DrillBit

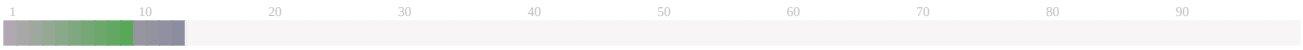
The Report is Generated by DrillBit Plagiarism Detection Software

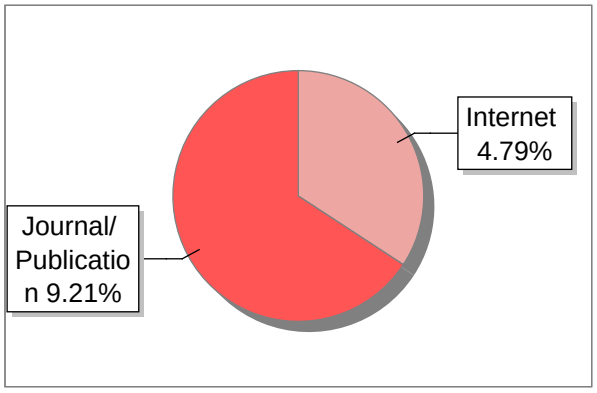
Submission Information

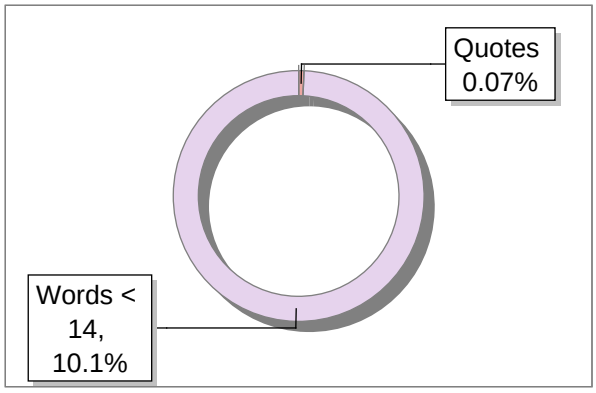
Author Name	Yogesh Saini
Title	PG REPORT
Paper/Submission ID	1422832
Submitted by	chieflibrarian@msrit.edu
Submission Date	2024-02-12 09:35:09
Total Pages	26
Document type	Project Work

Result Information

Similarity **14 %**







Exclude Information

Quotes	Not Excluded
References/Bibliography	Not Excluded
Sources: Less than 14 Words %	Not Excluded
Excluded Source	**0 %**
Excluded Phrases	Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	No

A Unique QR Code use to View/Download/Share Pdf File

DrillBit

DrillBit Similarity Report

14	32	B
SIMILARITY %	MATCHED SOURCES	GRADE

A-Satisfactory (0-10%)
B-Upgrade (11-40%)
C-Poor (41-60%)
D-Unacceptable (61-100%)

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	www.mdpi.com	2	Internet Data
2	Partial multi-label learning with noisy side information by Sun-2020	1	Publication
3	dspace.srmist.edu.in	1	Publication
4	profilelogin.admissione.online	1	Publication
5	scholarworks.umass.edu	1	Publication
6	Thesis Submitted to Shodhganga, shodhganga.inflibnet.ac.in	1	Publication
7	heliaelectronic.com	<1	Internet Data
8	www.mdpi.com	<1	Publication
9	ijisae.org	<1	Publication
10	a0af5022-c1e5-45af-9c8a-b11d448e331e.filesusr.com	<1	Publication
11	Fundamental Principles that Govern Retrofitting of Reinforced Concrete Columns b by Wu-2006	<1	Publication
12	Physics-Aware Processing of Rotational Micro-Doppler Signatures for DBN-Based UA by Madanayake-2020	<1	Publication
13	spie.org	<1	Publication

14	asbmr.onlinelibrary.wiley.com	<1	Internet Data
15	mdpi.com	<1	Internet Data
16	irp.fas.org	<1	Publication
17	sportdocbox.com	<1	Internet Data
18	ir.canterbury.ac.nz	<1	Publication
19	Machine learning for environmental monitoring by Hino-2018	<1	Publication
20	Not the End of the World Post-Classical Decline and Recovery in Rural by Roberts-2018	<1	Publication
21	plosjournal.deepdyve.com	<1	Internet Data
22	Recent trends in quantitative aspects of microscopic X-ray fluorescence analysis by Koe-2010	<1	Publication
23	scientific.net	<1	Internet Data
24	springeropen.com	<1	Internet Data
25	Thesis Submitted to Shodhganga Repository	<1	Publication
26	The perception of static colored noise Detection and masking described by CIE94 by Marce-2008	<1	Publication
27	www.academia.edu	<1	Internet Data
28	www.ijaeast.com	<1	Publication
29	www.karyailmiah.trisakti.ac.id	<1	Publication
30	www.mdpi.com	<1	Internet Data
31	www.mdpi.com	<1	Publication

32	ACM Press the 5th international symposium- San Diego, California (, by DeCoro, Christopher- 2007	<1	Publication

EXCLUDED PHRASES

1	**m s ramaiah institute of technology**
2	**msrit**
3	**bangalore**

1. Introduction

1.1 Overview

Machine learning can help humans solve problems which are not easily solvable by humans. Machine learning can be used to solve tasks like classification, prediction, identification, etc. and can be applied to a wide range of other fields such as agri- culture, sports, trade and business, and so on. This project aims at building a website focused on the agriculture sector, solving two significant issues crop recommendation and crop disease identification. The method used to solve these problems is by training models on datasets available over the internet and comparing them. Models with reasonable accuracy are embedded into the website, which can be then deployed on the cloud.

When it comes to problems beyond human comprehension, machine learning proves to be a potent ally. It performs exceptionally well in tasks like identification, prediction, and classification. Its adaptability crosses many fields, with notable effects in trade, business, sports, agriculture, and other areas. Specifically aimed at the agriculture industry, this research uses machine learning to address two critical problems: crop disease detection and crop recommendation.

The main goal is to develop an agriculture-focused website that is easy to use and incorporates models that are capable of tackling these problems. These models are compared to make sure the models that show excellent accuracy are chosen. The selected models are then smoothly incorporated into the website, improving its usefulness and functionality for individuals involved in agriculture. The benefits can be accessed remotely by farmers and other agricultural stakeholders thanks to the cloud deployment of these models.

Essentially, this initiative is an example of how machine learning is being applied forward-thinkingly in agriculture to provide concrete solutions to challenging issues. The website transforms into a useful resource for the agricultural community by providing a dependable platform for crop suggestion and disease identification. This gives farmers the ability to make educated decisions and promotes a more resilient and productive agricultural sector.

1.2 Problem Definition

Climate change has been quite effective over the past five years. Less knowledge about scientific ways of farming also leads to wrong decisions in the selection of crops. Farmers often tend to rely on experiences which are limited and also full of errors. Overall the agriculture industry suffers a huge loss due to improper usage of knowledge such as soil constituents present, pH of the soil, and early detection of diseases of plants. This problem can be solved by making proper use of technology. With the help of machine learning and the web, this solution can reach every individual having access to a mobile phone with an internet connection on it.

2. Literature Survey

There are many attempts made to tackle the problem of crop recommendation and plant disease classification.

1. G Chauhan and A. Chaudhary proposed the ways to recommend crops based on soil type and used random forest and decision trees to make the prediction. This showed that the task of predicting crops based on land patterns would be helpful via a random forest classifier.
2. S.P. Mohanty has an open-source dataset of plant leaf images along with their grayscale and segmented part. The paper published by him talks about the classification task done by using AlexNet.
3. This paper talks about using the above-mentioned dataset on various DL classification models, and Inspired by this state of the art result, and I decided to use the CNN approach for plant disease detection.
4. Department of Computer Science and Engineering, Lovely Professional University, Phagwara 144411, Punjab, India.
5. School of Engineering and Technology, CT University, Ludhiana 142024, Punjab, India.
6. School of Computer Science, SCS, Taylor's University, Subang Jaya 47500, Malaysia.
7. Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, P.O. Box 84428, Riyadh 11671, Riyadh, Saudi Arabia.

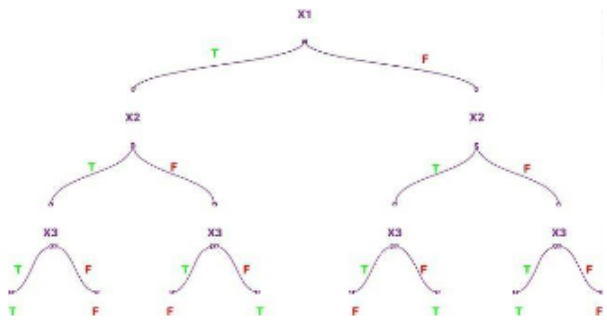


Figure 1 : Pictorial representation of the decision tree

3. Hardware and Software Requirements

3.1 Hardware Requirements

A desktop with 2.4GHz CPU, 8G memory, 32-bit or 64-bit processor, minimum 8-bit graphic adapter and running windows operation system. Basic hardware like monitor, mouse and keyboard are a must for input and output. A CD-ROM or a USB port to install the necessary software. An additional camera will be required for real-time video processing.

3.2 Software Requirements

Python with any version with the following modules installed, OpenCV, NumPy and machine learning packages like Pickle, TensorFlow, Scikit-learn, Keras and Windows operation system. These libraries will make it easier to put machine learning algorithms for crop recommendation and disease detection into practice. A strong backend architecture can be built using the Flask or Django frameworks, which will allow the machine learning models to be seamlessly integrated with the website. To create an interface that is easy to use and intuitive for users, HTML, CSS, and JavaScript are necessary on the front end. Pandas and NumPy are essential tools for data manipulation and analysis since datasets are an essential part of training and evaluating machine learning models. In addition, the website needs to have file upload features so that users can send in pictures of sick plants for examination.

4. Software Requirements Specification

4.1 System Features

4.1.1 User-Friendly Interface

It is one of the best features. We are aware of how crucial simplicity is, particularly for farmers who might not be tech-savvy. Users can easily submit photographs of their crops for disease detection or enter particular information for crop recommendation thanks to the interface's clear visuals and simple navigation. This promotes inclusivity in the agricultural sector by guaranteeing accessible for a broad spectrum of consumers. The system is a useful tool for farmers with different levels of technological skill since it attempts to close the gap between cutting-edge technology and practical usage.

4.1.2 Real-Time Analysis and Recommendations

It is another important aspect. There is no waiting for farmers with our system. Whether the incoming data is a photograph of a sick plant or specifics about the farming environment, it quickly processes it and produces instantaneous results. In order to make prompt decisions in agriculture, this real-time analysis is essential. Farmers are able to take fast action and avoid potential crop losses since they receive immediate input regarding the presence of diseases or crop recommendations. The practical value of the system is improved by the effectiveness of real-time analysis, which is in line with the dynamic and time-sensitive nature of agricultural practices. Our dedication to provide farmers with timely and useful insights for improved crop management is demonstrated by this feature.

5. System Design Description (SDD)

5.1 System Overview

The overall system can be divided into two parts a) Web server and b) a ML container. The whole web server can be connected to a database most probably a SQL database. The architecture is designed in a way to allow large number of requests. It is easily scalable and deployable.

The technologies used for this whole project are :

- **Docker :** Docker helps in containarization process. It creates a whole devel- opment environment which is easy to use and easily deployable across any server
- **NginX :** Nginx is used to create a load balancer. A load balancer help distrubute the traffic among multiple available servers using different algorithms.
- **Tensorflow :** It is an open-source machine learning framework provided by Google.
- **Keras :** Keras is a deep learning library which provides powerful deep learning solutions
- **MySQL :** A relational database system
- **Flask :** A light-weight python web server

5.1.1 System Architecture

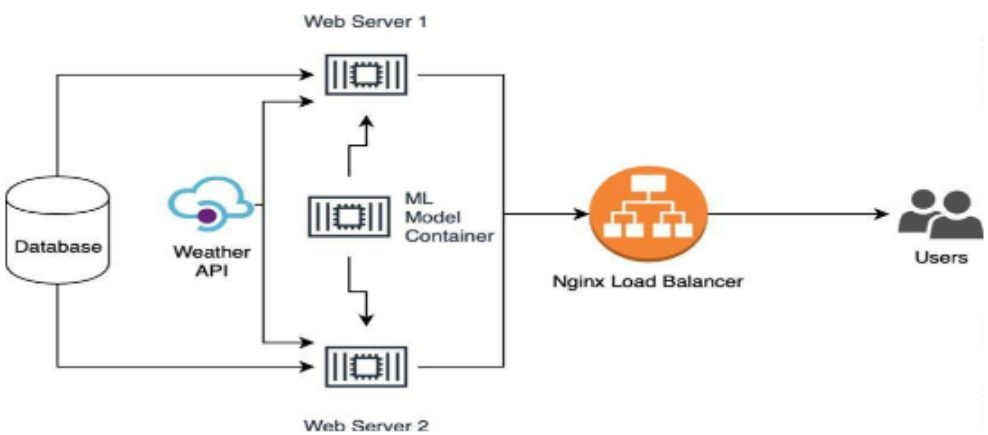


Figure 2 : System Architecture

5.2 Database Design/Data Set Description

The datasets for both the tasks were available online and were open sourced under open license to be used for anybody. The dataset for the first taskconsists of a CSV file which contains 2200 entries of the various factors such as soil condition, temperature, pH, humidity and rainfall and label output as the type of crop well produced in that type of conditions.

There are in total 22 types of crops available in the dataset and they are 'rice', 'maize', 'chickpea', 'kidneybeans', 'pigeonpeas', 'mothbeans', 'mungbean', 'blackgram', 'lentil', 'pomegranate', 'banana', 'mango', 'grapes', 'watermelon', 'muskmelon', 'apple', 'orange', 'papaya', 'coconut', 'cotton', 'jute' and 'coffee'.

The second dataset consists of 70,000 plant images having various diseases. It was a 5 GB data all of resolution 256x256. There are in total 38 classes available where there are 14 different plants and 26 diseases to be identified.

These datasets were used to train all the further mentioned ML algorithms and the one with best accuracy was chosen.

Crop Recommendation

The crop recommendation is basically a classification task. Standard ML algorithms were used to classify various plants. The features trained for the classification were the NPK value of the soil, temperature of the surroundings, pH of the soil and the rainfall in a particular area.

This dataset was trained on the following algorithms :

- Logistic Regression
- Decision Tree
- Support Vector Machine (SVM)
- Multi-Layer Perceptron
- Random Forest

These five algorithms were chosen because given the features and labeled dataset these algorithms can run faster and provide good result on labeled dataset for classification problem.

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

Figure 3 : Random data from the dataset for Crop Recommendation

Plant Disease Identification

The dataset available for this task is a collection of 70,000 images in total. Image classification is a hard task for normal ML algorithms since the feature detectionis not easy. Hence for major image classification problems convolutional neural networks(CNNs) are used which can detect features while training and provide better accuracy over the traditional ML algorithms.

There are 3 different state-of-the-art CNN architecture available for the image classification task.

- VGG16
- ResNet50
- Efficient Net

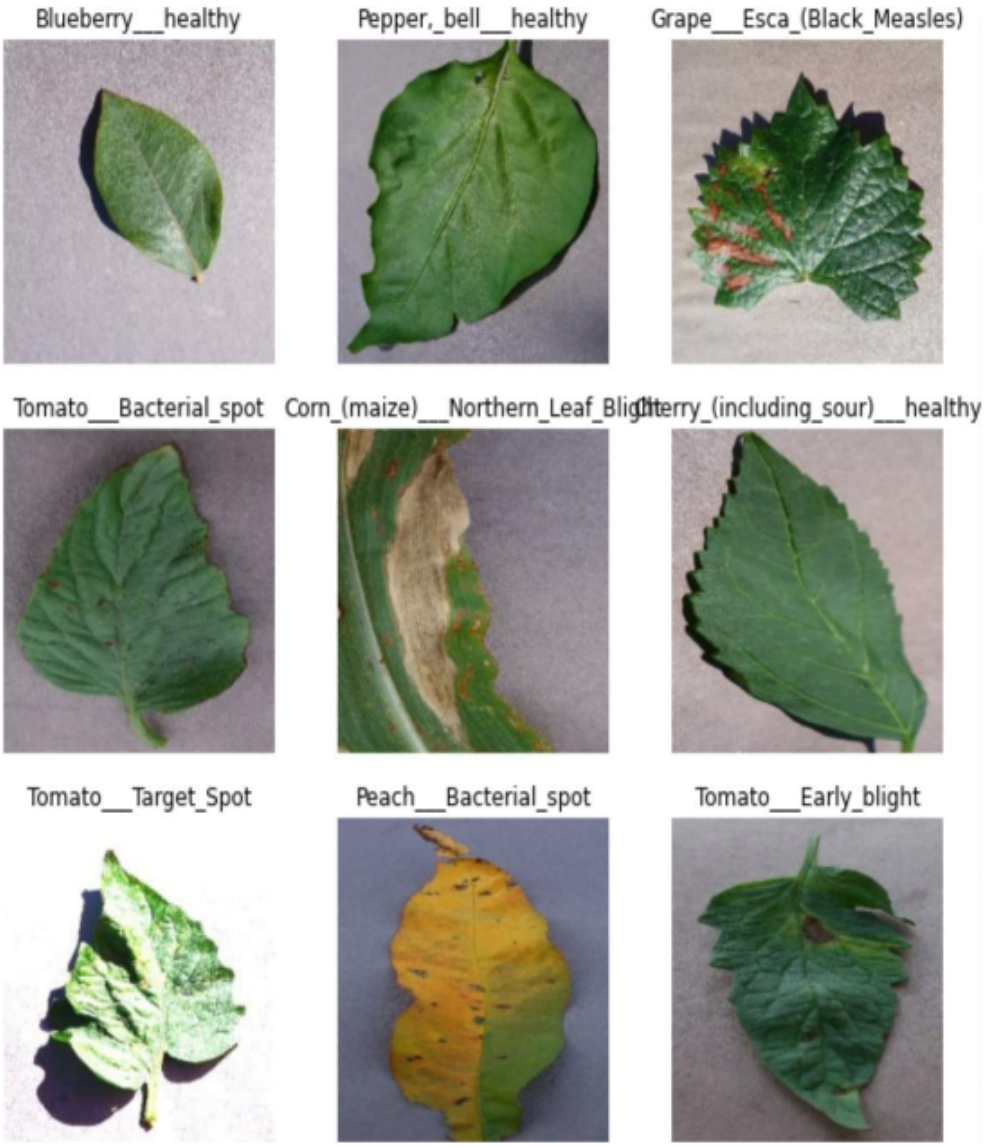


Figure 4 : Random data from the dataset for Plant Disease Identification

5.3 Functional Design

5.3.1 Describe the functionalities of the System

In contemporary agriculture, plant disease detection and crop recommendation systems are essential tools that use cutting-edge technologies to improve crop management techniques. Machine learning and image processing techniques are commonly employed in plant disease detection functions. To identify visual indications, such as discolouration, lesions, or deformities, associated with different illnesses, high-resolution photographs of plant leaves are taken and processed. These photos are analyzed by machine learning algorithms, which are frequently trained on large datasets, to precisely and early detect the presence of diseases.

Conversely, crop recommendation functions aim to offer farmers customized recommendations for the best crop choices, taking into account soil health, environmental conditions, and past crop performance information. These systems combine information from multiple sources, such as past crop rotation data, soil composition, and weather patterns. By analyzing this data, machine learning models produce recommendations that increase yield while lowering the risk of illness. To ensure a comprehensive approach to decision-making, the recommendations may additionally take into account variables like market demand, temperature, and the availability of water.

When combined, these features enable farmers to make knowledgeable choices, fostering effective and sustainable farming. Early disease identification reduces the demand for broad-spectrum insecticides by limiting the spread of illnesses and enabling the prompt use of specific remedies. Crop recommendations also help to maximize the use of available resources, enhance yield results, and promote a more robust and fruitful agricultural ecosystem. In addition to increasing production, technological integration in crop recommendation and disease detection facilitates the shift to precision agriculture, where data-driven insights inform sustainable and profitable farming methods.

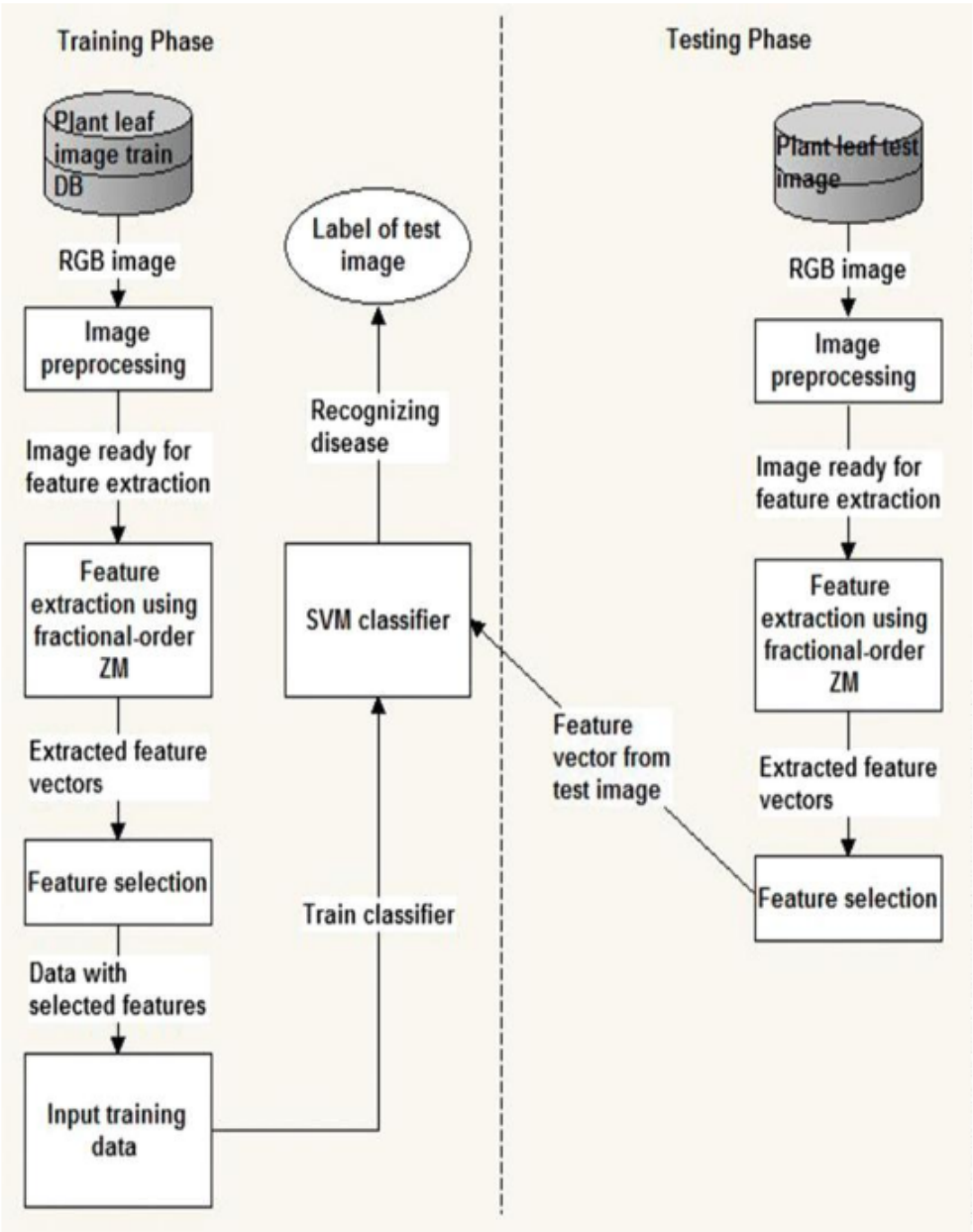


Figure 5 : Functional Architecture

5.3.2 Behavioral Design

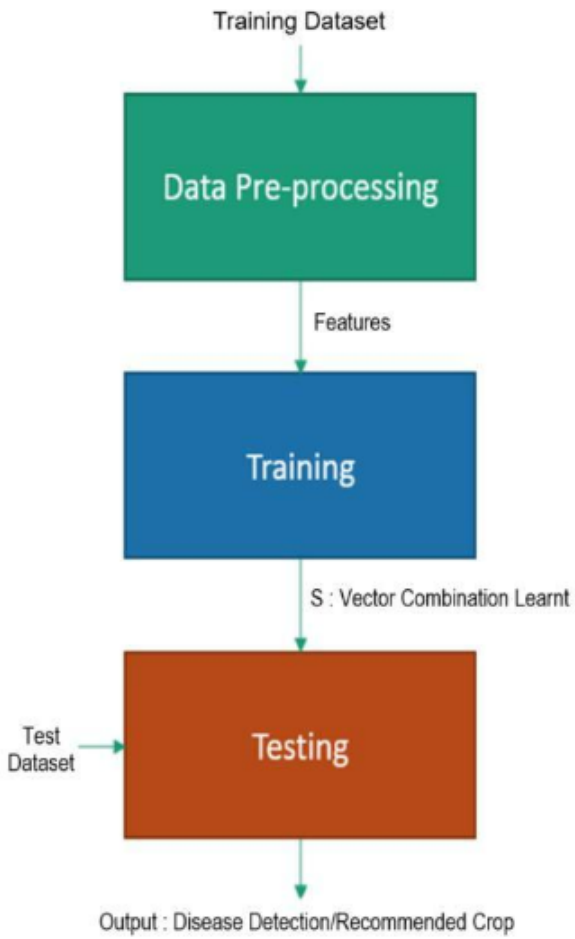


Figure 6 : Behavioral Design for Disease Detection and Crop Recommendation

6. Implementation

Code- app.py

```
from flask import Flask, render_template, request, url_for, flash

from werkzeug.utils import secure_filename

from werkzeug.utils import redirect

import os

import tensorflow

import numpy as np

import pickle


app = Flask(__name__)


BASE_DIR = os.path.dirname(os.path.realpath(__file__))

MODEL_DIR = os.path.join(BASE_DIR, 'model')


UPLOAD_FOLDER = os.path.join(BASE_DIR, 'bucket')

ALLOWED_EXTENSIONS = {'png', 'jpg', 'jpeg', 'JPG'}


MODEL = tensorflow.keras.models.load_model(os.path.join(MODEL_DIR, 'efficientnetv2s.h5'))

REC_MODEL = pickle.load(open(os.path.join(MODEL_DIR, 'RF.pkl'), 'rb'))


app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER


CLASSES = ['Apple___Apple_scab',

        'Apple___Black_rot',

        'Apple___Cedar_apple_rust',

        'Apple___healthy',

        'Blueberry___healthy',

        'Cherry_(including_sour)___Powdery_mildew',
```

'Cherry_(including_sour)___healthy',

'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot',

'Corn_(maize)___Common_rust_',

'Corn_(maize)___Northern_Leaf_Blight',

'Corn_(maize)___healthy',

'Grape___Black_rot',

'Grape___Esca_(Black_Measles)',

'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)',

'Grape___healthy',

'Orange___Haunglongbing_(Citrus_greening)',

'Peach___Bacterial_spot',

'Peach___healthy',

'Pepper,_bell___Bacterial_spot',

'Pepper,_bell___healthy',

'Potato___Early_blight',

'Potato___Late_blight',

'Potato___healthy',

'Raspberry___healthy',

'Soybean___healthy',

'Squash___Powdery_mildew',

'Strawberry___Leaf_scorch',

'Strawberry___healthy',

'Tomato___Bacterial_spot',

'Tomato___Early_blight',

'Tomato___Late_blight',

'Tomato___Leaf_Mold',

'Tomato___Septoria_leaf_spot',

'Tomato___Spider_mites Two-spotted_spider_mite',

'Tomato___Target_Spot',

'Tomato___Tomato_Yellow_Leaf_Curl_Virus',

```
        'Tomato___Tomato_mosaic_virus',
        'Tomato___healthy']

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

def allowed_file(filename):
    return '.' in filename and \
        filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

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

@app.route('/plantdisease/<res>')
def plantresult(res):
    print(res)
    corrected_result = ""
    for i in res:
        if i != '_':
            corrected_result = corrected_result+i
    return render_template('plantdiseaseresult.html', corrected_result=corrected_result)

@app.route('/plantdisease', methods=['GET', 'POST'])
def plantdisease():
    if request.method == 'POST':
        if 'file' not in request.files:
            flash('No file part')
            return redirect(request.url)
```

```
        file = request.files['file']
        # If the user does not select a file, the browser submits an
        # empty file without a filename.
        if file.filename == '':
            flash('No selected file')
            return redirect(request.url)
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
            model = MODEL
            imagefile = tensorflow.keras.utils.load_img(os.path.join(app.config['UPLOAD_FOLDER'], filename),
target_size=(224, 224, 3))
            input_arr = tensorflow.keras.preprocessing.image.img_to_array(imagefile)
            input_arr = np.array([input_arr])
            result = model.predict(input_arr)
            probability_model = tensorflow.keras.Sequential([model,tensorflow.keras.layers.Softmax()])
            predict = probability_model.predict(input_arr)
            p = np.argmax(predict[0])
            res = CLASSES[p]
            print(res)
            return redirect(url_for('plantresult', res=res))
    return render_template("plantdisease.html")


@app.route('/croprecommendation/<res>')
def cropresult(res):
    print(res)
    corrected_result = res
    return render_template('croprecresult.html', corrected_result=corrected_result)


@app.route('/croprecommendation', methods=['GET', 'POST'])
```

```
def cr():
    if request.method == 'POST':
        X = []
        if request.form.get('nitrogen'):
            X.append(float(request.form.get('nitrogen')))
        if request.form.get('phosphorous'):
            X.append(float(request.form.get('phosphorous')))
        if request.form.get('potassium'):
            X.append(float(request.form.get('potassium')))
        if request.form.get('temperature'):
            X.append(float(request.form.get('temperature')))
        if request.form.get('humidity'):
            X.append(float(request.form.get('humidity')))
        if request.form.get('ph'):
            X.append(float(request.form.get('ph')))
        if request.form.get('rainfall'):
            X.append(float(request.form.get('rainfall')))
        X = np.array(X)
        X = X.reshape(1, -1)
        res = REC_MODEL.predict(X)[0]
        # print(res)
        return redirect(url_for('cropresult', res=res))
    return render_template('croprec.html')


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

7. Testing

7.1 Description of Testing

Unit testing : A key component of software development is unit testing, which verifies the functionality of individual software system units or components. A function, method, or module that is the smallest tested portion of the code is referred to as a "unit" in this context. Ensuring that every code unit operates as intended when separated from the rest of the program is the main objective of unit testing. To verify that the code is correct, find errors, and make sure it complies with requirements; developers write and run unit tests, frequently with the aid of testing frameworks. Early error detection and prevention of difficulties from spreading to subsequent phases of development are made possible by unit testing.

Integration Testing: A critical stage of software testing is integration testing, which involves analysing the interfaces and interactions between various system units or components to make sure they work well together when integrated. Finding and fixing problems with data flow, communication breakdowns, and unforeseen dependencies are the main goals that may emerge from these components working together. Top-down and bottom-up are the two basic methods for conducting integration testing. The top-down method involves testing higher-level modules first, then lower-level ones. On the other hand, the bottom-up method starts with testing modules at a lower level and works its way up to higher ones. Furthermore, a third method that combines aspects of top-down and bottom-up tactics is called sandwich testing, or hybrid testing.

7.2 Description of Testing

Test case #	Test Case Name	Test Case Description	Inputs	Expected Output	Actual Output	Status
1.	Plant Disease Detection Test	We are predicting the Disease basis on the Plant Image.	Upload Test Image	Disease Detected	Disease Detected	Successfully Detected
2.	Crop Recommendation Test	We are entering the details for the Crop Recommendation	Fill up the Necessary Fields	Crop Name Detected	Crop Name Detected	Successfully Recommended

8. Results and Discussion

8.1 Model Results

For the first task, 5 classification algorithms were taken and then the accuracy over each of them were measured. For the task 2 the best image classification CNN architectures were trained. VGG16, ResNet50 and EfficientNetv2 are trained. The tables below show the results 2

Algorithm	Accuracy	Precision	Recall	F1-Score
Logistic Regression	94.54	0.95	0.95	0.94
Decision Tree	97.72	0.98	0.98	0.98
Support Vector Machine (SVM)	9.09	0.59	0.09	0.11
Multi-Layer Perceptron	95.22	0.96	0.95	0.95
Random Forest	99.31	0.99	0.99	0.99

Table 1 : Crop recommendation task accuracy over various algorithms. This table concludes that the random forest outperforms every other algorithm achieving 99.92% accuracy. A histogram representation of the accuracy vs models is also shown for the task 1of the project.

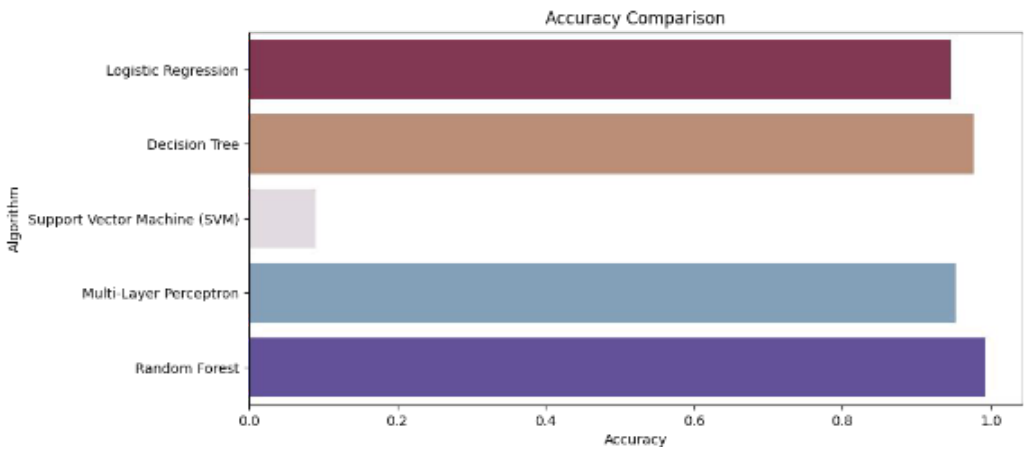


Figure 7 : Histogram representation of the table 1

For the task 2 i.e. the plant disease identification task three models are trained VGG16, ResNet50 and EfficientNetV2S. The accuracy and loss function graph over various epochs of both the models is shown below.

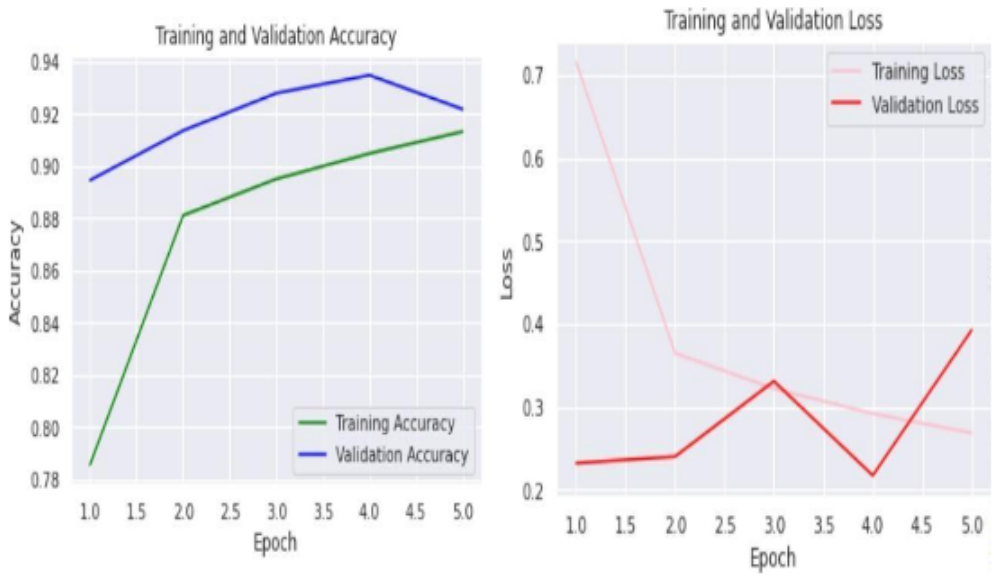


Figure 8 : Accuracy and Loss graphs of the VGG model

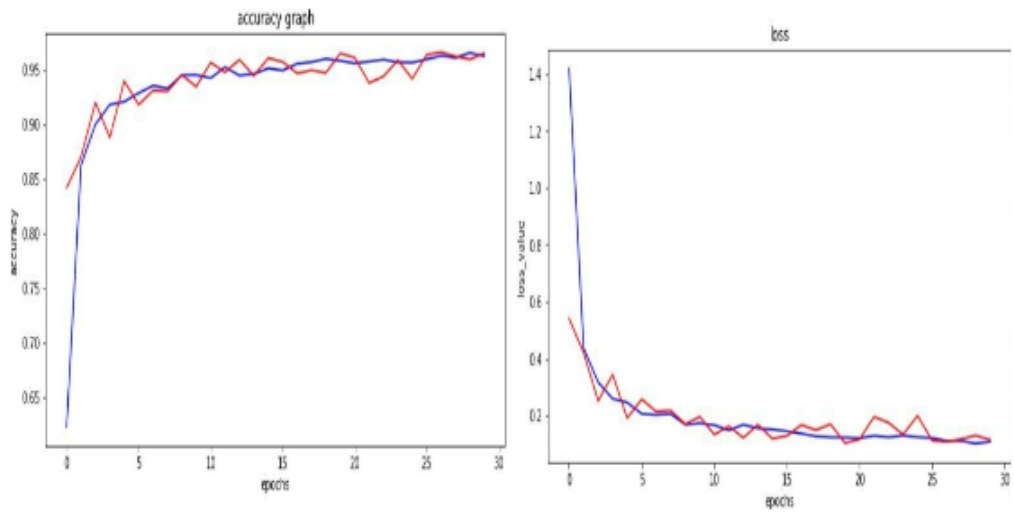


Figure 9 : Accuracy and Loss graphs of the ResNet50 model

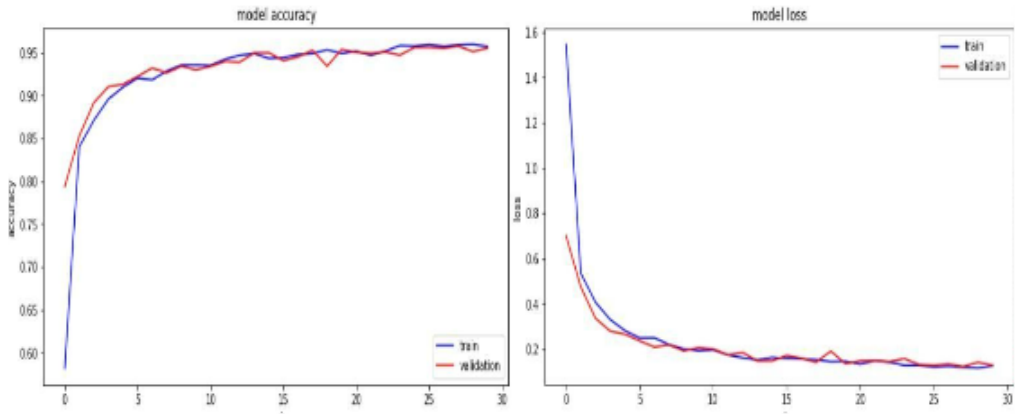


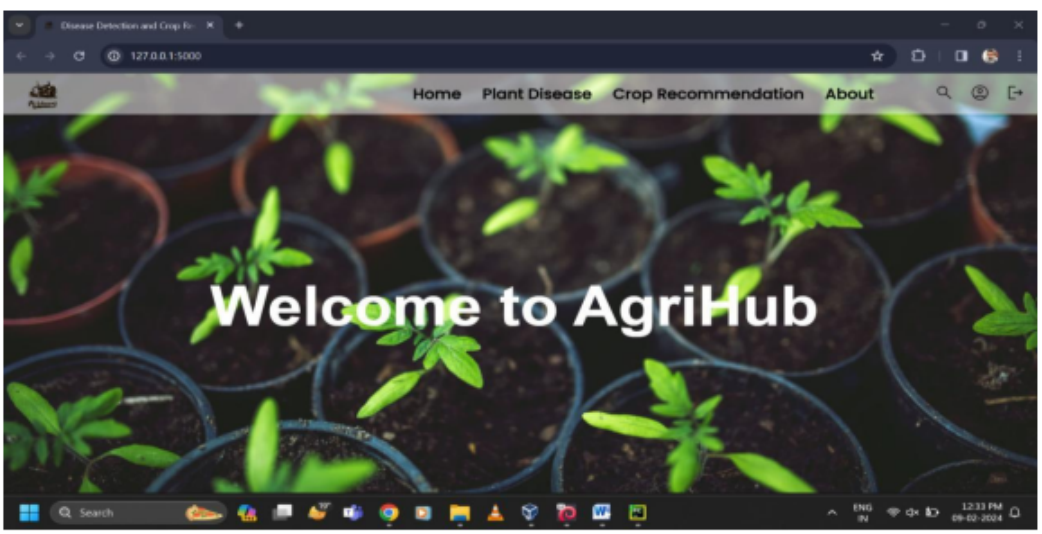
Figure 10 : Accuracy and Loss graphs of the EfficientNetV2S Model

Table 2 : Plant Disease Classification task accuracy over various architectures. This table concludes that EfficientNetV2 is efficient.

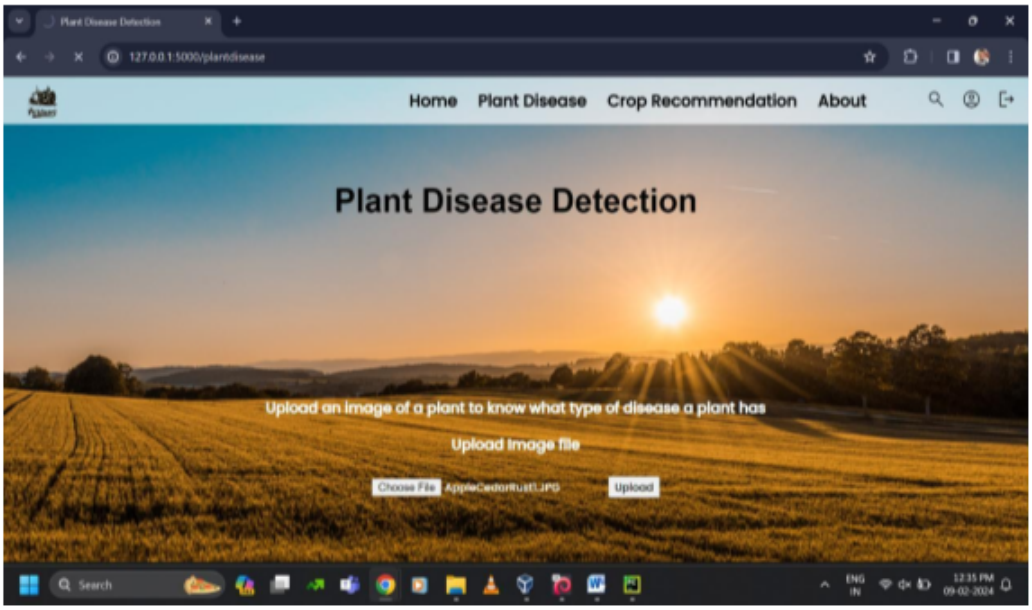
Architecture	Training Acc.	Validation Acc.	Test Accuracy.
VGG16	92.18	91.33	91.78
ResNet50	96.02	95.41	95.53
EfficientNetV2	96.06	95.53	95.83

8.2 Deployed Website

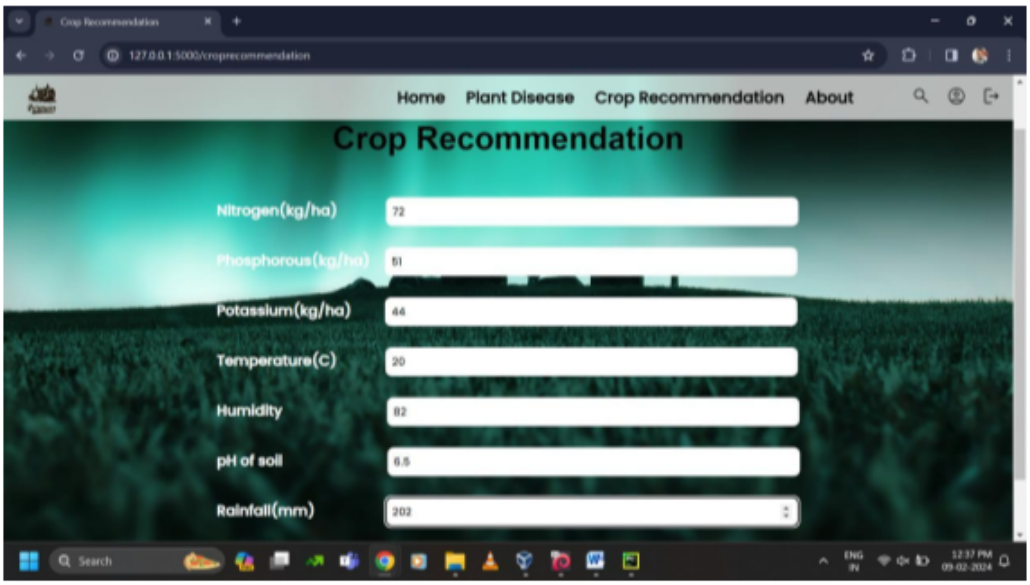
8.2.1 Home Page

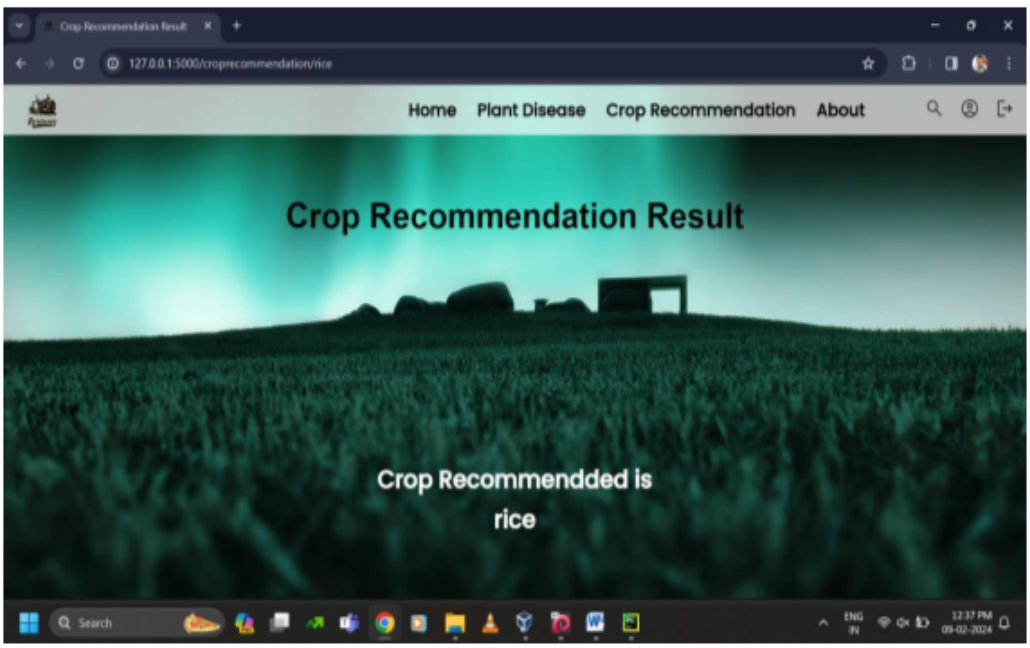


8.2.2 Plant Disease Detection



8.2.3 Crop Recommendation





8.2.4 About


About
127.0.0.1:5000/about
Home
Plant Disease
Crop Recommendation
About
About
Welcome to our innovative platform dedicated to revolutionizing agriculture through cutting-edge technology. Our website specializes in plant disease detection and crop recommendation using advanced machine learning algorithms. We empower farmers with accurate diagnoses of crop diseases, enabling timely intervention. Additionally, our platform provides personalized crop recommendations based on robust data analysis, ensuring optimal yields and sustainable farming practices.
Search
ENG
IN
12:37 PM
09-02-2024

9. Conclusion

The conclusion for Plant Disease Detection and Crop Recommendation We want to solve a major farming issue by simplifying the harvesting process. After experimenting with five alternative approaches to the first challenge, we discovered that Random Forest produced the best results for our particular set of data, with an astounding 99.3% accuracy. This indicates that it is highly proficient at determining crop conditions.

We then compared three models—VGG16, ResNet50, and EfficientNetV2S—for the second task. The winner was EfficientNetV2S, which outperformed the other two with an astounding 96.06% overall accuracy. ResNet50 achieved an overall accuracy of 95.53%, outperforming VGG16. These models are similar to intelligent assistants; they identify issues by analysing images of crops.

The best aspect is that we have now made these intelligent models publicly available on the internet. Just picture your farm with a virtual assistant! Farmers can upload photos of their crops, and our models will promptly inform them of any issues or whether everything is going well. It's like having an extremely intelligent agricultural companion at your fingertips, assisting farmers in making wiser choices and gathering crops more effectively. This is our attempt to make farming a little more tech-savvy and easy.