A Project Report

on

DETECTION AND CLASSIFICATION OF FRUIT DISEASES

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

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

17WH1A0540	M. GAYATRI
17WH1A0550	U. MOUNIKA
18WH5A0506	S. NISCHALA

Under the esteemed

guidance of

Ms. A. KRANTHI

Assistant Professor

Department of Computer Science & Engineering

BVRIT HYDERABAD

College of Engineering for Women

(NBA Accredited EEE.ECE.CSE.IT B.Tech Courses,

Accredited to NAAC with 'A' Grade)

(Approved by AICTE, New Delhi and Affiliated JNTUH,Hyderabad)

Bachupally, Hyderabad – 500090

MAY, 2021

BVRIT HYDERABAD

College of Engineering for Women

(NBA Accredited EEE.ECE.CSE.IT B.Tech Courses,
Accredited to NAAC with 'A' Grad
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Bachupally, Hyderabad – 500090

Department of Computer Science & Engineering

CERTIFICATE

This is to cerrtify that the Project Work entitled **"DETECTION AND CLASSIFICATION OF FRUIT DISEASES"** is a bonafide work carried out by **Ms. M.GAYATRI (17WH1A0540), Ms. U.MOUNIKA(17WH1A0550) Ms. S.NISCHALA (18WH1A0506),** in the partial fulfillment for the award of B.Tech. degree in **Computer Science and Engineering, BVRIT HYDERABAD College of Engineering for Women, Bachupally, Hyderabad**, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision.

The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Internal Guide
Ms.A.Kranthi
Asst. Professor, CSE

Head ofthe Department
Dr. Srinivasa Reddy Konda
Professor, CSE

External Examiner

DECLARATION

We hereby declare that the work presented in this project entitled **"DETECTION AND CLASSIFICATION OF FRUIT DISEASES"** submitted towards completion of Project Work in IV year of B.Tech , CSE at 'BVRIT HYDERABAD College of Engineering For Women Hyderabad' is an authentic record of our original work carried out under the guidance of Ms.A.Kranthi, Assistant Professor, Department of CSE.

Roll No.	Name	Signature
17WH1A0540	Ms. M. Gayatri	
17WH1A0550	Ms. U.Mounika	
18WH5A0506	Ms. S.Nischala	

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr.K.V.N.Sunitha**, **Principal, BVRIT HYDERABAD College of Engineering for Women** for her support by providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr. Srinivasa Reddy Konda, Head of Department of CSE, BVRIT HYDERABAD College of Engineering for Women** for all timely support and valuable suggestions during the period of our project.

We are extremely thankful to our Internal Guide, **Ms. A. Kranthi, Asst. Professor, CSE, BVRIT HYDERABAD College of Engineering for Women** for her constant guidance and encouragement throughout the project.

We express our gratitude to our Major Project coordinator **Dr. Ganti Naga Satish, Professor, CSE**, **BVRIT HYDERABAD College of Engineering for Women** for his valuable suggestions in completing our work successfully.

Finally, We would like to thank all our faculty and Staff of CSE department who helped us directly or indirectly. Last but not least, we wish to acknowledge our **Parents** and **Friends** for giving moral strength and constant encouragement.

M. Gayatri **(17WH1A0540)**
U.Mounika **(17WH1A0550)**
S.Nischala **(18WH5A0506)**

Contents

ABSTRACT

Fruit disease detection is most important that more than 70% of the people depend on agriculture for their livelihood in India. Nowadays the growth of productivity of plants, crops and fruits are normally affected by the diseases. This project aims at detection of fruit diseases at early stage since it will affect the agriculture field. The disease is a major problem arising in an agriculture field. In plants most of the fruits are affected by diseases due to bacteria and virus.

The fruit details and the identification of diseases from the feature extraction are stored in the database. The classification and segmentation of fruit images were performed using K-Means Algorithm and SVM technique. The various features of few fruits were initially extracted and segment the respective images. After comparison the various diseases are analyzed and the optimal disease for the image is identified and the disease name is displayed. It also helps farmers to do smart farming which helps to take time to time decisions which also save time and reduce loss of fruit due to diseases. The leading objective of our project is to enhance the value of fruit disease detection.

LIST OF FIGURES

1. INTRODUCTION

Agriculture has been the base for every person. It is most important that more than 70% of the people depend on agriculture for their livelihood in India. Nowadays the growth of productivity of plants, crops and fruits are normally affected by the diseases. The disease is a major problem arising in an agricultural field. In plants, most of the leaves and fruits are affected by diseases due to bacteria and viruses. This technique is used to determine the infection on leaves, and fruits. In order to generate an automated database to examine the infections using the proposed method. The database consists of data related to plant leaves, fruit conditions and the symptoms of disease to be affected.

The fruit details and the identification of disease from the feature extraction are stored in the database. The entire database is viewed and compared with the captured image. The project is developed for processing the data and providing intimation to the farmers. Thus the variation in image from the database also indicates the disease in the fruits.

1.1 Objectives:

Fruit disease detection is vital at an early stage since it will affect the agricultural field. It mainly considers the detection and analysis of fruit infections which is available in the plant areas and storage of data about the agricultural field and details in the database. The detected data from the plant area is determined by image processing and stored in the database.

The classification and segmentation of fruit images were performed using K-Means Algorithm and SVM technique. The various features of a few fruits were initially extracted and segment the respective images. After comparison with feature values, the various disease names are analysed and the optimal disease for the image is identified and the disease is indicated by an alert message.

The objectives are agricultural input systemization, profit hike and environmental damage reduction. So, in this work, a solution for the detection and

classification of fruit diseases is proposed and experimentally validated. This system takes input as an image of fruit and leaves and identifies it as infectious or noninfectious. The technique which helps the farmers to identify disease properly by using this proposed work.

1.2 Methodology:

To classify fruit diseases a large collection of the fruits and leaf images is required. The images are downloaded from the kaggle and data mendeley database. In this section the methodology followed is discussed in detail.

1.2.1 Dataset:

Proper and large dataset is required for all classification research during the training and the testing phase. The dataset for the experiment is downloaded from the Kaggle and data mendeley database which contains different fruit and leaf images and their labels. It contains a collection of images taken at different positions. A dataset containing approximately 2000 images. here we containing around 1500 images for training and 500 images for testing was downloaded and we are taken of five different classes including healthy leaves is downloaded. If the image is not affected by any disease that means that fruit was healthy.

The classical approach for detection and classification of fruit diseases is based on the naked eye observation by the experts. In some developing countries, consulting experts are expensive and time consuming due to the distant locations of their availability. Automatic detection of fruit diseases is essential to automatically detect the symptoms of diseases as early as they appear on the growing fruits.

Fruit diseases can cause major losses in yield and quality appeared in harvesting. To know what control factors to take next year to avoid losses, it is crucial to recognize what is being observed. Some disease also infects other areas of the tree causing diseases of twigs, leaves, and branches.

For example**,** some common diseases of orange fruits are orange scab, orange canker, and orange blackspot. orange scabs are gray or brown corky spots. orange rot infections produce slightly sunken, circular brown or black spots that may be covered by a red halo. orange canker is a fungal disease and appears on the surface of the fruit as dark blackdots. Visual inspection of orange is already automated in the industry by machine vision with respect to size and color.

However, detection of defects is still problematic due to natural variability of skin color in different types of fruits, high variance of defect types, and presence of stem/calyx. The studies of fruit can be determined by apparent patterns of specific fruit and it is critical to monitor health and detect disease within a fruit. Through proper management action such as pesticides, fungicides and chemical applications one can promote control of diseases which interns improve quality.

There are various approaches available such as spectroscopic and imaging technology, applied to achieve better plant disease control and management. The increased amount of commercialization of agricultural farms is always on the lookout to reduce manpower in whatever way possible without affecting productivity. A particular aspect to look upon is to use automatic harvesters which would significantly economize the entire process.

Fruit detection system has its major application in robotic harvesting. However the technology can be custom made to be suitable for other applications such as disease detection, maturity detection, tree yield monitoring and other similar operations. Varieties of fruits are being exported all over the world with the development of cold storage facilities and transportation. It becomes the necessity of maintaining the highest level export quality which is mainly carried out by visual checking by experts. This is expensive and time consuming due to the distant location of farms. Precision Agriculture helps the farmers to provide with sufficient and economical information and control technology due to the development and disclosure in various fields.

What you will learn:

- How to use Pycharm.
- How to use K-Means and SVM Algorithms.
- How to classify images with your trained classifier.

Download the Training Images:

Before you start any training, you'll need a set of images to teach the model about the new classes you want to recognize, which we have already downloaded a few images for the algorithm to predict.

- Place all the images in the folder structure as listed below.

C:\Users\HP\Desktop\MajProject\FruitDisease\FruitDataset
Under the FruitDataset, the folder structure should as shown below

- Black spot
- Canker
- Greening
- healthy
- Scab

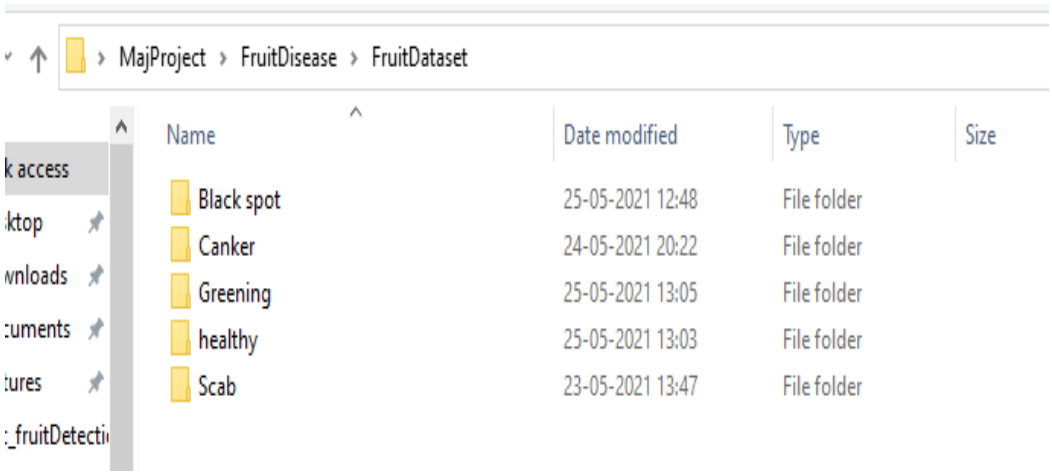

Figure 1.2.1: Dataset Contains fruit Diseases

1.2.2 The proposed model:

This project proposes a web based tool that helps farmers to identify fruit disease by uploading fruit and leaves images to the system. The system has an already trained dataset of images for the fruits and leaves. Input image given by the user undergoes several processing steps to detect the severity of disease by comparing with the trained dataset images.

The classification and segmentation of fruit images were performed using K-Means Algorithm and SVM technique. The various features of a few fruits were initially extracted and segment the respective images. After comparison with feature values, the various disease names are analysed and the optimal disease for the image is identified and the disease will be displayed.

K-Means Clustering Algorithm:

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science. Here K defines the number of predefined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

"It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs to only one group that has similar properties."It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.

It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.

The k-means clustering algorithm mainly performs two tasks:

- Determines the best value for K center points or centroids by an iterative process.
- Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

- Hence each cluster has data points with some commonalities, and it is away from other clusters.

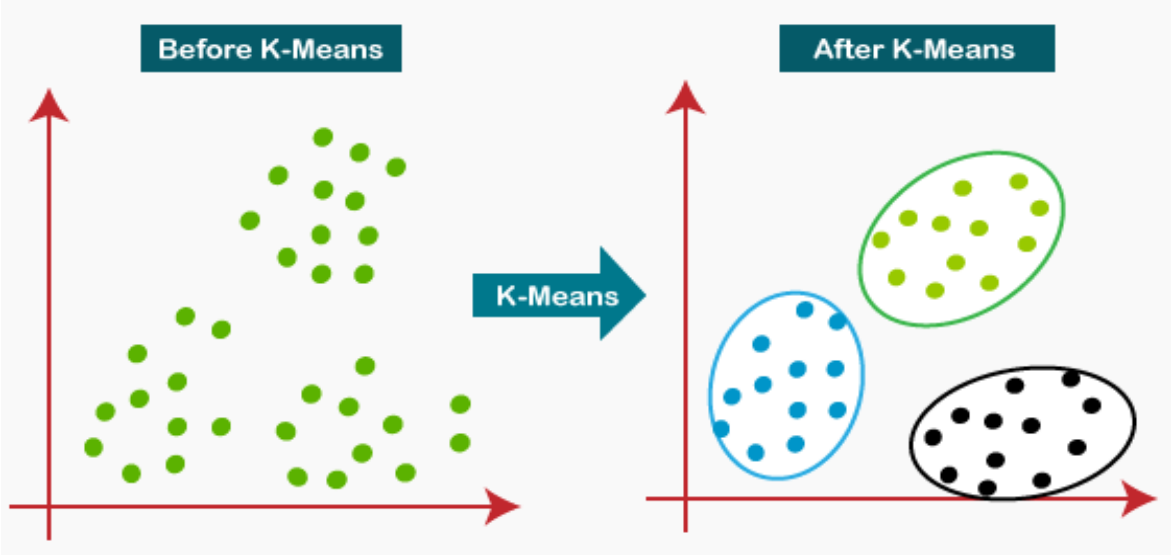

Figure 1.2.2 : Performance of K-Means Algorithm

Algorithm used for K-MEANS Clustering :

Input:

- Dataset (fruit image)
- K number of desired clusters.

Output:

- K set of clusters.

STEPS:

1.) Initialize the number of cluster k, and also pick initial centroid randomly.

2.) The squared Euclidean distance will be calculated from each image to each cluster is computed, and each object is assigned to the closest cluster.

3.) For each cluster, the new centroid is computed and each seed value is now replaced by the respective cluster centroid.

4.) Euclidean distance from an object to each cluster is calculated, and the image is allotted to the cluster with the smallest Euclidean distance.

5.) This process will be continue until image is in same cluster at every iteration.

Advantages of K-Means Algorithm:

- Relatively simple to implement.
- Scales to large data sets.
- Guarantees convergence.
- Can warm-start the positions of centroids.
- Easily adapts to new examples.
- Generalizes to clusters of different shapes and sizes, such as elliptical clusters.

Disadvantages of K-Means Algorithm:

- Choosing k manually.
- Being Dependent of initial values
- Clustering data of varying sizes and density
- Clustering outliers
- Scaling with no.of dimensions

SVM Model:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called support vectors, and hence the algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

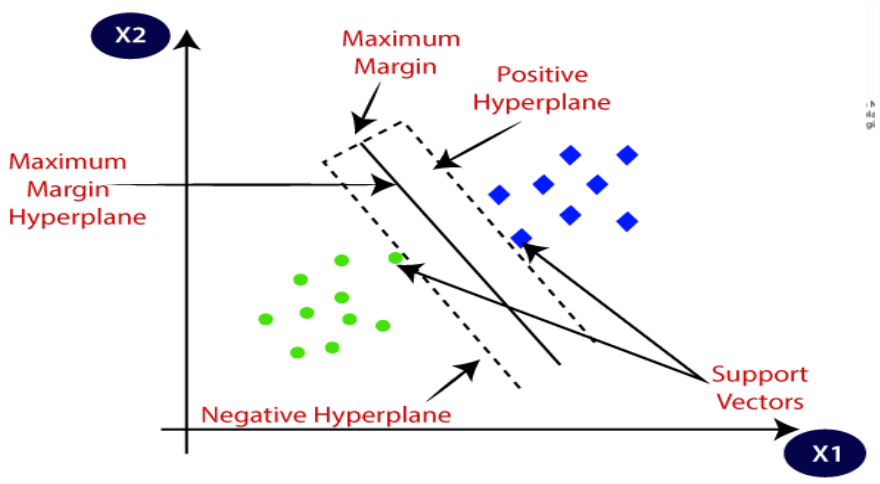

Figure 1.2.2.1: Working of SVM Model

SVM can be categories into two types:

- **Linear SVM:**

  It is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

- **Non-linear SVM:**

  It is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Advantages:

- SVM works relatively well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces.
- SVM is effective in cases where the number of dimensions is greater than the number of samples.
- SVM is relatively memory efficient

Disadvantages:

- SVM algorithm is not suitable for large data sets.
- SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.
- In cases where the number of features for each data point exceeds the number of training data samples, the SVM will underperform.

- As the support vector classifier works by putting data points above and below the classifying hyperplane there is no probabilistic explanation for the classification.

Applications:

- Used for Face Detection.
- Used for Image Classification.
- Used for Text Categorization, etc.

Hyperplane and Support Vectors in the SVM Algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then the hyperplane will be a straight line. And if there are 3 features, then the hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vectors. These vectors support the hyperplane, hence called a Support vector.

SVC:

The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

Confusion Matrix:

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model.

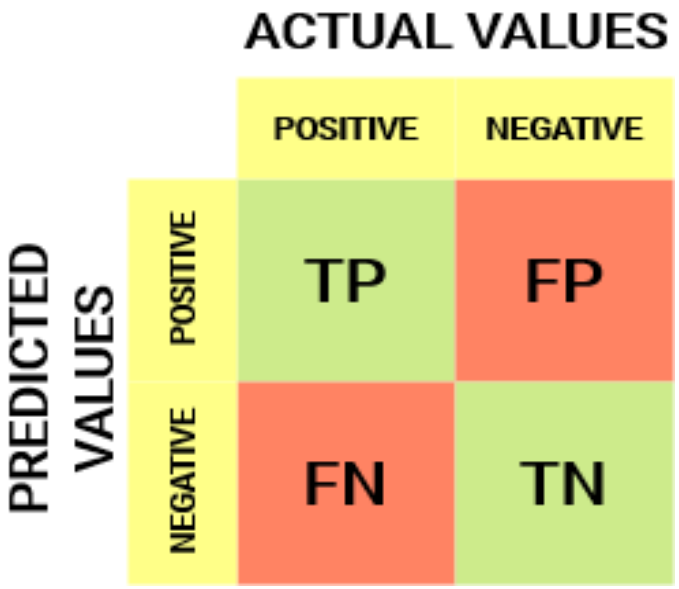

Figure 1.2.2.2: Basic confusion matrix

- The target variable has two values: **Positive** or **Negative.**
- The **columns** represent the **actual values** of the target variable.
- The **rows** represent the **predicted values** of the target variable.

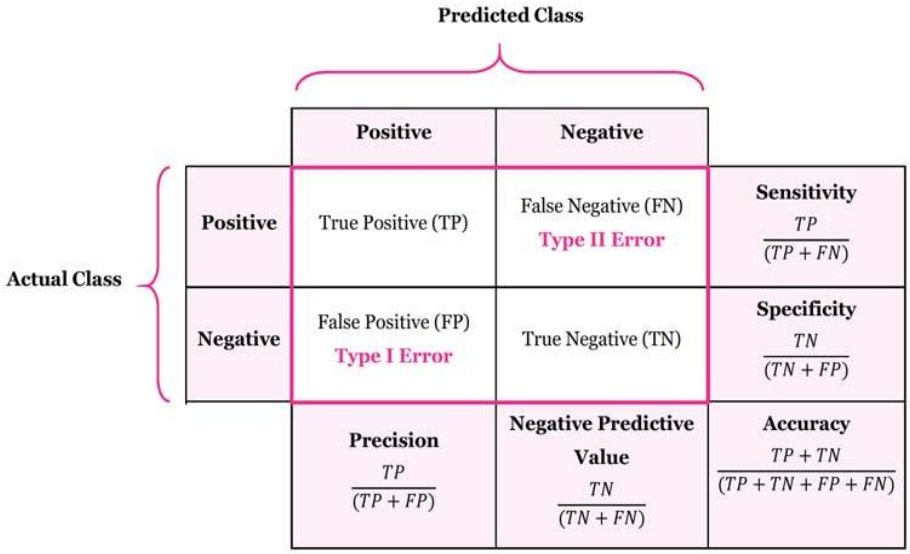

Figure 1.2.2.3: Confusion Matrix

True Positive (TP):-

- The predicted value matches the actual value
- The actual value was positive and the model predicted a positive value

True Negative (TN):-

- The predicted value matches the actual value
- The actual value was negative and the model predicted a negative value

False Positive (FP) – Type 1 error:-

- The predicted value was falsely predicted
- The actual value was negative but the model predicted a positive value
- Also known as the **Type 1 error**

False Negative (FN) – Type 2 error:-

- The predicted value was falsely predicted
- The actual value was positive but the model predicted a negative value
- Also known as the **Type 2 error.**

Accuracy:-

Classification accuracy is the ratio of correct predictions to total predictions made. It can be calculated as follows:

Accuracy = correct predictions / total predictions

It is often presented as a percentage by multiplying the result by 100.

Accuracy = correct predictions / total predictions * 100

1.3 Organization of Project

The technique which is developed is taking input as a fruit or leaf image and compares the uploaded image from the dataset using k-means and svm model. If the input image matches, then fruit disease as a result.

We have three modules in our project.

- Disease identification
- Disease Classification
- Disease prediction

2. THEORETICAL ANALYSIS OF THE PROPOSED PROJECT

2.1 Requirements Gathering :

2.1.1 Software Requirements

Programming Language	:	Python 3.7
Graphical User Interface	:	Tkinter
Dataset	:	Kaggle and Data Mendeley
Packages	:	Numpy, Matplotlib, opencv, Scikit-learn
Framework	:	python 3.7.3
Tool	:	Pycharm or Python ide 3.7 version

2.1.2 Hardware Requirements

Operating System	:	Windows 10/Linux
Processor	:	Intel i5 core processor
CPU Speed	:	2.30 GHz
Memory	:	8 GB (RAM)

2.2 Technologies Description

Python:-

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

- **Python is Interactive** − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Numpy

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

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc. via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis
- Extensions or modules for SciPy care conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

Pycharm

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language. It is developed by the Czech company JetBrains (formerly known as IntelliJ). It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License, and there is also Professional Edition with extra features – released under a proprietary license.

Tkinter

Python provides the standard library Tkinter for creating the graphical user interface for desktop based applications.

Developing desktop based applications with python Tkinter is not a complex task. An empty Tkinter top-level window can be created by using the following steps.

- import the Tkinter module
- Create the main application window.
- Add the widgets like labels, buttons, frames, etc. to the window.

- Call the main event loop so that the actions can take place on the user's computer screen

CV2:OpenCV-

Python is a library of Python bindings designed to solve computer vision problems. cv2.imread() method loads an image from the specified file. If the image cannot be read (because of missing file, improper permissions, unsupported or invalid format) then this method returns an empty matrix.

OpenCV is used to develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.To read an image in Python using OpenCV, use **cv2**. **imread**() function. **imread**() returns a 2D or 3D matrix based on the number of color channels present in the image.

OS:

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system dependent functionality. The *os* and *os.path* modules include many functions to interact with the file system.The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc.

Warning: package to handle warning and ignore is used when we need to filter out all the warnings.

Time: package to handle time manipulation

3. DESIGN

3.1 Introduction

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Once system requirements have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess quality. Design is the only way that we can accurately translate a customer's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed, reviewed and documented. System design can be viewed from either a technical or project management perspective. From the technical point of view, design consists of four activities – architectural design, data structure design, interface design and procedural design.

3.2 Block Diagram:

Here the infection in fruit is detected using MATLAB simulation and the corresponding result of disease name and details is given to the database. The complete system design infection detection is shown in the below:

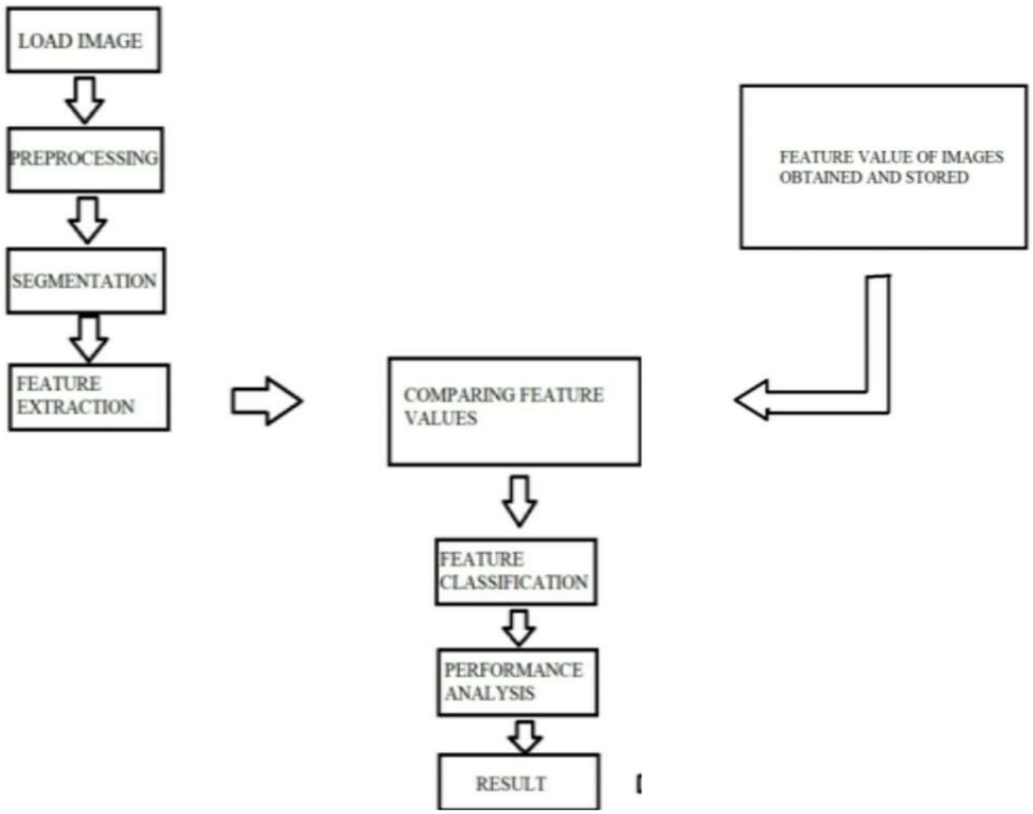

Figure 3.2: Block Diagram

Initially the images are trained in neural networks. The features are extracted and trained using a classifier. After training the images are stored in a database. The test images are given to the classifier and the images are tested and compared with trained images. If it is affected by disease it gives the classified disease in that fruit image.

3.3 Architecture Diagram

Web applications are by nature distributed applications, meaning that they are programs that run on more than one computer and communicate through a network or server. Specifically, web applications are accessed with a web browser and are popular because of the ease of using the browser as a user client. For the enterprise, software on potentially thousands of client computers is a key reason for their popularity. Web applications are used for web mail, online retail sales, discussion boards, weblogs, online banking, and more. One web application can be accessed and used by millions of people.

Like desktop applications, web applications are made up of many parts and often contain mini programs and some of which have user interfaces. In addition, web applications frequently require an additional markup or scripting language, such as HTML, CSS, or JavaScript programming language. Also, many applications use only the Python programming language, which is ideal because of its versatility.

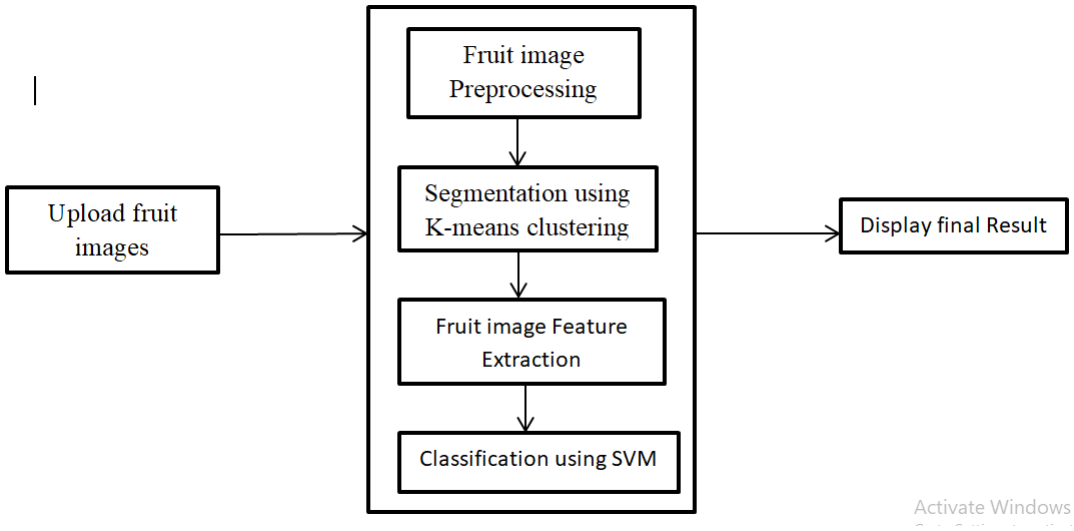

Figure 3.3.1: Architecture Diagram

Image Processing :

Image processing is one of the techniques used to process the natural image into digital image and also to perform few operations like segmentation, feature extraction, etc., in order to get the information about the image. The image of fruits parts is provided as the input to the system and the output is the clustered image which describes the features extracted.

Image Acquisition:

Image acquisition involves the process of acquiring the image from hardware source or by collecting the database available about the plant diseases. Here the image is acquired from the camera or normal image from the database.

Image Preprocessing:

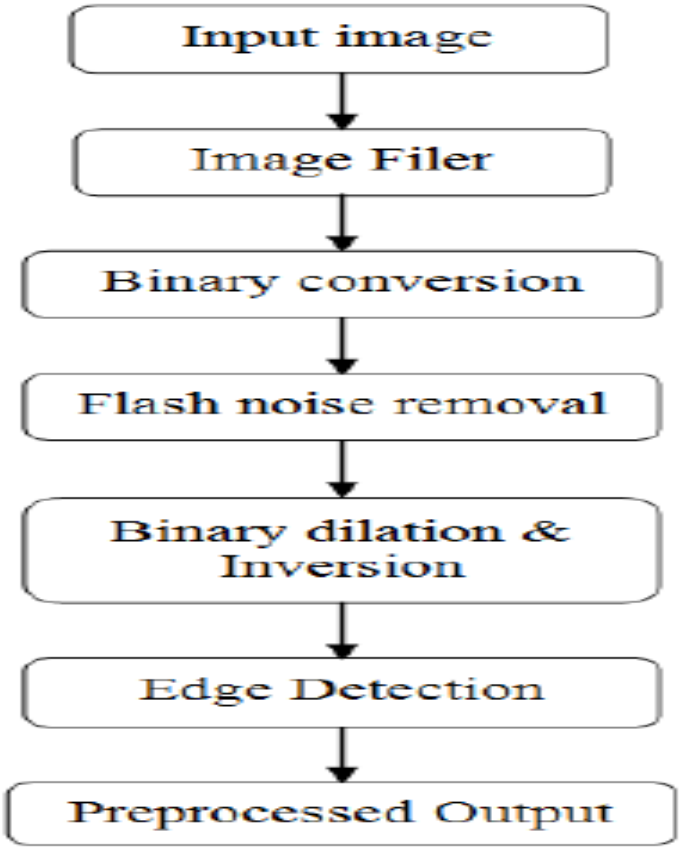

Figure 3.3.1: Block diagram of Image Preprocessing.

The process of enhancing the input image involves the noise elimination, edge detection and shape refinement to enhance the image. The main aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task.

There are 4 different types of Image Pre-Processing techniques and they are listed below.

- Pixel brightness transformations/ Brightness corrections
- Geometric Transformations
- Image Filtering and Segmentation
- Fourier transform and Image restauration.

Segmentation:

Image segmentation refers to the method of segmenting the image into multiple segments. It is performed to simplify the classification and analysis of features in the images. The boundary, area, edge and other features of the image are identified in the image.

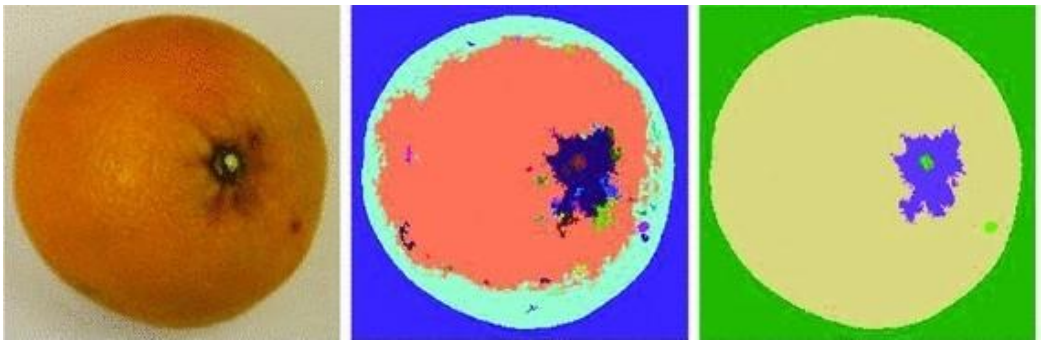

Figure 3.3.2: Segmentation

Feature Extraction:

Feature extraction is the enhancement of images for representing interesting parts of the image. The features such as spots, color, shape, area and other features are considered in the input images. The color is a main feature because it can separate one disease from another. Here, I intend to use color features such as mean and standard deviation.

Furthermore, each disease may have a different shape. Texture features such as Kurtosis, skewness, cluster prominence, and cluster shade. It is mainly performed to minimize the complexity in processing the image. The variations in the features indicate the infection in the fruit images and the disease is identified based on the threshold value.

Classification:

The images are classified and numerical properties of various features are analyzed and classified based on K-Mean, Neural Network and SVM classification techniques. Use a Support vector machine for the classification. SVM is a supervised learning approach. It classifies the training data based on the classes given as training class labels. This technique involves the feature classification from the image features extracted.

K-Means Clustering:

Input parameters:

- **samples** : It should be of np.float32 data type, and each feature should be put in a single column.
- **nclusters(K)** : Number of clusters required at end
- **criteria** : It is the iteration termination criteria. When this criteria is satisfied, algorithm iteration stops. Actually, it should be a tuple of 3 parameters. They are `( type, max_iter, epsilon )`:
    - type of termination criteria. It has 3 flags as below:
        - **cv.TERM_CRITERIA_EPS** - stop the algorithm iteration if specified accuracy, *epsilon*, is reached.

        - **cv.TERM_CRITERIA_MAX_ITER** - stop the algorithm after the specified number of iterations, *max_iter*.
        - **cv.TERM_CRITERIA_EPS + cv.TERM_CRITERIA_MAX_ITER** - stop the iteration when any of the above condition is met.
    - max_iter - An integer specifying maximum number of iterations.
    - epsilon - Required accuracy
- **attempts** : Flag to specify the number of times the algorithm is executed using different initial labellings. The algorithm returns the labels that yield the best compactness. This compactness is returned as output.
- **flags** : This flag is used to specify how initial centers are taken. Normally two flags are used for this **cv.kmeans_pp_centers** and **cv.kmeans_random_centers**.

Output parameters:

- **compactness** : It is the sum of squared distance from each point to their corresponding centers.
- **labels** : This is the label array (same as 'code' in previous article) where each element marked '0', '1'.....
- **centers** : This is array of centers of clusters.

3.4 UML Diagrams

3.4.1 Use Case Diagram

To model a system, the most important aspect is to capture the dynamic behavior. Dynamic behavior means the behavior of the system when it is running/operating.

Only static behavior is not sufficient to model a system; rather dynamic behavior is more important than static behavior. In UML, there are five diagrams available to model the dynamic nature and use case diagrams are one of them. Now as we have to discuss that the use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. Use case diagrams consist of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

Hence to model the entire system, a number of use case diagrams are used.

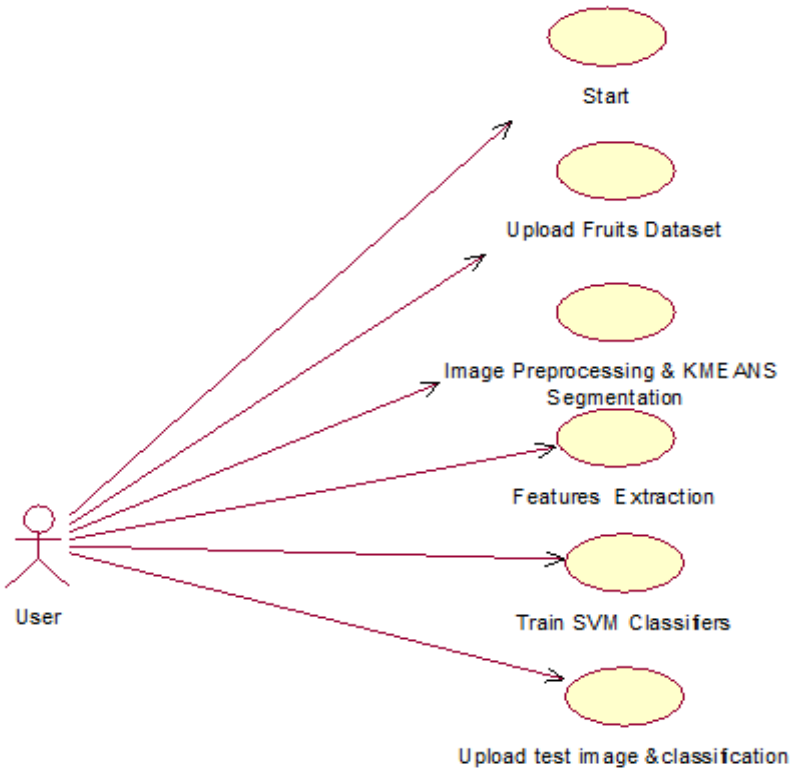

Fig 3.4.1: Use Case Diagram

3.4.2 Sequence Diagram

Sequence Diagrams Represent the objects participating in the interaction horizontally and time vertically. A Use Case is a kind of behavioral classifier that represents a declaration of an offered behavior. Each use case specifies some behavior, possibly including variants that the subject can perform in collaboration with one or more actors. Use cases define the offered behavior of the subject without reference to its internal structure. These behaviors, involving interactions between the actor and the subject, may result in changes to the state of the subject and communications with its environment. A use case can include possible variations of its basic behavior, including exceptional behavior and error handling.

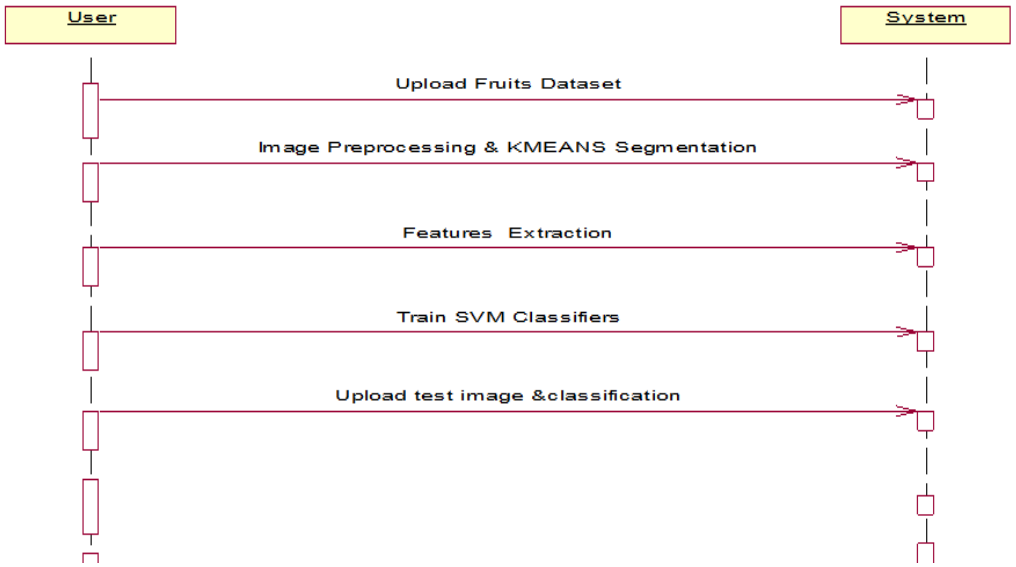

Fig 3.4.2: Sequence Diagram

3.4.3 Activity Diagram

Activity diagrams are graphical representations of Workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

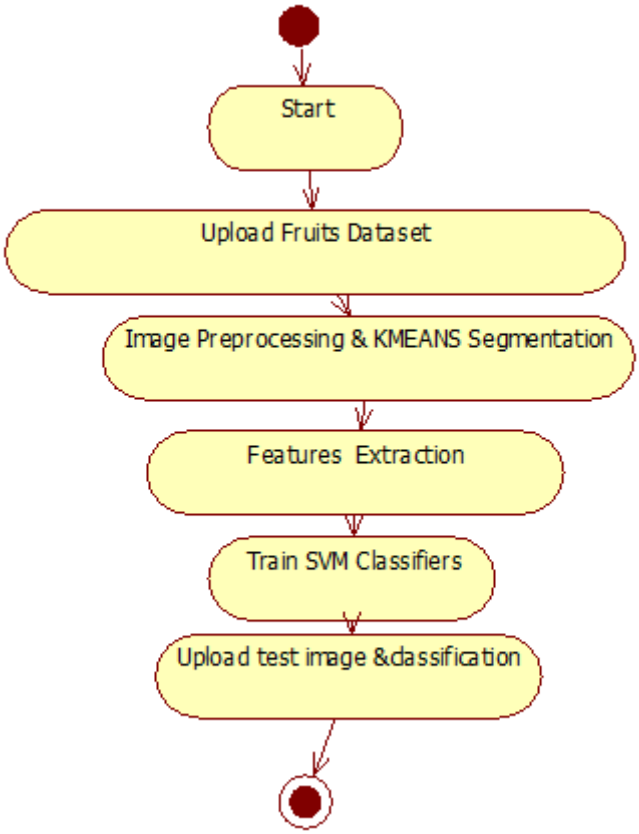

Fig 3.4.3: Activity Diagram

3.4.4 Collaboration Diagram

A collaboration diagram resembles a flowchart that portrays the roles, functionality and behavior of individual objects as well as the overall operation of the system in real time. Objects are shown as rectangles with naming labels inside. These labels are preceded by colons and may be underlined. The relationships between the objects are shown as lines connecting the rectangles. The messages between objects are shown as arrows connecting the relevant rectangles along with labels that define the message sequencing.

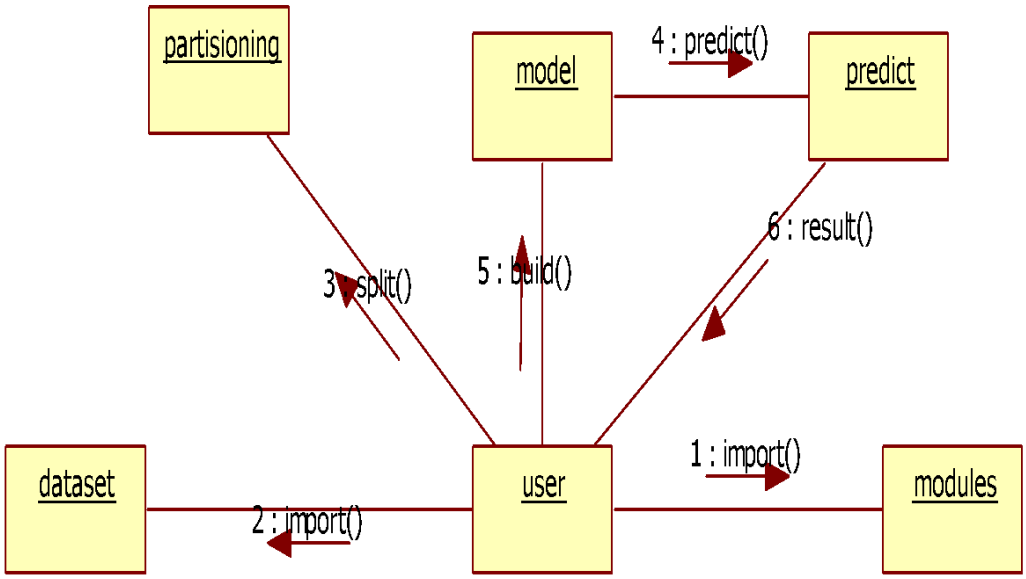

Fig 3.4.4: Collabration Diagram

3.4.5 Class Diagram

The class diagram is the main building block of object-oriented modeling. It is used for general conceptual modeling of the system of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

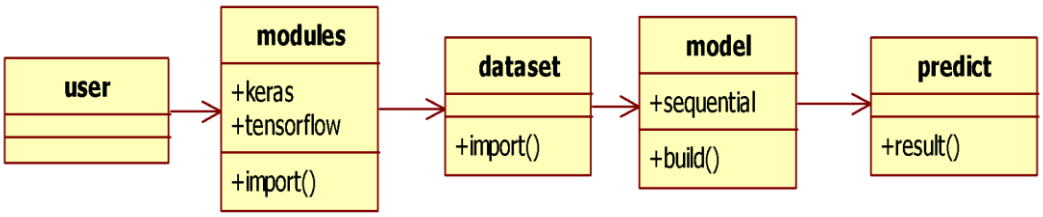

Fig 3.4.5: Class Diagram

4. IMPLEMENTATION

4.1 Coding

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
from tkinter.filedialog import askdirectory
import cv2
from sklearn.model_selection import train_test_split
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix
import random

import os
from sklearn.preprocessing import MinMaxScaler
from sklearn.svm import LinearSVC



main = tkinter.Tk()
main.title("DETECTION AND CLASSIFICATION OF FRUIT DISEASES ")
#designing main screen



fruits_disease = ['Black spot', 'Canker', 'Greening', 'healthy', 'scab']

global filename
```

```
global classifier
global X, Y
global X_train, X_test, y_train, y_test

def uploadFruitDataset():
    global filename
    filename = filedialog.askdirectory(initialdir=".")
    text.delete('1.0', END)
    text.insert(END, filename + " loaded\n");



def Preprocessing():
    global X, Y
    X, Y = [],[]

    for i in range(len(fruits_disease)):
        for root, dirs, directory in os.walk('FruitDataset/'+fruits_disease[i]):
            for j in range(len(directory)):
                img = cv2.imread('FruitDataset/'+fruits_disease[i] + "/" + directory[j])
                img = cv2.resize(img, (128, 128))
                img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
                pixel_vals = img.reshape((-1, 3))
                pixel_vals = np.float32(pixel_vals)
                criteria = (cv2.TERM_CRITERIA_EPS +
cv2.TERM_CRITERIA_MAX_ITER, 100, 0.85)
                retval, labels, centers = cv2.kmeans(pixel_vals, 6, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)
                centers = np.uint8(centers)
                segmented_data = centers[labels.flatten()]
                X.append(segmented_data.ravel())
                Y.append(i)
                print('FruitDataset/' + fruits_disease[i] + "/" + directory[j] + " " +
str(X[j].shape))
    np.save("features/features.txt.npy", X)
```

```
    np.save("features/labels.txt.npy", Y)

    X = np.load("features/features.txt.npy")
    Y = np.load("features/labels.txt.npy")
    text.insert(END,"Total preprocess images are : "+str(X.shape[0])+"\n\n")


def featuresExtraction():
    global X_train, X_test, y_train, y_test
    global X, Y
    ind = random.randint(0, (len(X) - 1))
    img = X[ind].reshape(128,128,3)
    cv2.imshow('Image after KMEANS & Feature Extraction',cv2.resize(img,(450,450)))
    cv2.waitKey(0)

    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    Y = Y[indices]
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2, random_state = 0)
    text.insert(END,"Features Extraction Process Completed\n")
    text.insert(END,"Total images used to train SVM is : "+str(X_train.shape[0])+"\n")
    text.insert(END,"Total images used to test  SVM is : "+str(X_test.shape[0])+"\n\n")


def svmClassifier():
    global classifier
    cls = svm.SVC(C=12, gamma='scale', kernel='rbf', random_state = 0)
    cls.fit(X, Y)
    prediction = cls.predict(X_test)
    svm_acc = accuracy_score(y_test,prediction)*100
    text.insert(END, "SVM Accuracy : "+str(svm_acc)+"\n")
```

```
    cm = confusion_matrix(y_test, prediction)
    total = sum(sum(cm))
    specificity = cm[1, 1]/(cm[1, 0]+cm[1, 1])
    text.insert(END, 'SVM Algorithm Specificity : '+str(specificity * 100)+"\n")
    classifier = cls


def Classification():
    name = filedialog.askopenfilename(initialdir="testImages")
    img = cv2.imread(name)
    img = cv2.resize(img, (128, 128))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    pixel_vals = img.reshape((-1, 3))
    pixel_vals = np.float32(pixel_vals)
    criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER,
100, 0.85)
    retval, labels, centers = cv2.kmeans(pixel_vals, 6, None, criteria, 10,
cv2.KMEANS_RANDOM_CENTERS)
    centers = np.uint8(centers)
    segmented_data = centers[labels.flatten()]
    temp = []
    temp.append(segmented_data.ravel())
    temp = np.array(temp)
    predict = classifier.predict(temp)[0]
    img = cv2.imread(name);
    img = cv2.resize(img, (500, 500))
    disease = 'Disease'
    if fruits_disease[predict] == 'healthy':
        disease = ''
    cv2.putText(img, 'Fruit classify as ' + fruits_disease[predict] + " " + disease, (10,
30), cv2.FONT_HERSHEY_SIMPLEX,
                0.6, (0, 255, 255), 2)
    cv2.imshow("Classification Result : " + 'Fruit classified as ' +
fruits_disease[predict] + " " + disease, img)
```

```
    cv2.waitKey(0)

def close():
    main.destroy()

font = ('times', 16, 'bold')
title = Label(main, text='DETECTION AND CLASSIFICATION OF FRUIT DISEASES')
title.config(bg='Yellow', fg='Red')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0, y=5)

font1 = ('times', 12, 'bold')
text = Text(main,height=20,width=150)
scroll = Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=50, y=120)
text.config(font=font1)


font1 = ('times', 13, 'bold')
uploadButton = Button(main, text="Upload Fruits Dataset", command=uploadFruitDataset)
uploadButton.place(x=50, y=550)
uploadButton.config(font=font1)

processButton = Button(main, text="Image Preprocessing & KMEANS Segmentation", command=Preprocessing)
processButton.place(x=250, y=550)
processButton.config(font=font1)

featuresButton = Button(main, text="Features Extraction", command=featuresExtraction)
```

```
featuresButton.place(x=650, y=550)
featuresButton.config(font=font1)

svmButton = Button(main, text="Train SVM Classifier", command=svmClassifier)
svmButton.place(x=50, y=600)
svmButton.config(font=font1)

classifyButton = Button(main, text="Upload Test Image & Classification", command=Classification)
classifyButton.place(x=250, y=600)
classifyButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close)
exitButton.place(x=650, y=600)
exitButton.config(font=font1)

main.config(bg='deep sky blue')
main.mainloop()
```

4.2 Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant costs associated with a software failure are motivating factors for we planned, through testing. Testing is the process of executing a program with the intent of finding an error. The design of tests for software and other engineered products can be as challenging as the initial design of the product itself.

There of basically two types of testing approaches.

One is Black-Box testing – the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operated.

The other is White-Box testing – knowing the internal workings of the product ,tests can be conducted to ensure that the internal operation of the

product performs according to specifications and all internal components have been adequately exercised.

White box and Black box testing methods have been used to test this package. The entire loop constructs have been tested for their boundary and intermediate conditions. The test data was designed with a view to check for all the conditions and logical decisions. Error handling has been taken care of by the use of exception handlers.

4.2.1 Testing Strategies

Testing is a set of activities that can be planned in advanced and conducted systematically. A strategy for software testing must accommodation low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements.

Software testing is one element of verification and validation. Verification refers to the set of activities that ensure that software correctly implements as specific function. Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements.

The main objective of software is testing to uncover errors. To fulfill this objective, a series of test steps unit, integration, validation and system tests are planned and executed. Each test step is accomplished through a series of systematic test technique that assist in the design of test cases. With each testing step, the level of abstraction with which software is considered is broadened.

Testing is the only way to assure the quality of software and it is an umbrella activity rather than a separate phase. This is an activity to be performed in parallel with the software effort and one that consists of its own phases of analysis, design, implementation, execution and maintenance.

UNIT TESTING:

This testing method considers a module as single unit and checks the unit at interfaces and communicates with other modules rather than getting into details at statement level. Here the module will be treated as a black box, which will take some input and generate output. Outputs for a given set of input combination are pre-calculated and are generated by the module.

SYSTEM TESTING:

Here all the pre-tested individual modules will be assembled to create the larger system and tests are carried out at system level to make sure that all modules are working in synchronous with each other. This testing methodology helps in making sure that all modules which are running perfectly when checked individually are also running in cohesion with other modules. For this testing we create test cases to check all modules once and then generated test combinations of test paths through out the system to make sure that no path is making its way into chaos.

INTEGRATED TESTING

Testing is a major quality control measure employed during software development. Its basic function is to detect errors. Sub functions when combined may not produce than it is desired. Global data structures can represent the problems. Integrated testing is a systematic technique for constructing the program structure while conducting the tests. To uncover errors that are associated with interfacing the objective is to make unit test modules and built a program structure that has been detected by design. In a non - incremental integration all the modules are combined in advance and the program is tested as a whole. Here errors will appear in an endless loop function. In incremental testing the program is constructed and tested in small segments where the errors are isolated and corrected.

Different incremental integration strategies are top – down integration, bottom – up integration, regression testing.

REGRESSION TESTING

Each time a new module is added as a part of integration as the software changes. Regression testing is an actually that helps to ensure changes that do not introduce unintended behavior as additional errors.

Regression testing maybe conducted manually by executing a subset of all test cases or using automated capture play back tools enables the software engineer to capture the test case and results for subsequent playback and compression. The regression suit contains different classes of test cases.

A representative sample of tests that will exercise all software functions.

Additional tests that focus on software functions that are likely to be affected by the change.

4.3 INPUT SCREENSHOTS

To run project open command prompt to get below screen

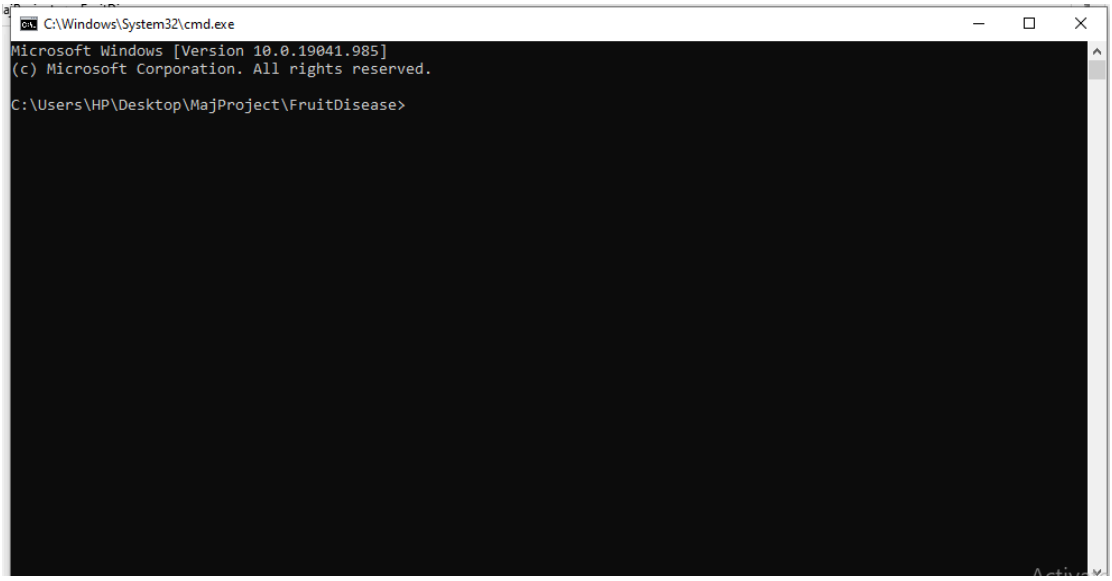

Figure 4.3.1: Command Prompt Screen

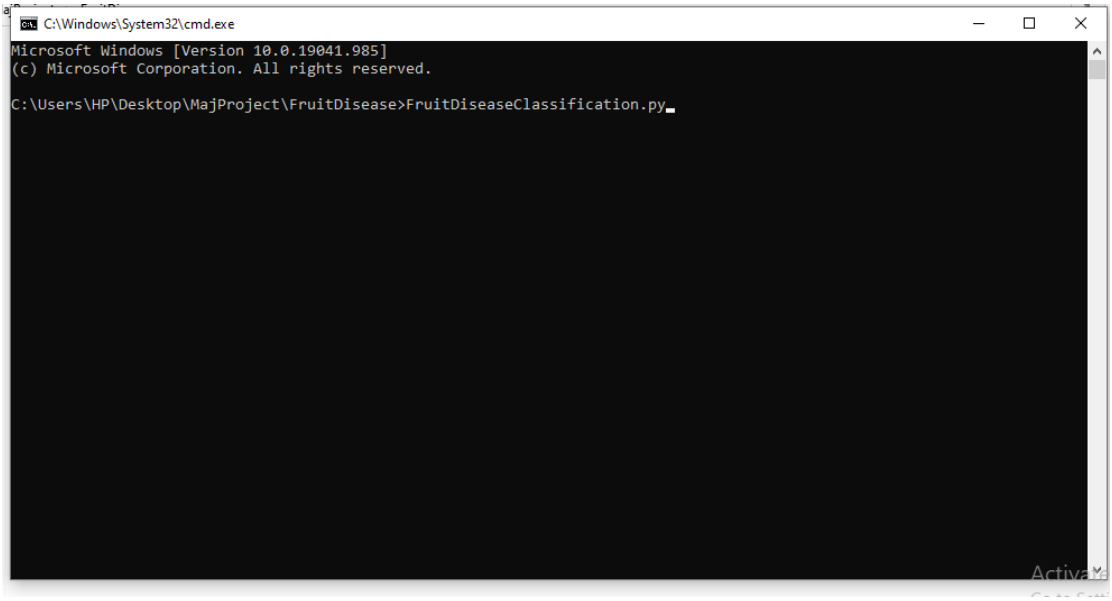

Fig 4.3.2: Running the fruit Disease Classification file

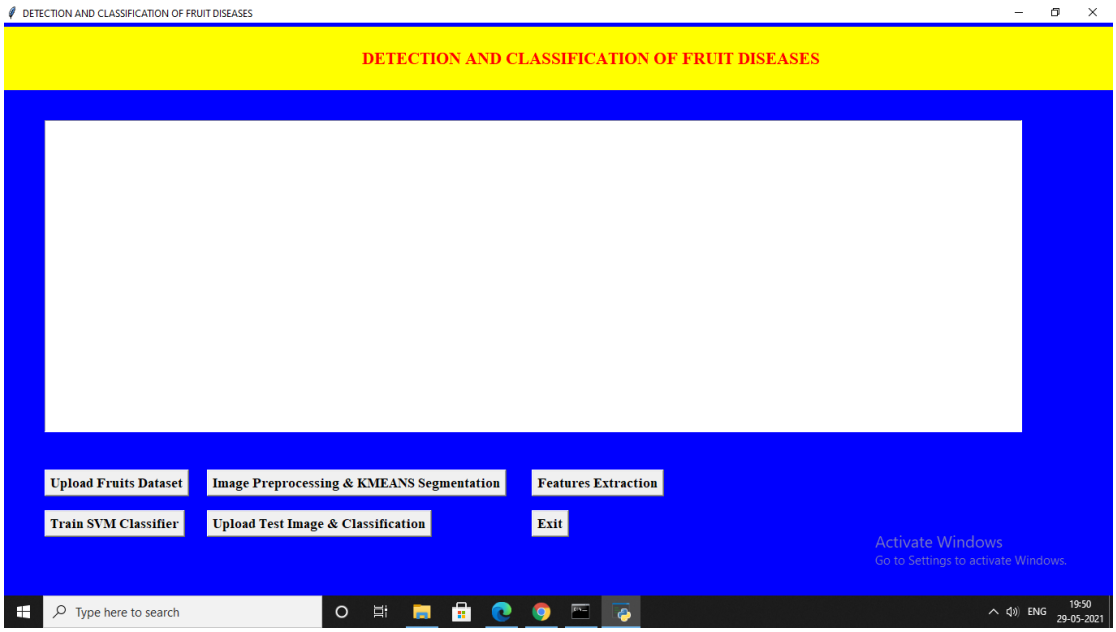

Figure 4.3.3: Tkinter GUI screen

4.4 OUTPUT SCREENSHOTS

Click on upload fruit dataset button to upload dataset

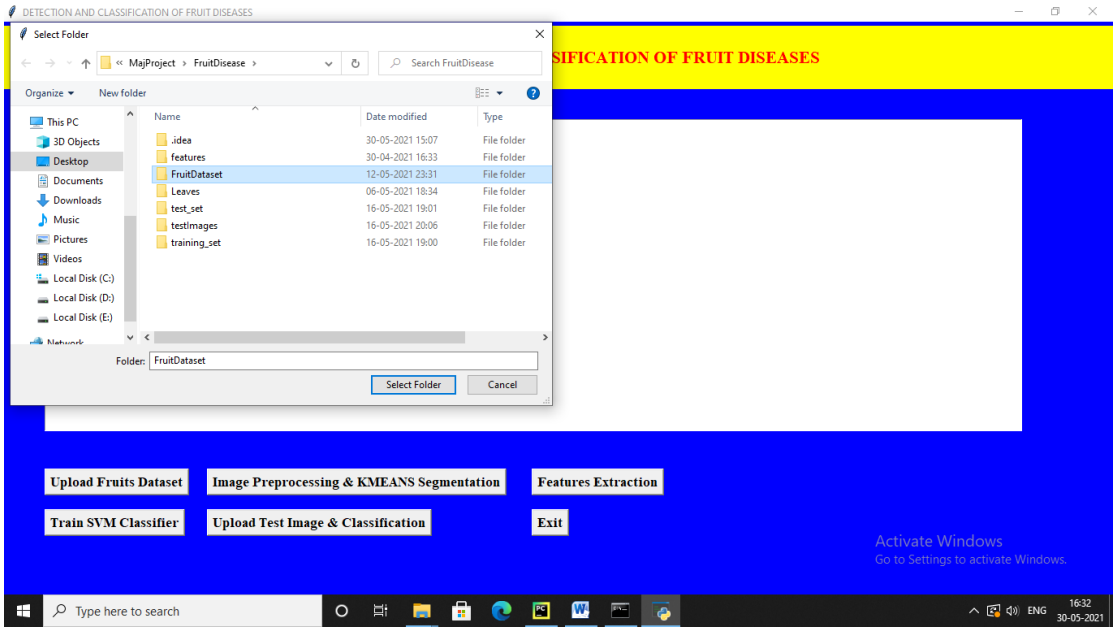

Figure 4.4.1: Upload fruit dataset

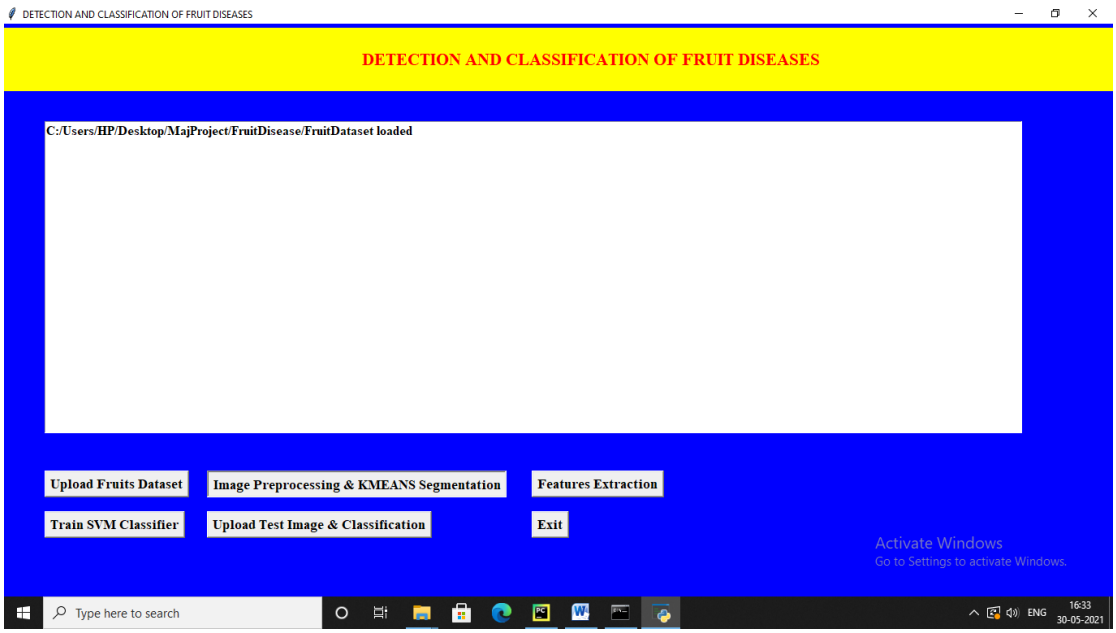

Figure 4.4.2: The fruit dataset is loaded

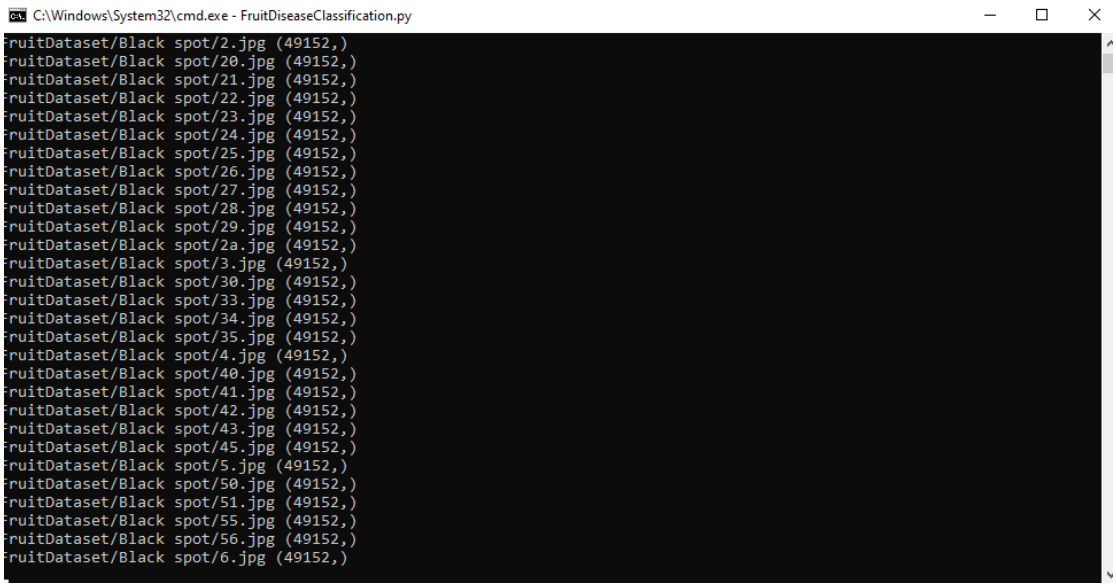

Figure 4.4.3: Images are preprocessing

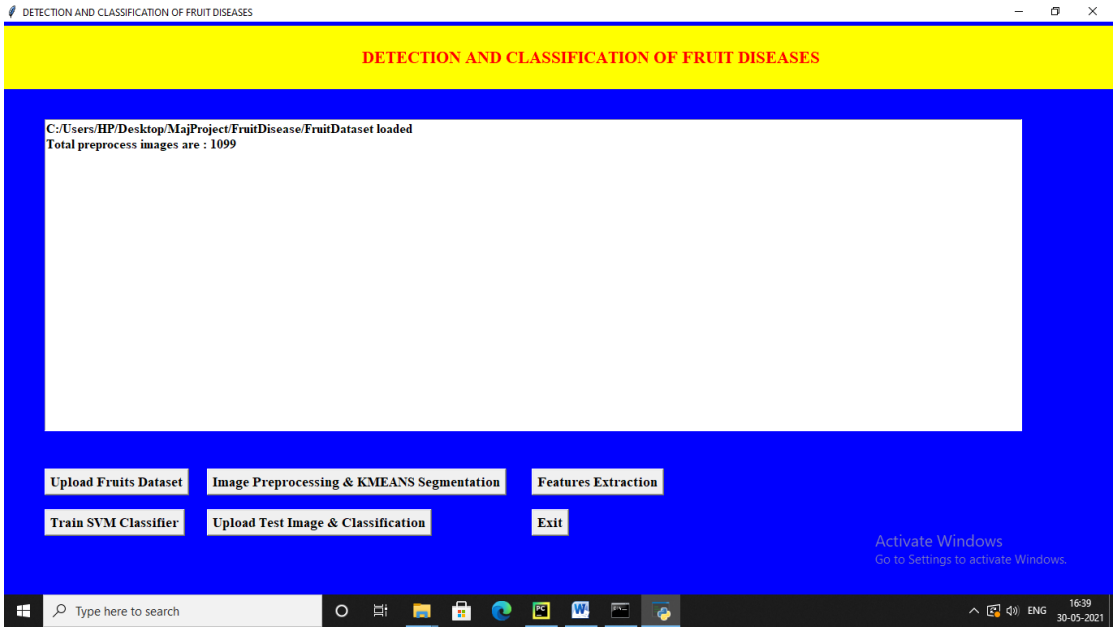

Figure 4.4.4: After Image Preprocessing & K-Means Segmentation

Image preprocessing and segment has been done to build train model and now click on ‘Features Extraction’ button.

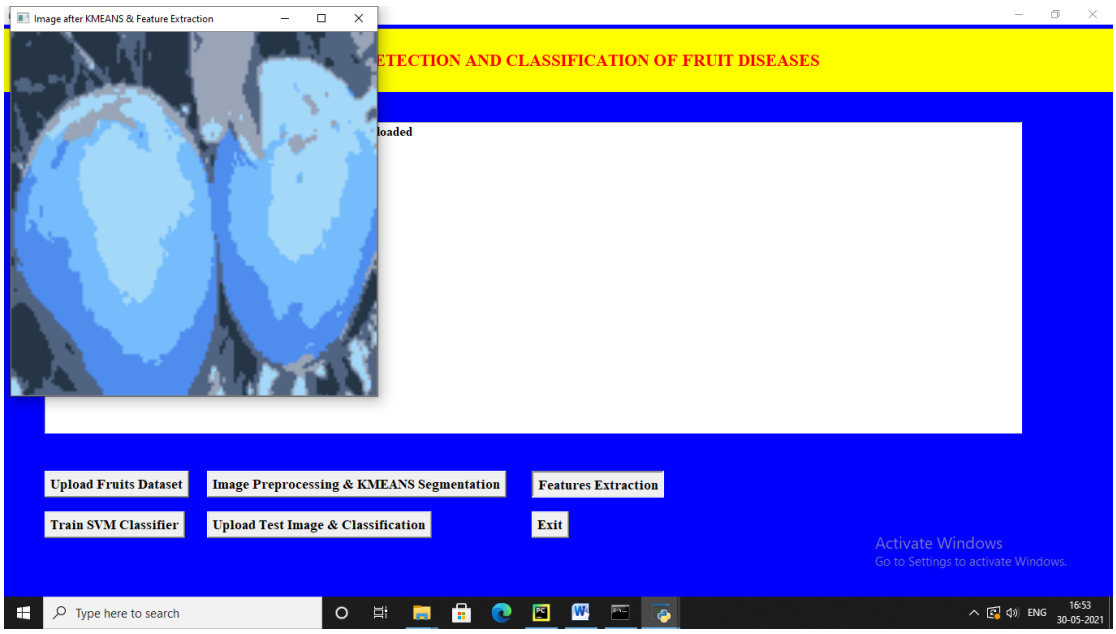

Figure 4.4.5: Feature extraction

In above screen, segmented image is displayed where you can see all similar colours pixels are in one place and now close above image to get train data and test details.

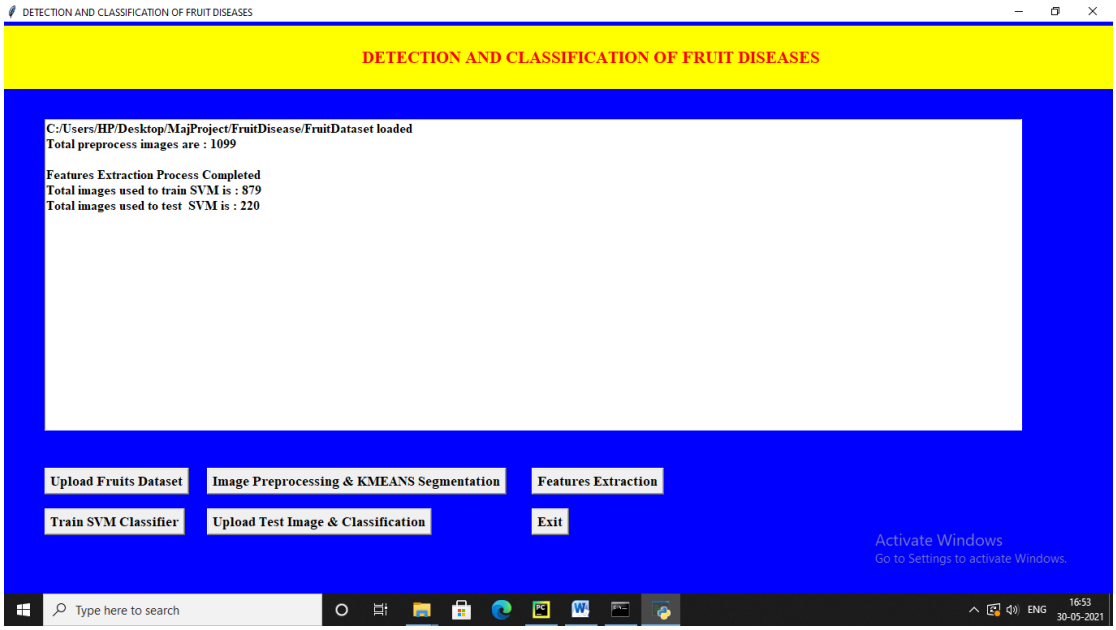

Figure 4.4.6: Train and Test Dataset

In above screen we can see that out of 1099 images application using 879(80%) images to train model and 220 (20%) images for testing.

Now click on ‘Train SVM Classifier’ button to train SVM with above train and test data.

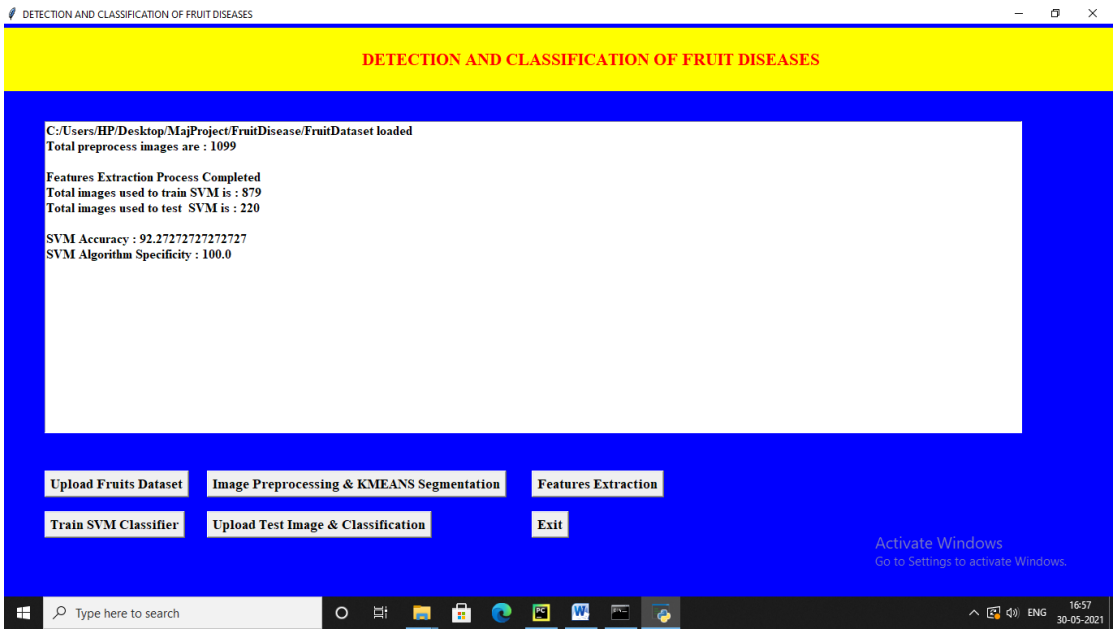

Figure 4.4.7: Display the accuracy

SVM classifier model is generated with prediction accuracy 92%.

Now click on ‘Upload Test Image & Classification’ button to upload test image and to predict the disease.

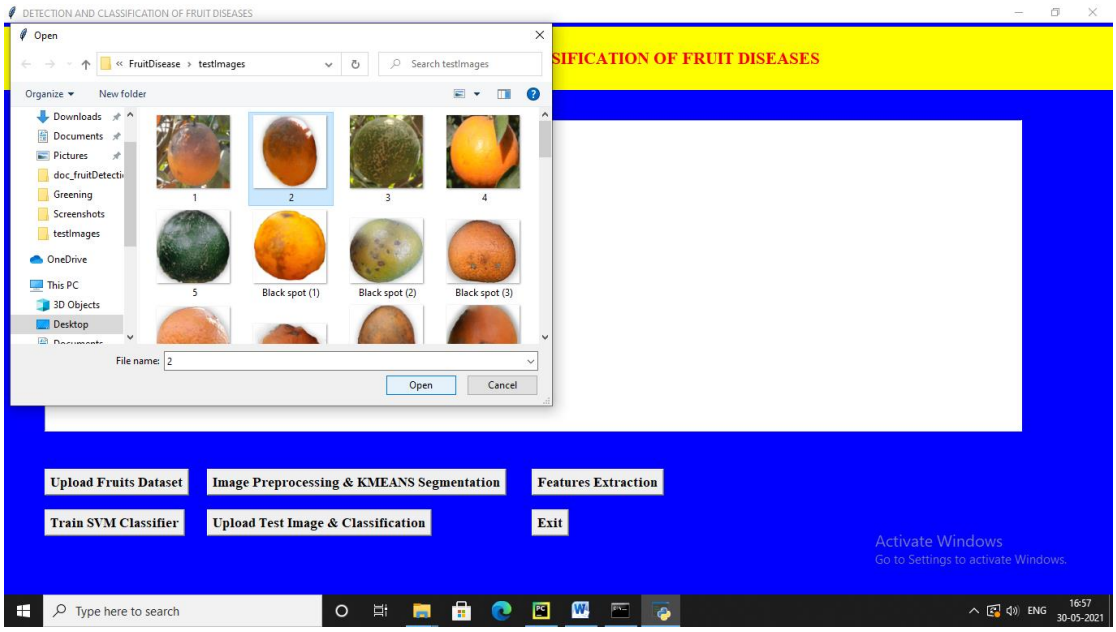

Figure 4.4.8: Upload the image for testing

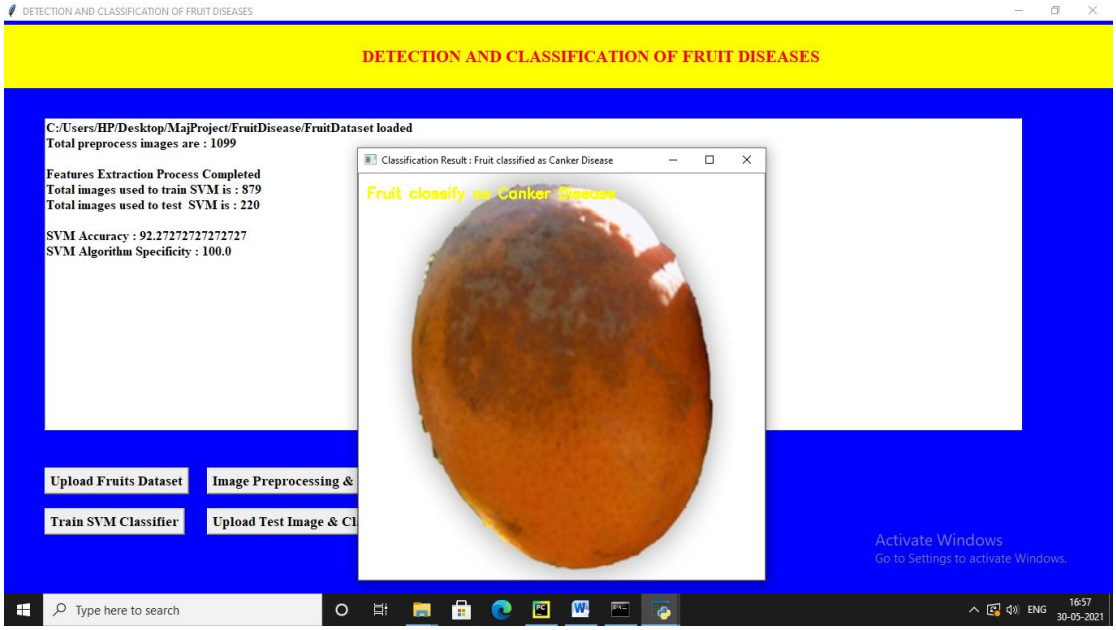

Fig 4.4.9: Test case showing fruit disease name.

5. CONCLUSION AND FUTURE SCOPE

The development of cloud based schemes for helping Indian farmers and agriculture, helps to analyze the agriculture data in a better way to reduce the hoardings and in bringing up a prosperous safe and peaceful farmer society in India. The classification and segmentation of fruit images were performed using K-Means Algorithm and SVM technique. The various features of a few fruits were initially extracted and segment the respective images. After comparison with feature values, the various disease names are analyzed and the optimal disease for the image is identified and the disease is displayed. In Future, store the data about the agricultural filed and details of farmers in database and recovering the data using Cloud computing. There are more fruit diseases which occur due to the surrounding conditions, mineral levels, insects in the farm area and other factors

6. REFERENCES

- https://ieeexplore.ieee.org/document/9104139
- https://www.irjet.net/archives/V4/i12/IRJET-V4I12213.pdf
- https://www.kaggle.com/mcbean/fruit-classification-w-nn
- https://data.mendeley.com/datasets/3f83gxmv57/2