
```

Figure 5.5 : Accuracy Metrics

Figure 5.4 shows the accuracy of the output given by the model.This result is printed on the console for developers reference,further this can alsbo be used to genarate a detailed report to give for farmers.

Mean Squared Error(MSE): the typical squared discrepancy between the estimated and actual values

Mean Absolute Error(MAE): A measure of mistakes between paired observations describing the same phenomenon is called mean absolute error (MAE).

Root Mean Squared Error(RMSE): It demonstrates how far predictions depart from actual measurements using Euclidean distance.

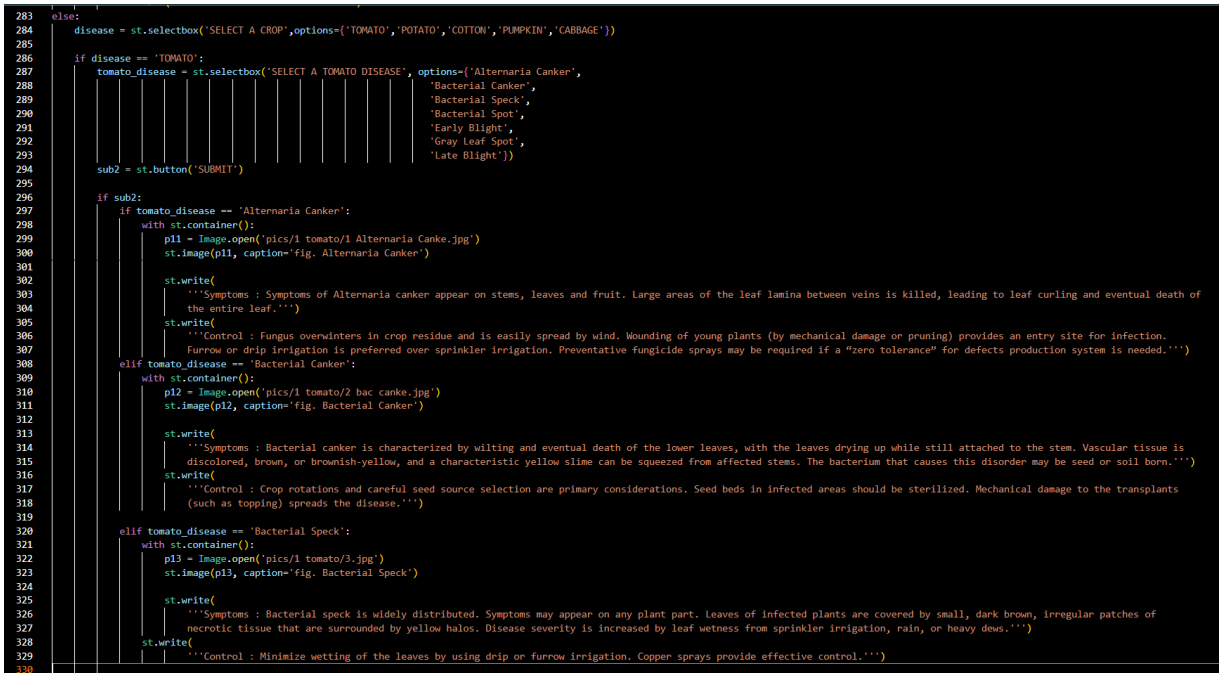
```
elif option == 'PROFIT PREDICTOR':
    dataset2 = pd.read_csv("YIELD & PROFIT.csv")
    dataset4 = pd.read_csv("YIELD & PROFIT.csv")
    temp = pd.read_csv("YIELD & PROFIT.csv")

    a1 = dataset2['State_Name'].unique().tolist()
    b1 = np.arange(1, len(a1) + 1, 1)
    c1 = {i: j for i, j in zip(a1, b1)}
    dataset2['State_Name'] = dataset2['State_Name'].map(c1)
    dataset4['State_Name'] = dataset4['State_Name'].map(c1)

    a2 = dataset2['District_Name'].unique().tolist()
    b2 = np.arange(1, len(a2) + 1, 1)
    c2 = {i: j for i, j in zip(a2, b2)}
    dataset2['District_Name'] = dataset2['District_Name'].map(c2)
    dataset4['District_Name'] = dataset4['District_Name'].map(c2)

    a3 = dataset2['Season'].unique().tolist()
    b3 = np.arange(1, len(a3) + 1, 1)
    c3 = {i: j for i, j in zip(a3, b3)}
    dataset2['Season'] = dataset2['Season'].map(c3)
    dataset4['Season'] = dataset4['Season'].map(c3)

    a4 = dataset2['Crop'].unique().tolist()
    b4 = np.arange(1, len(a4) + 1, 1)
    c4 = {i: j for i, j in zip(a4, b4)}
    dataset2['Crop'] = dataset2['Crop'].map(c4)
    dataset4['Crop'] = dataset4['Crop'].map(c4)

    dataset2 = dataset2.dropna()
    dataset4 = dataset4.dropna()

    dataset2.loc[dataset2['Profit'] <= 0, 'result'] = '0'
    dataset2.loc[dataset2['Profit'] > 0, 'result'] = '1'

    x3 = dataset2.iloc[:, :6].values
    y3 = dataset2.iloc[:, -1].values

    x3_train, x3_test, y3_train, y3_test = train_test_split(x3, y3, test_size=0.2, random_state=50000)
    sc = StandardScaler()
    x3_train = sc.fit_transform(x3_train)
    x3_test = sc.transform(x3_test)
    knn = KNeighborsClassifier()
    knn.fit(x3_train, y3_train)
    knn_pred = knn.predict(x3_test)

    dict2 = {'1': 'YOU HAVE CHOSEN A PROFITABLE CROP', '0': 'YOU HAVE CHOSEN A LOSSY CROP'}

    col1, col2, col3 = st.columns(3)
```

Figure 5.6: Profit Prediction(1)

```
    col1, col2, col3 = st.columns(3)
    with col1:
        state2 = st.selectbox('ENTER STATE NAME ', options=c1)
        state21=state2
        state2 = c1[state2]
    with col2:
        temp2=temp[temp["State_Name"]==state21]
        c21= temp2["District_Name"].unique()
        district2 = st.selectbox('ENTER DISTRICT NAME', options=sorted(c21))
        district2 = c2[district2]
    with col3:
        crop_year2 = st.number_input('ENTER CROP YEAR', min_value=1999, step=1)
        c31=c3
        season2 = st.selectbox('ENTER SEASON', options=sorted(c31))
    with col1:
        season2 = c3[season2]
        area2 = st.number_input('ENTER CULTIVATING AREA in acres', min_value=1, step=1)
    with col2:
        c41= c4
        crop2 = st.selectbox('ENTER CROP', options=sorted(c41))
        crop2 = c4[crop2]
        sub3 = st.button('PREDICT PROFITABILITY OF MY CROP')

    new3 = [[state2, district2, crop_year2, season2, crop2, area2]]

    x4 = dataset4.iloc[:, :6].values
    y4 = dataset4.iloc[:, -1].values
    x4_train, x4_test, y4_train, y4_test = train_test_split(x4, y4, test_size=0.1, random_state=11000)
    lr = DecisionTreeRegressor()
    lr.fit(x4_train, y4_train)
    lr_pred1 = lr.predict(x4_test)
    lr_res2 = lr.predict(new3)

    if sub3:
        knn_res = knn.predict(sc.transform(new3))
        st.header('PREDICTED GROSS YIELD : ')
        if(int(lr_res2[0])<=0):
            st.error('YOU HAVE CHOSEN A LOSSY  CROP')
            st.error('Rs ' + str(int(lr_res2[0])))
        else:
            st.success('YOU HAVE CHOSEN A PROFITABLE CROP')
            st.success('Rs ' + str(int(lr_res2[0])))
        st.header('MODEL PARAMETERS')
        st.success("ACCURACY : {0}%".format(round(accuracy_score(y3_test, knn_pred), 2) * 100))
        print("PRECISION : {0}%".format(round(precision_score(y3_test, knn_pred, pos_label='1'), 2) * 100))
        print("F1 SCORE  : {0}%".format(round(f1_score(y3_test, knn_pred, pos_label='1'), 2) * 100))
        print("R2 SCORE  : {0}%".format(round(recall_score(y3_test, knn_pred, pos_label='1'), 2) * 100))
```

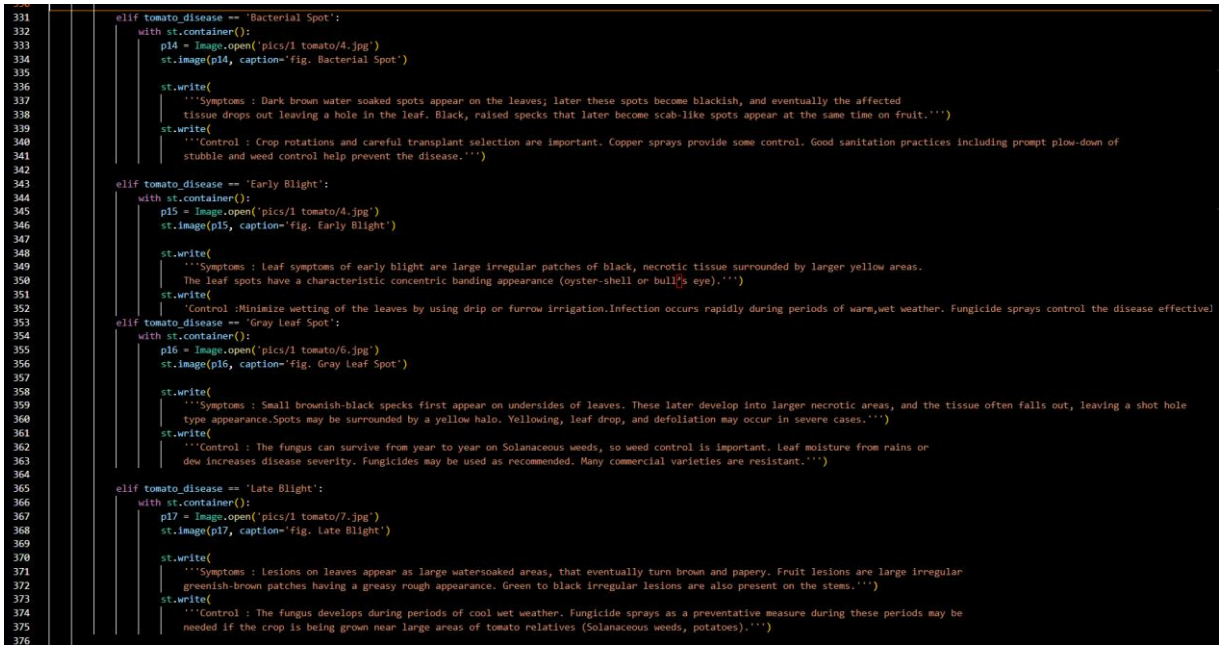
Figure 5.7: Profit Prediction(2)

Figure 5.5and 5.6 shows a code for predicting the profit. This function takes the crop chosen by a farmer,locality,and year as parameters and returns whether the chosen crop if grown results in profit or loss .



```
else:
    disease = st.selectbox('SELECT A CROP',options={'TOMATO','POTATO','COTTON','PUMPKIN','CABBAGE'})

    if disease == 'TOMATO':
        tomato_disease = st.selectbox('SELECT A TOMATO DISEASE', options={'Alternaria Canker',
                                                                          'Bacterial Canker',
                                                                          'Bacterial Speck',
                                                                          'Bacterial Spot',
                                                                          'Early Blight',
                                                                          'Gray Leaf Spot',
                                                                          'Late Blight'})
        sub2 = st.button('SUBMIT')

        if sub2:
            if tomato_disease == 'Alternaria Canker':
                with st.container():
                    p11 = Image.open('pics/1 tomato/1 Alternaria Canke.jpg')
                    st.image(p11, caption='fig. Alternaria Canker')

                    st.write(
                        '''Symptoms : Symptoms of Alternaria canker appear on stems, leaves and fruit. Large areas of the leaf lamina between veins is killed, leading to leaf curling and eventual death of
                        the entire leaf.''')
                    st.write(
                        '''Control : Fungus overwinters in crop residue and is easily spread by wind. Wounding of young plants (by mechanical damage or pruning) provides an entry site for infection.
                        Furrow or drip irrigation is preferred over sprinkler irrigation. Preventative fungicide sprays may be required if a "zero tolerance" for defects production system is needed.''')
            elif tomato_disease == 'Bacterial Canker':
                with st.container():
                    p12 = Image.open('pics/1 tomato/2 bac canke.jpg')
                    st.image(p12, caption='fig. Bacterial Canker')

                    st.write(
                        '''Symptoms : Bacterial canker is characterized by wilting and eventual death of the lower leaves, with the leaves drying up while still attached to the stem. Vascular tissue is
                        discolored, brown, or brownish-yellow, and a characteristic yellow slime can be squeezed from affected stems. The bacterium that causes this disorder may be seed or soil born.''')
                    st.write(
                        '''Control : Crop rotations and careful seed source selection are primary considerations. Seed beds in infected areas should be sterilized. Mechanical damage to the transplants
                        (such as topping) spreads the disease.''')

            elif tomato_disease == 'Bacterial Speck':
                with st.container():
                    p13 = Image.open('pics/1 tomato/3.jpg')
                    st.image(p13, caption='fig. Bacterial Speck')

                    st.write(
                        '''Symptoms : Bacterial speck is widely distributed. Symptoms may appear on any plant part. Leaves of infected plants are covered by small, dark brown, irregular patches of
                        necrotic tissue that are surrounded by yellow halos. Disease severity is increased by leaf wetness from sprinkler irrigation, rain, or heavy dews.''')
                    st.write(
                        '''Control : Minimize wetting of the leaves by using drip or furrow irrigation. Copper sprays provide effective control.''')
```

Figure 5.8: Image Encyclopedia 1



```
            elif tomato_disease == 'Bacterial Spot':
                with st.container():
                    p14 = Image.open('pics/1 tomato/4.jpg')
                    st.image(p14, caption='fig. Bacterial Spot')

                    st.write(
                        '''Symptoms : Dark brown water soaked spots appear on the leaves; later these spots become blackish, and eventually the affected
                        tissue drops out leaving a hole in the leaf. Black, raised specks that later become scab-like spots appear at the same time on fruit.''')
                    st.write(
                        '''Control : Crop rotations and careful transplant selection are important. Copper sprays provide some control. Good sanitation practices including prompt plow-down of
                        stubble and weed control help prevent the disease.''')

            elif tomato_disease == 'Early Blight':
                with st.container():
                    p15 = Image.open('pics/1 tomato/4.jpg')
                    st.image(p15, caption='fig. Early Blight')

                    st.write(
                        '''Symptoms : Leaf symptoms of early blight are large irregular patches of black, necrotic tissue surrounded by larger yellow areas.
                        The leaf spots have a characteristic concentric banding appearance (oyster-shell or bull's eye).''')
                    st.write(
                        'Control :Minimize wetting of the leaves by using drip or furrow irrigation.Infection occurs rapidly during periods of warm,wet weather. Fungicide sprays control the disease effectivel
            elif tomato_disease == 'Gray Leaf Spot':
                with st.container():
                    p16 = Image.open('pics/1 tomato/6.jpg')
                    st.image(p16, caption='fig. Gray Leaf Spot')

                    st.write(
                        '''Symptoms : Small brownish-black specks first appear on undersides of leaves. These later develop into larger necrotic areas, and the tissue often falls out, leaving a shot hole
                        type appearance.Spots may be surrounded by a yellow halo. Yellowing, leaf drop, and defoliation may occur in severe cases.''')
                    st.write(
                        '''Control : The fungus can survive from year to year on Solanaceous weeds, so weed control is important. Leaf moisture from rains or
                        dew increases disease severity. Fungicides may be used as recommended. Many commercial varieties are resistant.''')

            elif tomato_disease == 'Late Blight':
                with st.container():
                    p17 = Image.open('pics/1 tomato/7.jpg')
                    st.image(p17, caption='fig. Late Blight')

                    st.write(
                        '''Symptoms : Lesions on leaves appear as large watersoaked areas, that eventually turn brown and papery. Fruit lesions are large irregular
                        greenish-brown patches having a greasy rough appearance. Green to black irregular lesions are also present on the stems.''')
                    st.write(
                        '''Control : The fungus develops during periods of cool wet weather. Fungicide sprays as a preventative measure during these periods may be
                        needed if the crop is being grown near large areas of tomato relatives (Solanaceous weeds, potatoes).''')
```

Figure 5.9: Image Encyclopedia 2

Figure 5.7 & 5.8 shows the code for image encyclopedia. Image encyclopedia accepts an name of a crop, then shows the various disease that particular crop can get affected by and tells the preventive measures

CHAPTER-6
TESTING

CHAPTER-6

TESTING

Software testing is a process of inspecting a product or service to inform stakeholders of its quality. Software implementation risks can be better understood and appreciated by the business by providing it with an unbiased perspective on the program through the use of program testing. One of the test approaches is the practice of running a program or application with the intention of finding software defects (errors or other problems). Running a software or system component under test involves evaluating one or more useful properties. These qualities demonstrate how thoroughly the system or component is tested:

- Meets the requirements that guided its design and development
- It satisfies the overall goals of its stakeholders, responds accurately to all inputs, completes its tasks in a reasonable amount of time, is sufficiently useable
- can be installed in and used in the intended contexts.

All software testing employs some sort of method to choose tests that are doable given the time and resources available because the number of potential tests for even the simplest software components is almost unlimited. In order to detect software bugs (errors or other problems), software testing typically (but not always) tries to execute a program or application. Testing is an iterative process since after one bug is fixed, it may reveal deeper bugs or even reveal new ones.

Program testing can give users and/or sponsors unbiased, independent information on the caliber of program and danger of failure. As soon as there is executable software (even if it is only partially developed), testing can begin. When and how testing is done are frequently dictated by the general approach taken to software development. For instance, in a staged approach, the majority of testing happens following the definition and implementation of the system requirements in testable programs. In contrast, an agile strategy frequently combines requirements, programming, and testing.

THE MAIN AIM OF TESTING

Testing's primary objectives are performance analysis and error evaluation when the program is run in various operating conditions and with a variety of input sources.We created a crop recommendation and crop yield prediction in this project to aid in crop suggestion and help farmers increase their profit. The primary goals of this project's testing are to determine whether the suggested crop is accurate and to evaluate how well it performs when various environmental parameters are used as inputs.

The steps in testing are:

- Unit Testing.
- Integration Testing.
- Validation Testing.
- User Acceptance Testing.
- Output Testing

6.1 UNIT TESTING:

Tests that confirm the functionality of a particular area of code, typically at the function level, are referred to as unit testing, sometimes known as component testing. This occurs typically at the class level in an object-oriented system, and the constructors and destructors are covered by the bare minimum unit tests. In order to lower the risks, expenses, and duration of software development, unit testing is a software development method that comprises the coordinated use of a wide range of defect prevention and detection strategies. The functions that were tested at the time of programming are displayed in the following unit testing table. All the modules that were tested are listed in the first column, and the test results are listed in the second. The results of the tests show whether the functions are producing correct results for the specified inputs.

Name of Function Tests Outcomes crop dataset parameters and values are uploaded. crop advice using testing several agricultural metrics under various circumstances. If the message is correctly shown, check the output.

Function Name	Tests Results
Uploading csv data	Tested for uploading data set and model.
Crop Prediction	For the uploaded model file, Crops yield and market prices are predicted
Display result	Output is displayed successfully

Table 6.1: Unit Testing

6.2 INTEGRATION TESTING:

Any sort of software testing that aims to validate the interfaces between components in comparison to a software design is known as integration testing. Software components can either be put together incrementally or all at once ("big bang"). The former is typically seen as a better practice because it makes it possible to identify and address interface problems more rapidly. Defects in the interfaces and communication between integrated components (modules) are revealed through integration testing. Up until the software functions as a system, ever-larger groupings of tested software components that match to components of the architectural design are merged and tested.

6.3 VALIDATION TESTING

After integration testing, the software is finished and put together as a whole. Interfacing mistakes have been found and fixed. Numerous definitions exist for validation testing; in this instance, the testing verifies that the software performs as the client may reasonably anticipate. Verification and validation (V&V) is the process of ensuring that a software system complies with requirements and serves the intended purpose in the contexts of software project management, software testing, and software engineering. Another name for it is software quality control.

6.4 USER ACCEPTANCE TESTING

An acceptance test's performance is really the user's performance. User motivation and expertise are essential for the system to operate well. The newly constructed system performed as expected during the tests mentioned above. The following test case designs were used to execute all the aforementioned testing methodologies.

6.5 TEST CASES

Data set collection Test case:

Test Case	1
Name of Test	**Data set Collection**
Input	Collection of crop Parameter dataset
Expected output	Csv data format file
Actual output	Csv data format file
Result	Successful

Table 6.2: Data Set Collection Test Case

Training Test case:

Test Case	1
Name of Test	**Training**
Input	Load the dataset from dataset folder i.e csv file
Expected output	Training of csv file to get model file
Actual output	Training of csv file to get model file
Result	Successful

Table 6.3: Training Test case

Prediction or Recommendation Test case:

Test Case	1
Name of Test	**Prediction or Recognition**
Input	Parameters
Expected output	Ragi
Actual output	Ragi
Result	Successful

Table 6.4: Prediction or Recommendation Test case-1

Test Case	2
Name of Test	**Prediction or Recognition**
Input	Parameters
Expected output	Maize
Actual output	Maize
Result	Successful

Table 6.5: Prediction or Recommendation Test case-2

CHAPTER-7
EXPERIMENTAL RESULTS

CHAPTER-7

EXPERIMENTAL RESULTS

PROJECT OUTCOMES

7.1. CROP RECOMMENDATION

This seamless UI lets the user connect to backend to predict the result.

Ui components-

- Form Parameters- The user must fill in the form that is displayed on the page,
  The data collected from this form is passed into the model to predict the most likely outcome based on the given parameters.
    1. **Nitrogen** - Enter the nitrogen amount found by testing the soil sample in mg/kg with min value = 0 and max value =100
    2. **Potassium** - Enter the potassium amount found by testing the soil sample in mg/kg with min value = 0 and max value =50
    3. **Phosphorus** - Enter the Phosphorus amount found by testing the soil sample in mg/kg with min value = 0 and max value =100
    4. **Rainfall** - Entering the annual rainfall in mm
    5. **Temperature**- Entering the Temperature in degree Celsius
    6. **Ph**- Enter the pH of soil ,
    7. **Humidity**- Entering the moisture content in atmosphere in gm/m^3

- OUTPUT- The output is predicted by the model (SVM),KNN model and Naïve Bayes Model is displayed in the output section
  The Accurate Prediction of crop is decided based upon the accuracy values displayed in the output section.

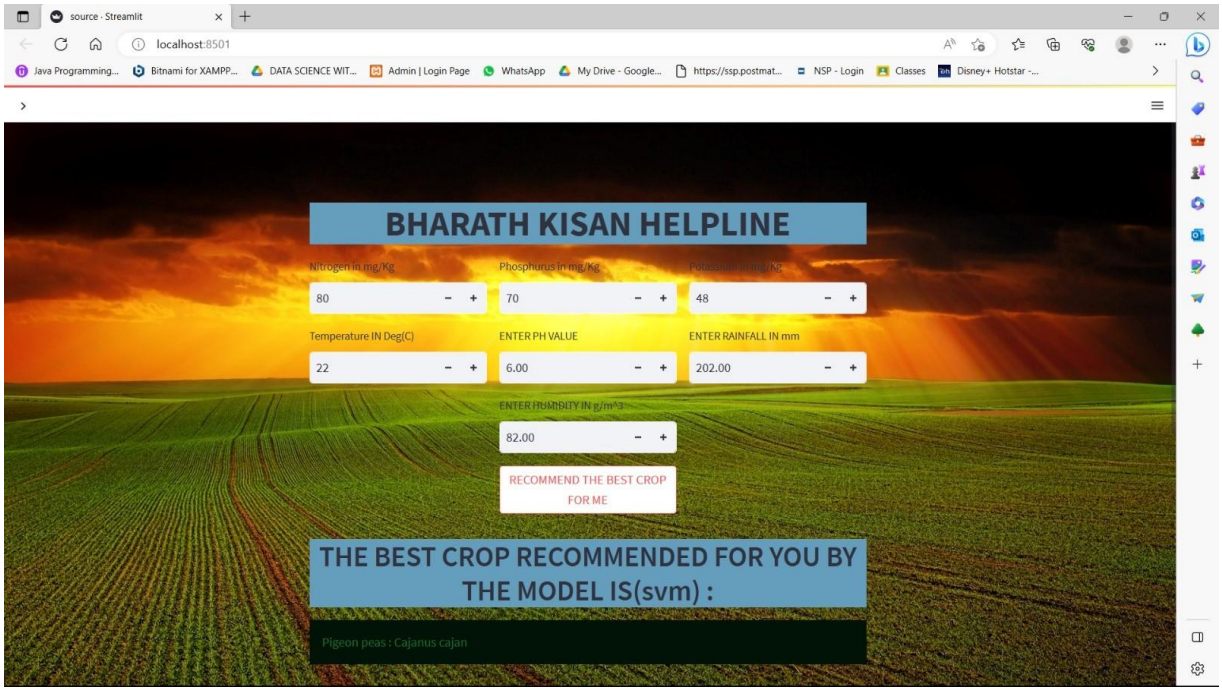


Figure 7.1: Home page of Crop Recommendation System

7.2. YIELD PREDICTOR

This function predicts the gross yield (in tons) for the selected crop.

Ui components-

- Parameter form-

  1.STATE - Enter the state name.

  2.DISCTICT- Enter the name of the district.

  3.YEAR- select the year for prediction.

  4. AREA- enter the cultivation area in acres.

  5. CROP- name of the crop.

  6.SEASON- enter the season of planting.

- OUTPUT- The output is predicted by the model is displayed in the output section along with a report describing the dataset.

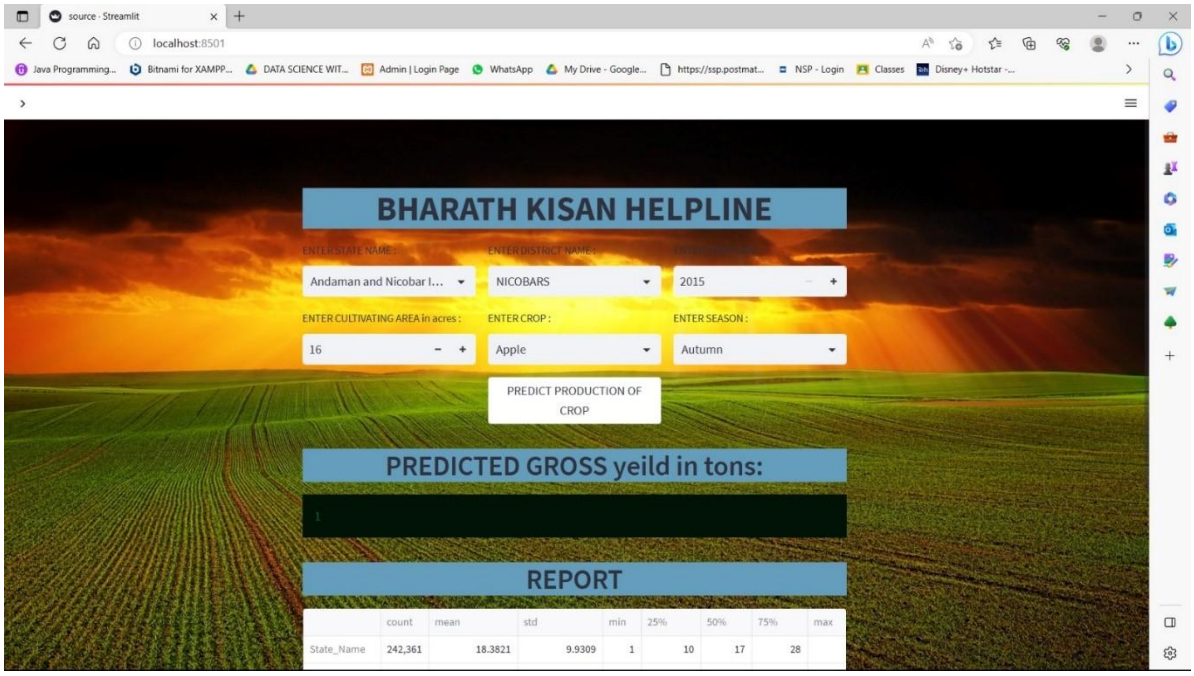


Figure 7.2: Home Page of Yield Predictor

7.3. PROFIT PREDICTOR

This function predicts the profit/loss incurred on planting a crop...

Ui components-

- Parameter form-

  1.STATE - Enter the state name.

  2.DISCTICT- Enter the name of the district.

  3.YEAR- select the year for prediction.

  4. AREA- enter the cultivation area in acres.

  5. CROP- name of the crop.

  6.SEASON- enter the season of planting.

- OUTPUT- Displays whether the crop is a loss/profit along with the numbers to prove it.

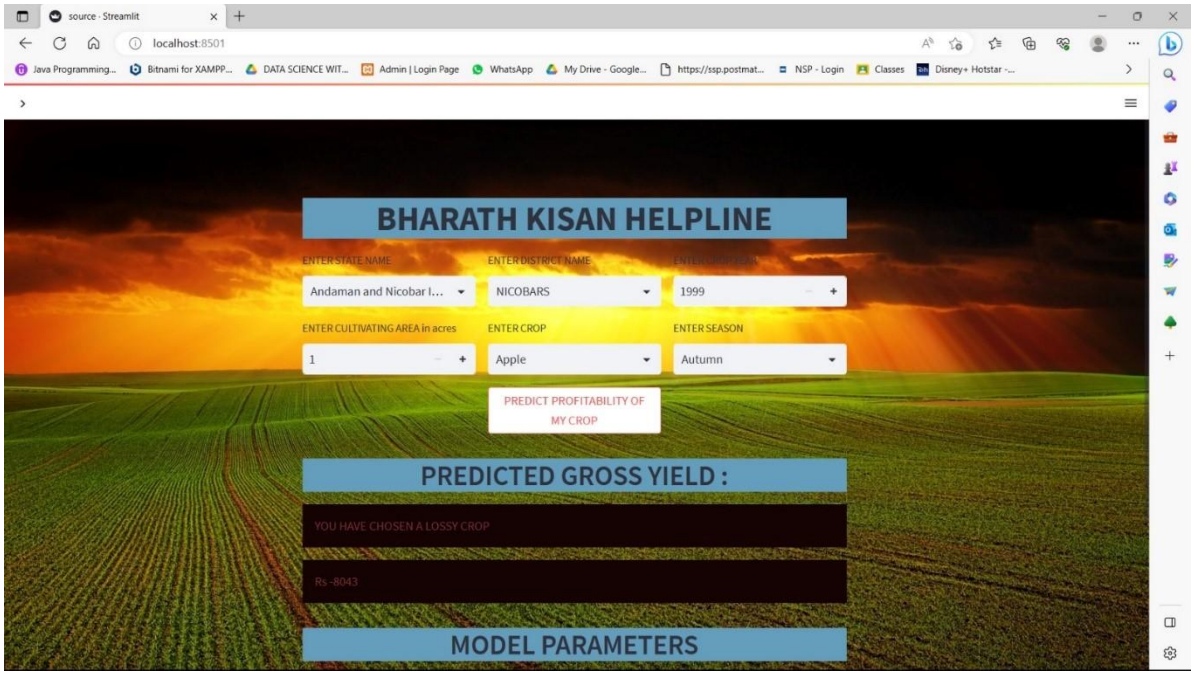


Figure 7.3: Home Page of Profit Predictor

7.4. DISEASE ENCYCLOPEDIA

This function displays information of various plant diseases.

Ui components-

FORM-

1.CROP – select the crop

2.DISEASE- Options of various disease that occurs in the selected crop.

OUTPUT- Displays images of the disease plant specimen along with the reason, prevention and cure.

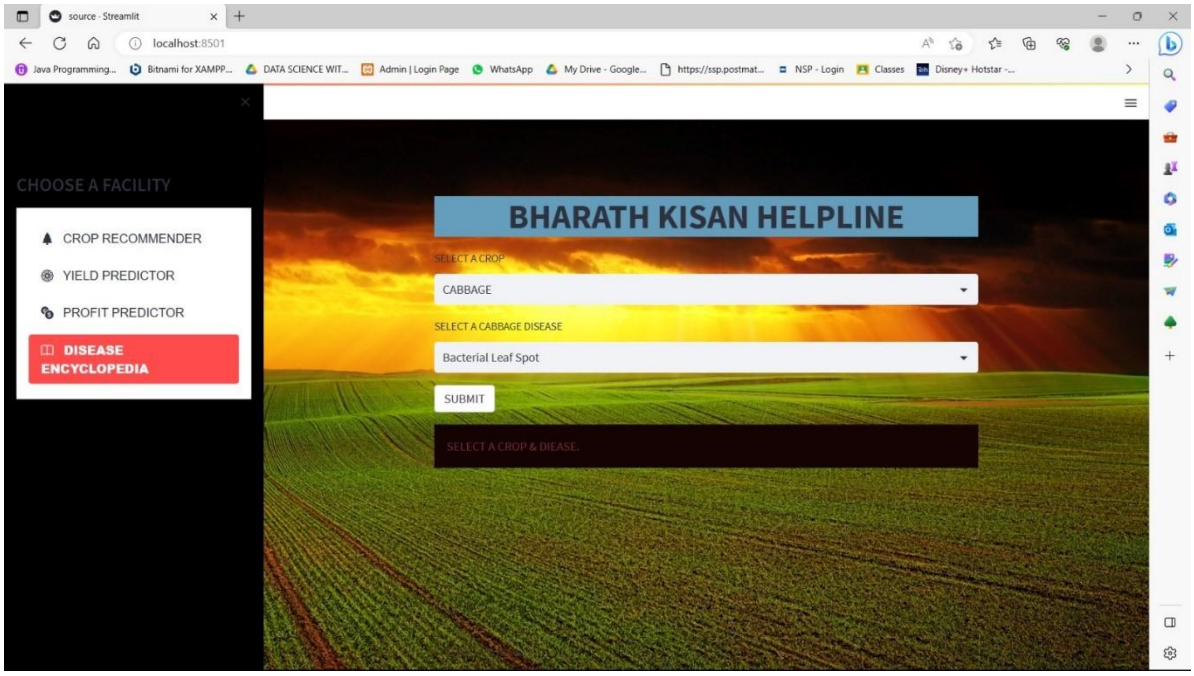


Figure 7.4: Home Page of Disease Encyclopedia

7.5 METRICS OUTPUTS

Below are the outcomes of the model metrics which can be interpreted to gather more information about the models and scope for improvement in the future.

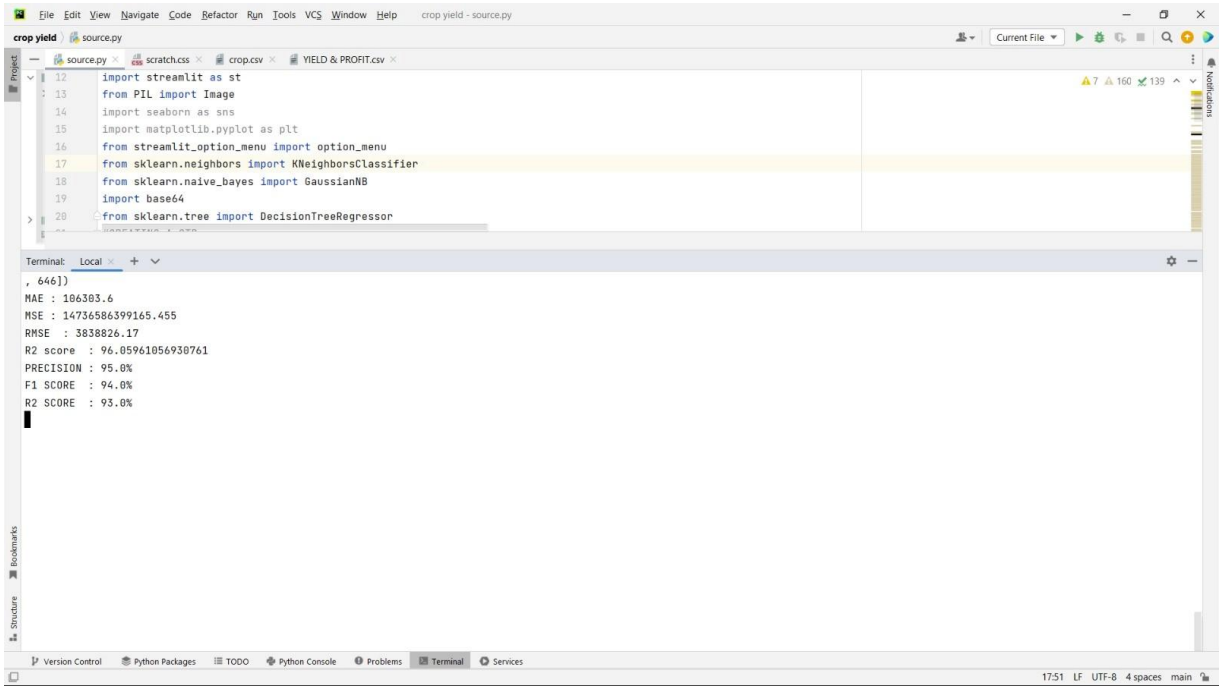


Figure 7.5: Unsupervised Learning Accuracy Parameters

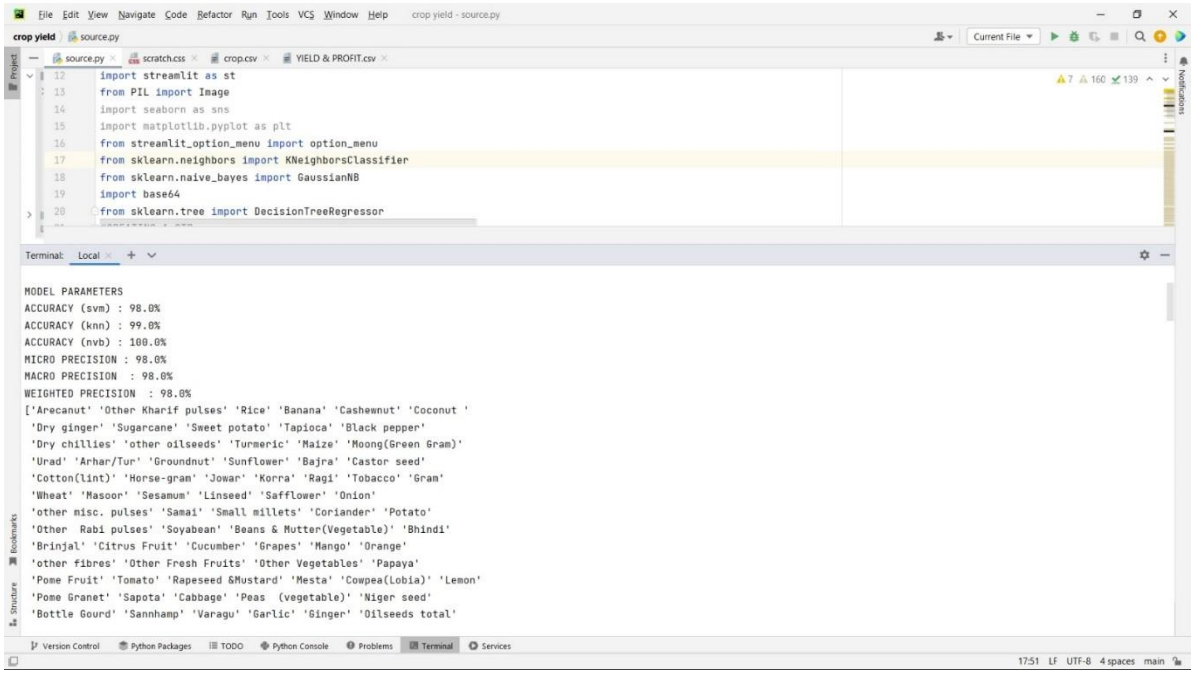


Figure 7.6: Supervised Learning Accuracy Parameters

CHAPTER-8
CONCLUSION

CHAPTER-8

CONCLUSION

The purpose of this project is to suggest and put into practice a piece of software that uses Python and machine learning to forecast agricultural yield production using historical data. Applying association rule mining to agricultural data from 1996 to 2015 has enabled this. This study shown that, depending on the scope of the research and the accessibility of data, the selected papers employ a number of features. Each paper examines yield prediction using machine learning, however the features vary. The scale, geological site, and crop of the research also vary. The dataset's accessibility and the study's objectives influence the attributes used.

Studies have also shown that models with additional features do not necessarily perform well for predicting yield. Models with more and fewer features should be evaluated in order to determine which one performs the best. Various research have employed a variety of algorithms. The findings indicate that it is impossible to pinpoint the optimal model, but they do make it very evident that some models of machine learning are employed more frequently than others.

FUTURE ENHANCEMENTS

Our current system just recommends crops, but this might be expanded in the future to include other things like fertilizer suggestions so that someone could learn which fertilizer is best for their particular crop. This might also be used to classify plant diseases using CNN and to provide a treatment for each illness. The potential for such a project is enormous because agriculture is a relatively uncharted territory. Although we have created a functional website for our idea, it may also be converted into an app in the future to make it more accessible to farmers. Regional language versions of the website and app can be created to make them more user-friendly.

We are Submitting a review paper on this project in the **"International Conference on Security, Surveillance and Artificial Intelligence"** organized by Techno India University West Bengal.

REFERENCES

[1] "Using machine learning techniques for crop yield prediction and climate change assessment." Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). PloS one, 11(5), e0153632. DOI: 10.1371/journal.pone.0153632

[2] "Crop yield prediction using machine learning algorithms." Neginhal, S. G., & Seetharam, M. R. (2018). In 2018 3rd International Conference on Computational Systems and Information Technology for Sustainable Solutions (CSITSS) (pp. 148-151). IEEE. DOI: 10.1109/CSITSS.2018.8526049

[3] "Machine learning approaches for crop yield prediction using weather parameters." Padhee, S., Mishra, S., Sharma, N., & Mukherjee, A. (2019). In 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA) (pp. 25-30). IEEE. DOI: 10.1109/ICBDA.2019.8747341

[4] "Crop yield prediction using machine learning techniques": Jain, D., Kumar, S., & Saini, R. (2020). In 2020 2nd International Conference on Advances in Sustainable Engineering and Applications (ASEA) (pp. 1-6). IEEE. DOI: 10.1109/ASEA51159.2020.9316733

[5] "Crop yield prediction using machine learning and optimization techniques:" Tanwar, A., Kumar, R., & Bhattacharya, S. (2021). A comprehensive review. Computers and Electronics in Agriculture, 184, 106042. DOI: 10.1016/j.compag.2021.106042

[6] "Improvised Extreme Learning Machine for Crop Yield Prediction"Swati Vashisht;Praveen Kumar;Munesh Chandra Trivedi2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)

[7] "Early Prediction of Crop Yield in India using Machine Learning"Ankita Sharma;Anushtha Tamrakar;Sourajita Dewasi;Nenavath Srinivas Naik2022 IEEE Region 10 Symposium (TENSYMP)

[8] "An Improved Machine Learning based Crop Recommendation System"B. Swathi Sri;G. Pavani;B.Y.S. Sindhuja;V. Swapna;P.L. Priyanka2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)

[9] "Accelerating Crop Yield: Multisensor Data Fusion and Machine Learning for Agriculture Text Classification"A. Reyana;Sandeep Kautish;P. M. Sharan Karthik;Ibrahim Ahmed Al-Baltah;Muhammed Basheer Jasser;Ali Wagdy Mohamed,2023,Volume:11,Journal Article

[10] "Crop Yield Prediction Enhancement Utilizing Deep Learning and Ensemble Algorithms" Ohnmar Khin;Sung Keun Lee 2022 13th International Conference on Information and Communication Technology Convergence (ICTC)

[11] “Analysis of Crop Yield Prediction using Machine Learning algorithms” VamsiKrishna; ThejeswarReddy;SaiHarsha;KaladeviRamar;ShanmugasundaramHariharan;Bhanuprasad A2022nd International Conference on Innovative Sustainable Computational Technologies (CISCT)

[12] “Crop prediction using machine learning” Arushi Singh**;N V** Subba Reddy**;**Dinesh U. Acharya