MULTI CLASS CLASSIFIER FOR CROP YIELD PREDICTION BASED ON NUTRIENTS FEATURES OF THE SOIL

Submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in Computer Science

by

PREETHI V (Reg.No - 42290074) SWATHI S (Reg.No - 42290116)

DEPARTMENT OF COMPUTER SCIENCE SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY (DEEMED TO BE UNIVERSITY) CATEGORY -1 UNIVERSITY BY UGC
Accredited with Grade "A++" by NAAC I 12B Status by UGC I Approved by AICTE JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL 2025

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
CATEGORY - 1 UNIVERSITY BY UGC
Accredited with Grade "A++" by NAAC I Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **PREETHI V (Reg.No - 42290074), SWATHI S (Reg.No - 42290116)** who carried out the project entitled "**MULTI CLASS CLASSIFIER FOR CROP YIELD PREDICTION BASED ON NUTRIENTS FEATURES OF THE SOIL"** under my / our supervision from November 2024 to April 2025.

INTERNAL GUIDE
P. DEVA SHAILU, MSC.(COMPUTER SCIENCE &TECHNOLOGY).

HEAD OF THE DEPARTMENT & DEAN
Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for viva voce Examination held on

Internal Examiner **External Examiner**

DECLARATION

We, **PREETHI V (Reg.No - 42290074), SWATHI S (Reg.No -42290116)** hereby declare that the Project Report entitled **"MULTI CLASS CLASSIFIER FOR CROP YIELD PREDICTION BASED ON NUTRIENTS FEATURES OF THE SOIL"** done by us under the guidance of **P.DEVA SHAILU MSC.(Computer Technology).** is submitted in partial fulfillment of the requirements for the award of Bachelor of Science degree in **Computer Science .**

DATE:

PLACE: Chennai **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

We pleased to acknowledge my sincere thanks to the **Board of Management** of **Sathyabama Institute of Science and Technology** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey my thanks to **Dr. Rekha Chakarvathi M.E., Ph.D**, **Dean**, **School of Science and Humanities,** and **Head of the Department of Computer Science** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project guide **P.DEVA SHAILU MSC.(Computer Technology),** Department of Computer Science for her valuable guidance, suggestions, and constant encouragement paved way for the successful completion of my project work.

We wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science** who were helpful in many ways for the completion of the project.

-

.

ABSTRACT

Agriculture serves as the backbone of many economies, providing food security and livelihoods to millions worldwide. However, traditional farming systems face several challenges, including unreliable crop selection methods and limited market access, which hinder productivity and profitability. Farmers often lack the tools to determine the most suitable crops for their land, leading to reduced yields. Additionally, the presence of intermediaries in the supply chain limits fair pricing for fresh produce, while access to quality fertilizers and pesticides tailored to specific needs remains a significant challenge. To address these issues, the project leverages cutting-edge Machine Learning and Artificial Intelligence technologies to revolutionize traditional farming practices. The system features a crop prediction module powered by the Support Vector Machine (SVM) algorithm, enabling farmers to make informed decisions based on soil properties, climate data, and historical trends. It includes a digital marketplace for fresh produce, allowing farmers to sell directly to consumers and businesses, ensuring fair pricing. A procurement platform for fertilizers and pesticides offers tailored recommendations, while an interactive farmers' forum fosters collaboration and knowledge sharing. Additionally, a plant leaf disease prediction feature, utilizing Convolutional Neural Networks (CNN), identifies diseases from uploaded images, providing timely intervention. The system further recommends appropriate fertilizers and pesticides based on disease diagnostics and crop needs. This proposed solution integrates real-time data analysis, user-friendly interfaces, and advanced algorithms to empower farmers, improve productivity, and promote sustainable agricultural practices. By addressing critical challenges, this project aims to transform the agricultural ecosystem, making it more efficient, profitable, and future-ready.

TABLE OF CONTENTS

PAGE NO

TITLE **ABSTRACT**

CHAPTER 1

INTRODUCTION

Crops are plants grown by the farmers. Agriculture plays a very important role in the Indian economy. It is the backbone of our country. 70% of the Indian population depends on agriculture for food and money. It is the major occupation in the rural areas. The cultivation of crops depends primarily on the weather and soil conditions. Animals and Plants are the two major sources of food. People have always collected plants and hunted animals to fulfil the needs of food and nutrition. Later, people who started agriculture became dependent on it for their nutritional needs. “Agriculture” is a complex term encompassing all the human activities in which the Earth’s resources are used appropriately to meet human needs for food, fiber, feed, fuel, etc. Therefore, the classification of crops has been done to utilize the resources properly.In agriculture, a crop is a plant that can be grown and harvested on a large scale for profit or subsistence purposes. The majority of crops are grown in agriculture or aquaculture. Among the crops are macroscopic fungi (such as mushrooms) and marine macroalgae (such as seaweed).

Most crops are harvested for human consumption or as feed for livestock. Wild crops are often collected in an intensive manner (e.g. ginseng, Yohimbe, and eucommia). Non-food crops include horticulture, floriculture, and industrial crops. The plants used in horticulture include those used in other crops (for example, fruit trees). Among the floriculture crops are bedding plants, houseplants, flowering gardens, pot plants, cultivated greens, and cut flowers. In the industrial world, crops are grown for clothing (fibre crops like cotton), biofuel (energy crops like algae), and medicine (medicinal plants).

1.1. Problem Statement

In farming, different weather factors such as Rainfall, temperature, and humidity play an important role. Due to pollution, sometimes climate varies abruptly, and hence it becomes difficult for farmers to make proper decisions for harvesting, sowing seeds, and soli preparing. For a better crop, it is necessary that the soil should be productive and have the required nutrition, such as Nitrogen, Phosphorous, and Potassium. If these nutrients are not present in effective way in the soil, then it may lead to poor quality crops. But it is difficult to identify these soil-quality with traditional ways. In the agriculture lifecycle, it is required that we save our crops from weeds. Else it may increase the production cost, and it also absorbs nutrients from the soil. But by traditional ways, identification and prevention of crop from weeds is not efficient. Farmers can barely talk about yields when bad weather occurs or crops are afflicted by illness. Or, if a worldwide epidemic strikes, managing numerous procedures becomes more difficult because most are not digital. At the same time, the world's population continues to rise, and urbanization continues. Consumption patterns are shifting as disposable income rises. Farmers are under a lot of pressure to meet rising demand, and they're looking for a method to boost output. We need to find strategies to assist farmers in reducing their risks, or at the very least making them more manageable. One of the most intriguing potentials is to use artificial intelligence in agriculture on a worldwide scale. AI has the potential to change the way we think about agriculture, enabling farmers to do more with less .

Aim and Objective

The aim of the project "Crops2Go: Web based Crops to grow Prediction with Temperature, Humidity, pH, Rainfall, Soil N P K Features using LSTM" is to develop a web-based application that predicts the best crops to grow based on climate, soil, and other relevant features. The project aims to provide an accurate and reliable crop recommendation system to farmers and other end-users, enabling them to make informed decisions on what to grow and when to plant.

The objectives of the project include:

1. Collecting relevant data on climate, soil, and other features that affect crop growth.
2. Pre-processing the data to remove null, missing values, redundant data, and misspelled data.

3. Extracting relevant features using the confusion matrix algorithm to identify the most important variables affecting crop yield.
4. Building a classification model using LSTM to predict the best crops to grow based on the extracted features.
5. Providing an intuitive web interface for end-users to interact with the application.
6. Integrating the application with relevant notification systems to provide farmers with real-time alerts and recommendations.
7. Evaluating the performance of the model and continuously improving the system to ensure accuracy and reliability.

This can help farmers to make informed decisions about crop selection and optimize their yield and profits. The project involves data collection, pre-processing, feature extraction, classification using LSTM, and performance analysis.

1.2. Scope of the Project

The scope of the project is to develop a web-based application that predicts the crop that can be grown in a particular region based on various environmental factors such as temperature, humidity, pH, rainfall, and soil nutrients like nitrogen, phosphorus, and potassium. The web application will be user-friendly, and end-users will be able to input environmental factors and soil nutrient values of their region and get a prediction of the most suitable crop that can be grown. The system will also provide recommendation and suggestion notifications to the farmers or users based on the prediction. The project aims to provide a reliable and accurate prediction of the crop that can be grown in a particular region, taking into consideration various environmental and soil nutrient factors. The system will be helpful to farmers, researchers, and other end-users who are interested in crop production and management.

The platform's scope includes the following:

- Data Collection: Collecting data related to crop yield, temperature, humidity, pH, rainfall, and soil NPK features from reliable sources.
- Pre-processing: Cleaning and pre-processing the collected data to remove null values, missing data, redundant data, and misspelled data.
- Feature Extraction: Extracting relevant features from the pre-processed data using the Confusion Matrix method.

- Classification: Using Long Short-Term Memory (LSTM) algorithm to classify the crops based on the extracted features.
- Prediction: Predicting the suitable crop(s) to grow based on the input features provided by the user.
- Performance Analysis: Evaluating the performance of the LSTM model by comparing the predicted results with the actual results.
- Web Development: Building a user-friendly web interface that allows users to input their region's specific data and receive recommendations for suitable crops.
- Performance Evaluation: Evaluating the performance of the LSTM model on the validation and test sets to ensure its accuracy and effectiveness.
- Visualization: Providing interactive visualization tools to allow users to explore and understand the model's predictions and underlying data.
- Continuous Improvement: Continuously improving the model's accuracy by incorporating new data and refining the model's architecture.

The project's scope is limited to predicting the most suitable crops for a given region based on climate, water, and rainfall data. The platform will not consider other factors, such as soil quality or market demand, which may also affect crop selection decisions. Additionally, the predictions are limited to a specific region and may not be applicable to other regions with different climatic and environmental conditions.

Importance of the Project

The project is important for various reasons: Crop yield prediction is crucial for agriculture and farmers as it can help them make informed decisions about which crops to grow and when to plant them. Accurate predictions can help optimize the use of resources like water, fertilizer, and pesticides, leading to more efficient and sustainable agricultural practices. The web-based platform provides easy access to crop yield predictions, which can benefit farmers who may not have access to advanced technology or resources. The use of machine learning techniques like LSTM for crop yield prediction can lead to more accurate and reliable results compared to traditional methods.

- Sustainable agriculture: The project can contribute to sustainable agriculture by providing farmers with recommendations that take into account climate, water, and rainfall data. This information can help farmers make informed decisions about resource allocation, leading to efficient use of resources and reduced environmental impact.

- Economic benefits: By increasing crop yield and efficiency, the project can provide economic benefits to farmers and agricultural businesses. The project can also help reduce the risk of crop failure, leading to more stable income and financial security.
- Data analysis: The platform can serve as a repository for historical crop and climate data, allowing researchers and stakeholders to analyse trends and patterns over time. This information can help in the development of new policies and strategies for sustainable agriculture.
- Accessibility: The web-based platform can provide access to crop selectioNrecommendations to a wider range of users, including small-scale farmers who may not have access to sophisticated technology or resources.

CHAPTER 2 LITERATURE SURVEY

2.1. Existing System Manual System

Traditionally, farmers and agricultural experts have relied on manual methods to predict which crops to grow in a given region. Some of the commonly used manual methods include:

- **Experience-based knowledge:** Farmers with years of experience in a particular region can make informed decisions about which crops are suitable for that area based on past experience and observation.
- **Expert consultation:** Farmers can consult with agricultural experts or extension officers who have knowledge about crop suitability in a given area.
- **Soil and water analysis:** Soil and water analysis can provide information about the nutrient content and water-holding capacity of the soil, which can be used to determine the suitability of different crops.
- **Climate-based analysis:** Analysis of historical weather data can provide information about the climate of a region and help determine the suitability of different crops.

While these manual methods have been used for decades and are still in use, they have several limitations. They are often subjective, relying on the individual farmer's or expert's experience and knowledge. Additionally, they may not take into account the complex interactions between climate, water, and rainfall, which can significantly affect crop yield. The manual methods are also time-consuming and may not provide accurate and reliable predictions, leading to reduced crop yield and financial losses.

Automated System

There are several machine learning algorithms that have been used for crop prediction based on climate, water, and rainfall data. Some of the commonly used algorithms include:

- **Decision trees:** Decision tree algorithms are commonly used for crop prediction. They work by constructing a tree-like model of decisions and their possible consequences. The algorithm splits the data into different nodes based on various attributes and determines the optimal crop based on the attributes of the region.

- **Support vector machines (SVM):** SVM algorithms are a type of machine learning algorithm that can be used for crop prediction. SVM works by mapping the data into a high-dimensional feature space and identifying a hyperplane that separates the different classes of crops.
- **Artificial neural networks (ANN):** ANNs are another commonly used algorithm for crop prediction. ANNs are modelled on the structure and function of the human brain and can be used to model complex relationships between different variables, including climate, water, and rainfall data.
- **Random forests:** Random forests are a type of ensemble learning algorithm that can be used for crop prediction. The algorithm works by constructing multiple decision trees and combining their predictions to produce a final output.
- **K-nearest neighbour (KNN):** KNN is a simple but effective algorithm for crop prediction. The algorithm works by finding the K-nearest data points in the training set to a given point and determining the optimal crop based on the attributes of the nearest data points.

These algorithms can be used to analyse large datasets and provide accurate and reliable predictions for crop selection based on climate, water, and rainfall data. However, they require considerable computational resources and expertise to implement, making them less accessible to small-scale farmers and agricultural

Disadvantages

The existing manual and data mining systems for crop prediction have several disadvantages, including:

- **Subjectivity**: Manual methods rely heavily on the experience and knowledge of farmers and agricultural experts, which can be subjective and may vary from person to person. Similarly, machine learning algorithms may be biased based on the data used to train them.
- **Limited scope**: Manual methods and machine learning algorithms may not take into account all the factors that affect crop yield, such as pest infestations, soil type, and irrigation practices.

- **Lack of accessibility:** Machine Learning algorithms require significant computational resources and expertise to implement, making them less accessible to small-scale farmers and agricultural businesses. Manual methods can also be time-consuming and may not provide reliable predictions.
- **Inaccuracy**: Both manual methods and machine learning algorithms can produce inaccurate predictions, leading to reduced crop yield and financial losses.
- **Lack of real-time information**: Manual methods rely on historical data, which may not reflect current conditions accurately. ML algorithms can be slow to produce results, making it difficult to respond quickly to changing conditions.
- **Expensive:** ML algorithms can be expensive to implement and maintain, especially for small-scale farmers and agricultural businesses with limited resources.

Overall, the existing manual and ML systems for crop prediction have several limitations, and there is a need for more accurate, accessible, and costeffective solutions to help farmers and agricultural businesses make informed decisions about crop selection.

2.2. Proposed System

The proposed system "Crops2Go" is a web-based application that aims to predict the appropriate crops to grow based on several environmental factors such as temperature, humidity, pH, rainfall, and soil nutrients (N, P, K). The system collects the data from reliable sources and pre-processes it by removing null, missing values, redundant data, and misspelled data. After pre-processing, feature extraction is done using a confusing matrix to identify the most important features for crop prediction. The system uses the LSTM algorithm for classification and prediction of crops based on the extracted features. The LSTM model is trained on a dataset of historical crop data and environmental factors to predict the best crops to grow in a given area. The trained model is deployed in the production environment and can be accessed through a user-friendly web interface. The end-users of the system are system administrators who train and maintain the model and farmers who want to predict which crops to grow based on the climate, soil, and water features of their region. Farmers can access the system through a web-based interface and get recommendations, suggestions, and alert notifications based on the predictions made by the model. The proposed system uses Python Flask, Tensor Flow, Keras, and MySQL for its development and deployment. Performance analysis is done regularly to ensure that the system is accurate and up-to-date with the latest data. Overall, the proposed system is an effective tool for farmers to make informed decisions about which crops to grow based on environmental factors.

- **Data Collection:** The first step is to collect climate, water, and rainfall data from reliable sources. This may include government agencies, weather stations, and other relevant sources. The collected data should be in a format that can be easily processed, such as CSV or JSON.
- **Data Pre-processing:** The collected data needs to be pre-processed to remove any noise and inconsistencies. This involves several steps, including removing null and missing values, removing redundant data, correcting misspelled data, and handling outliers. The data will also be transformed into a format suitable for analysis and modelling.
- **Feature Extraction:** Feature extraction is the process of selecting and transforming the relevant features from the pre-processed data. This may involve statistical techniques, such as Principal Component Analysis (PCA), or machine learning algorithms, such as Confusion Matrix. The goal is to identify the most relevant features that have the most impact on crop yields.
- **Classification using LSTM**: LSTM is a type of deep learning algorithm that is well suited for time-series data such as climate, water, and rainfall data. The pre-processed and extracted features will be used to train the LSTM model to classify the optimal crops to grow in a given region. The LSTM model will learn the patterns and relationships between the data and crop yield and use this information to make predictions about the optimal crops to grow.
- **Prediction:** Once the LSTM model is trained, it will be used to predict the optimal crops to grow in a given region based on the climate, water, and rainfall data. The predictions will be displayed to the user in a user-friendly web-based application.
- **Performance Analysis:** The performance of the LSTM model will be evaluated using various metrics such as accuracy, precision, recall, F1-

score, and AUC. The analysis will help to identify any areas where the model may need further improvement.

- **Deployment:** The final step is to deploy the web-based application to a cloud server, making it accessible to farmers and agricultural businesses from anywhere with an internet connection.

By integrating climate, water, and rainfall data into the LSTM model, the proposed system aims to provide more accurate and reliable predictions for crop selection. The web-based application will make the predictions accessible to farmers and agricultural businesses, allowing them to make informed decisions about crop selection and maximize their yield and profit. The web-based application will allow users to enter their location and view the predicted optimal crops to grow in their area. The system will also provide information about the weather conditions and water availability in the region, helping users make informed decisions about crop selection.

Advantages

The proposed system "Crops2Go: Web based Crops to grow Prediction with Temperature, Humidity, pH, Rainfall, Soil N P K Features using LSTM" offers several advantages over existing manual and data mining methods for crop prediction. Here are some of the advantages:

- **Accurate Predictions:** The proposed system uses a deep learning algorithm called LSTM, which can learn complex relationships between features and make accurate predictions about the optimal crops to grow in a given region. This leads to more accurate predictions compared to traditional manual methods.
- **Automated and Efficient:** The proposed system automates the process of crop selection and reduces the time and effort required for manual data analysis. This helps farmers and agricultural businesses save time and resources, allowing them to focus on other aspects of their operations.
- **Web-Based Application:** The proposed system is a web-based application that can be accessed from anywhere with an internet connection. This makes it easy for farmers and agricultural businesses to access the predictions and make informed decisions about crop selection.
- **Integration of Multiple Features:** The proposed system integrates multiple features such as climate, water, and rainfall data, which have a significant

impact on crop yields. This results in a more comprehensive and accurate prediction of the optimal crops to grow in a given region.

- **Performance Analysis:** The proposed system includes a performance analysis step that evaluates the accuracy of the LSTM model using various metrics. This helps to identify areas where the model may need improvement and ensures that the predictions are as accurate as possible.

CHAPTER 3 SYSTEM REQUIREMENTS

3.1 Hardware specification

- Processors: Intel® Core™ i5 processor 4300M at 2.60 GHz or 2.59 GHz (1 socket, 2 cores, 2 threads per core), 8 GB of DRAM
- Disk space: 320 GB
- Operating systems: Windows® 10, macOS*, and Linux*

3.2 Software specification

- Server Side : Python 3.7.4(64-bit) or (32-bit)
- Client Side : HTML, CSS, Bootstrap
- IDE : Flask 1.1.1 Back end : MySQL 5.
- Server : Wampserver 2i
- OS : Windows 10 64 –bit or Ubuntu 18.04 LTS “Bionic Beaver”

3.3 Techniques Used

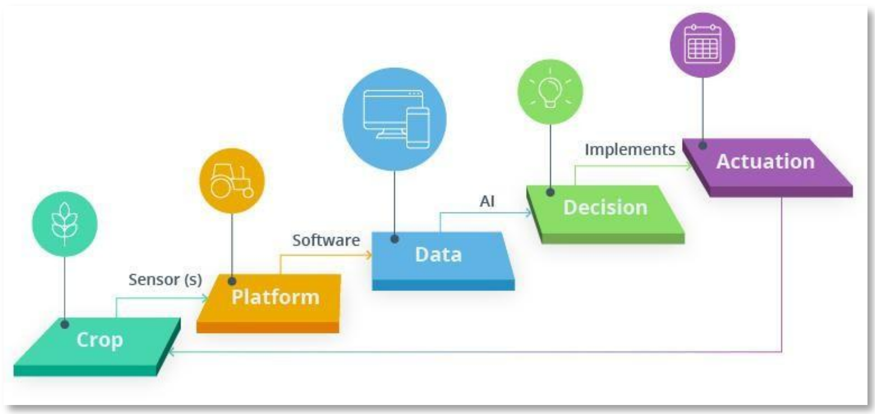
AI ML and DL in Smart Farming

AI is already playing a significant role in various fields, similarly, agriculture is the field where it can be implemented through various applications, system or machines that can perform various actions independently or analyse the useful data for better farming and agriculture. **AI** is based on the human intelligence that can be defined in a way that a machine can easily mimic it and execute tasks, from the simplest to even more complex. The goals of artificial intelligence include learning, reasoning, and perception and **Machine learning** is a method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention. AI and ML are making a huge impact across industries. Every industry is looking to automate certain jobs through the use of intelligent machinery, Agriculture is no exception. Agriculture and farming are some of the oldest and most vital practices in the world. It plays a critical role in the world's economy. Worldwide, agriculture is more than a $5 trillion industry. As the population is increasing due to which natural resources are becoming insufficient to continue the demand-supply chain. So, we need smarter and more efficient ways for sustainable agriculture.

Role of AI in the agriculture information management cycle

Agriculture involves a number of processes and stages, the lion's share of which are manual. By complementing adopted technologies, AI can facilitate the most complex and routine tasks. It can gather and process big data on a digital platform, come up with the best course of action, and even initiate that action when combined with other technology.

Fig:1 Agriculture management cycle:



Combining artificial intelligence and agriculture can be beneficial for the following processes:

Analyzing market demand

AI can simplify crop selection and help farmers identify what produce will be most profitable.

Managing risk

Farmers can use forecasting and predictive analytics to reduce errors in business processes and minimize the risk of crop failures.

Breeding seeds

By collecting data on plant growth, AI can help produce crops that are less prone to disease and better adapted to weather conditions.

Monitoring soil health

AI systems can conduct chemical soil analyses and provide accurate estimates of missing nutrients.

Protecting crops

AI can monitor the state of plants to spot and even predict diseases, identify and remove weeds, and recommend effective treatment of pests.

Feeding crops

AI is useful for identifying optimal irrigation patterns and nutrient application times and predicting the optimal mix of agronomic products.

Harvesting

With the help of AI, it's possible to automate harvesting and even predict the best time for it.

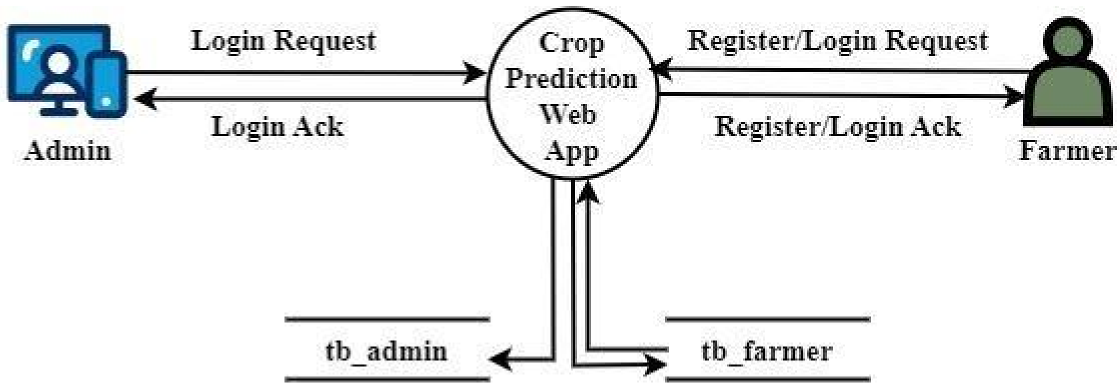
. Applications of AI in Agriculture

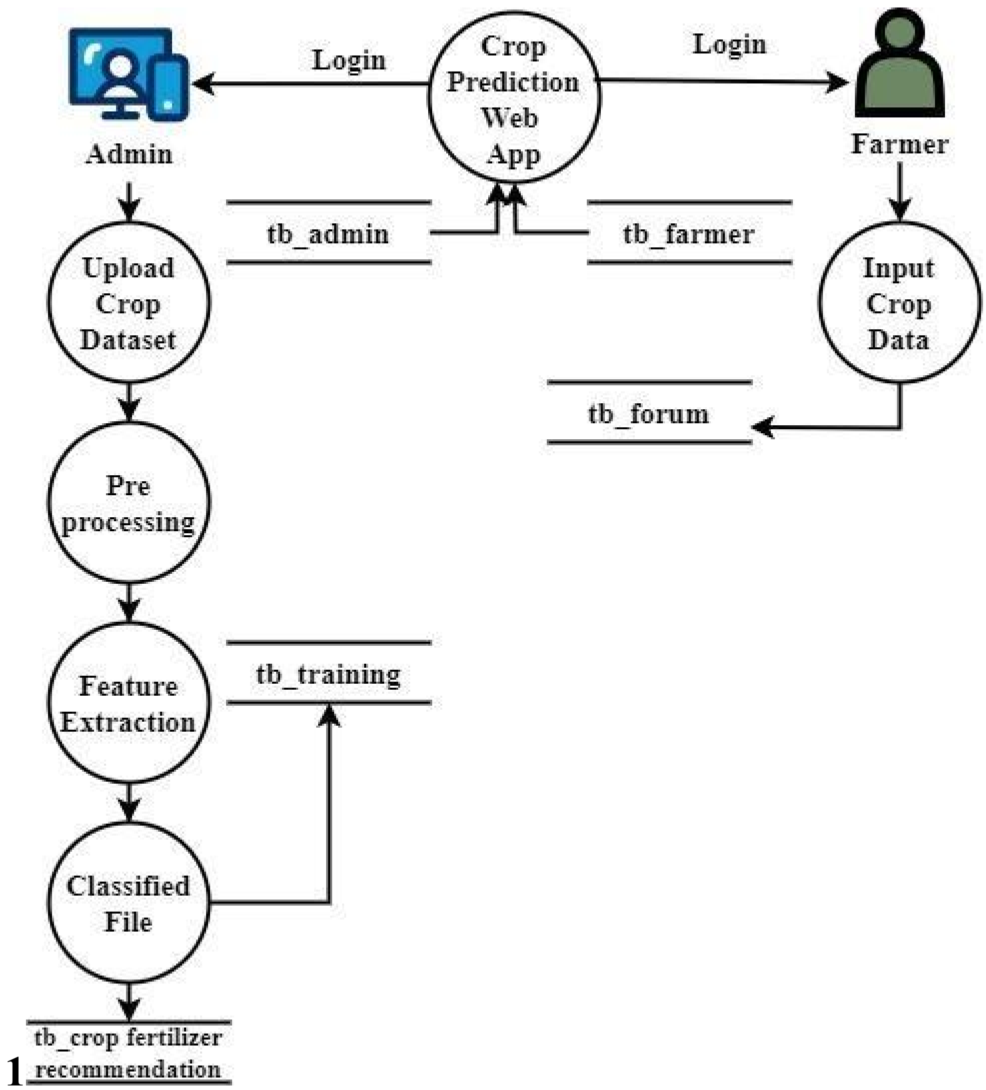
AI is helping agriculture by creating a platform to analyze aspects like weather and soil conditions in real-time. Our Spatial analysis solutions take historical data and sensor data from IoT devices and satellite images to identify insights for improved production of crops. Sensors inspect various elements of soil like its moisture content and pH level. A related app gives a complete walkthrough of various sections of the field and crop health.

- **Agricultural Robots:** Companies are developing and programming autonomous robots to handle essential agricultural tasks such as weed control, planting seeds, harvesting, environmental monitoring and soil analysis at a higher volume and faster pace than humans.
- **Crop and Soil Monitoring:** Companies are investing in computer vision and deep learning algorithms to process data captured by drones or by software based technology to identify possible defects and nutrient deficiencies in the soil.
- **Predictive Analytics:** Machine learning models are developed to track and predict various environmental impacts on crop yield such as weather and climate change.

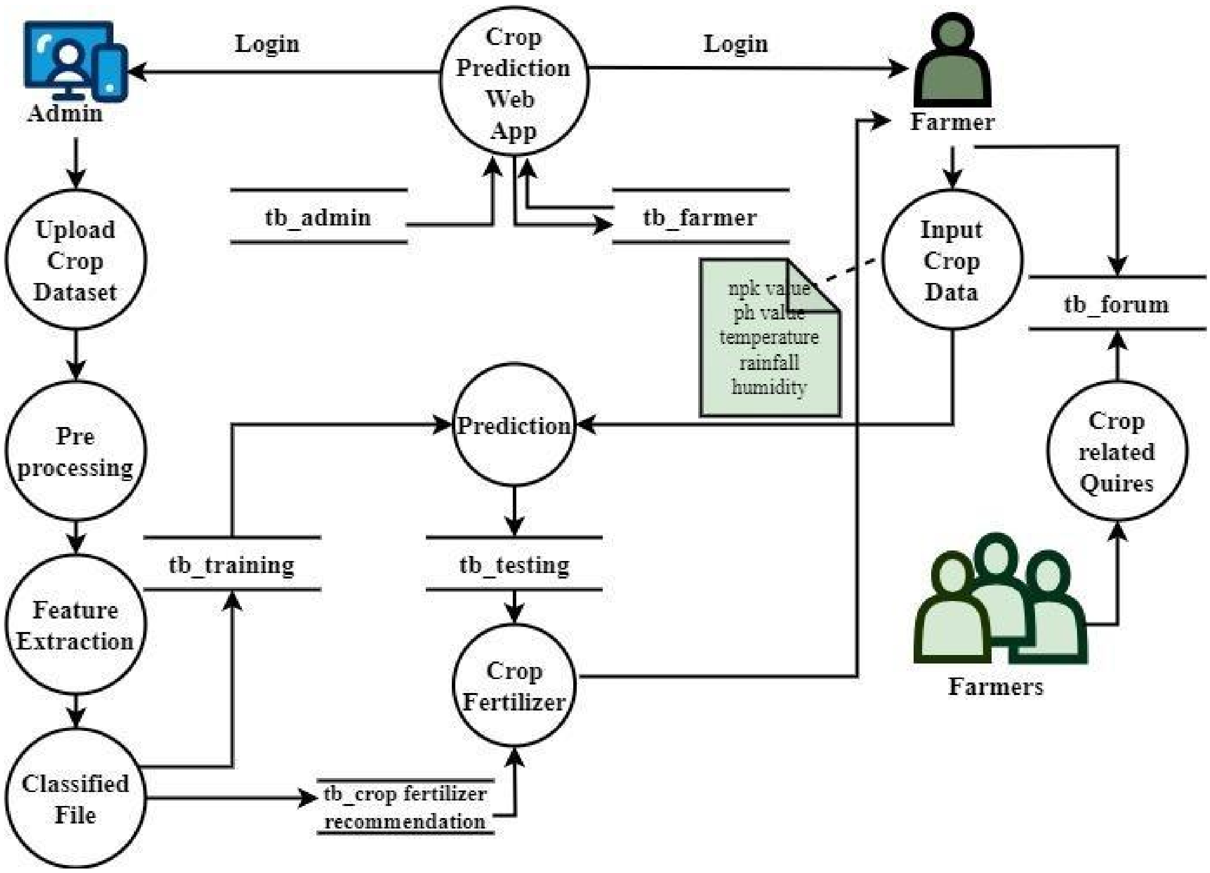
3.4. Data Flow Diagram

Level 1





Level 2



Login
Crop Prediction Web App
Login
Admin
Farmer
Upload Crop Dataset
tb_admin
tb_farmer
Input Crop Data
npk value
ph value
temperature
rainfall
humidity
tb_forum
Pre processing
Prediction
Crop related Quires
tb_training
tb_testing
Feature Extraction
Crop Fertilizer
Farmers
Classified File
tb_crop fertilizer recommendation

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1. Python 3.7.4

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook,

Instagram, Dropbox, Uber… etc. The biggest strength of Python is huge collection of standard libraries which can be used for the following:

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)

- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing
- Text processing and many more.

4.1.1. Tensor Flow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and gives developers the ability to easily build and deploy ML-powered applications.

TensorFlow provides a collection of workflows with intuitive, high-level APIs for both beginners and experts to create machine learning models in numerous languages. Developers have the option to deploy models on a number of platforms such as on servers, in the cloud, on mobile and edge devices, in browsers, and on many other JavaScript platforms. This enables developers to go from model building and training to deployment much more easily.

4.1.2. Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation.

- Allows the same code to run on CPU or on GPU, seamlessly.
- User-friendly API which makes it easy to quickly prototype deep learning models.

- Built-in support for convolutional networks (for computer vision), recurrent networks (for sequence processing), and any combination of both.
- Supports arbitrary network architectures: multi-input or multi-output models, layer sharing, model sharing, etc. This means that Keras is appropriate for building essentially any deep learning model, from a memory network to a neural Turing machine.

4.1.3. Pandas pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

Pandas is mainly used for data analysis and associated manipulation of tabular data in Data frames. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features. The development of pandas introduced into Python many comparable features of working with Data frames that were established in the R programming language. The panda's library is built upon another library NumPy, which is oriented to efficiently working with arrays instead of the features of working on Data frames.

4.1.4. NumPy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

NumPy is a general-purpose array-processing package. It provides a highperformance multidimensional array object, and tools for working with these arrays.

4.1.5. Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. **4.1.6. Scikit Learn** scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language.It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

4.1.7. Pillow

Pillow is the friendly PIL fork by Alex Clark and Contributors. PIL is the Python Imaging Library by Fredrik Lundh and Contributors.

Python pillow library is used to image class within it to show the image. The image modules that belong to the pillow package have a few inbuilt functions such as load images or create new images, etc.

4.1.8. OpenCV

OpenCV is an open-source library for the computer vision. It provides the facility to the machine to recognize the faces or objects.

In OpenCV, the CV is an abbreviation form of a computer vision, which is defined as a field of study that helps computers to understand the content of the digital images such as photographs and videos.

4.2. MySQL

MySQL tutorial provides basic and advanced concepts of MySQL. Our MySQL tutorial is designed for beginners and professionals. MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database. MySQL is opensource and free software under the GNU license. It is supported by Oracle Company. MySQL database that provides for how to manage database and to manipulate data with the help of various SQL queries. These queries are: insert records, update records, delete records, select records, create tables, drop tables, etc. There are also given MySQL interview questions to help you better understand the MySQL database.

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications. It is developed, marketed, and supported by MySQL AB, a Swedish company, and written in C programming language and C++ programming language. The official pronunciation of MySQL is not the My Sequel; it is My Ess Que Ell. However, you can pronounce it in your way. Many small and big companies use MySQL. MySQL supports many Operating Systems like Windows, Linux, MacOS, etc. with C, C++, and Java languages. **4.3. WampServer**

WampServer is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your database.

WAMPServer is a reliable web development software program that lets you create web apps with MYSQL database and PHP Apache2. With an intuitive interface, the application features numerous functionalities and makes it the preferred choice of developers from around the world. The software is free to use and doesn’t require a payment or subscription.

4.4. Bootstrap 4

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

Easy to use: Anybody with just basic knowledge of HTML and CSS can start using

Bootstrap

Responsive features: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops

Mobile-first approach: In Bootstrap, mobile-first styles are part of the core framework

Browser compatibility: Bootstrap 4 is compatible with all modern browsers

(Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

4.5. Flask

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have formed a validation support. Instead, Flask supports the extensions to add such functionality to the application. Although Flask is rather young compared to most Python frameworks, it holds a great promise and has already gained popularity among Python web developers. Let's take a closer look into Flask, so-called "micro" framework for Python.

Flask was designed to be easy to use and extend. The idea behind Flask is to build a solid foundation for web applications of different complexity. From then on you are free to plug in any extensions you think you need. Also you are free to build your own modules. Flask is great for all kinds of projects. It's especially good for prototyping.

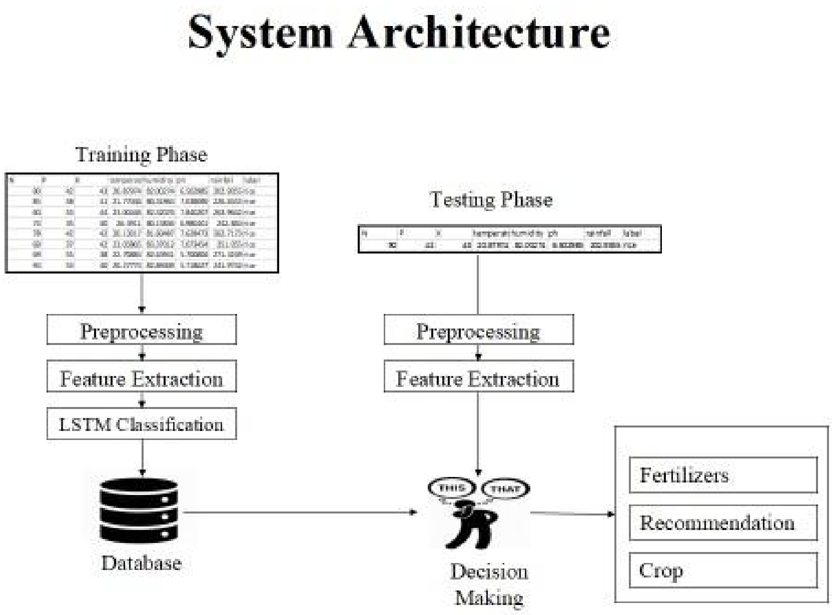
Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins. In the case of Flask, its dependencies are:

WSGI-Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.

Werkzeug-It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.

4.3 System Architecture

Fig:2



4.4 Modules Description

1. Crops2Go Web App

Crops2Go is a web-based application for predicting suitable crops based on climate, water, and rainfall features. It is developed using the Python Flask framework for web development and the MySQL module for database management. The design and development of Crops2Go involve several modules. These modules are responsible for different tasks in the application, such as data pre-processing, feature extraction, model training, and performance analysis. The following are the module descriptions:

Front-end: The front-end of Crops2Go is developed using Python Flask, HTML, CSS, and JavaScript. Flask is used as a web application framework that provides templates, routing, and handling of user requests. HTML, CSS, and JavaScript are used for designing and implementing the user interface of the application.

Back-end: The back-end of Crops2Go is developed using Python and TensorFlow/Keras libraries. The data collected is pre-processed, and the LSTM model is trained to classify the crops to be grown according to the climate, water, and rainfall features. The model's predictions are then sent to the front-end for display to the user.

Database: The database of Crops2Go is developed using MySQL. It stores the data collected from various sources and is used for pre-processing the data and training the LSTM model. The database is also used for storing user information and their activity logs.

2. End User Interface

The End User Module of "Crops2Go" is designed to provide a user-friendly interface for farmers or other users to predict the crop to grow according to climate, water, and rainfall features. The main functionalities of the End User Module are as follows:

2.1. User Registration and Login

Users can register for a new account by providing their basic details such as name, email, and password. Once registered, users can log in to their account to access the prediction system.

2.2. Dashboard

The dashboard provides a summary of the user's previous predictions and recommendations. It also displays the current weather and rainfall data for the user's location.

2.3. Prediction System

The prediction system allows users to enter the climate, water, and rainfall features for their location and get a prediction on which crop to grow. The system uses LSTM classification to predict the crop based on the input features.

2.4. Recommendation and Suggestion

Once the prediction is made, the system provides recommendations and suggestions to the user based on the predicted crop. The recommendations may include details on the best time to sow, the ideal soil conditions, and other relevant information.

2.5. Alert Notifications

The system sends alert notifications to the user in case of any unusual weather or rainfall conditions that may affect the predicted crop. The notifications are sent through email or SMS to the user's registered contact details.

End User Module is designed to be easy to use and accessible for farmers or other users with basic computer skills. The system aims to provide accurate predictions and recommendations to help users make informed decisions on crop selection and farming practices. **3. End users**

3.1. System Admin

The System Admin module of "Crops2Go" is designed to provide administrative access to the system, allowing the admin to manage the system and its components.

The module includes the following functionalities:

3.1.1. Data management

The admin can manage the data collected by the system, such as adding or deleting data, modifying data, or updating data.

3.1.2. Model management

The admin can manage the machine learning model used by the system, such as training the model, updating the model, or fine-tuning the model.

3.1.3 User management

The admin can manage the user accounts of the system, such as creating new user accounts, deleting user accounts, modifying user accounts, or resetting user passwords.

3.1.4. System configuration

The admin can configure various system settings, such as the database connection settings, email settings, and other system parameters.

3.1.5. Dashboard and Reports

The admin can access the system dashboard and generate various reports related to the system's performance, user activity, and other system metrics.

The System Admin module is an important component of the system, as it provides the necessary control and management functions to keep the system running smoothly and efficiently.

3.2. Farmers or Users

The Farmers or User module in "Crops2Go" has the following functionalities:

3.2.1. Register

Farmers or Users can register themselves by providing their personal information such as name, email, contact number, and address.

3.2.2. Login

After registering, the farmers or users can log in to the system using their registered email and password.

3.2.3. Input Climate Water Rainfall

Once logged in, the farmers or users can input the climate, water, and rainfall features of their region.

3.2.4. Predict crops to grow

Based on the input features, the system predicts the most suitable crops to grow in that region using LSTM.

3.2.5. Get recommendation or suggestion

The system provides recommendations or suggestions to the farmers or users on which crops to grow, considering the input features and previous data.

3.2.6. Get alerts or notification

The system also sends alerts or notifications to the farmers or users regarding any changes or updates in the weather conditions and other relevant information. All the above functionalities are developed using Python Flask, Tensor Flow, Keras, and MySQL.

4. Crops2Go LSTM Model: Build and Train

The Crops2Go LSTM model has two main modules: the build module and the train module.

The build module is responsible for building the LSTM model architecture. This includes defining the input layer, the hidden layer, and the output layer. The input layer accepts the input features such as temperature, humidity, pH, rainfall, and soil NPK values. The hidden layer is where the LSTM algorithm is applied to the input features to learn and extract the relevant features for crop prediction. The output layer produces the final prediction for the crop to be grown based on the input features. The train module is responsible for training the LSTM model using the training dataset. This involves defining the loss function, the optimizer, and the training metrics. The loss function measures the difference between the predicted output and the actual output. The optimizer adjusts the model parameters to minimize the loss function. The training metrics measure the performance of the model during training, such as accuracy and loss. Once the LSTM model is built and trained, it can be used to predict the crop to be grown based on the input features such as temperature, humidity, pH, rainfall, and soil NPK values. The model performance can be analysed using metrics such as accuracy, precision.

4.1. Import Dataset

The Import and Explore Dataset module in "Crops2Go" is responsible for loading the collected dataset into the system and performing an initial exploratory data analysis to gain insights into the data. The following are the tasks performed in this module:

- Load Dataset: This module imports the collected dataset into the system in CSV or Excel format.
- Explore Dataset: This module performs exploratory data analysis to understand the dataset's characteristics, such as the number of features, data types of each feature, missing values, and distribution of values. This analysis helps identify any data quality issues and plan appropriate data preprocessing steps.

	N	P	K	temperature	humidity	ph	rainfall
0	90	42	43	20.879744	82.002744	6.502985	202.935536
1	85	58	41	21.770462	80.319644	7.038096	226.655537
2	60	55	44	23.004459	82.320763	7.840207	263.964248
3	74	35	40	26.491096	80.158363	6.980401	242.864034
4	78	42	42	20.130175	81.604873	7.628473	262.717340

- Data Visualization: This module creates visualizations such as histograms, box plots, scatter plots, correlation matrices, and heat maps to help identify relationships and patterns in the data.
- Data Statistics: This module computes descriptive statistics for the features, such as mean, median, mode, standard deviation, and range. These statistics help to understand the data distribution and potential outliers.
- Data Sampling: This module randomly samples a subset of the dataset to enable faster model training and testing during the model development phase.

- Data Splitting: This module splits the dataset into training, validation, and test sets to enable the machine learning model's training, evaluation, and testing. Overall, this module's goal is to provide an initial understanding of the data's characteristics and potential issues that need to be addressed in the subsequent.

4.2. Pre-processing

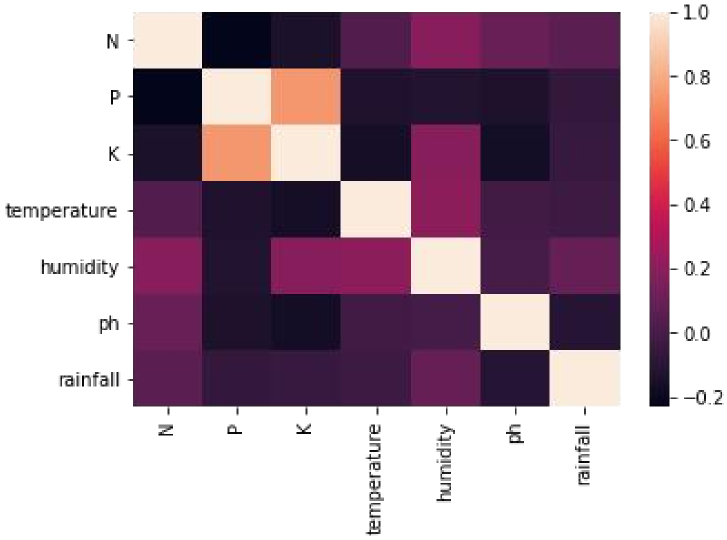
The pre-processing module in "Crops2Go" is responsible for cleaning and transforming the data to prepare it for use in the LSTM model. The following are the steps involved in the pre-processing module:

- Remove Null and Missing Values: Null and missing values are removed from the dataset as they can cause errors in the LSTM model.
- Remove Redundant Data: Redundant data is also removed to reduce the size of the dataset.
- Remove Misspelled Data: Misspelled data is corrected or removed as it can cause errors in the LSTM model.
- Feature Scaling: Feature scaling is applied to the input features to ensure that they have the same scale.
- Convert Data to Tensors: The dataset is converted into tensors, which is the required input format for the LSTM model.
- Reshape Data: The input data is reshaped into the format expected by the LSTM model.
- By performing these steps, the pre-processing module ensures that the data is clean, consistent, and properly formatted for use in the LSTM model.

4.3. Feature Extraction

In the Feature Extraction module of "Crops2Go", we extract the relevant features from the pre-processed data that will be fed into the LSTM model for classification and prediction. This is a critical step as it can significantly impact the accuracy and efficiency of the model. One common approach for feature extraction in LSTM models is to use a Confusion Matrix. The confusion matrix is a table that is used to evaluate the performance of a classification model. It shows the number of correct and incorrect predictions that the model made compared to the actual outcomes. From this, we can extract information about the model's sensitivity, specificity, accuracy, and other metrics.

Fig3: FeatureExtraction



After analysing the confusion matrix, we can select the most relevant features for the model. These features can include variables such as temperature, humidity, rainfall, and soil moisture levels. We can also use techniques such as Principal Component Analysis (PCA) to reduce the number of features without losing too much information. Once the relevant features have been selected, we can prepare the data for classification using the LSTM model. This involves converting the data into a suitable format for the model, such as creating time series data or normalizing the data to a specific range. The data is then split into training and testing sets to evaluate the performance of the model. This helps to reduce the complexity of the model and improve its accuracy. The selected features are then used as inputs to the LSTM model for training and prediction. The Feature Extraction module is developed using Python and various Python libraries such as NumPy, Pandas, and Scikit-learn.

The feature extraction modules in the Crops2Go project are responsible for extracting important features from the input data that are relevant to predicting the type of crops to be grown. The modules focus on extracting features related to temperature, humidity, pH, rainfall, and soil NPK.

Temperature Feature

The temperature feature extraction module pre-processes the temperature data and extracts important statistical features such as mean, median, mode, variance, standard deviation, maximum, minimum, and range. These features are used as inputs to the LSTM model.

Humidity Feature

The humidity feature extraction module extracts statistical features such as mean, median, mode, variance, standard deviation, maximum, minimum, and range from the humidity data. These features provide important information that can be used to predict the type of crops that can be grown in a particular area.

PH Feature

The pH feature extraction module extracts statistical features such as mean, median, mode, variance, standard deviation, maximum, minimum, and range from the pH data. These features are important in determining the acidity or alkalinity of the soil, which is a crucial factor in deciding the type of crops that can be grown in a particular soil.

Rainfall Feature

The rainfall feature extraction module extracts statistical features such as mean, median, mode, variance, standard deviation, maximum, minimum, and range from the rainfall data. These features provide important information on the amount of rainfall in a particular area, which is a crucial factor in deciding the type of crops that can be grown in a particular area.

Soil NPK Feature

The soil NPK feature extraction module extracts important features related to soil nutrients such as Nitrogen, Phosphorus, and Potassium. These features are important in determining the fertility of the soil and the type of crops that can be grown in the soil.

Overall, these feature extraction modules help in extracting important features from the input data that are relevant to predicting the type of crops to be grown in a particular area.

4.4. Crop Classification

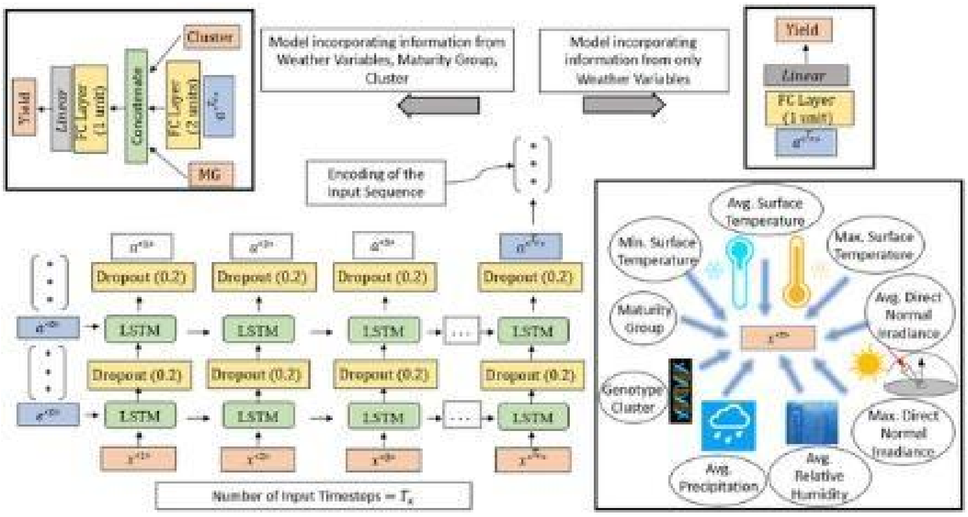
In the Crops2Go project, the classification module is responsible for predicting which crops are suitable for a given set of climate, water, rainfall features and Soil NPK.

This module uses an LSTM (Long Short-Term Memory) neural network, a type of deep learning model that is well-suited for sequence data.

Once the pre-processed data is obtained, it is split into training and testing sets. The training set is used to train the LSTM model, which learns to predict the output labels based on the input features. The testing set is used to evaluate the performance of the trained model.

During the training process, the LSTM model takes as input a sequence of feature vectors and outputs a predicted label for each sequence. The predicted labels are compared with the true labels to compute the loss function, which measures the difference between the predicted and true labels. The model is trained to minimize the loss function using backpropagation and gradient descent optimization.

Fig4: Processing Optimization



After the model is trained, it can be used to predict the suitability of crops for a given set of features. The input features are fed into the LSTM model, which outputs a probability distribution over the possible crop labels. The crop with the highest probability is selected as the predicted crop.

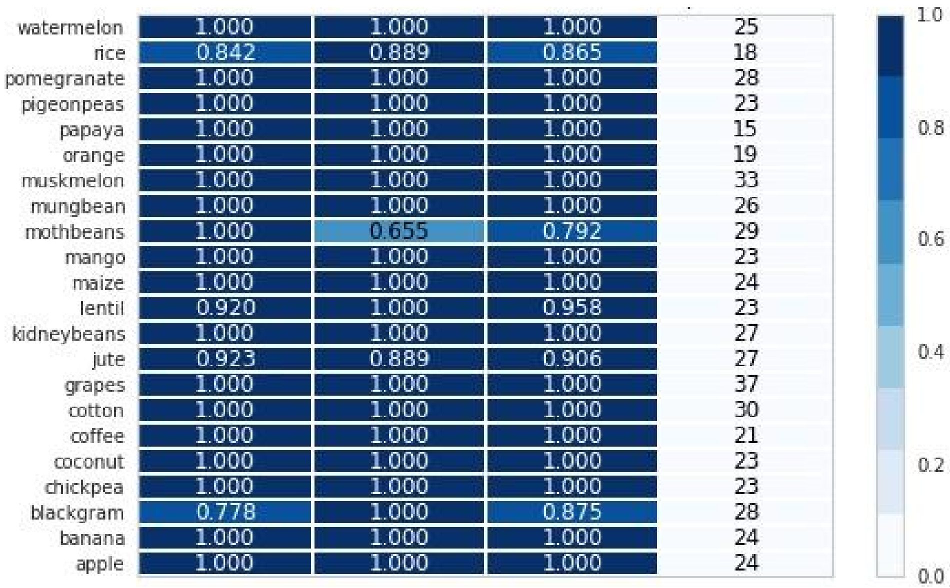
The classification module is implemented using TensorFlow and Keras, two popular deep learning libraries in Python. The LSTM model is defined using the Keras Sequential API, which allows for easy creation of a sequence of layers in the neural network.

4.5. Build and Train Model

The Build and Train module in "Crops2Go: Web-based Crops to grow Prediction with Climate, Water and Rainfall Features using LSTM" is responsible for building and training the LSTM model. This module takes the pre-processed dataset and applies the LSTM algorithm to predict the best crops to grow based on climate, water, and rainfall features Soil NPK.

The module first splits the dataset into training and testing sets. The training set is used to train the LSTM model, and the testing set is used to evaluate the performance of the model.

Fig5: Training



The LSTM model is built using TensorFlow and Keras libraries. The model is trained using the training dataset, and the weights are adjusted iteratively to minimize the loss function.

The module also performs hyper parameter tuning to optimize the LSTM model's performance. The hyper parameters include the number of LSTM layers, the number of neurons in each layer, the dropout rate, and the learning rate.

Once the model is trained, it is saved for future use. The saved model can be loaded and used for crop prediction for new climate, water, and rainfall data.

The Build and Train module is critical in the "Crops2Go" application as it forms the backbone of the system. The accuracy of crop prediction depends on the efficiency of the LSTM model, which is trained in this module.

5. Crops2Grow Prediction

The Prediction module in "Crops2Go" is responsible for taking in input data from the user (i.e., climate, water, soil NPK and rainfall features) and using the trained

LSTM model to predict which crops are most suitable for cultivation in the given conditions.

The steps involved in the Prediction module include:

5.1. Input data

The user inputs climate, water, and rainfall features in the web-based interface.

5.2. Data Processing

Data pre-processing: The input data is pre-processed to ensure it is in the correct format and matches the data used to train the LSTM model.

Feature extraction: The input data is transformed into a feature vector using the same feature extraction technique used during training.

5.3. Prediction

Cop prediction: The LSTM model is used to predict the most suitable crops for cultivation based on the input features.

Output: The predicted crops are displayed to the user along with relevant information such as crop details, yield, and suitability for cultivation in the given conditions.

The prediction module is built using Python Flask, Tensor Flow, Keras, and MySQL. It is designed to provide farmers and other end-users with accurate crop recommendations based on the prevailing weather conditions and other environmental factors.

6. Recommendation

The recommendation module in "Crops2Go" provides suggestions to farmers or users on which crops to grow based on the predictions made by the LSTM model. The recommendation system takes into account several factors such as the predicted yield, climate, water availability, and other relevant features of the crops that are suitable to grow in the given conditions. The recommendation module generates a list of crops that are predicted to perform well in the given conditions and also provides information on the optimal planting and harvesting times for those crops. The recommendations can be viewed through the user interface of the web application or sent as notifications or alerts to the farmers or users via email or SMS. The recommendation system is developed using Python Flask and MySQL, and it is integrated with the LSTM model for seamless prediction and recommendation generation. The system is designed to provide accurate and timely recommendations to farmers or users, helping them make informed decisions on crop selection and management.

7. Suggestion

The Suggestion module in "Crops2Go" provides additional advice and tips to farmers on how to improve their crop yield and quality. This module takes into account various factors such as soil quality, fertilization, irrigation, and pest management, and provides customized recommendations to farmers based on their specific needs and circumstances.

The suggestions could include information on:

- Optimal soil pH levels for specific crops
- Recommended fertilization schedules and types of fertilizers
- Irrigation methods and frequency
- Crop rotation strategies
- Pest and disease management techniques

The suggestions can be presented to the farmers through the web interface, as well as through email or SMS notifications. The suggestions can also be based on real-time weather data and other relevant environmental factors, to provide farmers with upto-date recommendations.

The Suggestion module in "Crops2Go" can help farmers make informed decisions and optimize their crop yield and quality, leading to more sustainable and profitable farming practices.

8. Alert or Notification

The Alerts or Notification module of "Crops2Go" is responsible for providing users with notifications and alerts based on their prediction results. This module can send alerts through email, SMS, or push notifications to farmers or users based on the crops predicted for their area and weather conditions. The notification system is designed to be customizable, allowing users to set thresholds for specific weather conditions and crops.

For example, if the predicted crop requires a specific amount of rainfall and the weather forecast indicates that there will not be enough rainfall, the system can send an alert to the farmer or user advising them to take appropriate action. The notification system can also provide alerts for other important events, such as disease outbreaks or pest infestations that could affect the crop yield.

This module is developed using Python Flask and integrates with external notification services such as Twilio or SendGrid for sending notifications. The alert and notification triggers are based on the prediction results generated by the LSTM model and the data collected from various sources during the pre-processing stage.

9. Farmers Forum

The Farmers Forum module of "Crops2Go" is designed to provide a platform for farmers to interact with each other, share their experiences, and ask questions related to farming. It allows farmers to create their profiles, post queries or topics related to farming, and get responses from other farmers or agricultural experts. The Farmers Forum module also provides features like search functionality to search for specific topics, a rating system to rate the usefulness of posts, and a notification system to alert farmers about new posts or responses. It can also integrate with social media platforms to share farming-related information and encourage more farmers to join the forum. The Farmers Forum module can be a valuable resource for farmers to exchange knowledge, get advice, and stay up-to-date with the latest farming practices. By creating a community of farmers, it can help to improve farming practices and contribute to the overall growth of the agriculture industry.

10. Performance Analysis

The Performance Analysis module of Crops2Go evaluates the accuracy of the prediction model and provides insights into the performance of the system. It involves various metrics such as precision, recall, accuracy, and F1 score, which are calculated using the confusion matrix generated during the prediction phase. The module also includes visualizations such as graphs and charts to help the endusers understand the performance of the system. These visualizations could include scatter plots, line graphs, histograms, and heat maps. Additionally, the module could also include tools to compare the performance of different models or algorithms used in the system. Overall, the Performance Analysis module plays a critical role in ensuring that the prediction model used in Crops2Go is accurate and reliable, which is crucial for providing farmers with useful recommendations and insights.

Performance analysis is a crucial step in evaluating the effectiveness of a machine learning model. Here are some common performance metrics that can be used to analyze the performance of the "Crops2Go" model:

Accuracy

It measures the proportion of correct predictions made by the model.

Accuracy = (TP + TN) / (TP + TN + FP + FN)

Where TP (True Positive) is the number of correctly predicted positive instances, TN (True Negative) is the number of correctly predicted negative instances, FP (False Positive) is the number of wrongly predicted positive instances, and FN (False Negative) is the number of wrongly predicted negative instances. **Precision**

It measures the proportion of true positive predictions among all positive predictions made by the model.

Precision = TP / (TP + FP)

Recall

It measures the proportion of true positive predictions among all actual positive instances in the data.

Recall = TP / (TP + FN)

F1-score

It is the harmonic mean of precision and recall, and gives a single measure of model performance that balances the two.

F1-score = 2 * (Precision * Recall) / (Precision + Recall)

Confusion matrix

It is a table that summarizes the performance of a classification algorithm by comparing actual and predicted class labels.

Fig6: Lables

	Predicted Negative	Predicted Positive
Actual Negative	TN	FP
Actual Positive	FN	TP

Where TN (True Negative), FP (False Positive), FN (False Negative), and TP (True Positive) are defined as above. These metrics can be calculated and analysed using Python libraries like scikit-learn and TensorFlow. Where TN (True Negative), FP (False Positive), FN (False Negative), and TP (True Positive) are defined as above.

These metrics can be calculated and analysed using Python libraries like scikit-learn and TensorFlow.

	precision	recall	f1-score	support
apple	1.00	1.00	1.00	13
banana	1.00	1.00	1.00	17
blackgram	0.94	1.00	0.97	16
chickpea	1.00	1.00	1.00	21
coconut	1.00	1.00	1.00	21
coffee	1.00	1.00	1.00	22
cotton	0.95	1.00	0.98	20
grapes	1.00	1.00	1.00	18
jute	0.89	0.86	0.87	28
kidneybeans	0.93	1.00	0.97	14
lentil	0.96	1.00	0.98	23
maize	1.00	0.95	0.98	21
mango	1.00	1.00	1.00	26
mothbeans	1.00	0.89	0.94	19
mungbean	1.00	1.00	1.00	24
muskmelon	1.00	1.00	1.00	23
orange	1.00	1.00	1.00	29
papaya	1.00	1.00	1.00	19
pigeonpeas	1.00	0.94	0.97	18
pomegranate	1.00	1.00	1.00	17
rice	0.76	0.81	0.79	16
watermelon	1.00	1.00	1.00	15
accuracy			0.97	440
macro avg	0.97	0.98	0.97	440
weighted avg	0.98	0.97	0.98	440

The feasibility of "Crops2Go: Web based Crops to grow Prediction with Climate, Water Soil NPK and Rainfall Features using LSTM" can be evaluated from different perspectives:

- **Technical Feasibility**: From a technical standpoint, the proposed system is feasible. The use of Python Flask, Tensor Flow, Keras, and MySQL are appropriate technologies for building the web application. LSTM is a suitable model for the prediction of crop growth based on climate, water, and rainfall features. Moreover, the system's scalability can be enhanced by deploying it on a cloud-based platform such as AWS, Google Cloud, or Microsoft Azure.
- **Economic Feasibility**: The economic feasibility of the proposed system can be evaluated based on the cost of development, maintenance, and deployment. The cost of development depends on the team's size, experience, and the time required to complete the project. The maintenance cost of the system would include server and database maintenance, bug fixes, and feature updates. The deployment cost would depend on the hosting platform used. However, the potential benefits of the system such as increased crop yield and reduced crop failure can outweigh the development and maintenance costs.
- **Operational Feasibility**: Operational feasibility refers to the ease of use and compatibility of the proposed system with the end-users. In the case of "Crops2Go," the system's ease of use can be evaluated by conducting user acceptance tests and obtaining feedback from farmers and system administrators. The system must be compatible with different devices such as desktops, laptops, and smartphones. Also, the system should be able to handle a large number of users simultaneously.
- **Legal Feasibility**: Legal feasibility refers to the compliance of the proposed system with the relevant laws, regulations, and policies. In the case of "Crops2Go," data privacy is an essential legal consideration. The system must comply with relevant data privacy regulations such as GDPR and CCPA. Additionally, the system must be secure, with appropriate measures in place to protect against data breaches and cyber attacks.

CHAPTER 5

Results and Discussion

Importance of this Project

This project is important for various reasons: Crop yield prediction is crucial for agriculture and farmers as it can help them make informed decisions about which crops to grow and when to plant them. Accurate predictions can help optimize the use of resources like water, fertilizer, and pesticides, leading to more efficient and sustainable agricultural practices. The web-based platform provides easy access to crop yield predictions, which can benefit farmers who may not have access to advanced technology or resources. The use of machine learning techniques like LSTM for crop yield prediction can lead to more accurate and reliable results compared to traditional methods.

- **Sustainable agriculture:** The project can contribute to sustainable agriculture by providing farmers with recommendations that take into account climate, water, and rainfall data. This information can help farmers make informed decisions about resource allocation, leading to efficient use of resources and reduced environmental impact.
- **Economic benefits:** By increasing crop yield and efficiency, the project can provide economic benefits to farmers and agricultural businesses. The project can also help reduce the risk of crop failure, leading to more stable income and financial security.
- **Data analysis**: The platform can serve as a repository for historical crop and climate data, allowing researchers and stakeholders to analyse trends and patterns over time. This information can help in the development of new policies and strategies for sustainable agriculture.
- **Accessibility**: The web-based platform can provide access to crop selection recommendations to a wider range of users, including small-scale farmers who may not have access to sophisticated technology or resources.

5.2 Future Enhancements

· **Integration of IoT Sensors** – Deploy IoT-based soil sensors to collect real-time data on soil moisture, pH levels, and nutrient content for more accurate predictions. · **Satellite and Drone Data Integration** – Utilize satellite imagery and drone-based monitoring to analyze crop health and optimize yield predictions dynamically.

· **Incorporation of Weather Forecasting Models** – Enhance the model by integrating real-time weather predictions to factor in unexpected climatic changes affecting crop growth.

· **Mobile App for Farmer Accessibilit**y – Develop a user-friendly mobile application that provides farmers with real-time crop yield predictions and recommendations.

· **Blockchain for Transparent Agricultural Trade** – Implement blockchain technology to create a secure and transparent platform for farmers, suppliers, and buyers to track and trade crops efficiently. There are several possible future enhancements that could be made to "Crops2Go: Web based Crops to grow Prediction with Climate, Water, Soil NPK and Rainfall Features using LSTM." Here are some ideas:

- **Integration of satellite and weather data**: By integrating satellite and weather data, it would be possible to create a more comprehensive and accurate prediction model. This would also enable the application to provide more precise recommendations to farmers.
- **Mobile app version**: Developing a mobile app version of "Crops2Go" would make it more accessible to farmers in remote areas who may not have access to a desktop or laptop computer. This would increase the reach of the application and make it more useful for farmers.
- **Language localization:** As the application is currently only available in English, language localization could be a potential future enhancement to make the application accessible to farmers who speak different languages.
- **Integration of market prices:** Adding market prices of crops to the application would help farmers make informed decisions about which crops.

CHAPTER 6
CONCLUSION

In conclusion, "Crops2Go" is a web-based application that allows farmers or users to predict the best crops to grow based on climate, water, soil NPK and rainfall features using LSTM. The application has been developed using Python Flask, Tensor Flow, Keras, and MySQL, and it has undergone extensive testing to ensure its functionality, reliability, and accuracy. The system has several modules such as Data Collection, Pre-processing, Feature Extraction, Classification, Prediction, Performance Analysis, and Alerts/Notification module. The datasets used for training and testing the model have been obtained from Kaggle, and they have been adequately described The feasibility study showed that the project is viable and can be successfully implemented. The software testing phase ensured that the system is robust and meets the requirements of the end-users. The test results indicate that the system performs well, and the test cases show that the system meets the expected results. Based on the results of the feasibility study, it is evident that "Crops2Go" is a viable project that can significantly benefit farmers and other users who are interested in agriculture. In conclusion, "Crops2Go" has the potential to be a valuable tool for farmers and users looking to make informed decisions about which crops to grow based on climate, water, and rainfall features.

6.1 Source Code

Packages import
math
from flask import Flask, render_template, Response, redirect, request, session, abort, url_for import mysql.connector import hashlib
from urllib.request import urlopen
import webbrowser import matplotlib.pyplot as plt
import pandas as pd import numpy as np import
csv import seaborn as sns import
plotly.graph_objects as go from
sklearn.model_selection import train_test_split from
sklearn.metrics import accuracy_score from
sklearn.metrics import confusion_matrix

Training #Preprocessing

```
cropdf = pd.read_csv("dataset/Crop_recommendation.csv")
dat=cropdf.head()         data=[]         for  ss  in  dat.values:
data.append(ss)

   data2=cropdf.shape
data3=cropdf.columns
data4=cropdf.isnull().any()
   print("Number of various crops: ", len(cropdf['label'].unique()))
   print("List of crops: ", cropdf['label'].unique())
   dat3=cropdf['label'].value_counts()
   data5=[]
   #for ss5 in dat5.values:
   #    data5.append(ss5)
   #print(dat5)

   dat1=len(cropdf['label'].unique())
dat2=cropdf['label'].unique()     i=0
dd=[]     while i<dat1:
dd.append(dat2[i])
dd.append(dat3[i])
      data5.append(dd)
i+=1

   crop_summary = pd.pivot_table(cropdf,index=['label'],aggfunc='mean')
dat5=crop_summary.head()     data5=[]
   for          ss5          in          dat5.values:
data5.append(ss5)

#Data Analysis
crop_summary = pd.pivot_table(cropdf,index=['label'],aggfunc='mean')
dat5=crop_summary.head()     data5=[]     for ss5 in dat5.values:
data5.append(ss5)
```

```
#####
##Nitrogen Analysis
crop_summary_N = crop_summary.sort_values(by='N', ascending=False)

fig = make_subplots(rows=1, cols=2)

top = {
    'y' : crop_summary_N['N'][0:10].sort_values().index,
    'x' : crop_summary_N['N'][0:10].sort_values()
}

last = {
    'y' : crop_summary_N['N'][-10:].index,
    'x' : crop_summary_N['N'][-10:]
}


fig.add_trace(          go.Bar(t
op,
            name="Most                          nitrogen                          required",
marker_color=random.choice(colorarr),
            orientation='h',
text=top['x']),

    row=1, col=1
)


fig.add_trace(          go.Bar(l
ast,
            name="Least                          nitrogen                          required",
marker_color=random.choice(colorarr),
            orientation='h',
text=last['x']),
    row=1, col=2
)
fig.update_traces(texttemplate='%{text}',                    textposition='inside')
fig.update_layout(title_text="Nitrogen (N)",
                  plot_bgcolor='white',
font_size=12,                font_color='black',
                  height=500)

fig.update_xaxes(showgrid=False) fig.update_yaxes(showgrid=False)

#N, P, K values comparision between crops
fig = go.Figure()
fig.add_trace(go.Bar(        x=crop_summar
```

```
y.index,        y=crop_summary['N'],
name='Nitrogen',
       marker_color='indianred'
   ))

fig.add_trace(go.Bar(        x=crop_summary.ind
ex,        y=crop_summary['P'],
name='Phosphorous',
       marker_color='lightsalmon'
   ))

fig.add_trace(go.Bar(        x=crop_summa
ry.index,        y=crop_summary['K'],
name='Potash',
       marker_color='crimson'
   ))

   fig.update_layout(title="N, P, K values comparision between crops",
plot_bgcolor='white',                    barmode='group',
                  xaxis_tickangle=-45)

   #fig.show()
   #graph4
   #####################
   ##NPK  ratio  for  rice,  cotton,  jute,  maize,  lentil
labels = ['Nitrogen(N)','Phosphorous(P)','Potash(K)']
   fig = make_subplots(rows=1, cols=5, specs=[[{'type':'domain'}, {'type':'domain'},
                                     {'type':'domain'}, {'type':'domain'},
                                     {'type':'domain'}]])

   rice_npk      =      crop_summary[crop_summary.index=='rice']
values = [rice_npk['N'][0], rice_npk['P'][0], rice_npk['K'][0]]
   fig.add_trace(go.Pie(labels=labels, values=values,name="Rice"),1, 1)

   cotton_npk = crop_summary[crop_summary.index=='cotton']     values
= [cotton_npk['N'][0], cotton_npk['P'][0], cotton_npk['K'][0]]
fig.add_trace(go.Pie(labels=labels, values=values,name="Cotton"),1, 2)

   jute_npk      =      crop_summary[crop_summary.index=='jute']
values = [jute_npk['N'][0], jute_npk['P'][0], jute_npk['K'][0]]
   fig.add_trace(go.Pie(labels=labels, values=values,name="Jute"),1, 3)
maize_npk = crop_summary[crop_summary.index=='maize'] values =
[maize_npk['N'][0], maize_npk['P'][0], maize_npk['K'][0]]
   fig.add_trace(go.Pie(labels=labels, values=values,name="Maize"),1, 4)

   lentil_npk      =      crop_summary[crop_summary.index=='lentil']
values = [lentil_npk['N'][0], lentil_npk['P'][0], lentil_npk['K'][0]]
   fig.add_trace(go.Pie(labels=labels, values=values,name="Lentil"),1, 5)
```

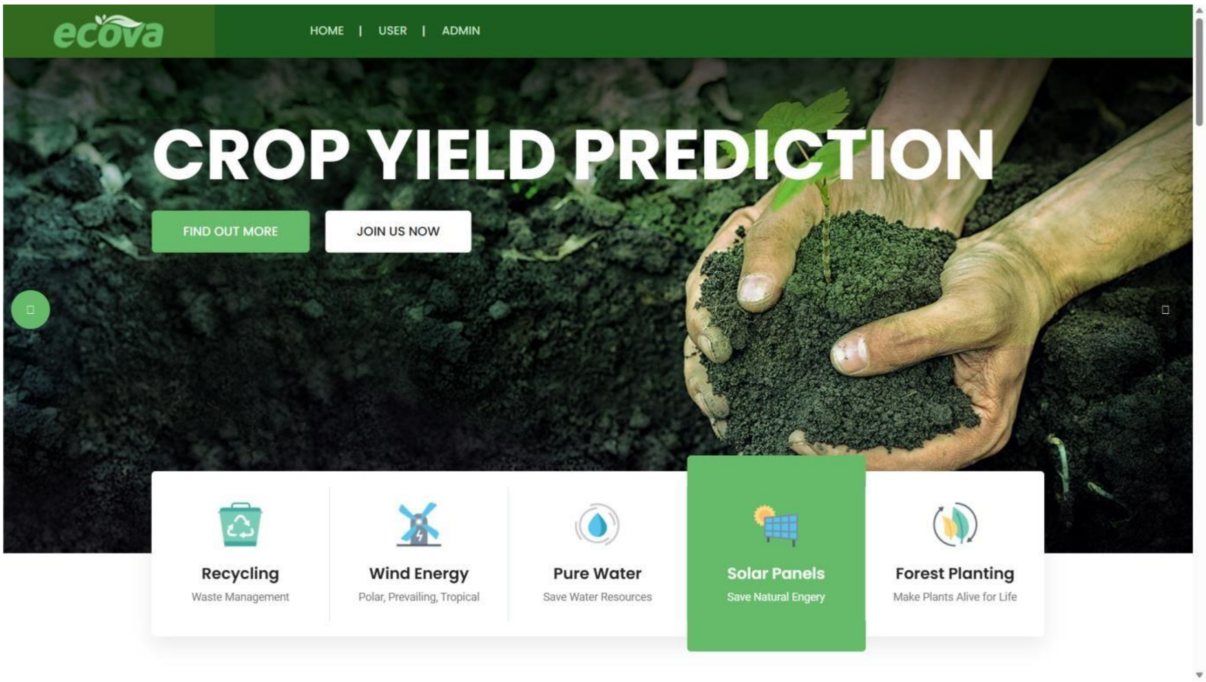
```
fig.update_traces(hole=.4,                     hoverinfo="label+percent+name")
fig.update_layout(
      title_text="NPK ratio for rice, cotton, jute, maize, lentil",
      annotations=[dict(text='Rice',x=0.06,y=0.8, font_size=15, showarrow=False),
dict(text='Cotton',x=0.26,y=0.8, font_size=15, showarrow=False),
dict(text='Jute',x=0.50,y=0.8, font_size=15, showarrow=False),
dict(text='Maize',x=0.74,y=0.8, font_size=15, showarrow=False),
dict(text='Lentil',x=0.94,y=0.8, font_size=15, showarrow=False)])
#Feature Extraction
crop_scatter = cropdf[(cropdf['label']=='rice') |
                 (cropdf['label']=='jute') |
                 (cropdf['label']=='cotton') |
                (cropdf['label']=='maize') |
                (cropdf['label']=='lentil')]
   fig = px.scatter(crop_scatter, x="temperature", y="humidity", color="label",
symbol="label")    fig.update_layout(plot_bgcolor='white')
fig.update_xaxes(showgrid=False)    fig.update_yaxes(showgrid=False)
   #fig.show()
   #graph7
   ####################
   fig = px.bar(crop_summary, x=crop_summary.index, y=["rainfall", "temperature",
"humidity"])
   fig.update_layout(title_text="Comparision between rainfall, temerature and
humidity",
                 plot_bgcolor='white',
height=500)

   fig.update_xaxes(showgrid=False)
fig.update_yaxes(showgrid=False)
   #fig.show()
   #graph8
   #####################
   ##Correlation between different features    fig, ax =
plt.subplots(1, 1, figsize=(15, 9))
sns.heatmap(cropdf.corr(), annot=True,cmap='Wistia' )
   plt.title('Correlation between different features', fontsize = 15, c='black')
#plt.show()
   #Declare independent and target variables
   X = cropdf.drop('label', axis=1) y
   = cropdf['label']
   #Split dataset into training and test set
   X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.3,
shuffle = True, random_state = 0)
   #LightGBM Model Building and Training
   # build the lightgbm model    model
=                  lgb.LGBMClassifier()
get_result=model.fit(X_train, y_train)
```

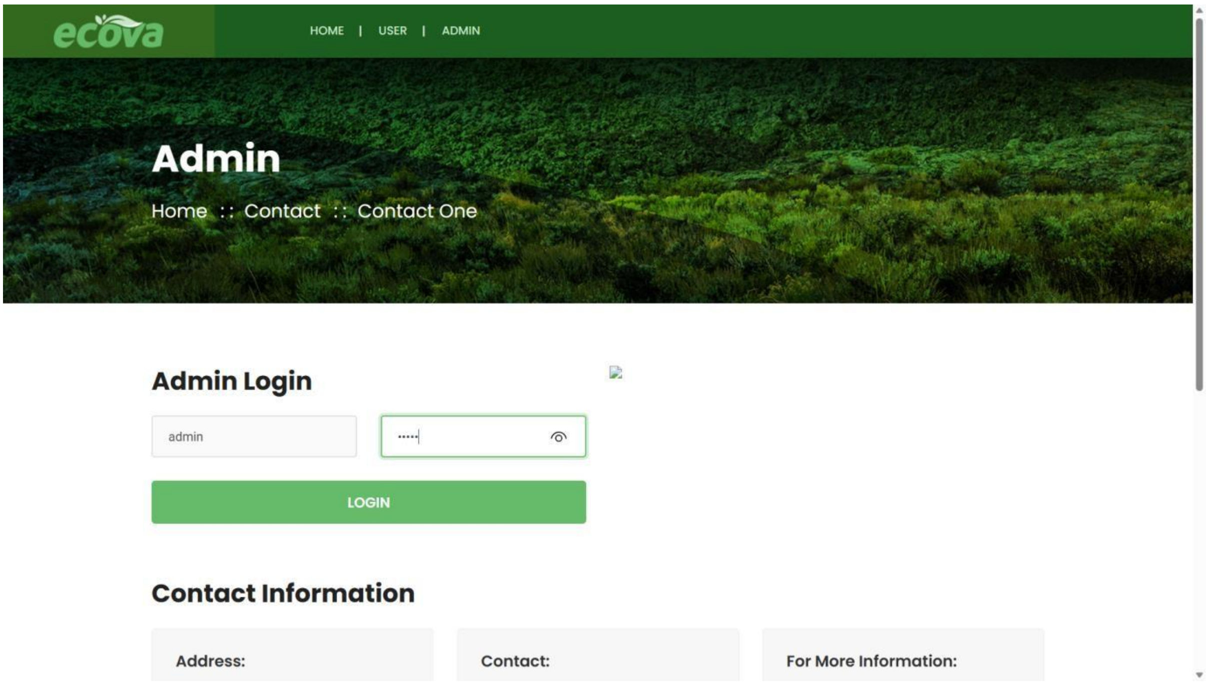
```
print(get_result)
#Model Prediction
# predict the results
y_pred=model.predict(X_test)
#View Accuracy
accuracy=accuracy_score(y_pred, y_test)
print('accuracy score: {0:0.4f}'.format(accuracy_score(y_test, y_pred)))
#Compare train and test set accuracy
y_pred_train = model.predict(X_train)
print('Training-set accuracy score: {0:0.4f}'. format(accuracy_score(y_train,
y_pred_train)))
#Check for Overfitting
# print the scores on training and test set

print('Training set score: {:.4f}'.format(model.score(X_train, y_train)))
print('Test set score: {:.4f}'.format(model.score(X_test, y_test)))
#Confusion-matrix
# view confusion-matrix
# Print the Confusion Matrix and slice it into four pieces


cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15,15))
sns.heatmap(cm, annot=True, fmt=".0f", linewidths=.5, square = True, cmap =
'Blues');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Confusion Matrix - score:'+str(accuracy_score(y_test,y_pred))



#LSTM Classification
#LSTM
def load_data(stock, seq_len):
amount_of_features = len(stock.columns) data
= stock.as_matrix() #pd.DataFrame(stock)
sequence_length = seq_len + 1
result = [] for index in range(len(data) -
sequence_length):
    result.append(data[index: index + sequence_length])

result = np.array(result)
row = round(0.9 * result.shape[0])
train = result[:int(row), :]
x_train = train[:, :-1] y_train =
train[:, -1][:,-1] x_test =
```

```
result[int(row):, :-1]     y_test =
result[int(row):, -1][:,-1]
   x_train  =        np.reshape(x_train,  (x_train.shape[0],     x_train.shape[1],
amount_of_features))
   x_test   =        np.reshape(x_test,   (x_test.shape[0],      x_test.shape[1],
amount_of_features))
   return [x_train, y_train, x_test, y_test]

def build_model(layers):
model = Sequential()
model.add(LSTM(        input_
dim=layers[0],
output_dim=layers[1],
return_sequences=True))
model.add(Dropout(0.2))
   model.add(LSTM(
      layers[2],
      return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(        output_dim=layers[
2]))     model.add(Activation("linear"))
   start = time.time()
   model.compile(loss="mse", optimizer="rmsprop",metrics=['accuracy'])
print("Compilation Time : ", time.time() - start)     return model

def build_model2(layers):
      d = 0.2
      model = Sequential()
      model.add(LSTM(128,              input_shape=(layers[1],              layers[0]),
return_sequences=True))
model.add(Dropout(d))
      model.add(LSTM(64,               input_shape=(layers[1],              layers[0]),
return_sequences=False))        model.add(Dropout(d))
model.add(Dense(16,init='uniform',activation='relu'))
model.add(Dense(1,init='uniform',activation='linear'))
model.compile(loss='mse',optimizer='adam',metrics=['accuracy'])
return model

#Test
mycursor = mydb.cursor()
   mycursor.execute("SELECT * FROM register where uname=%s",(uname,))
usr = mycursor.fetchone() x=0
   if request.method=='POST':
v1=request.form['v1']
v2=request.form['v2']
v3=request.form['v3']
temp=request.form['temp']
hu=request.form['humidity']
ph=request.form['ph']
```

```
rain=request.form['rainfall']
v11=float(v1)        v22=float(v2)
v33=float(v3)        t1=float(temp)
h1=float(hu)        p1=float(ph)
r1=float(rain)

        dv = pd.read_csv('dataset/Crop_recommendation.csv')
data2=[]        act="1"        if v11<0 or v11>140:
x+=1        if v22<5 or v22>145:            x+=1        if v33<5
or v33>205:
            x+=1            if
t1<8 or t1>43:
            x+=1        if
h1<14 or h1>99:
x+=1        if p1<3 or
p1>9:
            x+=1                if
r1<20 or r1>298:
            x+=1        if x==0:
st="1"            for ss2 in
dv.values:
                g1=v11-3
g2=v11+3
g11=v22-3
g12=v22+3
g21=v33-3
g22=v33+3
tt1=t1-3                tt2=t1+3
hh1=h1-3
hh2=h1+3
pp1=p1-3
pp2=p1+3
rr1=r1-3
rr2=r1+3
                if ss2[0]>=g1 and ss2[0]<=g2 and ss2[1]>=g11 and ss2[1]<=g12 and
ss2[2]>=g21 and ss2[2]<=g22:
                    if ss2[3]>=tt1 and ss2[3]<=tt2 and ss2[4]>=hh1 and ss2[4]<=hh2 and
ss2[5]>=pp1    and    ss2[5]<=pp2    and    ss2[6]>=rr1    and    ss2[6]<=rr2:
result="1"                    crop=ss2[7]
                        mycursor.execute("SELECT count(*) FROM fert_data where
crop=%s",(crop,))
                        cc = mycursor.fetchone()[0]
if cc>0:                        st2="1"
                            mycursor.execute("SELECT * FROM fert_data where
crop=%s",(crop,))
                            data = mycursor.fetchone()
break                            else:
result="2"
```

6.2 Output Screenshots





Data Preparation

	N	P	K	Temperature	Humidity	ph	rainfall
1	90	42	43	20.87974371	82.00274423	6.502985292	202.9355362
2	85	58	41	21.77046169	80.31964408	7.038096361	226.6555374
3	60	55	44	23.00445915	82.3207629	7.840207144	263.9642476
4	74	35	40	26.49109635	80.15836264	6.980400905	242.8640342
5	78	42	42	20.13017482	81.60487287	7.628472891	262.7173405

(2200, 8)

Check Null and NA values

Index(['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall', 'label'], dtype='object')

List of various crops

N False P False K False temperature False humidity False ph False rainfall False label False dtype: bool

K	N	P	Humidity	ph	rainfall	temperature
199.89	20.8	134.22	92.3333828756	5.929662931809999	112.654779275	22.6309424132
50.05	100.23	82.01	80.35812258109999	5.98389318024	104.6269804001	27.3767983057
19.24	40.02	67.47	65.1184255887	7.13395162948	67.8841511832	29.9733396789
79.92	40.09	67.79	16.8604394237	7.33695662374	80.0589772605	18.8728467519

	N	P	K	Temperature	Humidity	ph	rainfall
1	90	42	43	20.87974371	82.00274423	6.502985292	202.9355362
2	85	58	41	21.77046169	80.31964408	7.038096361	226.6555374
3	60	55	44	23.00445915	82.3207629	7.840207144	263.9642476
4	74	35	40	26.49109635	80.15836264	6.980400905	242.8640342
5	78	42	42	20.13017482	81.60487287	7.628472891	262.7173405

(2200, 8)

Check Null and NA values

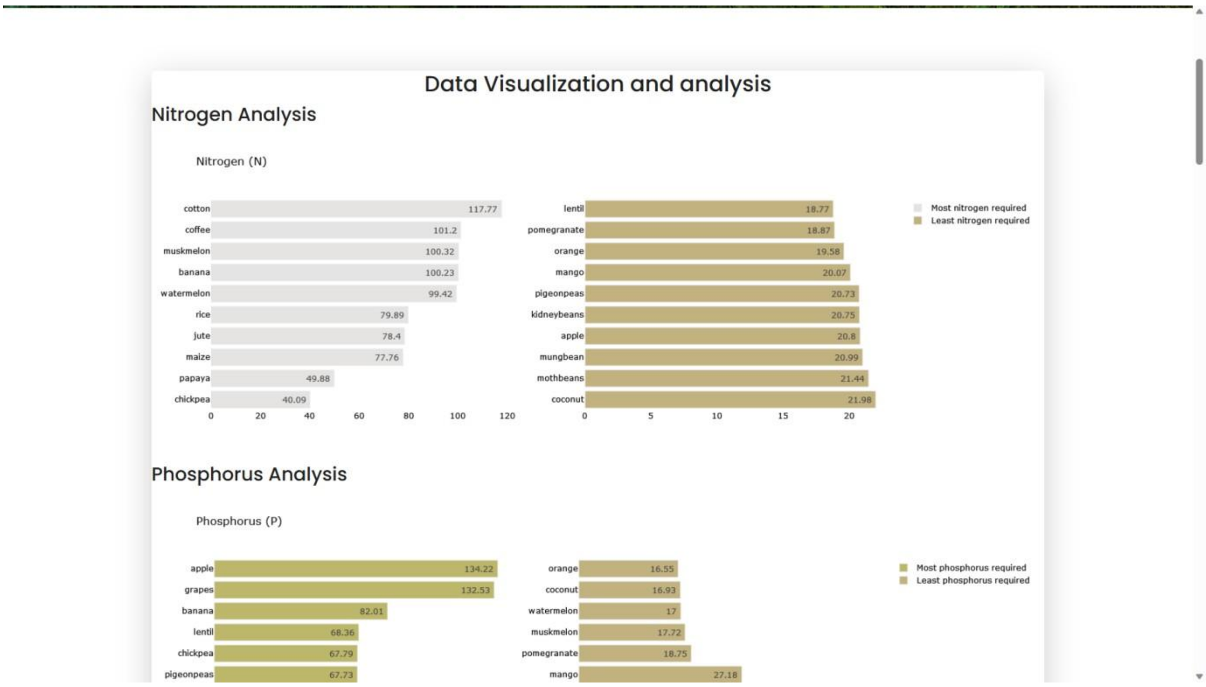
Index(['N', 'P', 'K', 'temperature', 'humidity', 'ph', 'rainfall', 'label'], dtype='object')

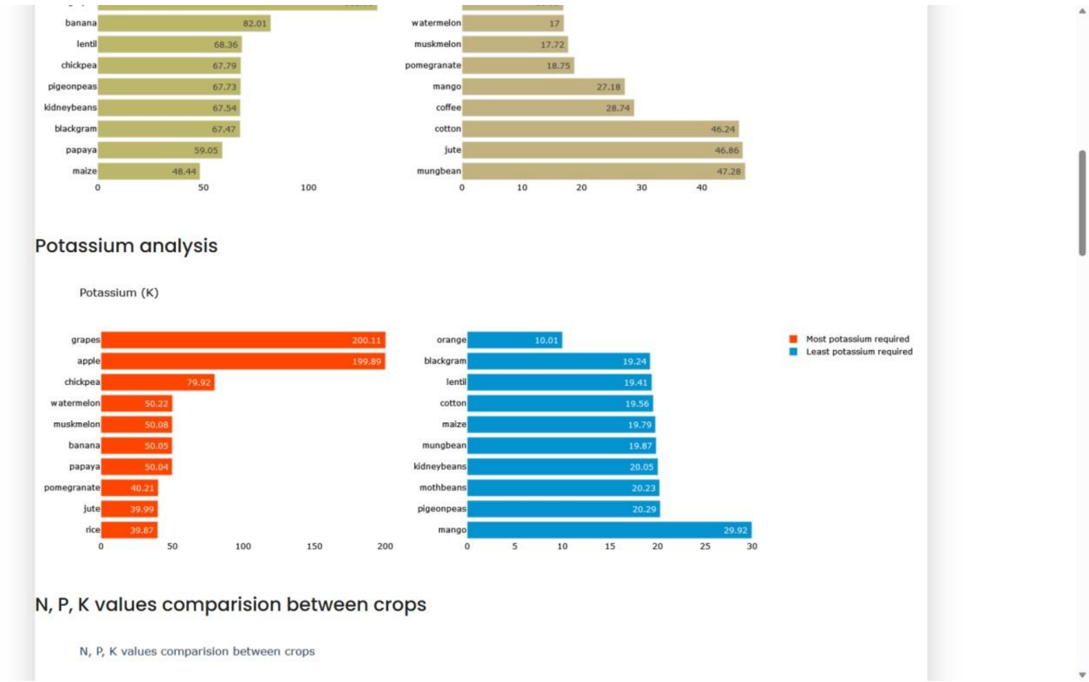
List of various crops

N False P False K False temperature False humidity False ph False rainfall False label False dtype: bool

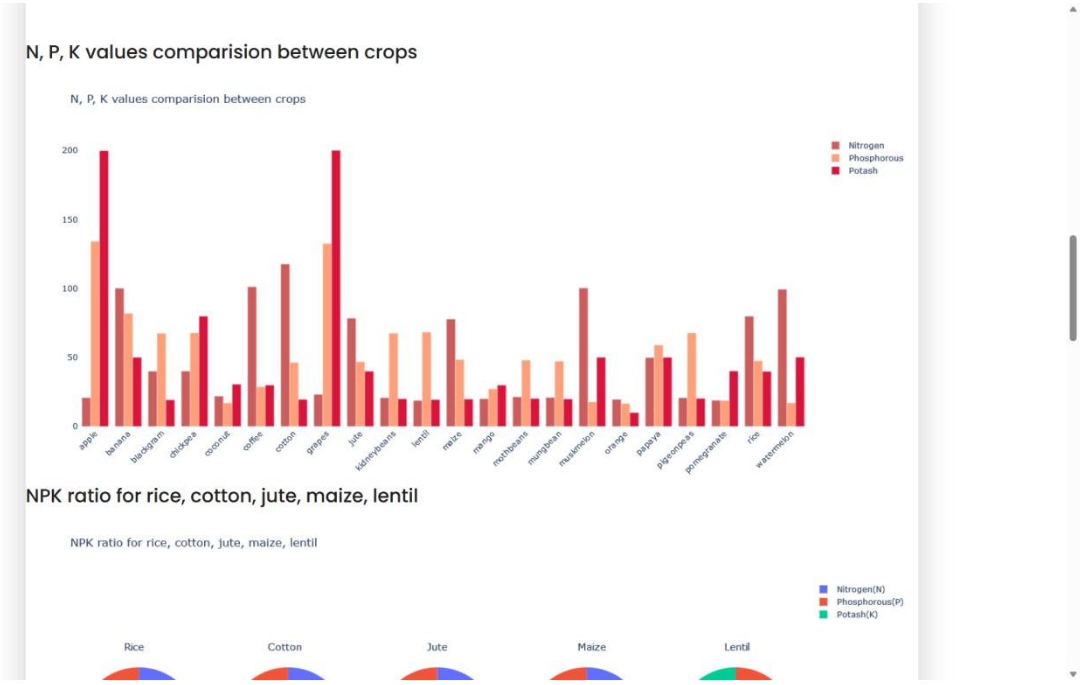
K	N	P	Humidity	ph	rainfall	temperature
199.89	20.8	134.22	92.3333828756	5.929662931809999	112.654779275	22.6309424132
50.05	100.23	82.01	80.35812258109999	5.98389318024	104.6269804001	27.3767983057
19.24	40.02	67.47	65.1184255887	7.13395162948	67.8841511832	29.9733396789
79.92	40.09	67.79	16.8604394237	7.33695662374	80.0589772605	18.8728467519
30.59	21.98	16.93	94.84427180610001	5.97656212619	175.686645804	27.4098921723

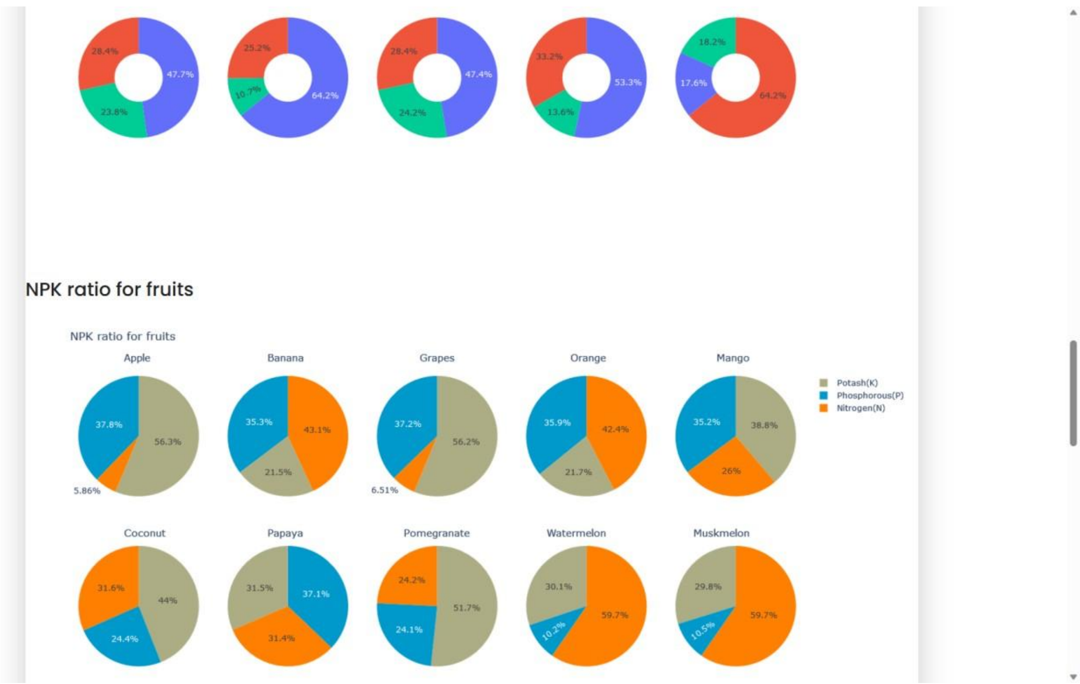
DATA VISUALIZATION

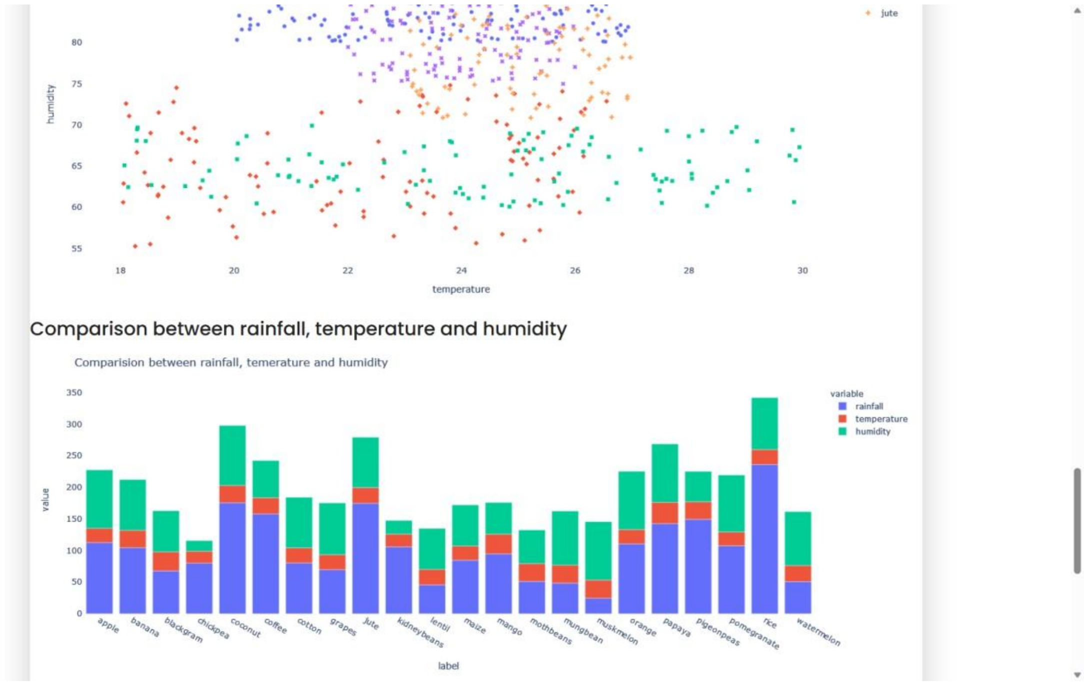

Data Visualization and analysis
Nitrogen Analysis
Nitrogen (N)
cotton 117.77
coffee 101.2
muskmelon 100.32
banana 100.23
watermelon 99.42
rice 79.89
jute 78.4
maize 77.76
papaya 49.88
chickpea 40.09
lentil 18.77
pomegranate 18.87
orange 19.58
mango 20.07
pigeonpeas 20.73
kidneybeans 20.75
apple 20.8
mungbean 20.99
mothbeans 21.44
coconut 21.98
Most nitrogen required
Least nitrogen required
Phosphorus Analysis
Phosphorus (P)
apple 134.22
grapes 132.53
banana 82.01
lentil 68.36
chickpea 67.79
pigeonpeas 67.73
orange 16.55
coconut 16.93
watermelon 17
muskmelon 17.72
pomegranate 18.75
mango 27.18
Most phosphorus required
Least phosphorus required


banana 82.01
lentil 68.36
chickpea 67.79
pigeonpeas 67.73
kidneybeans 67.54
blackgram 67.47
papaya 59.05
maize 48.44
watermelon 17
muskmelon 17.72
pomegranate 18.75
mango 27.18
coffee 28.74
cotton 46.24
jute 46.86
mungbean 47.28
Potassium analysis
Potassium (K)
grapes 200.11
apple 199.89
chickpea 79.92
watermelon 50.22
muskmelon 50.08
banana 50.05
papaya 50.04
pomegranate 40.21
jute 39.99
rice 39.87
orange 10.01
blackgram 19.24
lentil 19.41
cotton 19.56
maize 19.79
mungbean 19.87
kidneybeans 20.05
mothbeans 20.23
pigeonpeas 20.29
mango 29.92
Most potassium required
Least potassium required
N, P, K values comparision between crops
N, P, K values comparision between crops

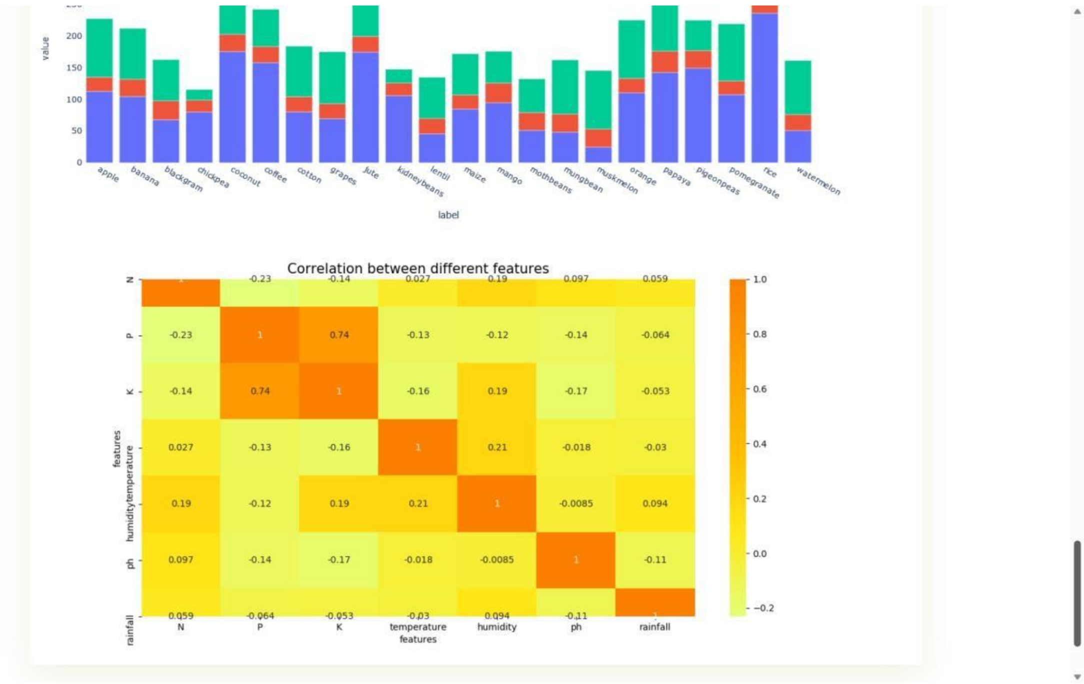
:


N, P, K values comparision between crops
N, P, K values comparision between crops
Nitrogen
Phosphorous
Potash
200
150
100
50
0
apple
banana
blackgram
chickpea
coconut
coffee
cotton
grapes
jute
kidneybeans
lentil
maize
mango
mothbeans
mungbean
muskmelon
orange
papaya
pigeonpeas
pomegranate
rice
watermelon
NPK ratio for rice, cotton, jute, maize, lentil
NPK ratio for rice, cotton, jute, maize, lentil
Nitrogen(N)
Phosphorous(P)
Potash(K)
Rice
Cotton
Jute
Maize
Lentil

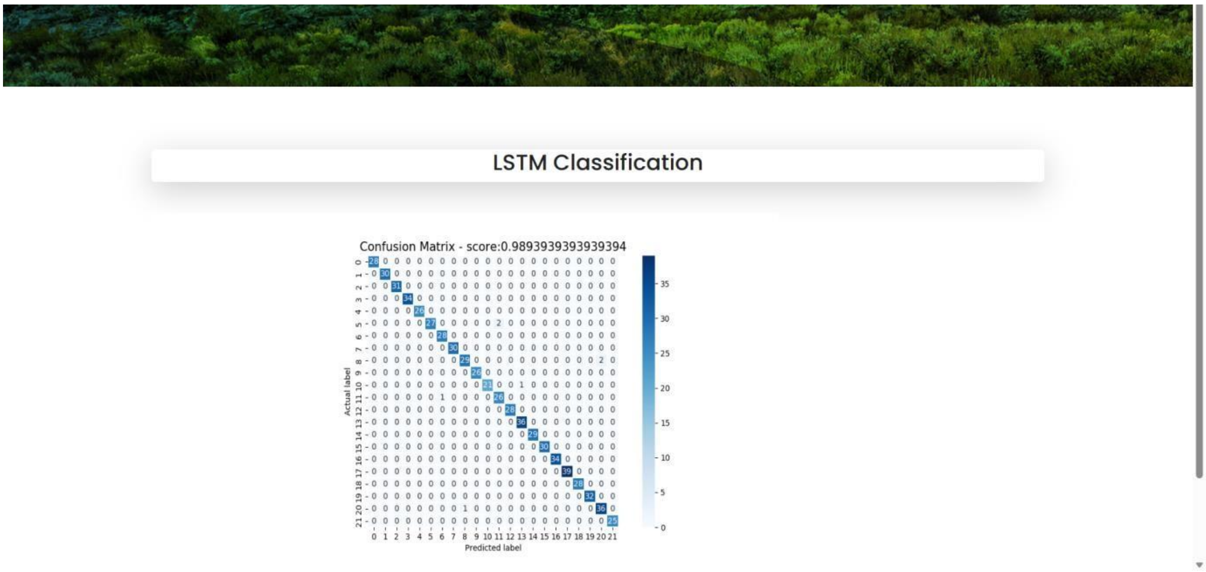

28.4%
47.7%
23.8%
25.2%
10.7%
64.2%
28.4%
47.4%
24.2%
33.2%
53.3%
13.6%
18.2%
17.6%
64.2%
NPK ratio for fruits
NPK ratio for fruits
Apple
Banana
Grapes
Orange
Mango
Potash(K)
Phosphorous(P)
Nitrogen(N)
37.8%
56.3%
5.86%
35.3%
43.1%
21.5%
37.2%
56.2%
6.51%
35.9%
42.4%
21.7%
35.2%
38.8%
26%
Coconut
Papaya
Pomegranate
Watermelon
Muskmelon
31.6%
44%
24.4%
31.5%
37.1%
31.4%
24.2%
51.7%
24.1%
30.1%
10.2%
59.7%
29.8%
10.5%
59.7%

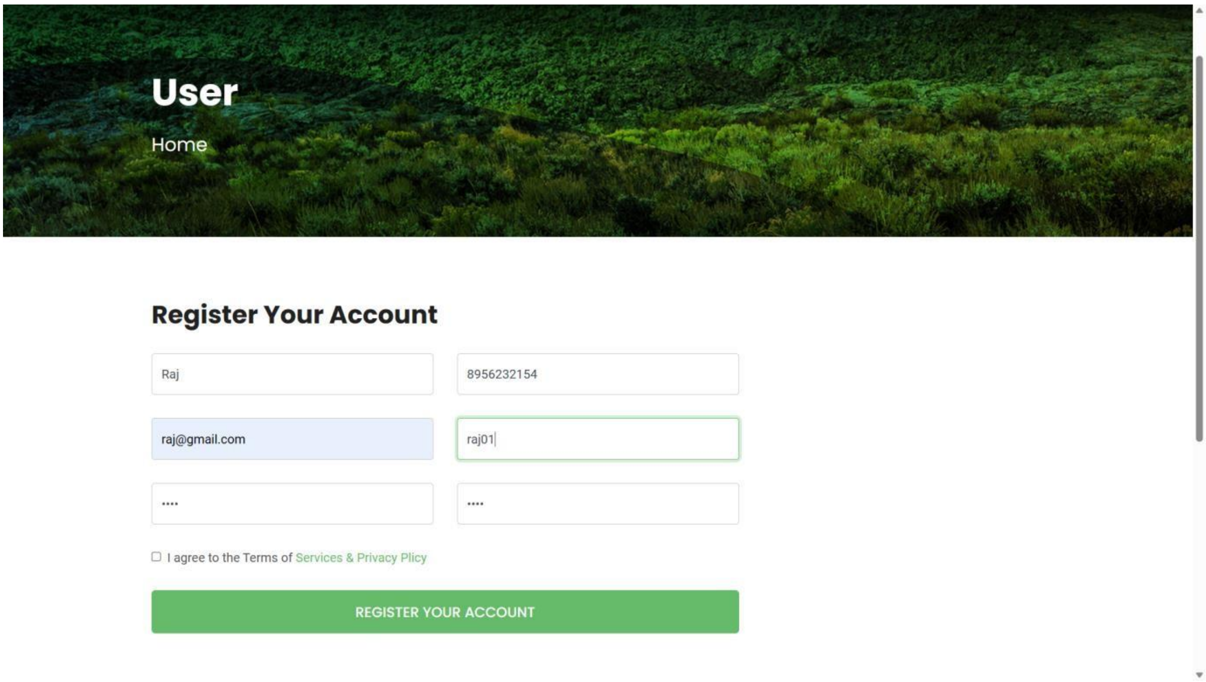

jute
humidity
temperature
Comparison between rainfall, temperature and humidity
Comparision between rainfall, temerature and humidity
variable
rainfall
temperature
humidity
value
label
apple
banana
blackgram
chickpea
coconut
coffee
cotton
grapes
jute
kidneybeans
lentil
maize
mango
mothbeans
mungbean
muskmelon
orange
papaya
pigeonpeas
pomegranate
rice
watermelon

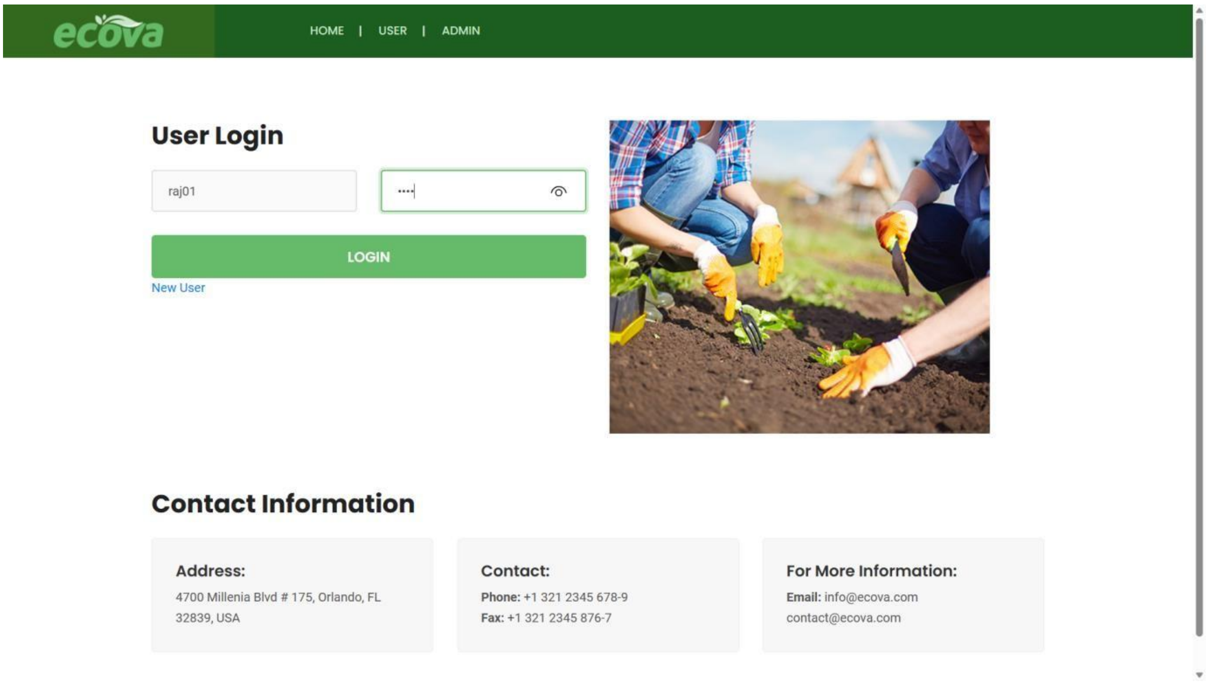
:

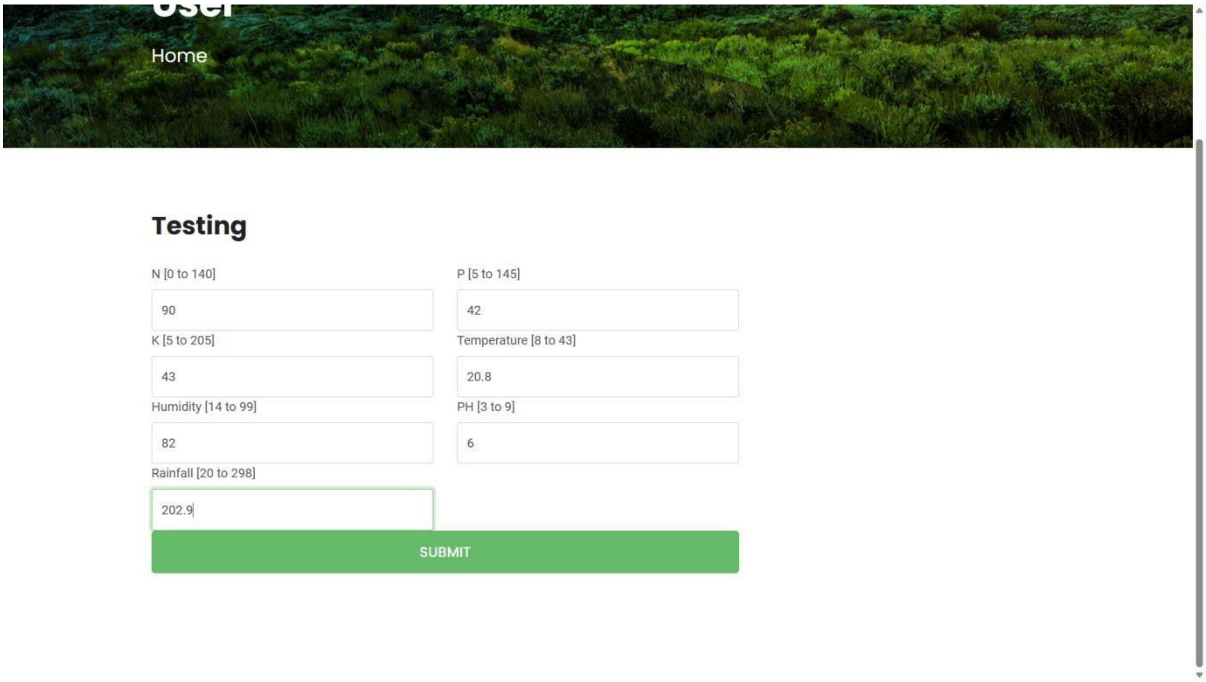

value
label
Correlation between different features
features
N
P
K
temperature
humidity
ph
rainfall

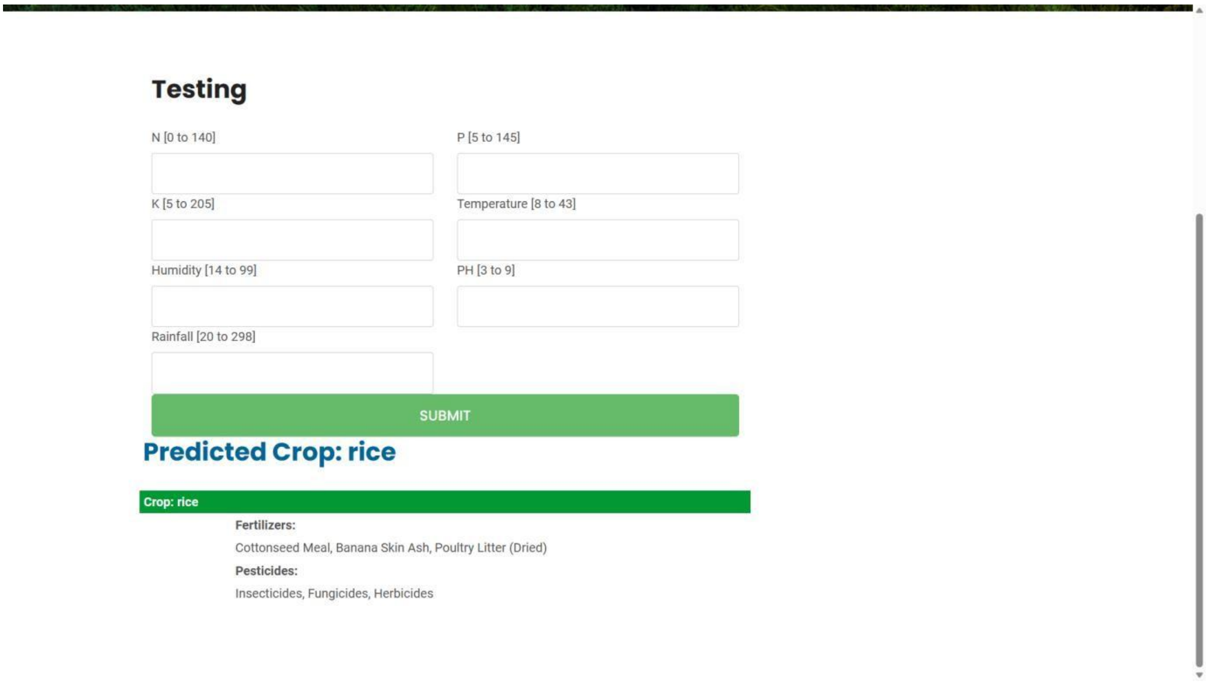
:


LSTM Classification
Confusion Matrix - score:0.9893939393939394
Actual label
Predicted label


User
Home
Register Your Account
Raj
8956232154
raj@gmail.com
raj01
I agree to the Terms of Services & Privacy Plicy
REGISTER YOUR ACCOUNT


ecova
HOME | USER | ADMIN
User Login
raj01
LOGIN
New User
Contact Information
Address:
4700 Millenia Blvd # 175, Orlando, FL 32839, USA
Contact:
Phone: +1 321 2345 678-9
Fax: +1 321 2345 876-7
For More Information:
Email: info@ecova.com
contact@ecova.com


Home
Testing
N [0 to 140]
90
P [5 to 145]
42
K [5 to 205]
43
Temperature [8 to 43]
20.8
Humidity [14 to 99]
82
PH [3 to 9]
6
Rainfall [20 to 298]
202.9
SUBMIT


Testing
N [0 to 140]
P [5 to 145]
K [5 to 205]
Temperature [8 to 43]
Humidity [14 to 99]
PH [3 to 9]
Rainfall [20 to 298]
SUBMIT
Predicted Crop: rice
Crop: rice
Fertilizers:
Cottonseed Meal, Banana Skin Ash, Poultry Litter (Dried)
Pesticides:
Insecticides, Fungicides, Herbicides

References

1. Zeng, Y., Ma, X., Wang, J., & Guo, J. (2022). A deep learning-based crop yield prediction model with feature selection. Computers and Electronics in Agriculture, 201, 106319.
2. Liu, M., & Liu, G. (2022). Combining Landsat 8 and Sentinel-2 data for crop yield prediction using machine learning methods. Journal of Applied Remote Sensing, 16(1), 016516.
3. Wang, J., Luo, Y., Wu, L., Shi, Y., & Gu, B. (2022). An improved crop yield prediction model based on deep learning and image segmentation. Journal of Applied Remote Sensing, 16(2), 026510.
4. Silva, T. C., Maia, A. H., Rodrigues, L. N., & Fernandes, H. B. (2022). Use of Sentinel-2 satellite images and machine learning techniques for soybean yield prediction in Brazil. Remote Sensing, 14(3), 426.
5. Shakeri, S., & Ahmadi, M. (2021). Evaluation of satellite-based and groundbased methods for crop yield prediction: a case study in Iran. International Journal of Remote Sensing, 42(9), 3076-3095.
6. Zhang, C., Yu, L., Cao, W., Zhu, L., & Lu, J. (2021). A crop yield prediction method based on deep learning and feature fusion. Computers and Electronics in Agriculture, 183, 106033.
7. Huang, S., Li, S., Li, G., Li, J., Li, X., & Li, Y. (2021). A hybrid deep learning model for crop yield prediction with remote sensing data. International Journal of Applied Earth Observation and Geoinformation, 105, 102390.
8. Kisi, O., Yaseen, Z. M., & Şen, Z. (2021). A novel machine learning method for crop yield prediction using atmospheric and climate variables. Computers and Electronics in Agriculture, 185, 106110.
9. Zhang, Y., Chen, X., Chen, H., Li, D., & Yang, G. (2020). Maize yield prediction using machine learning algorithms based on weather and remote sensing data. Journal of Applied Remote Sensing, 14(3), 036518.
10. Hu, Z., Chen, S., Huang, W., Hu, W., & Li, Y. (2020). Crop yield prediction using deep learning and remote sensing data. Remote Sensing, 12(1), 146.
11. Li, X., Hu, J., Li, J., Liu, L., Li, J., & Xie, J. (2020). A deep learning-based method for predicting soybean yield using multisource data. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 13, 4420-4430.

12. Yang, Y., Liu, Q., Liu, Y., & Cui, T. (2020). A spatiotemporal machine learning model for predicting soybean yield using high‐resolution remote sensing data. Journal of Geophysical Research: Biogeosciences, 125(8), e2020JG005648.
13. Li, Y., Zhang, X., Liu, F., & Li, Q. (2019). Prediction of rice yield using machine learning algorithms based on climate and soil variables. Agricultural and Forest Meteorology, 264, 240-251.
14. Ma, Y., He, X., & Wang, Y. (2019). Crop yield prediction using deep residual networks. Computers and Electronics in Agriculture, 165, 104963.
15. Patidar, S., & Srinivasan, R. (2019). Machine learning approaches for yield prediction in precision agriculture: A review. Computers and Electronics in Agriculture, 157, 324-337.
16. Zhang, B., Chen, J., & Wei, X. (2019). A novel approach to estimate the yield of rice from UAV-acquired images and machine learning techniques. Remote Sensing, 11(5), 584.
17. Baniya, B. K., Dahal, K., & Maharjan, K. L. (2018). A review of machine learning techniques for crop yield prediction. Computers and Electronics in Agriculture, 154, 37-49.
18. Gopal, R., Jadhav, A. A., & Pawar, V. M. (2018). Crop yield prediction using machine learning techniques. In 2018 3rd International Conference for Convergence in Technology (I2CT) (pp. 1-6). IEEE.
19. Husson, O., Sinoquet, H., & Andrieu, B. (2018). Deep learning for precision agriculture: a review. Agronomy for Sustainable Development, 38(4), 41.
20. Li, Y., Chen, D., & Zhu, J. (2018). Deep learning for crop yield prediction based on remote sensing data. International Journal of Remote Sensing, 39(21), 73087325.

Web References Python:

- Official Python Documentation: https://www.python.org/doc/
- Python Tutorial by W3Schools: https://www.w3schools.com/python/
- Python Tutorial by Tutorialspoint: https://www.tutorialspoint.com/python/index.htm **TensorFlow:**
- Official TensorFlow Documentation: https://www.tensorflow.org/api_docs
- TensorFlow Tutorial by TensorFlow: https://www.tensorflow.org/tutorials

TensorFlow Tutorial by DataCamp: https://www.datacamp.com/community/tutorials/tensorflow-tutorial **Pandas:**

- Official Pandas Documentation: https://pandas.pydata.org/docs/

Pandas Tutorial by W3Schools: https://www.w3schools.com/python/pandas/default.asp Pandas Tutorial by DataCamp: https://www.datacamp.com/community/tutorials/pandas-tutorial-dataframepython **MySQL:**

- Official MySQL Documentation: https://dev.mysql.com/doc/
- MySQL Tutorial by W3Schools: https://www.w3schools.com/mysql/
- MySQL Tutorial by Tutorialspoint: https://www.tutorialspoint.com/mysql/index.htm