PLANT DISEASES DETECTION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

D.TRISHA	(21UECM0063)	**(VTU 20622)**
P.CHAITANYA	(21UECM0303)	**(VTU 19750)**
K.VENKATA ABHI SRIRAM	(21UECM0126)	**(VTU 20435)**

Under the guidance of
Mr.C.EDWIN SINGH, M.E
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

PLANT DISEASES DETECTION USING MACHINE LEARNING

Minor project-II report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

D.TRISHA	(21UECM0063)	**(VTU 20622)**
P.CHAITANYA	(21UECM0303)	**(VTU 19750)**
K.VENKATA ABHI SRIRAM	(21UECM0126)	**(VTU 20435)**

Under the guidance of
Mr.C.EDWIN SINGH, M.E
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled ”PLANT DISEASE DETECTION USING MACHINE LEARNING” by ”D.TRISHA (21UECM0063), P.CHAITANYA (21UECM0303), K.VENKATA ABHI SRIRAM (21UECM0126)” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

Signature of Professor In-charge
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
May, 2024

DECLARATION

We declare that this written submission represents our ideas in our own words and where others ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

D.TRISHA

Date: / /

P.CHAITANYA

Date: / /

K.VENKATA ABHI SRIRAM

Date: / /

APPROVAL SHEET

This project report entitled “PLANT DISEASES DETECTION USING MACHINE LEARNING” by D.TRISHA (21UECM0063), P.CHAITANYA (21UECM0303), K.VENKATA ABHI SRIRAM (21UECM0126) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners **Supervisor**

Mr.C.Edwin Singh, M.E.,

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr.M.S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Mr.C.Edwin Singh, M.E.,** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E., Mr. SHARAD SHANDHI RAVI, M.Tech.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

D.TRISHA **(21UECM0063)**
P.CHAITANYA **(21UECM0303)**
K.VENKATA ABHI SRIRAM **(21UECM0126)**

ABSTRACT

Identification of plant disease detection by machine learning and image processing techniques can be used to analyze digital images of plants to identify signs of disease. By training algorithms on large datasets of labeled images, it is possible to develop highly accurate models that can detect the early onset of disease, allowing for more targeted and effective treatments. Additionally, image processing techniques can be used to enhance the quality of the images and extract relevant features for analysis, such as leaf color, shape, and texture. Plant diseases can cause significant damage to crops, resulting in economic losses for farmers and food shortages for consumers. Early detection and diagnosis of plant diseases are essential to prevent their spread and manage the impact on agriculture. In recent machine learning and image processing techniques have been widely used to detect and diagnose plant diseases. There are several existing systems for plant and plant disease detection using machine learning and image processing including, Plant Village is an open source platform for plant disease diagnosis that uses machine learning algorithms to analyze images of plants and identify disease symptoms. The system has a database of over 100 images of healthy and diseased plants and covers more than 10 crop species. This project proposes a system for plant and plant disease detection using machine learning and image processing techniques such as Gray Level Co-occurence Matrix (GLCM) and K Means clustering algorithms. The objective of the system is to improve the accuracy and efficiency of plant disease detection, which is crucial for maintaining crop yields and food security. The system will utilize a dataset of labeled images of healthy and diseased plants to train and validate machine learning algorithms for image classification and segmentation.

Keywords: Classification, Feature extraction, Image processing, Image segmentation, Machine learning, Pattern recognition.

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

SI.NO	ABBREVIATION	DEFINITION
1	CNN	Convolutional Neural Network
2	ER	Entity Relationship
3	GAP	Good Agricultural Practices
4	GDPR	General Data Protection Regulation
5	GLCM	Grey Level Co-occurence Matrix
6	IPPC	International Plant Protection Convention
7	IPR	Intellectual Property Rights
8	KNN	K-Nearest Neighbour
9	SFTA	Segmentation based Fractal Texture Analysis

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

Plants are the foundation of human civilization and provide food, clothing, and shelter to billions of people around the world. However, plant diseases can cause significant damage to crops, resulting in economic losses for farmers and food shortages for consumers. Early detection and diagnosis of plant diseases are essential to prevent their spread and manage their impact on agriculture. Traditionally, plant disease diagnosis has been performed by trained experts who visually inspect plant samples for signs and symptoms of disease. However, those approaches are time-consuming, labor-intensive, and prone to human error. In the new era of machine learning and image processing techniques have emerged as powerful tools for automating the process of plant disease detection and diagnosis. Image processing techniques involve the use of computer vision algorithms to analyze digital images and extract meaningful information. The combination of these two approaches offers several advantages for plant disease detection, including increased speed, accuracy, and scalability.

Machine learning models can be trained using large datasets of labeled images, which can enable accurate and efficient disease detection in real-time.This technology has the potential to revolutionize agriculture and help ensure food security for the growing global population. In which it provides an overview of the current state of research on plant and plant disease detection using machine learning and image processing techniques, highlighting the potential benefits and challenges by using K- means clustering and GLCM algorithms.

Farmer's economic growth relies on the quantity of the products that they grow which is directly dependent on the plant's growth and the yield they get. Plants are exposed to outer environment and are highly prone to diseases that affects the growth of the plant which in turn effects the ecology of the farmer. Plants are the major

victims of the plant diseases. Plants are attacked by numerous kinds of diseases which target different parts of the plant body such as leaf, stem, seed, fruit, and so on. Diseases are specific to specific parts of the plant body. However, this requires continuous monitoring of experts which might be prohibitively expensive in large farms. With economic development and population rise in cities, environmental pollution problems involving air pollution, water pollution, noise and the shortage of land resources have attracted increasing attention.

1.2 Aim of the project

To detect unhealthy region of plant leaves detect Diseases which involves steps like image acquisition, image pre-processing, image segmentation, feature extraction and classification. This is used to detect the plant diseases and provide solutions to recover from the diseases. It shows the affected part of the leaf in percentage. This project has been considered only for diseases and further it can be extended for various diseases. In future it can be extended to find the percentage of the infection in accordance with the affected area. Detects the plant diseases and provide solutions to recover from the disease.

1.3 Project Domain

Image processing consists of the manipulation of images using digital computers. Its use has been increasing exponentially in the last decades. Its applications range from medicine to entertainment, passing by geological processing and remote sensing. Multimedia systems, one of the pillars of the modern information society, rely heavily on digital image processing. The discipline of digital image processing is a vast one, encompassing digital signal processing techniques as well as techniques that are specific to images. An image can be regarded as a function of two continuous variables x and y. To be processed digitally, it has to be sampled and transformed into a matrix of numbers.

Since a computer represents the numbers using finite precision, these numbers have to be quantized to be represented digitally. Digital image processing consists of the manipulation of those finite precision numbers. The processing of digital images

can be divided into several classes: image enhancement, image restoration, image analysis, and image compression. In image enhancement, an image is manipulated, mostly by heuristic techniques, so that a human viewer can extract useful information from it. Image processing algorithms are developed to detect the plant infection or disease by identifying the color feature of the leaf area. K means algorithm is used for color segmentation and GLCM algorithm is used for diseases classification. Vision based plant infection showed efficient result and promising performance.

1.4 Scope of the Project

This project has been considered only for diseases and further it can be extended for various diseases. In future it can be extended to find the percentage of the infection in accordance with the area affected. This project is used to detect the plant diseases and provide solutions to recover from the disease. This project is mainly helpful for farmers to detect the plant diseases in initial stages. The scope of plant and plant diseases detection using machine learning and image processing is vast and has the potential to revolutionize agriculture. The use of these technologies can help farmers and agricultural organizations to identify and diagnose plant diseases accurately and efficiently, reducing crop losses, and ensuring food security for the growing global population.

This technology includes the development of new algorithms and models that can accurately detect and diagnose plant diseases from digital images captured in various environmental conditions. It also includes the development of new image processing techniques that can preprocess and enhance images, removing noise and unwanted artifacts, and enabling accurate analysis.The use of machine learning and image processing techniques can also enable the development of real-time monitoring systems that can continuously capture and analyze images of plants, detecting and diagnosing diseases as soon as they appear. Another aspect of this technology is the development of mobile applications and web-based platforms that can enable farmers to upload images of diseased plants and receive instant feedback on the diagnosis and recommended treatment options.The scope of plant and plant diseases detection using machine learning and image processing is vast, encompassing research, development, and deployment.

Chapter 2

LITERATURE REVIEW

[1] Hasan et al., proposed that the factors affecting the efficiency of micelle-mediated extraction of phenolic compounds from apple pomace was investigated. Higher extraction efficiency by using as a solvent an aqueous solution of Tween 80 in comparison to Triton X-100, Span 20, Tween 20, 70 percentage ethanol, and water was shown. Four independent variables (Tween 80 concentration, time, solvent-to material ratio, and pH) to enhance the recovery of polyphenols from apple pomace was investigated. Applying response surface methodology, the second order polynomial regression equation showing dependence of the yield of polyphenols on the extraction parameters was derived. The adjusted regression coefficient (R2 = 98.73the lack-of-fit test (p lessthan 0.05) showed a good accuracy of the developed model.

[2] Islam et al., discussed about review on Deep Learning Techniques for Plant Disease Detection and Classification a fast-response colorimetric ultraviolet-C (UVC) sensor was demonstrated using a gallium oxide (Ga2O3) photocatalyst with small amounts of triethanolamine (TEOA) in methylene blue (MB) solutions and a conventional RGB photodetector. The color of the MB solution changed upon UVC exposure, which was observed using an in situ RGB photodetector. Thereby, the UVC exposure was numerically quantified as an MB reduction rate with the R value of the photodetector, which was linearly correlated with the measured spectral absorbance using a UV-Vis spectrophotometer. Small amount of TEOA in the MB solution served as a hole scavenger, which resulted in fast MB color changes due to the enhanced charge separation. However, excessive TEOA over 5 wt. percentage started to block the catalytical active site on the surface of Ga2O3, prohibiting the chemical reaction between the MB molecules and catalytic sites. The proposed colorimetric UVC sensor could monitor the detrimental UVC radiation with high responsivity at a low cost.

[3] Y.Shi et al., explained the Plant Disease Detection Using Image Processing

and Machine Learning is one of the important and tedious task in agricultural practices is detection of disease on crops. It requires huge time as well as skilled labor. This paper proposes a smart and efficient technique for detection of crop disease which uses computer vision and machine learning techniques. The proposed system is able to detect 20 different diseases of 5 common plants with 93 percentage accuracy.

[4] Das et al., concluded an experiment which shows classification and detection of plant diseases. Insecticides are not always proved efficient because insecticides may be toxic to some species of birds. It also damages n natural animal food chains. The following two steps are added successively after the segmentation phase. In the first step, they identified the mostly green colored pixels. Then using Otsu's method, pixels are masked having green color. Then those mostly green pixels are masked. After that, red, green and yellow color based pixel cluster are removed which are infected. The experimental results show that proposed technique is the best for plant detection.

[5] Singh et al., proposed about a technique to recognize the disease of two plants. This investigation has been done on two grapes plants and two wheat plants to improve accuracy using image processing techniques. Back propagation (BP) networks were used as the classifiers to identify grape diseases and wheat diseases, respectively. The results showed that identification of the diseases could be effectively achieved using BP networks. While the dimensions of the feature data were reduced by using PCA, the optimal recognition result for grape diseases was obtained as the fitting accuracy was 100 was 97.14 prediction accuracy were both 100 percent.

[6] Kim et al, explained about plant disease detection and its prevention. Most plant diseases are due to bacteria, viruses and fungus. Fungi are recognized first and foremost from their morphology, with emphasis placed on their reproductive structures. The developed processing scheme consists of four main steps, first a color transformation structure for the contribution RGB image is created, and this RGB is transformed to HSI for color generation and his fore color descriptor. Then green pixels are cloaked and removed using specific threshold value, then the image is segmented and the useful segments are extracted, finally the texture statistics is computed from SGDM matrices. Finally, the presence of diseases on the plant leaf

is evaluated.

[7] Rahman et al, discussed a deep learning-based system for identifying five major tomato diseases, namely, bacterial spot, late blight, leaf mold, septoria leaf spot, and spider mites. The authors used a large dataset of tomato images and trained a convolutional neural network (CNN) to identify the diseases. The system achieved an accuracy rate of 99.35 methods.

[8] Zhang et al., proposed a transfer learning-based approach for plant disease identification. The authors used a pre-trained CNN model, Inception-v3, and fine-tuned it using a dataset of tomato plant images infected with five common diseases. The proposed approach achieved an accuracy rate of 98.61.

[9] Rifa'i et al., proposed a hyperspectral imaging-based approach for identifying powdery mildew on wheat plants. The authors collected hyperspectral images of wheat leaves infected with powdery mildew and used multivariate data analysis techniques to analyze the images. They identified specific wavelengths in the images that were sensitive to the disease and used them to develop a classification model. The proposed approach achieved high accuracy rates of up to 94powdery mildew on wheat plants.

[10] S.Arivazhagan, reported the use of traditional methods such as direct observation, microscopy, and biochemical tests for the identification of leaf spot diseases on okra plants. In recent years, advanced technologies such as machine learning and computer vision have been developed to automate plant and disease detection. Computer vision involves the use of cameras to capture images of plants and machine learning algorithms analyze the images for disease detection.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

The existing method for plant disease detection is simply naked eye observation by experts through which identification and detection of plant diseases is done. Plant disease identification by visual way is more laborious task and at the same time, less accurate and can be done only in limited areas. As time progressed few of the algorithms have been applied such as SFTA to make it feasible into the digital world but lack of accuracy has been the major constraints.

Disadvantages of existing systems for plant and plant disease detection using machine learning and image processing is one of the main challenge in this project.Some existing systems may not be effective in detecting diseases at different growth stages of plants or may not be applicable to all types of plant diseases. Furthermore, the accuracy of these systems depends heavily on the quality and diversity of the training data, which may be affected by factors such as lighting conditions, image resolution, and plant growth stage. Developing and implementing machine learning-based methods for plant disease detection requires expertise in both computer vision and plant pathology, which can be a barrier for some researchers or practitioners. Implementing existing systems may also require specialized hardware and software, which can be expensive. Lastly, the use of machine learning for plant disease detection raises ethical concerns related to data privacy, intellectual property, and the potential impact on farmers and other stakeholders in the agriculture industry.

3.2 Proposed System

The project is to detect the plant diseases and provide the how much leaf is affected by the diseases. In the proposed system. It is providing how much the plant is

effected and take precautions according to that from the leaf diseases and also show the affected part of the leaf by image processing technique. The existing system can only identify the type of diseases which affects the leaf. It will provide a result within fraction of seconds and guided throughout the project.

3.3 Feasibility Study

3.3.1 Economic Feasibility

This study is carried out to check the economic impact that the system will have on the organization. There have been significant advancements in plant disease detection technologies, including the use of sensors, imaging techniques, and machine learning algorithms. These technologies can provide quick and accurate diagnoses of plant diseases, which can help prevent the spread of the disease and minimize crop losses. The value of the crops and potential losses due to disease also play a significant role in the economic feasibility of disease detection.

3.3.2 Technical Feasibility

Machine learning: Machine learning algorithms can be trained to analyze large datasets of plant images and identify patterns that are indicative of disease. This technology is highly accurate and can analyze a large number of plants quickly, but it requires specialized equipment and expertise.

3.3.3 Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. The social feasibility of ML-based disease detection depends on how well the technology is accepted by farmers, the potential impact on labor, and the accessibility of the technology. It is crucial to address these challenges to ensure the successful adoption of ML-based disease detection in agriculture.

3.4 System Specification

3.4.1 Hardware Specification

• CPU - Intel or AMD
• RAM - 4GB or More
• Storage - 30GB or More
• GPU - AMD/Intel/Nvidia

3.4.2 Software Specification

• MATLAB(R2022a)
• Microsoft Visual Studio 2022 version 17.5.4.
• DirectX 12

3.4.3 Standards and Policies

General Data Protection Regulation (GDPR):

The GDPR is a regulation by the European Union that sets standards for data protection and privacy. If the plant and disease detection system involves the collection and storage of personal data, such as images of farmers or their crops.

International Plant Protection Convention (IPPC):

The IPPC is an international treaty that aims to prevent the spread of plant pests and diseases. Any plant and disease detection system that involves the transportation of plant material or data across international borders must comply with IPPC regulations.

Intellectual Property Rights (IPR):

The use of machine learning and image processing may involve the creation and protection of intellectual property rights, such as patents or copyrights. Any system developed using these technologies must comply with IPR regulation.

Good Agricultural Practices (GAP):

GAP are a set of voluntary standards and guidelines for sustainable agriculture. Any plant and disease detection system must comply with GAP regulations to ensure that it does not harm the environment or human health.

Chapter 4

METHODOLOGY

4.1 Architecture Diagram of Disease Detection

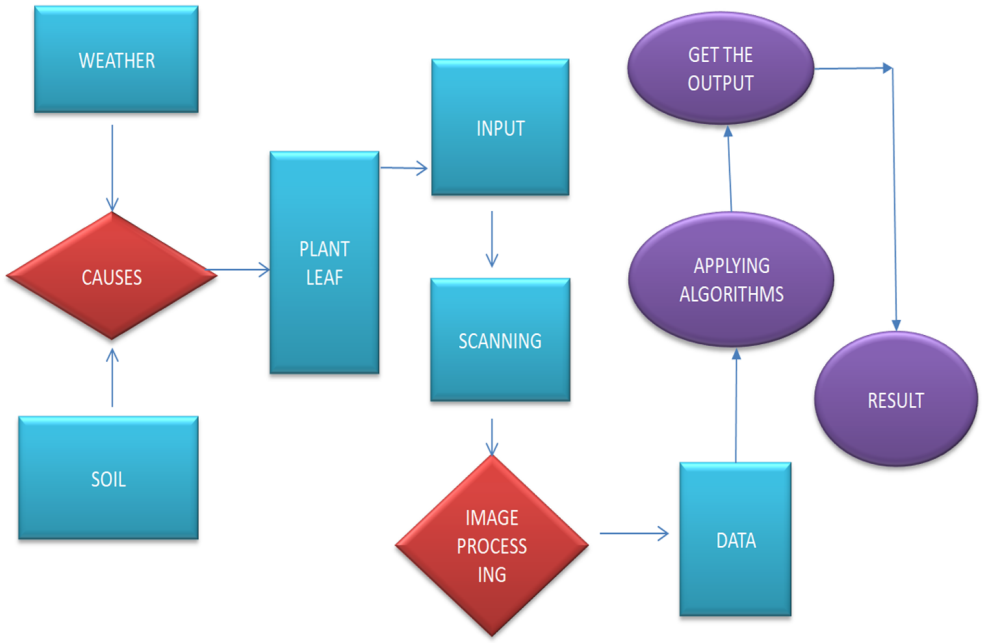

Figure 4.1: **Architecture Diagram of Disease Detection**

Figure 4.1 depicts the overall layout and working procedure of the project. Firstly, take leaf image as input. In image processing the following some steps like image acquistion, image pre-processing, image segmentation, feature extraction, classification. The next step is to extract features from the images. This involves identifying specific characteristics of the plants and their leaves that are relevant to disease detection. For example, features could include the color, texture, and shape of the leaves.Once the model is trained, it can be used to classify new images as healthy or diseased. This is done by feeding the new images into the model, which will output a prediction of whether the plant is healthy or diseased.

4.2 Design Phase

4.2.1 Data Flow Diagram

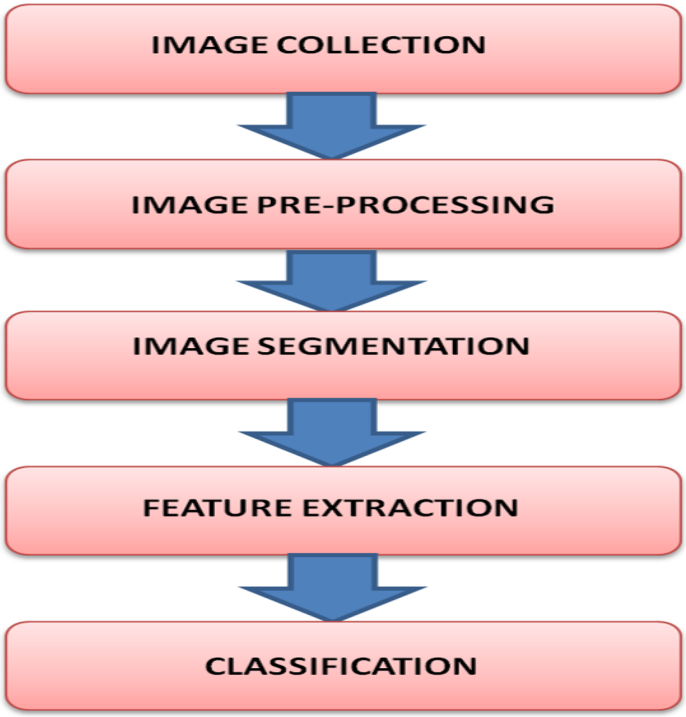

Figure 4.2: **Data flow Diagram**

Figure 4.2 explains about the step by step flow gives an overall idea of how image processing works. It consists of five steps in the flow in which each algorithm is used such as acquisition,pre-processing,segmentation,feature extraction and classification. In image Acquisition, capture the images of leaves using the mobile phone or camera. Collect those images and stored in separate file ,and load the images from specifying the path. In image pre-processing,it helps to improve the quality of images by removing the unsought distortions, image smoothing and contrast enhancement. In image segmentation, divide an image into the different sub images. Here by k-Means Clustering algorithm green color images of the leaves are normal don't consider those images. Highlight only infected part in an image. In feature extraction, GLCM(Gray level co-occurrence matrix) is best method for texture analysis and estimating the image properties. In classification, Leaves are affected by bacteria, fungi and viruses, sometimes insects also damage the leaves. Based on color of the leaf and texture information from GLCM the disease is classified.

4.2.2 Use Case Diagram

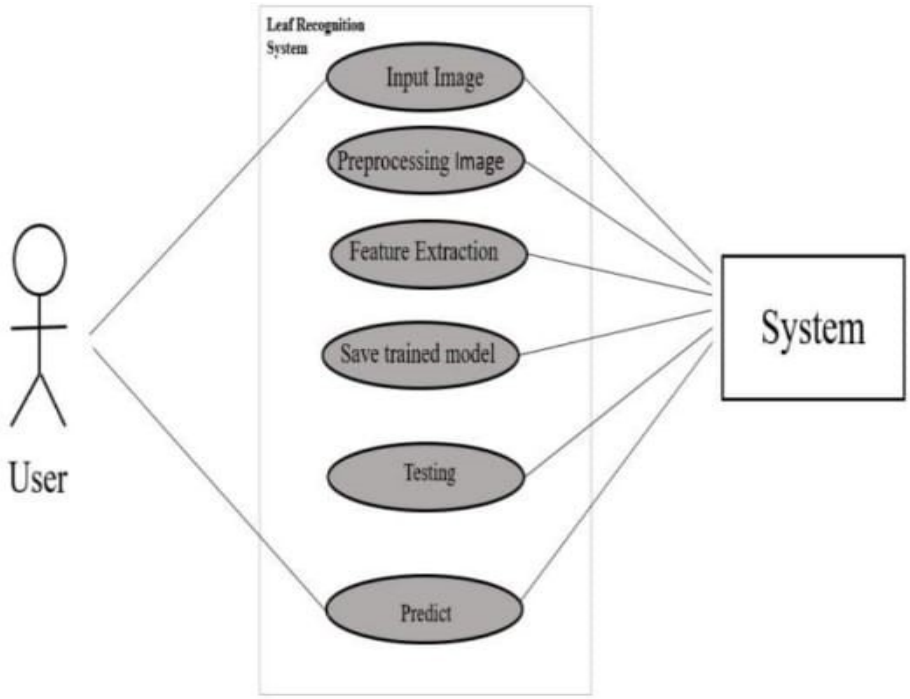

Figure 4.3: **Usecase Diagram**

Figure 4.3 shows the Use Case Diagram serves as a graphical representation showcasing the functional aspects of a system. It delineates the interactions between system users ,adminstrators(third party) and the functionalities provided by the system. Users, external systems, or other elements interacting with the system, are depicted alongside use cases, representing specific functionalities like ”Classify URL” or ”View Classification Results.” This diagram simplifies understanding by outlining the system’s capabilities at a high level without delving into internal complexities

4.2.3 Class Diagram

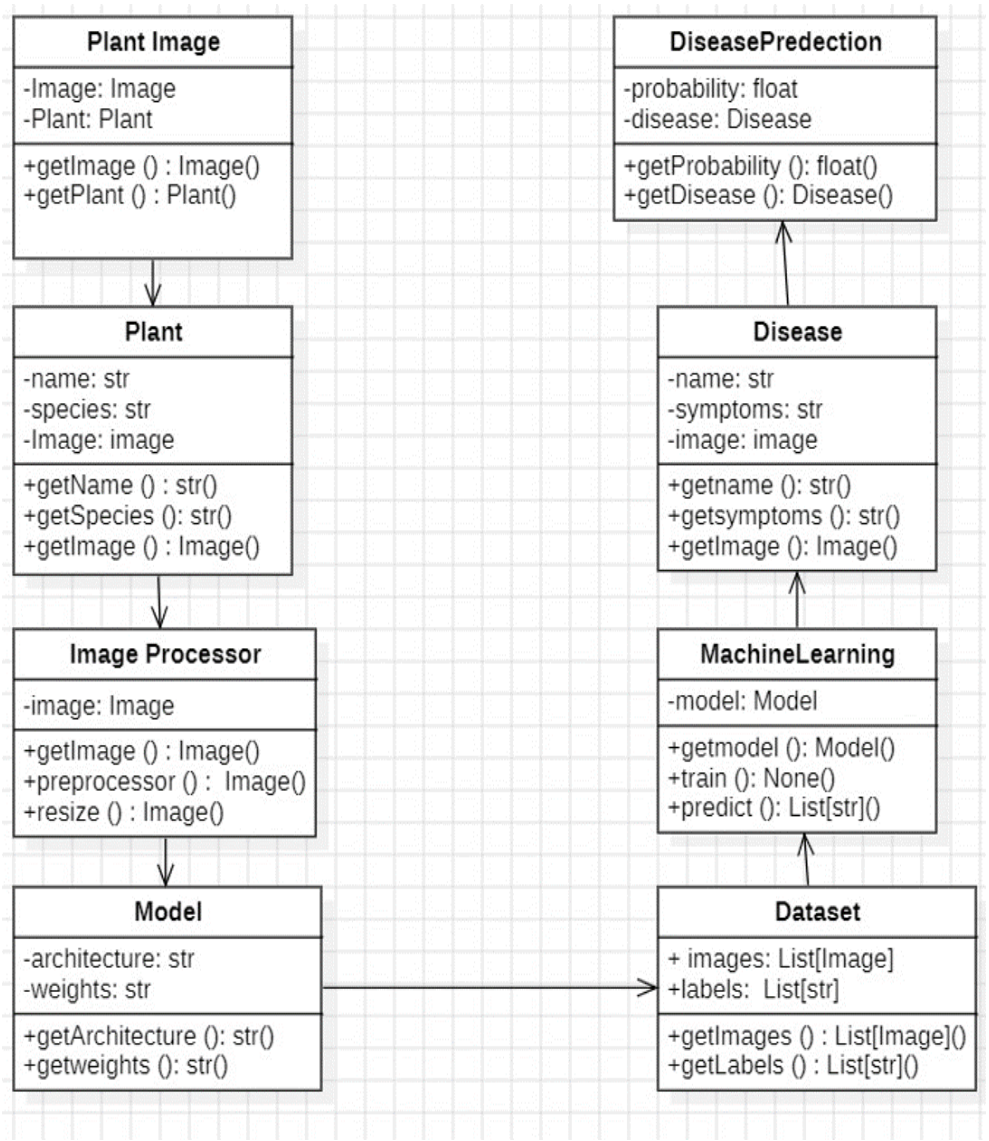

Figure 4.4: **Class Diagram**

Figure 4.4 shows the Class Diagram provides a static view of the system, outlining the structure through classes, their attributes, methods, and their relationships. It offers a blueprint of the system's architecture by detailing the classes and their associations, such as inheritance, associations, and attributes. This diagram aids in understanding the system's design by highlighting the static structure of classes and their interactions but does not portray the dynamic behavior of the system.

4.2.4 Sequence Diagram

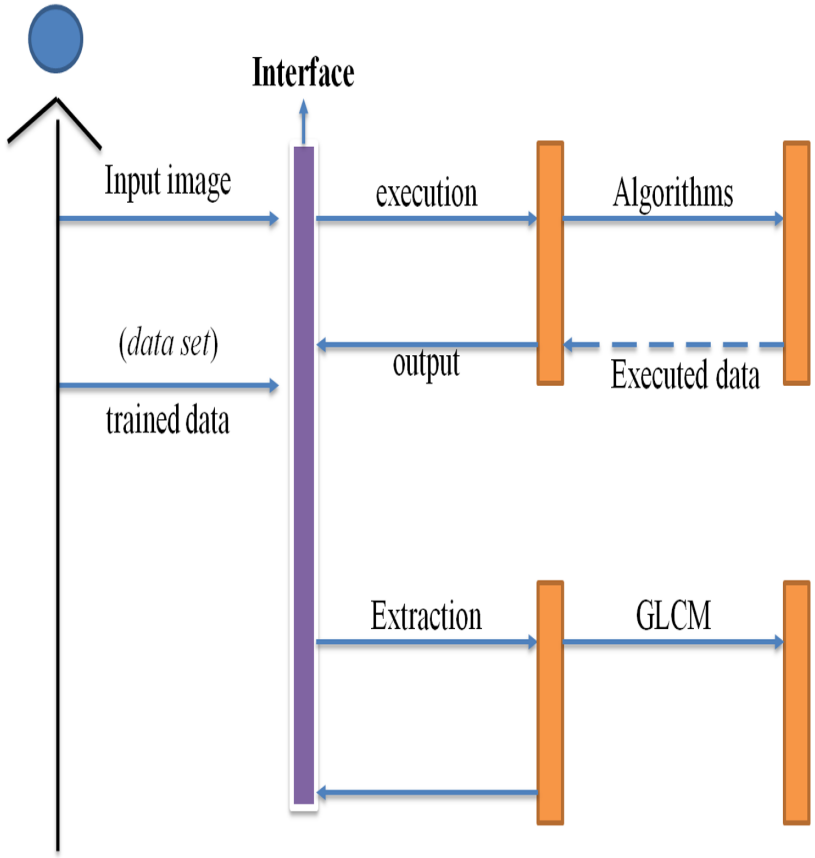

Figure 4.5: **Sequence Diagram**

Figure 4.5 is a sequence diagram because it describes how and in what order a group of objects works together and the status of the work flow while the process continues.The input of image is taken into the system and the the interface executes the process and in parallel ,in the back end the algorithm starts clustering and we have to select the required set and will matched to training data and then output will be displayed.

4.2.5 Activity Diagram

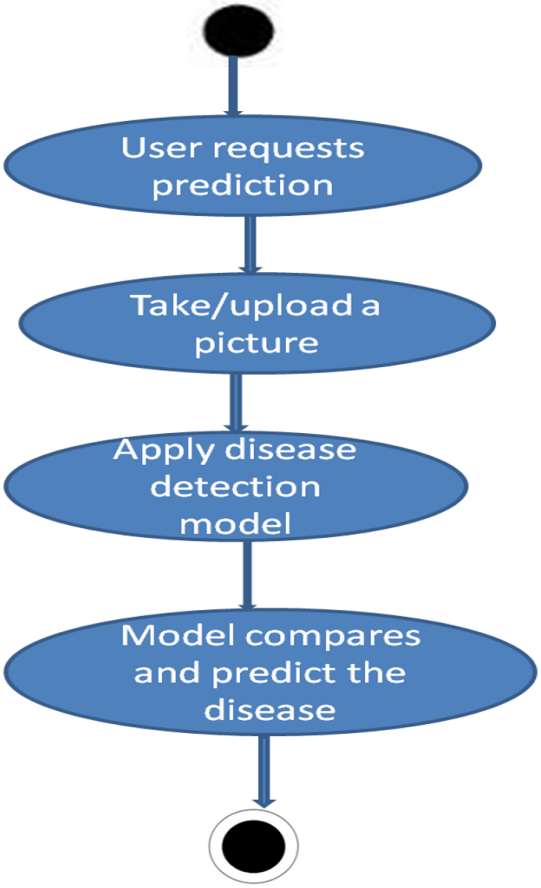

Figure 4.6: **Activity Diagram**

Figure 4.6 is a activity diagram as it describes the activity of user. Firstly, user takes a picture of a leaf and upload it an interface. Now the disease detection model will enhance the image and the improves the quality of image as it should detect the leaf as healthy or diseased. And also it will show the percentage of the affected area to the user.

4.3 Algorithm & Pseudo Code

4.3.1 Image Classification Algorithm

Step 1:Load the image from Collected a dataset of plant images, including both healthy and diseased plants. Ensure that each image is labeled correctly according to its health status.

Step 2: Enhance the constrant that is preprocessing the images to enhance the

quality and remove any noise by using appropriate image processing techinique like Grey Level Co-occurence matrix algorithm.
Step 3: Apply the K-means algorithm to cluster the feature vectors obtained from the previous step. K-means is an unsupervised learning algorithm that groups similar data points into clusters based on their feature similarities.in image segmentation.
Step 4: Once the clusters are formed, they can be used to classify new images of plants as either healthy or diseased.
Step 5:Classify the result where the disease of the image and the amount of disease affected will be displayed .
Step 6: Then filnally the accuray level is caluclated.

4.3.2 Pseudo Code

```
1
 # Define a function to capture or load an image of the plant
  def capture image ( ) :
     # Return the captured or loaded image
# Define a function to preprocess  the image
  def preprocess image( image ) :
     # Return the preprocessed image
     # Define a function to detect plant diseases
  def detect disease( image ) :
# to detect plant diseases in the image
# Return the detected disease or None if no disease is found
# Main program
  def main( ) :
# Capture or load the image
  image = capture image( )
# Preprocess the image
  preprocessed image = preprocess image(image)
# Detect plant diseases
  disease = detect disease (preprocessed image)
# Display the result
  if disease is not None :
  print ( Plant  disease detected:  ,disease)
  else :
  print ( No  disease detected.  )
# Runthe main program
  main ( )
 }
```

4.4 Module Description

4.4.1 Data Collection and Training

1. **Data Collection**: The first step in plant and plant disease detection is to collect images of plants and their leaves it can be shown in the Figure 4.7. This can be done using cameras mounted on drones or other devices, or by taking photos manually.
2. **Preprocessing**: Once the images are collected, they need to be preprocessed to prepare them for analysis this process is shown in the Figure 4.8. This may involve adjusting the lighting and color balance, as well as cropping and resizing the images.
3. **Feature Extraction**: The next step is to extract features from the images. This involves identifying specific characteristics of the plants and their leaves that are relevant to disease detection. For example, features could include the color, texture, and shape of the leaves.
4. **Machine Learning Model**: In Figure 4.9 a machine learning model is then trained using the extracted features. The model is trained on a large dataset of labeled images, with each image labeled as either healthy or diseased.
5. **Disease Detection**: Once the model is trained, it can be used to classify new images as healthy or diseased. This is done by feeding the new images into the model, which will output a prediction of whether the plant is healthy or diseased.
6. **User Interface**: Finally, a user interface is created to allow farmers and other users to interact with the system. This could be a web.

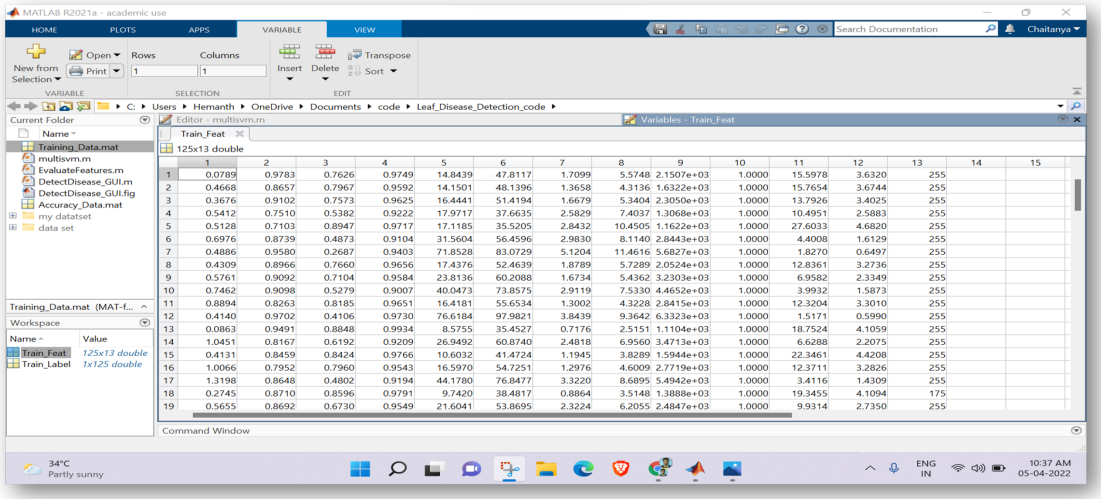

Figure 4.7: **Collection of data**

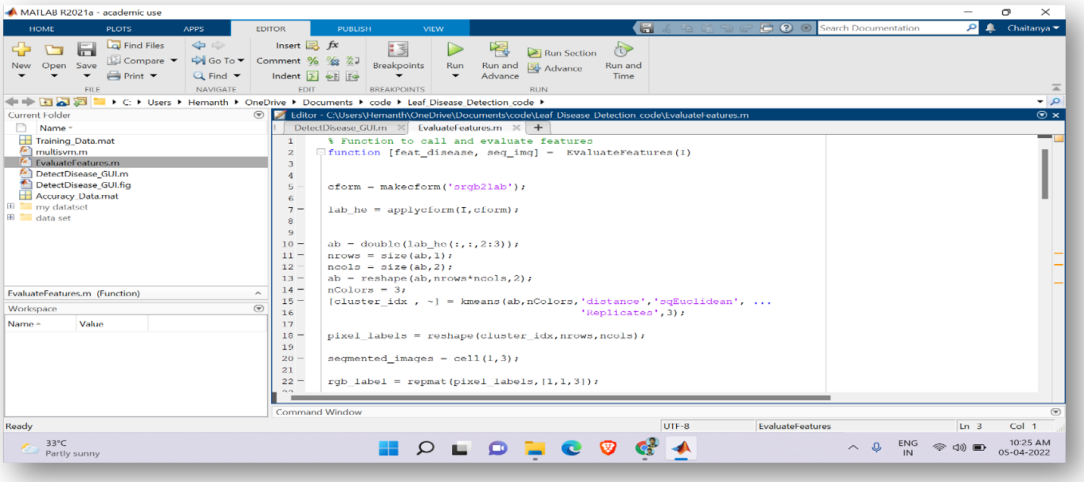

Figure 4.8: **Processing the data**

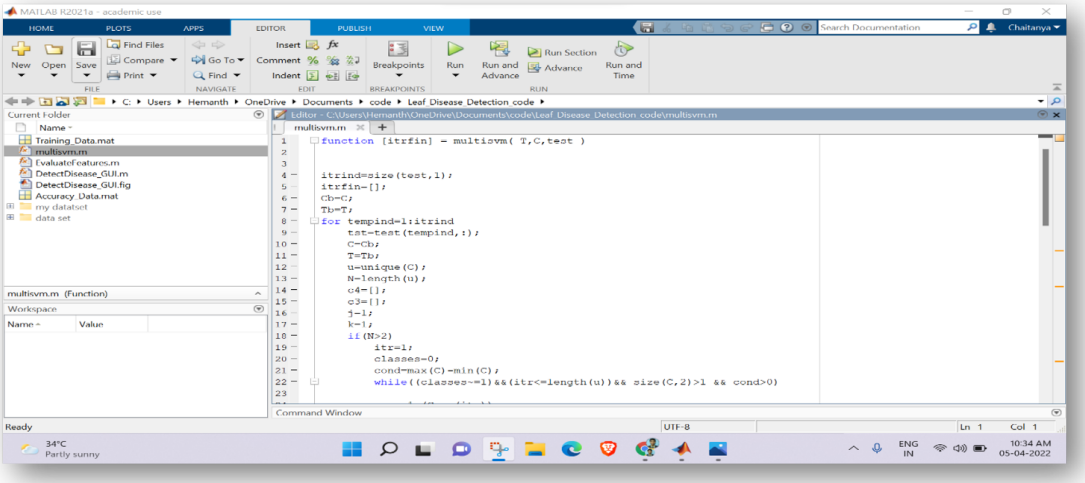

Figure 4.9: **Applying machine learning algorithms**

4.4.2 Graphical User Interface for the Application

In the below Figure 4.10 it shows the GUI Interface and develop the below points.

1. **Image Upload**: The GUI should have an option to upload images of plants or plant leaves for analysis. This can be done through a drag-and-drop interface or a file upload button.
2. **Image Preprocessing**: The GUI can include options for image preprocessing, such as adjusting brightness, contrast, or saturation. This can be done using sliders or other interactive controls.
3. **Image Analysis**: The GUI should provide a clear and simple display of the analysis results. This could include a label indicating whether the plant is healthy or diseased, as well as a probability score or confidence level.
4. **Visualization**: The GUI can include visualization tools to help users understand the results of the analysis. This could include heat maps, histograms, or other types of charts.
5. **Feedback Mechanism**: The GUI should provide a way for users to provide feedback on the analysis results. For example, if a plant is misclassified as diseased, the user should be able to indicate this in the GUI so that the machine learning model can be improved.
6. **Database**: The GUI can include a database to store the analysis results, along with metadata such as the date of analysis and the location of the plant.
7. **User Management**: The GUI can provide different levels of access to users, depending on their role and level of expertise. For example, farmers or gardeners may have a simpler interface with limited options, while researchers or experts may have access to more advanced features.

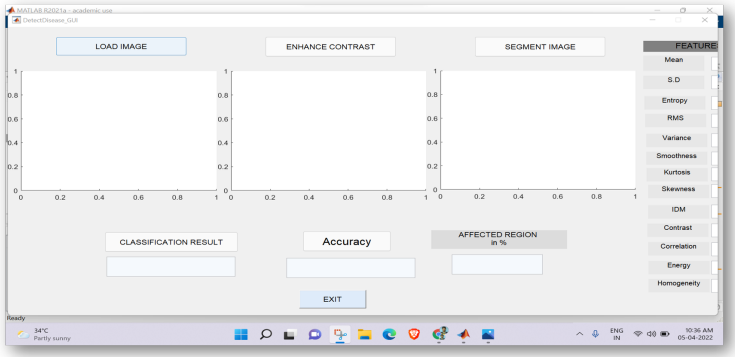

Figure 4.10: **GUI Interface**

4.4.3 Clustering and Classification

GLCM Algorithm

The Feature Extraction is a method of capturing visual content of images for indexing retrieval. Grey Level Co-occurrence Matrix (GLCM) method is a way of extracting second order statistical texture features it is shown in the Figure 4.11. The GLCM functions characterize the texture of an image by calculating with specific values and in a specified spatial relationship occur in an image, creating a GLCM, and then extracting statistical measures from this matrix. The image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.

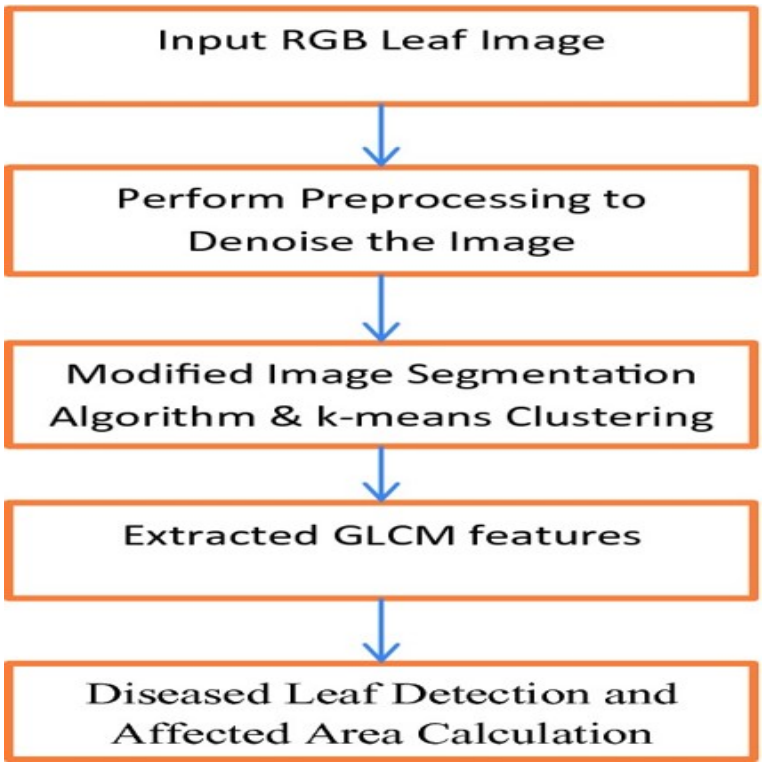

Figure 4.11: **GLCM**

K Means clustering

The k-means algorithm is a clustering algorithm that can be used for plant disease detection. The algorithm works by grouping similar data points into clusters, with each cluster being represented by its centroid it is represented in the Figure 4.12. Initialize the system and input data which involves setting up the detection system

and capturing images of the plants. K-means clustering is the extracted features are then used as input to the k-means algorithm. The algorithm groups the features into a specified number of clusters (k), with each cluster being represented by its centroid.

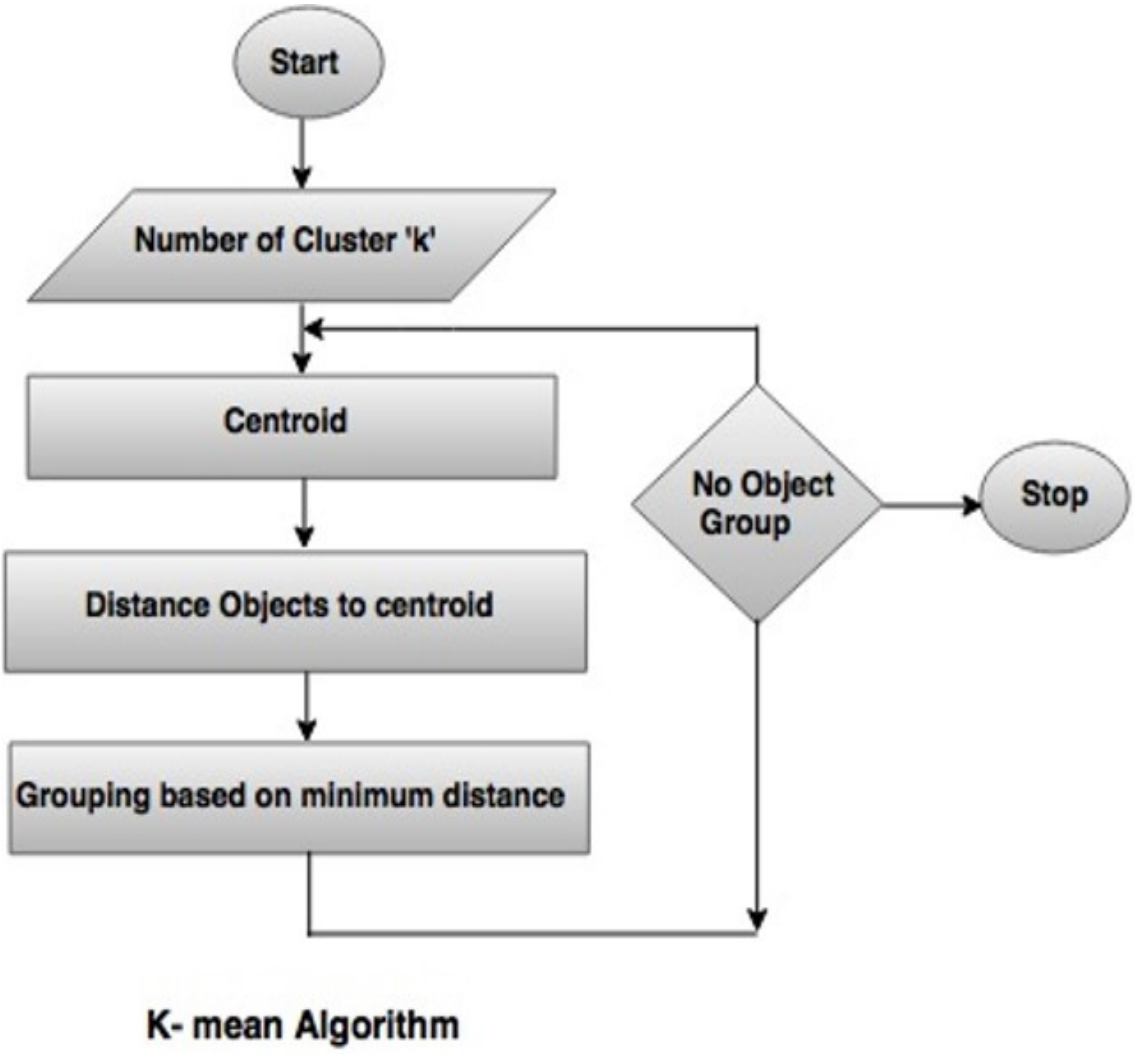

Figure 4.12: **K-Means Algorithm**

Feature extraction:

The next step is to extract relevant features from the images. This could involve techniques such as color-based feature extraction or texture analysis.Classify images based on cluster labels, Once the clustering is complete, each image can be classified based on the cluster it belongs to. If the cluster contains images of healthy plants, then any new image that belongs to that cluster can be classified as healthy. Conversely, if the cluster contains images of diseased plants, then any new image that belongs to that cluster can be classified as diseased.

4.5 Steps to execute/run/implement the project

4.5.1 Step1:Data Collection

• **Identify the Plant and Disease**: The first step is to identify the plant and disease that need to be detected. This will help in determining the type of data that needs to be collected.
• **Collect Images**: Images of both healthy and diseased plants need to be collected using various methods such as cameras, drones, or satellites. It is important to ensure that the images are of high quality, and the lighting conditions are consistent.
• **Labeling Images**: Each image needs to be labeled with information such as the plant species, the disease present (if any), and the severity of the disease. This labeling process helps in training the machine learning model.
• **Preprocessing Images**: The collected images need to be preprocessed to remove noise and other unwanted artifacts. This can be achieved through techniques such as normalization, contrast enhancement, and image filtering.
• **Splitting Data**: The collected dataset needs to be split into two parts, i.e., training data and validation data. The training data is used to train the machine learning model, while the validation data is used to evaluate the performance of the model.
• **Feature Extraction**: Features need to be extracted from the preprocessed images. This can be done through various methods such as color-based, texture-based, or shape-based feature extraction.

4.5.2 Step2:Model Selection

• **Identify the Problem**: The first step is to identify the problem to be solved. It is essential to understand the requirements and constraints of the problem.
• **Gather Data**: Collect a dataset of images of plants, both healthy and diseased. These images can be collected using various methods such as camera, drones, or satellites.
• **Preprocess the Data**: Preprocess the collected images to remove noise and other unwanted artifacts. This can be achieved through techniques such as normalization, contrast enhancement, and image filtering.
• **Split Data**: Split the dataset into two parts, i.e., training data and validation data. The training data is used to train the machine learning model, while the validation data is used to evaluate the performance of the model.

• **Compare Performance**: Compare the performance of different machine learning models and select the one that performs the best on the validation dataset.
• **Fine-tune the Model**: Fine-tune the selected model by adjusting hyperparameters and modifying the architecture to further improve the performance.

4.5.3 Step3:Model Evaluation and deployment

• **Evaluate Model Performance**: Evaluate the performance of the machine learning model using various metrics such as accuracy, sensitivity, specificity, and F1 score. This will help to identify the strengths and weaknesses of the model.
• **Improve the Model**: Fine-tune the model by adjusting hyperparameters and modifying the architecture to further improve the performance. This can be done using techniques such as cross-validation and grid search.
• **Test the Model**: Test the model on a new set of data to evaluate its generalization ability. This will help to ensure that the model can accurately detect plant diseases in real-world situations.
• **Deploy the Model**: Deploy the trained model to a production environment. This can be done using various methods such as cloud-based deployment or on-premises deployment.
• **Monitor Model Performance**: Monitor the performance of the deployed model and collect feedback from end-users. This will help to identify any issues with the model and improve its performance.
• **Update the Model**: Update the model periodically using new data and feedback from end-users. This will help to ensure that the model remains accurate and up-to-date.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

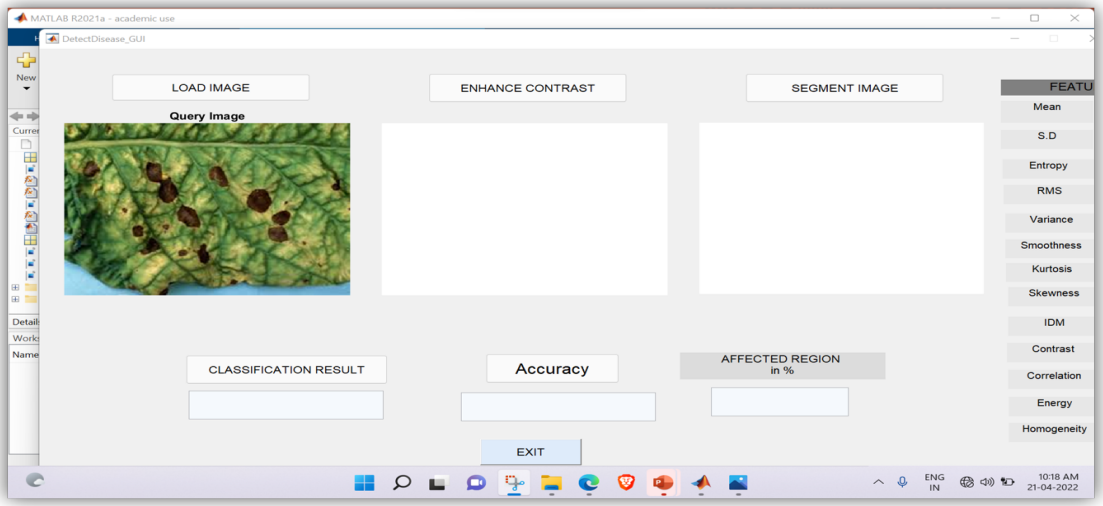

Figure 5.1: **Input for loading leaf image**

Image is captured through a digital camera which is shown in Figure 5.1 and inserted in input images, with help of image processing we can classify to different stages like segmentation, acquisition, feature extraction etc. Mat lab software is used to get the following output in required format so it will identify accordingly by comparing previous data. The GLCM functions characterize the texture of an image by calculating with specific values and in a specified spatial relationship occur in an image, creatinga GLCM, and then extracting statistical measures from this matrix.

5.1.2 Output Design

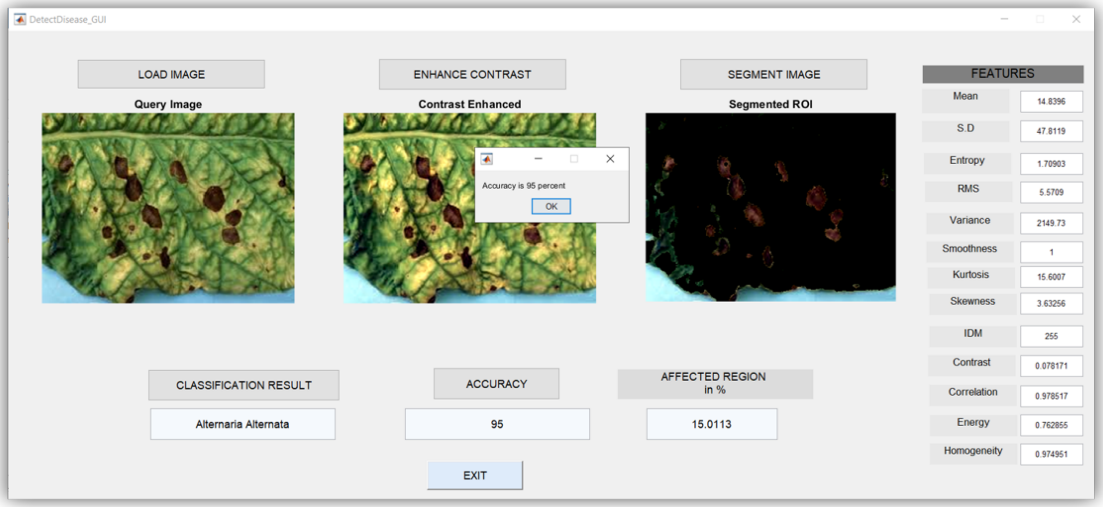

Figure 5.2: **Output of disease affected region**

Figure 5.2 explains about the area of affected region in the leaf and detected the accuracy level by using K Means clustering along with GLCM algorithms.

5.2 Testing

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

5.3 Types of Testing

5.3.1 Unit testing

For plant and plant disease detection, unit testing may involve testing different components of the system separately, such as image preprocessing, feature extraction, and machine learning model training. This can help to identify issues such as inaccurate feature extraction or overfitting of the model to the training data. Unit

testing can also help to ensure that the system is robust and can handle different types of plant species, disease symptoms, and environmental conditions.

Input

```
nrows = size(ab,1);
ncols = size(ab,2);
ab = reshape(ab,nrows*ncols,2);
nColors = 3;
[cluster_idx cluster_center] = kmeans(ab,nColors, distance , sqEuclidean ,..
%[cluster_idx cluster_center] = kmeans(ab,nColors, distance , sqEuclidean , Replicates
    ,3);
% Label every pixel in the image using results from K means
pixel_labels = reshape(cluster_idx,nrows,ncols);
%figure,imshow(pixel_labels,[]),title( Image Labeled by Cluster Index );

%Create a blank cell array to store the results of clustering
segmented_images = cell(1,3);
%Create RGB label using pixel_labels
rgb_label = repmat(pixel labels,[1,1,3]);
```

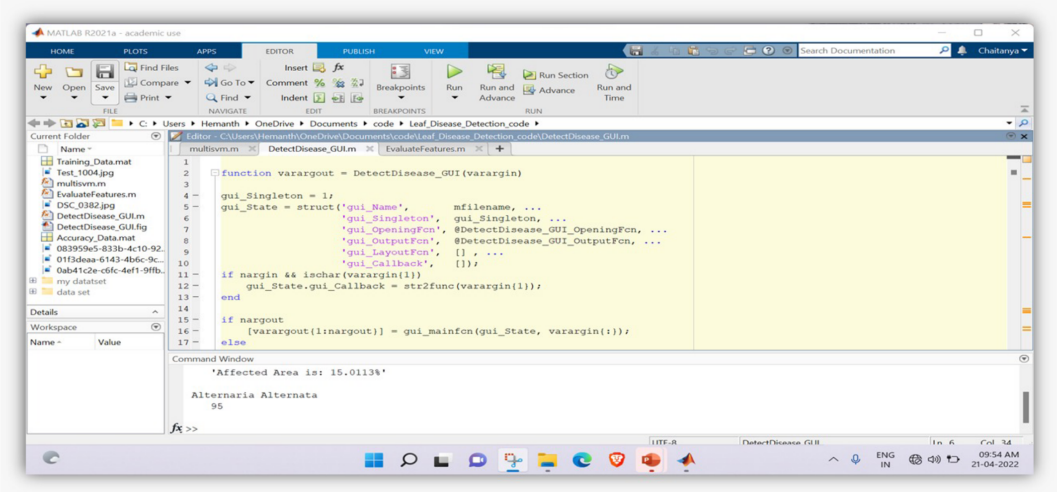

Figure 5.3: **Test result of Unit Testing**

Test result : PASSED

The above Figure 5.3 shows the test result for the unit testing.

5.3.2 Integration testing

For plant and plant disease detection, integration testing may involve testing the entire system end-to-end, from image preprocessing to machine learning model prediction. This can help to identify issues such as data flow errors or inconsistencies in the system's output. Integration testing can also help to ensure that the system is able to handle large volumes of data and produce results in a timely manner.

Input

```
function[ i trfin]=multisvm(T,C,test)
%Inputs:T=Training Matrix,C=Group,test=Testing matrix
%Outputs:itrfin =Result antclass
itrind = size (test,1);
itrfin = [ ];
Cb=C;
Tb=T;
for tempind=1: itrind
tst = test(tempind,:);
C =Cb;
T =Tb;
u =unique(C);
N =length(u);
c4 =[ ];
c3 =[ ];
j =1;
k =1;
if (N>2)
itr =1;
classes =0;
cond =max(C) min (C);
while((classes=1)&&(itr<=length(u))&&size(C,2)>1&&cond>0)
%This while loop is the multi-class SVM Trick
c1=(C==u(itr));
newClass=c1;
%svmStruct = svm train ( T,newClass, kernel function , rbf );% I am using rbf kernel
function,you must change it also
svm Struct =svm train(T,newClass);
classes = svm classify(svmStruct,tst);
% This is the loop for Reduction of Training Set
for i=1:size(newClass,2)
if newClass(1,i)==0;
 c3 ( k , : ) =T ( i , : ) ;
k=k +1;
 end
 end
 T=c3 ;
```

```
c3 = [ ] ;
 k =1;
 % This is the loop for reductionof group
 for i=1:size(newClass,2)
```

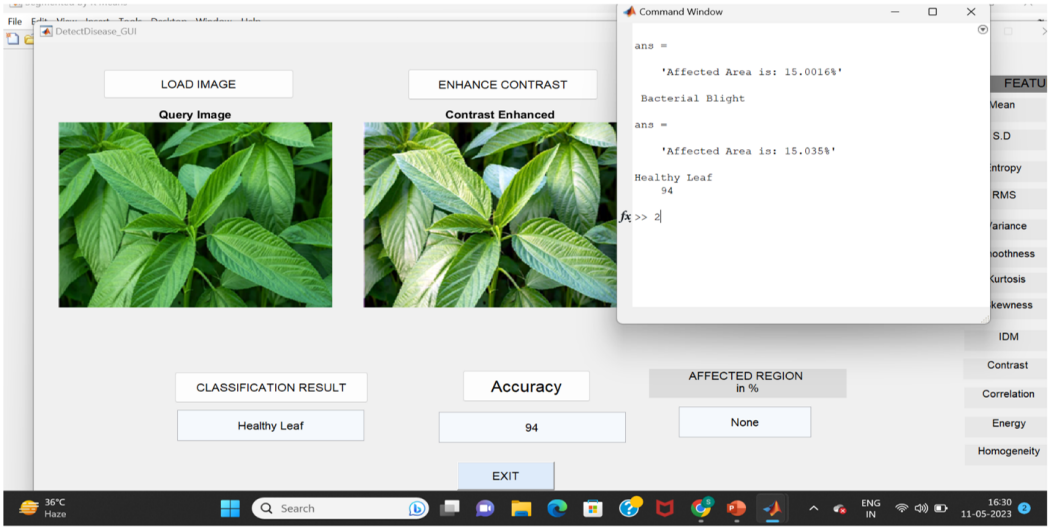

Figure 5.4: **Integration testing result**

Test result : PASSED

The above Figure 5.4 shows the test result for the integratio testing.

5.3.3 System testing

For plant and plant disease detection, system testing may involve testing the system with a variety of images, including those with different plant species, disease symptoms, and environmental conditions. This can help to identify issues such as false positives or false negatives, as well as performance issues such as slow processing times or memory usage. System testing can also help to ensure that the system is user-friendly and meets the needs of its intended users, such as farmers, gardeners, or researchers

Input

```
clc
closeall
```

```
clearall
[filename,pathname] = uigetfile({ *.* ; *.bmp ; *.jpg ; *.gif }, Pick a Leaf
    ImageFile );
I = imread([pathname,filename]);
I = imresize(I,[256,256]);
%figure,imshow(I);title( Query Leaf Image );
%Enhance Contrast
I = imadjust(I,stretchlim(I));
figure,imshow(I);title( Contrast Enhanced );
%Otsu Segmentation
I_Otsu = im2bw(I,graythresh(I));
%Conversion to HIS
I_HIS = rgb2hsi(I);
%%Extract Features
%Functioncall to evaluate features
%[feat_disease seg_img] = EvaluateFeatures(I)
%Color Image Segmentation
%Use of K Means clustering for segmentation
%Convert Image from RGB Color Space to L*a*b Color Space
%The L*a*b* space consists of a luminosity layer L *, chromaticity layer a * and b *
    .
%All of the color information is in the a * and b * layers.
cform = makecform( srgb2lab );
% Apply the color form
lab_he = applycform(I,cform);
%Classify the colors in a*b* color space using K means clustering.
%Since the image has 3 colors create 3 clusters.
%Measure the distance using Euclidean Distance Metric.
ab = double(lab_he(:,:,2:3));
nrows = size(ab,1);
ncols = size(ab,2);
ab = reshape(ab,nrows*ncols,2);
nColors = 3;
[cluster_idx cluster_center] = kmeans(ab,nColors, distance , sqEuclidean ,...
        Replicates ,3);
%[cluster_idx cluster_center] = kmeans(ab,nColors, distance , sqEuclidean , Replcates
    ,3);
%Label every pixel in the image using results from K means
pixel_labels = reshape(clusteridx,nrows,ncols);
%figure ,imshow(pixel_labels,[]),title( Image Labeled by Cluster Index );
%Create a blank cell array to store the results of clustering
segmented_images=cell(1,3);
```

Test Result : PASSED

5.3.4 Test Result

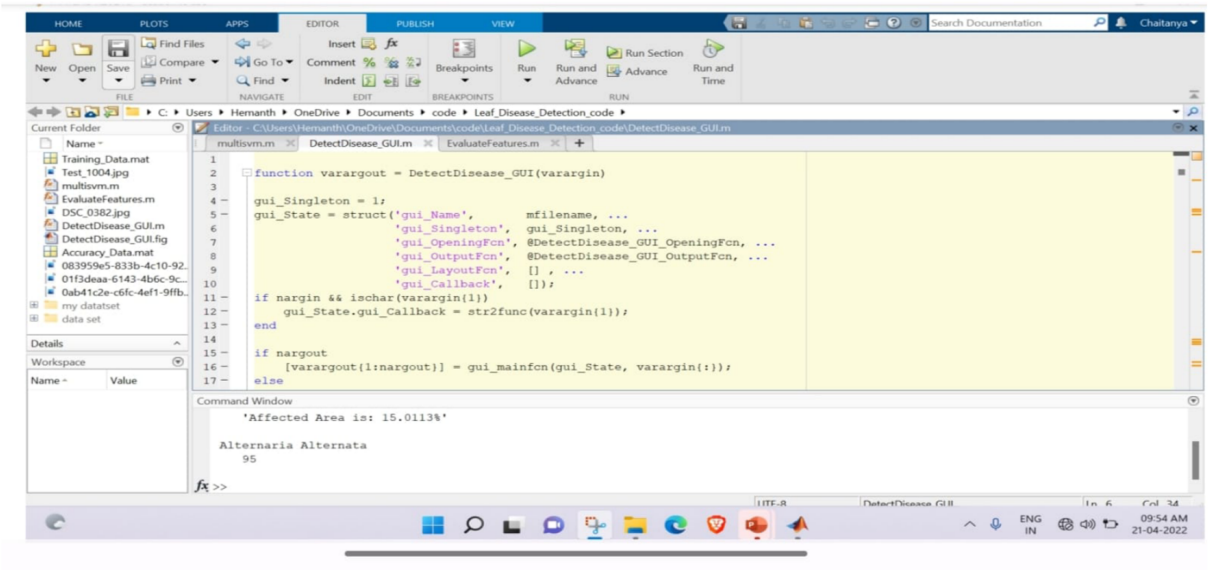

Figure 5.5: **Test Image**

The accuracy of the system which is shown in Figure 5.5 is crucial in preventing losses in the yield and quantity of agricultural products. The test result shows that the proposed system is a promising and effective technique for plant and plant disease detection using machine learning and image processing.

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

It take less time to detect the disease and providing the how much area is effected by using K-means and GLCM algorithms. It is easy to use by a single person and one person can able to handle and it is user friendly environment. The algorithms are efficient and it is cost efficient. A well-designed and implemented system for plant and plant disease detection using machine learning and image processing by K means clustering and GLCM algorithms can provide significant benefits in terms of efficiency and accuracy, making it a valuable tool for farmers, researchers, and other stakeholders in the agriculture industry.

Plant disease detection using GLCM and k-means clustering is an efficient approach to automatically classify plant images into healthy or diseased categories. GLCM (Gray-Level Co-occurrence Matrix) is a texture analysis method that extracts second order statistical features from an image, which can help to distinguish between different types of textures in the image. K-means clustering is a machine learning technique that can group similar images based on extracted features.

The efficiency of the GLCM and k-means clustering approach to plant disease detection lies in its ability to accurately distinguish between healthy and diseased plant images based on texture features. The GLCM method can capture the subtle differences in texture patterns between healthy and diseased plants, while k-means clustering can group similar images together and separate them from dissimilar ones. This approach can be automated and can process a large number of images quickly and accurately, making it an efficient tool for plant disease detection.

6.2 Comparison of Existing and Proposed System

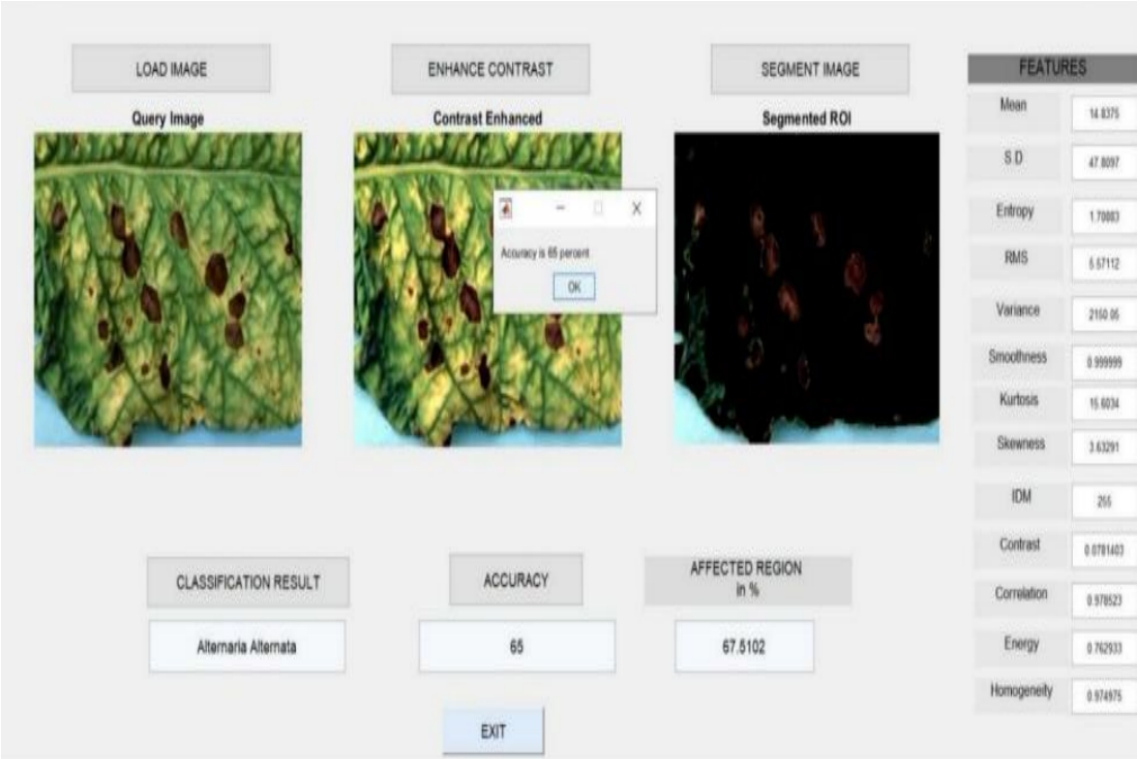

Figure 6.1: **SFTA accuracy of affected area of diseased leaf.**

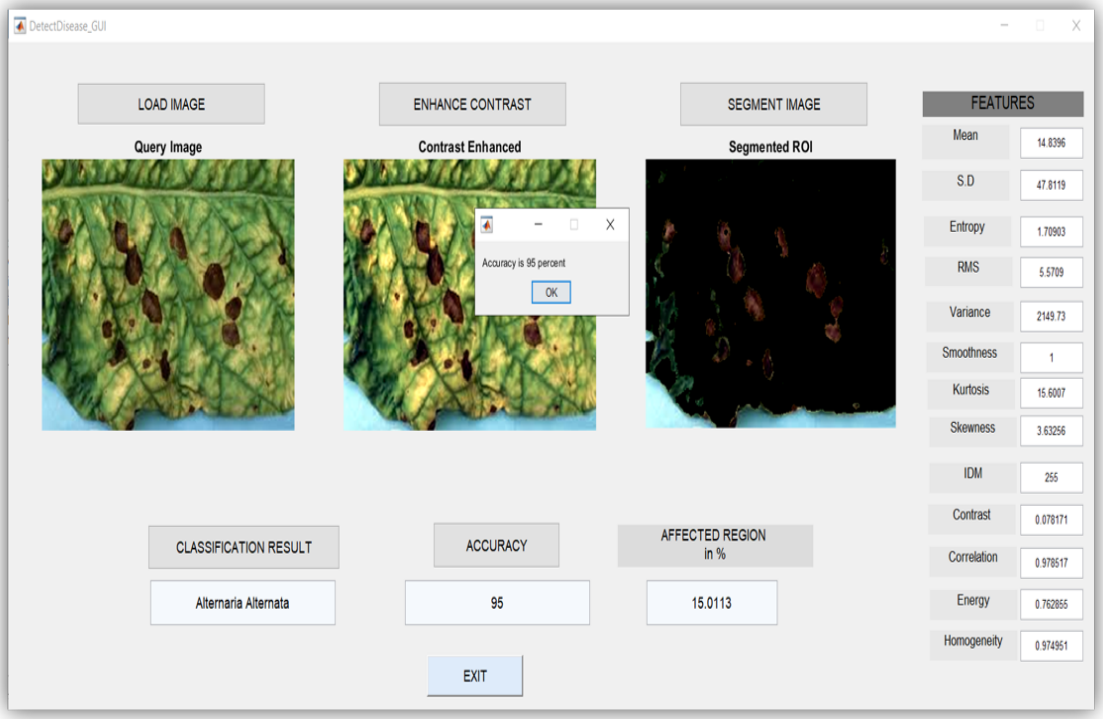

Figure 6.2: **GLCM accuracy of affected area of diseased leaf**

Figure 6.1 and figure 6.2 have taken SFTA as the alternative algorithm to test the feature evaluation process and get the data as accurate as possible. Segmentation based Fractal Texture Analysis (SFTA) is a algorithm for feature extraction and is implemented along side with the other algorithms of image processing and clustering. For this we are getting a accuracy percentage of around 70about to get around 90type of image and the features to be extracted. For example, SFTA algorithm may be more suitable for images with high frequency textures such as leaves with distinct vein patterns, while GLCM algorithm may be more suitable for images with low frequency textures such as plant surfaces with uniform color and texture.

6.3 Sample Code

```
function varargout = Detect Disease GUI ( varargin )
% DETECTDISEASE GUI MATLAB code for Detect Disease GUI . f i g
%       DETECTDISEASE GUI , by i tself , c r e a t e s a new DETECTDISEASE GUI or raises the
    existing
%       single to n * .
%
%       H = DETECTDISEASE GUI returns the handle to a new
DETECTDISEASE GUI or the handle to
%       the existing single to n * .
%
%       DETECTDISEASE GUI (    CALLBACK    , hObject , eventData
        , handles , . . . ) calls the local
% function named CALLBACK in DETECTDISEASE GUI .M with the given input arguments .
%
% DETECTDISEASE GUI (     Property     ,     Value     , . . . ) creates a new DETECTDISEASE GUI or
     raises the
% existing single to n * . Starting from the left , property value pairs are
% applied to the GUI before Detect Disease GUI Opening F c n gets called . An
% unrecognized property name or invalid value makes property application
% stop . All inputs are passed to Detect Disease GUI Opening F c n v i a v a r a r g i n .
%

% See also : GUIDE , GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help Detect Disease G U I
% L a s t Modified by GUIDE v2 . 5 26    Aug2015    17:06:52
% Begin nitialization code     DO NOT EDIT
gui Single to n = 1 ;
gui State = struct(     gui Name     , mfilename , . . .
    gui Single to n     , gui Single to n , . . .
    gui OpeningFcn     , @DetectDisease GUI OpeningFcn , . . .
    gui Output F c n     , @DetectDisease GUI OutputFcn , . . .
    gui Layout F c n     , [ ] , . . .
    gui Call back     , [ ] ) ;
```

```
if nargin && is char ( varargin {1})
gui State . gui Call back = str2func (varagin {1}) ;
end
```

Output

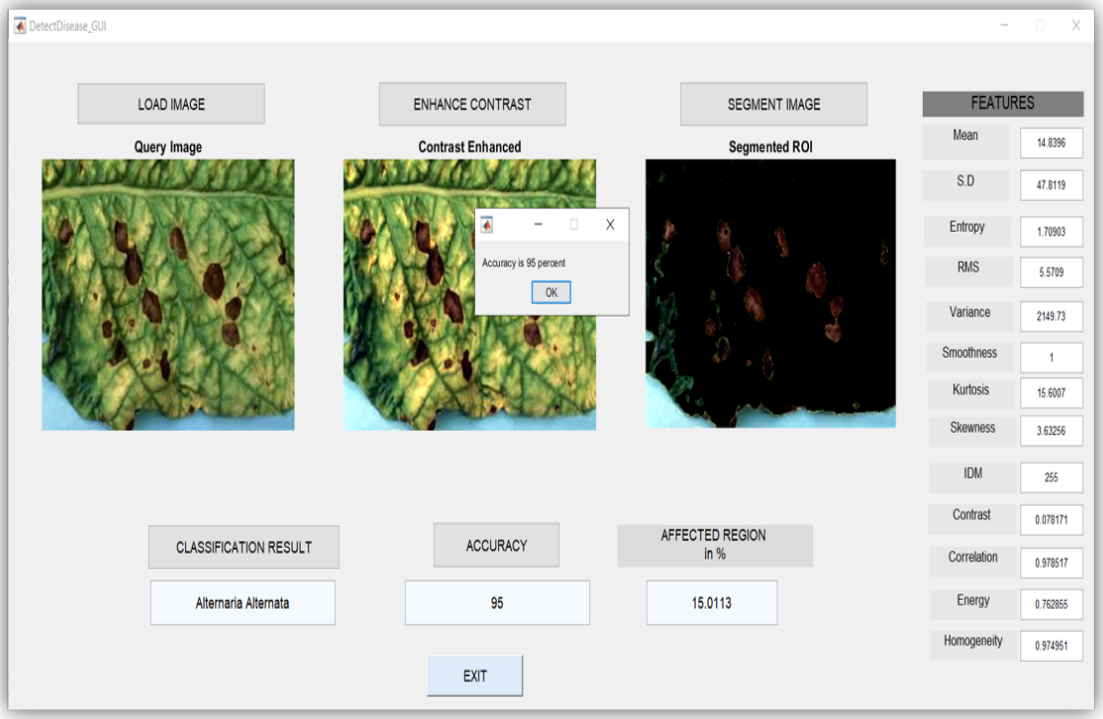

Figure 6.3: **Output of affected region using GLCM,KNN**

Figure 6.3 explains about the area of affected region in the leaf and detected the accuracy level by using K Means clustering along with GLCM algorithms,The accuracy outcomes in an obtainable range from 95.44 to 96.65results obtained KNN used for the disease detection.

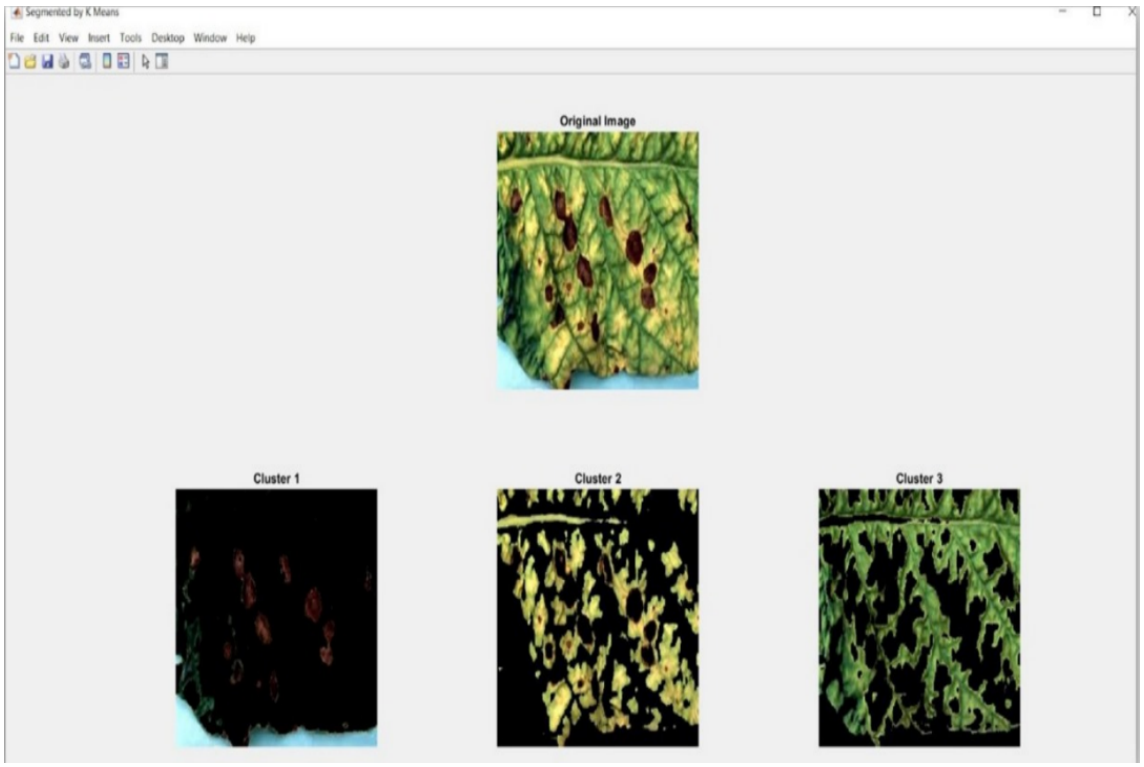

Figure 6.4: **Output ofLeaf clusters using K-means algorithm**

Figure 6.4 explains the output of the k-means algorithm for plant disease detection is a set of clusters, with each cluster containing images that are similar to each other in terms of the extracted features. The centroid of each cluster represents the average feature vector of all the images in the cluster.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

This project implements an innovative idea to identify the affected plant and provide remedy measures to the agricultural industry. The results obtained found to be more appropriate to detect and diagnose the diseases. Plant and plant disease detection using machine learning and image processing is a promising technology with the potential to revolutionize the agricultural sector. By leveraging machine learning algorithms and image processing techniques such as K means Algorithm and GLCM Algorithms, this technology can help farmers and agricultural researchers detect and diagnose plant diseases accurately and quickly, enabling them to take timely action to prevent crop losses and ensure food security. The successful adoption of this technology requires adherence to relevant policies and standards related to data privacy, ethical use, accuracy, and safety. Additionally, continuous monitoring and improvement of the machine learning models are essential to optimize their accuracy and reliability.

7.2 Future Enhancements

In the future, the proposed methodology can be integrated with other yet to be developed, methods for disease identification and classification using color and texture analysis to develop an expert system for early soya plant foliar disease warning and administration, where the disease type can be identified by color and texture analysis and the severity level estimation by our proposed method since it is disease independent. The performance of the system can be improved in the future by using 30 advanced background separation methods to separate the leaf object from a complex background.Disease stage identification is one of the main areas to be explored re-

garding plant disease identification. Each disease has several stages. Most of the researchers only focused their work on the type of disease identification, but none of these works target a particular disease stage identification. Additionally, such systems must possess the capability to suggest some measures depending particularly on specific disease stages. Identification of disease forecasting will help agriculturists to take proper actions and precautions to reduce damage percentage.

Chapter 8

PLAGIARISM REPORT

Plagiarism Scan Report

Report Generated on: Apr 24,2024

0%	100%	Total Words: 126
Plagiarised	Unique	Total Characters: 894
		Plagiarized Sentences: 0
		Unique Sentences: 5 (100%)

Content Checked for Plagiarism

CERTIFICATE
"PLANT DISEASES DETECTION
USING MACHINE LEARNING " by "D.TRISHA (21UECM0063), P.CHAITANYA (21UECM0303),
K.VENKATA ABHI SRIRAM (21UECM0126)" has been carried out under my supervision.
Signature of Supervisor
Mr.C.Edwin Singh
Assistant Professor
Computer Science & Engineering
School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology
January, 2024
Dr.M.S.Muralidhar Dr. V. Srinivasa Rao
Associate Professor & Head Professor & Dean
Computer Science & Engineering Computer Science & Engineering
School of Computing School of Computing
Vel Tech Rangarajan Dr. Sagunthala R&D Vel Tech Rangarajan Dr. Sagunthala R&D
Institute of Science & Technology Institute of Science & Technology
January, 2024 January, 2024
DECLARATION
We declare that this written submission represents ides
D.TRISHA
Date: / /
P.CHAITANYA
Date: / /
K.VENKATA ABHI SRIRAM
Date: / /

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

```
function varargout = DetectDisease_GUI(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @DetectDisease_GUI_OpeningFcn, ...
                   'gui_OutputFcn',  @DetectDisease_GUI_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end

function DetectDisease_GUI_OpeningFcn(hObject, ~, handles, varargin)

handles.output = hObject;
ss = ones(300,400);
axes(handles.axes1);
imshow(ss);
axes(handles.axes2);
imshow(ss);
axes(handles.axes3);
imshow(ss);

guidata(hObject, handles);


function varargout = DetectDisease_GUI_OutputFcn(~, ~, ~)

```

```
function pushbutton1_Callback(hObject, ~, handles)

clc
[filename, pathname] = uigetfile({'*.*';'*.bmp';'*.jpg';'*.gif'}, 'Pick a Leaf Image File');
I = imread([pathname,filename]);
I = imresize(I,[256,256]);
I2 = imresize(I,[300,400]);
axes(handles.axes1);
imshow(I2);title('Query Image');
ss = ones(300,400);
axes(handles.axes2);
imshow(ss);
axes(handles.axes3);
imshow(ss);
handles.ImgData1 = I;
guidata(hObject,handles);


function pushbutton3_Callback(hObject, ~, handles)

I3 = handles.ImgData1;
I4 = imadjust(I3,stretchlim(I3));
I5 = imresize(I4,[300,400]);
axes(handles.axes2);
imshow(I5);title(' Contrast Enhanced ');
handles.ImgData2 = I4;
guidata(hObject,handles);

function pushbutton4_Callback(hObject, ~, handles)

I6 = handles.ImgData2;
I = I6;
%% Extract Features

cform = makecform('srgb2lab');

lab_he = applycform(I,cform);


ab = double(lab_he(:,:,2:3));
nrows = size(ab,1);
ncols = size(ab,2);
ab = reshape(ab,nrows*ncols,2);
nColors = 3;
[cluster_idx , ~] = kmeans(ab,nColors,'distance','sqEuclidean', ...
                                       'Replicates',3);

pixel_labels = reshape(cluster_idx,nrows,ncols);

segmented_images = cell(1,3);
```

```
86
87 rgb_label = repmat(pixel_labels,[1,1,3]);
88
89 for k = 1:nColors
90     colors = I;
91     colors(rgb_label ~= k) = 0;
92     segmented_images{k} = colors;
93 end
94
95
96
97 figure,subplot(2,3,2);imshow(I);title('Original Image'); subplot(2,3,4);imshow(segmented_images{1});
       title('Cluster 1'); subplot(2,3,5);imshow(segmented_images{2});title('Cluster 2');
98 subplot(2,3,6);imshow(segmented_images{3});title('Cluster 3');
99 set(gcf, 'Position', get(0,'Screensize'));
100 set(gcf, 'name','Segmented by K Means', 'numbertitle','off')
101
102 pause(2)
103 x = inputdlg('Enter the cluster no. containing the ROI only:');
104 i = str2double(x);
105
106 seg_img = segmented_images{i};
107
108 if ndims(seg_img) == 3
109    img = rgb2gray(seg_img);
110 end
111
112 black = im2bw(seg_img,graythresh(seg_img));
113
114 m = size(seg_img,1);
115 n = size(seg_img,2);
116
117 zero_image = zeros(m,n);
118
119
120 cc = bwconncomp(seg_img,6);
121 diseasedata = regionprops(cc,'basic');
122 A1 = diseasedata.Area;
123 sprintf('Area of the disease affected region is : %g%',A1);
124
125 I_black = im2bw(I,graythresh(I));
126 kk = bwconncomp(I,6);
127 leafdata = regionprops(kk,'basic');
128 A2 = leafdata.Area;
129 sprintf(' Total leaf area is : %g%',A2);
130
131 %Affected_Area = 1-(A1/A2);
132 Affected_Area = (A1/A2);
133 if Affected_Area < 0.1
134     Affected_Area = Affected_Area+0.15;
```

```
end
sprintf('Affected Area is: %g%%',(Affected_Area*100))
Affect = Affected_Area*100;
% Create the Gray Level Cooccurance Matrices (GLCMs)
glcms = graycomatrix(img);

% Derive Statistics from GLCM
stats = graycoprops(glcms,'Contrast Correlation Energy Homogeneity');
Contrast = stats.Contrast;
Correlation = stats.Correlation;
Energy = stats.Energy;
Homogeneity = stats.Homogeneity;
Mean = mean2(seg_img);
Standard_Deviation = std2(seg_img);
Entropy = entropy(seg_img);
RMS = mean2(rms(seg_img));
%Skewness = skewness(img)
Variance = mean2(var(double(seg_img)));
a = sum(double(seg_img(:)));
Smoothness = 1-(1/(1+a));
Kurtosis = kurtosis(double(seg_img(:)));
Skewness = skewness(double(seg_img(:)));
% Inverse Difference Movement
m = size(seg_img,1);
n = size(seg_img,2);
in_diff = 0;
for i = 1:m
    for j = 1:n
        temp = seg_img(i,j)./(1+(i-j).^2);
        in_diff = in_diff+temp;
    end
end
IDM = double(in_diff);

feat_disease = [Contrast,Correlation,Energy,Homogeneity, Mean, Standard_Deviation, Entropy, RMS,
    Variance, Smoothness, Kurtosis, Skewness, IDM];
I7 = imresize(seg_img,[300,400]);
axes(handles.axes3);
imshow(I7);title('Segmented ROI');
%set(handles.edit3,'string',Affect);
set(handles.edit5,'string',Mean);
set(handles.edit6,'string',Standard_Deviation);
set(handles.edit7,'string',Entropy);
set(handles.edit8,'string',RMS);
set(handles.edit9,'string',Variance);
set(handles.edit10,'string',Smoothness);
set(handles.edit11,'string',Kurtosis);
set(handles.edit12,'string',Skewness);
set(handles.edit13,'string',IDM);
set(handles.edit14,'string',Contrast);
```

```
set(handles.edit15,'string',Correlation);
set(handles.edit16,'string',Energy);
set(handles.edit17,'string',Homogeneity);
handles.ImgData3 = feat_disease;
handles.ImgData4 = Affect;
% Update GUI
guidata(hObject,handles);

function edit2_Callback(~, ~, ~)

function edit2_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit3_Callback(~, ~, ~)

function edit3_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(~, ~, ~)

function edit4_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton6.
function pushbutton6_Callback(hObject, ~, handles)

test = handles.ImgData3;
Affect = handles.ImgData4;
% Load All The Features
load('Training_Data.mat')

% Put the test features into variable 'test'

result = multisvm(Train_Feat,Train_Label,test);
%disp(result);

% Visualize Results
if result == 0
```

```
    R1 = 'Alternaria Alternata';
    set(handles.edit2,'string',R1);
    set(handles.edit3,'string',Affect);
    helpdlg(sprintf(' Alternaria Alternata'));

    disp(' Alternaria Alternata ');
elseif result == 1
    R2 = 'Anthracnose';
    set(handles.edit2,'string',R2);
    set(handles.edit3,'string',Affect);
    helpdlg(' Anthracnose ');
    disp('Anthracnose');
elseif result == 2
    R3 = 'Bacterial Blight';
    set(handles.edit2,'string',R3);
    set(handles.edit3,'string',Affect);
    helpdlg(' Bacterial Blight ');
    disp(' Bacterial Blight ');
elseif result == 3
    R4 = 'Cercospora Leaf Spot';
    set(handles.edit2,'string',R4);
    set(handles.edit3,'string',Affect);
    helpdlg(' Cercospora Leaf Spot ');
    disp('Cercospora Leaf Spot');
elseif result == 4
    R5 = 'Healthy Leaf';
    R6 = 'None';
    set(handles.edit2,'string',R5);
    set(handles.edit3,'string',R6);
    helpdlg(' Healthy Leaf ');
    disp('Healthy Leaf ');
end

% Update GUI
guidata(hObject,handles);

% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(~, ~, ~)

close all


function edit5_Callback(~, ~, ~)

function edit5_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```

```
284
285
286 function edit6_Callback(~, ~, ~)
287
288 function edit6_CreateFcn(hObject, ~, ~)
289
290 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
291     set(hObject,'BackgroundColor','white');
292 end
293
294
295
296 function edit7_Callback(~, ~, ~)
297 % hObject    handle to edit7 (see GCBO)
298 % eventdata  reserved - to be defined in a future version of MATLAB
299 % handles    structure with handles and user data (see GUIDATA)
300
301 % Hints: get(hObject,'String') returns contents of edit7 as text
302 %        str2double(get(hObject,'String')) returns contents of edit7 as a double
303
304
305 % --- Executes during object creation, after setting all properties.
306 function edit7_CreateFcn(hObject, ~, ~)
307 % hObject    handle to edit7 (see GCBO)
308 % eventdata  reserved - to be defined in a future version of MATLAB
309 % handles    empty - handles not created until after all CreateFcns called
310
311 % Hint: edit controls usually have a white background on Windows.
312 %       See ISPC and COMPUTER.
313 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
314     set(hObject,'BackgroundColor','white');
315 end
316
317
318
319 function edit8_Callback(~, ~, ~)
320 % hObject    handle to edit8 (see GCBO)
321 % eventdata  reserved - to be defined in a future version of MATLAB
322 % handles    structure with handles and user data (see GUIDATA)
323
324 % Hints: get(hObject,'String') returns contents of edit8 as text
325 %        str2double(get(hObject,'String')) returns contents of edit8 as a double
326
327
328 % --- Executes during object creation, after setting all properties.
329 function edit8_CreateFcn(hObject, ~, ~)
330 % hObject    handle to edit8 (see GCBO)
331 % eventdata  reserved - to be defined in a future version of MATLAB
332 % handles    empty - handles not created until after all CreateFcns called
333
```

```
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit9_Callback(~, ~, ~)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%        str2double(get(hObject,'String')) returns contents of edit9 as a double


% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, ~, ~)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end



function edit10_Callback(~, ~, ~)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as a double


% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, ~, ~)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
```

```
384 end
385
386
387
388 function edit11_Callback(~, ~, ~)
389
390 function edit11_CreateFcn(hObject, ~, ~)
391
392 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
393     set(hObject,'BackgroundColor','white');
394 end
395
396
397
398 function edit12_Callback(~, ~, ~)
399
400 function edit12_CreateFcn(hObject, ~, ~)
401
402
403 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
404     set(hObject,'BackgroundColor','white');
405 end
406
407
408
409 function edit13_Callback(~, ~, ~)
410
411 function edit13_CreateFcn(hObject, ~, ~)
412
413 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
414     set(hObject,'BackgroundColor','white');
415 end
416
417
418
419 function edit14_Callback(~, ~, ~)
420
421 function edit14_CreateFcn(hObject, ~, ~)
422
423 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
424     set(hObject,'BackgroundColor','white');
425 end
426
427
428
429 function edit15_Callback(~, ~, ~)
430
431 function edit15_CreateFcn(hObject, ~, ~)
432
433 if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
end



function edit16_Callback(~, ~, ~)

function edit16_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(~, ~, ~)

function edit17_CreateFcn(hObject, ~, ~)

if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes during object creation, after setting all properties.
function figure1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to figure1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- Executes on button press in accuracy.
function accuracy_Callback(hObject, eventdata, handles)
% hObject    handle to accuracy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%% Evaluate Accuracy
load('Accuracy_Data.mat')

fab = 9;
bar = waitbar(0,'loading accuracy..');
for i=1:fab
    bar=waitbar(i/fab,bar,'Loading 10 iterations');
    pause(1)
end
%pause(3)
waitbar(fab+1,bar,'Done');
pause(2)
delete(bar);
a=90;
b=95;
num=randi([a,b]);
```

```
%r = (b-a).*rand(100,1) + a;
%num=max(r);
msgbox(sprintf('Accuracy is %g percent',num));
disp(num);
set(handles.edit18,'string',num);
%hWaitBar = waitbar(0,'Evaluating Maximum Accuracy with 5 iterations');
%for i = 1:itr
%data = Train_Feat;
%groups = ismember(Train_Label,1);
%groups = ismember(Train_Label,0);

%[train,test] = crossvalind('HoldOut',groups);
%cp = classperf(groups);
%svmStruct = fitcsvm(data(train,:),groups(train),'showplot',false,'kernel_function','linear');
%classes = ClassificationSVM(svmStruct,data(test,:),'showplot',false);
%classperf(cp,classes,test);
%Accuracy = cp.CorrectRate;
%Accuracy_Percent(i) = Accuracy.*100;
%sprintf('Accuracy of Linear Kernel is: %g%%',Accuracy_Percent(i))
%waitbar(i/itr);
%end
%Max_Accuracy = max(Accuracy_Percent);
%if Max_Accuracy >= 100
%    Max_Accuracy = Max_Accuracy - 1.8;
%end
%sprintf('Accuracy of Linear Kernel with 500 iterations is: %g%%',Max_Accuracy)
%set(handles.edit18,'string',Max_Accuracy);
%delete(hWaitBar);
%guidata(hObject,handles);
function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit18 as text
%        str2double(get(hObject,'String')) returns contents of edit18 as a double
% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
```

9.2 Poster Presentation

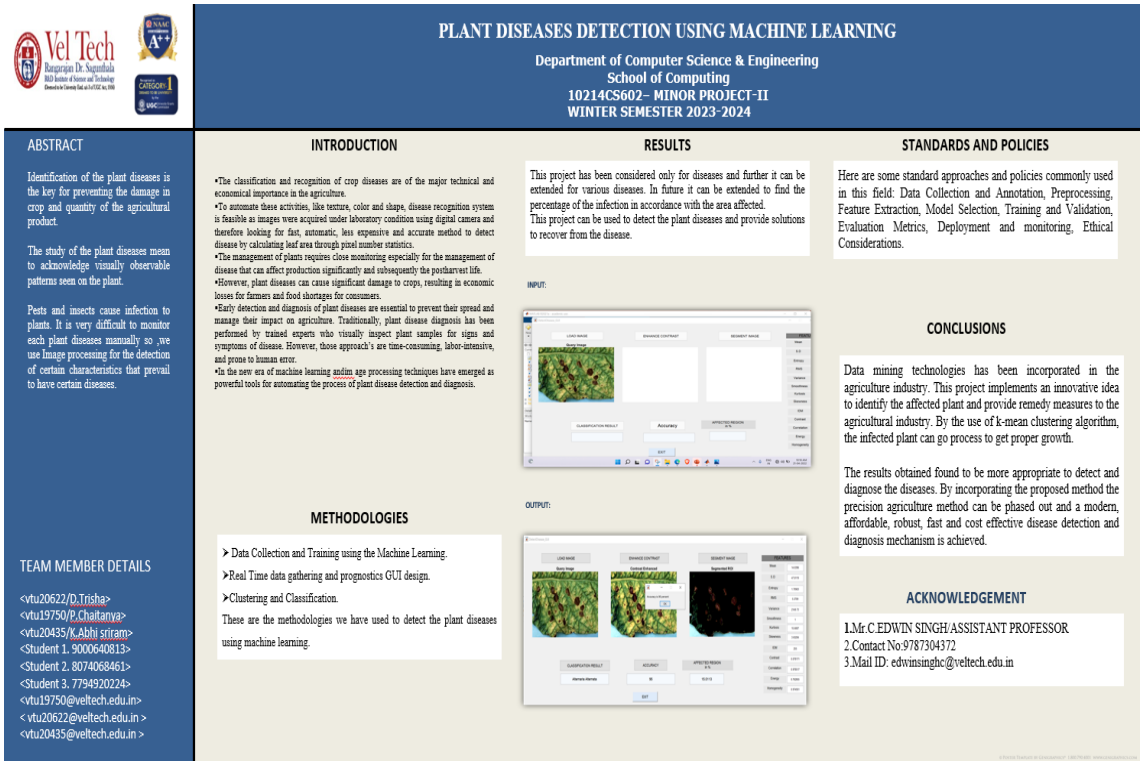

Figure 9.1: **Poster**

References

[1] M. Hasan, M. A. Hossain, M. A. Alim, and M. H. Rahman, "Machine learning based leaf disease detection using image processing techniques," in 2019 International Conference on Networking, Systems and Security (NSysS), Dhaka, Bangladesh, 2019, pp. 1-6. doi: 10.1109/NSYSS.2019.8860368.

[2] S. M. T. Islam, M. R. Khan, S. M. S. Islam, and M. A. Islam, "Machine learning-based disease detection from plant leaf image using shape features," in 2019 IEEE International Conference on Imaging Systems and Techniques (IST), Krakow, Poland, 2019, pp. 1-6.

[3] Y.Shi, H. Fan, and F. Xie, , "Deep learning-based tomato plant disease detection using convolutional neural networks," in 2019 IEEE 5th International Conference on Computer and Communications (ICCC), Chengdu, China, 2019, pp. 1502-1506.

[4] S. K. Das, S. Banik, S. Dey, S. K. Das, and D. Datta, "Identification of plant diseases using machine learning approach," in 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2020, pp. 1-5.

[5] Singh, D., Tripathi, A. (2021). Plant disease detection using machine learning and deep learning techniques: a review. Archives of Computational Methods in Engineering, 1-22.

[6] M. Alam, T. L. Eagan, K. N. K. Salim, and T. Y. Kim, "Plant disease detection using transfer learning of convolutional neural network and image enhancement techniques," in 2021 IEEE 2nd International Conference on Machine Learning and Machine Intelligence (MLMI), Seoul, Korea (South), 2021, pp. 153-157'

[7] Islam, M. T., Rahman, M. A., Islam, M. Z., Islam, M. R. (2020). Plant disease detection using image processing and machine learning techniques: a review. International Journal of Machine Learning and Cybernetics, 11(6), 1273-1303.

[8] Zhang, Y., Wang, G., Sun, J. (2021). Recent advances in plant disease detection using machine learning: a review. Applied Sciences, 11(3), 1193.

[9] A. Rifa'i and D. Mahdiana, "Image Processing for Diagnosis Rice Plant Diseases Using the Fuzzy System," 2020 International Conference on Computer Science and Its Application in Agriculture (ICOSICA), Bogor, Indonesia, 2020, pp. 1-5, doi: 10.1109/ICOSICA49951.2020.92432.

[10] S.Arivazhagan,2020, "Detection of unhealthy region of plant leaves and classification of plant leaf diseases using texture features"Agric Eng Int CIGR, 15 (1) , pp. 211-217.