GRAPE LEAF DISEASE IDENTIFICATION USING MACHINE LEARNING TECHNIQUES

A project report submitted to ANU, Guntur in partial fulfillment of the requirement for the award of

MASTER OF COMPUTER APPLICATIONS

Submitted By

JYESHTA DEVAKUMARI

Y23MC47034

Under the esteemed guidance of

Mr.G.RAGHU RAM

Asst.Professor

DEPARTMENT OF M.C.A.

K.CHANDRAKALA P.G. COLLEGE

(Affiliated to ANU, Guntur)

Burripalem Road, Tenali, Guntur – 522 201.

(Potti Sree Ramulu Educational Society's)

K.CHANDRAKALA P.G. COLLEGE

BURRIPALEM ROAD, TENALI – 522 201.

CERTIFICATE

This is to certify that the project work entitled "GRAPE LEAF DISEASE IDENTIFICATION USING MACHINE LEARNING TECHNIQUES" done JYESHTA DEVAKUMARI is bearing Regd.No. Y23MC47034 submitted in partial fulfillment of the requirements for the award of Master of Computer Applications is a bonafide work carried out by him / her under my guidance and supervision during theacademic year 2022-24.

Internal Guide **Head of the Department**

Mr.G.RAGHU RAM **Mrs.G.BHARATHI**

External Examiner

ACKNOWLEDGEMENT

I express my sincere thanks to **Mr.G.Vijaya Krishna**, **Principal** of **K.Chandrakala P.G.College, Tenali** for giving me opportunity to do this project.

My sincere thanks to **Mrs.G.BHARATHI, Head of the Department, Dept.of M.C.A., K.Chandrakala P.G.College, Tenali,** for providing excellent environment and encouragement in completing the project.

I acknowledge with thanks the valuable guidance of **Mr.G.RAGHU RAM, Asst.Professor, Dept.of M.C.A.**, for their kind co-operation in times need.

I am very thankful to the lab faculty who always readily solved the problems during development by giving their valuable suggestions. Their encouragement assisted me in making this project a success.

I express thanks to all the teaching and non-teaching staff members of **Dept.of M.C.A., K.Chandrakala P.G.College, Tenali.**

Last but not the least I also acknowledge with humble gratitude to my parents, my team members and my dearest friends who helped me to complete this project.

JYESHTA DEVAKUMARI

Y23MC47034

DECLARATION

I **JYESHTA DEVAKUMARI,** declare that this project report entitled as **"GRAPE LEAF DISEASE IDENTIFICATION USING MACHINE LEARNING TECHNIQUES"** has been prepared by me during the year **2022-24**, in partial fulfillment of the requirement, for the award of **Master of Computer Applications** in **K.Chandrakala P.G.College**, **Tenali** affiliated to **ANU, Guntur.**

I also declare that this project is the outcome of my own effort and has not been submitted toany other universities for the award of any other degree

Place: Tenali

Date:

JYESHTA DEVAKUMARI
Y23MC47034

ABSTRACT

Having diseases is quite natural in crops due to changing climatic and environmental conditions. Diseases affect the growth and produce of the crops and often difficult to control. To ensure good quality and high production, it is necessary to have accurate disease diagnosis and control actions to prevent them in time. Grape which is widely grown crop in India and it may be affected by different types of diseases on leaf, stem and fruit. Leaf diseases which are the early symptoms caused due to fungi, bacteria and virus. So, there is a need to have an automatic system that can be used to detect the type of diseases and to take appropriate actions. We have proposed an automatic system for detecting the diseases in the grape vines using image processing and machine learning technique. The system segments the leaf (Region of Interest) from the background image using grab cut segmentation method. From the segmented leaf part the diseased region is fruther segmented based on two different methods such as global thresholding and using semi-supervised technique. The features are extracted from the segmented diseased part and it has been classified as healthy, rot, esca,and leaf blight using different machine learning techniques such as Support Vector Machine (SVM), adaboost and Random Forest tree. Using SVM we have obtained a better testing accuracy of 93%.

INDEX

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

1.1 INTRODUCTION

Indian Economy is highly dependent on agricultural productivity of the country. Grape is very commercial fruit of India. It can easily be grown in all tropical, sub-tropical and temperate climatic regions. India has got different types of climate and soil in different parts of the country. This makes grapevines a major vegetative propagated crop with high socioeconomic importance. The grape plant will cause poor yield andgrowth when affected by diseases. The diseases are due to the viral, bacteria and fungi infections which are caused by insects, rust and nematodes etc., These diseases are judged by the farmers through their experience or with the help of experts through naked eye observationwhich is not accurate and time consuming process. Early detection of disease is then very much needed in the agriculture and horticulture field to increase the yield of the crops. We have proposed a system that can detect and identify diseases in the leaves of the grape plants.

1.2 MOTIVATION

Web enabled disease detection system have been proposed in [8]. The system proposed a segmentation method which has used mean based strategy for computing threshold and textual features were extracted and classification was done by SVM. The survey proposed by Vijai et al. in [12], discuses about different disease classification techniques used for plant leaf disease and used genetic algorithm for image segmentation. An integrated approach of particle swarm optimization and SVM for plant leaf disease detection and classification was proposed in [10].

1.3 EXISTING SYSTEM

The system proposed a segmentation method which has used mean based strategy for computing threshold and textual features were extracted and classification was done by SVM. The survey proposed by Vijai et al. in , discuses about different disease classification techniques used for plant leaf disease and used genetic algorithm for image segmentation. An integrated approach of particle swarm optimization and SVM for plant leaf disease detection and classification was proposed in. Disease detection system for

pomegranate leaves was proposed in which used colour-based segmentation and features like color, morphology and texture for classifying the leaves.

1.3.1 Limitations of existing system

Used for plant leaf disease and used genetic algorithm for imagesegmentation.

1.4 OBJECTIVES

The objective of project is to identify grape leaf disease.

1.5 APPLICATIONS

It can be used in agriculture.

1.6 SOFTWARE DEVELOPMENT LIFE CYCLE

What is SDLC?

SDLC stands for Software Development Life Cycle. A Software Development Life Cycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.

SDLC is the process consisting of a series of planned activities to develop or alter the software products.

Benefits of the SDLC Process

The intent of a SDLC process it to help produce a product that is cost-efficient, effective, and of high quality. Once an application is created, the SDLC maps the proper deployment and decommissioning of the software once it becomes a legacy. The SDLC methodology usually contains the following stages: Analysis (requirements and design), construction, testing, release, and maintenance (response). Veracode makes it possible to integrate automated security testing into the SDLC process through use of its cloud based platform.

STRUCTURE OF PROJECT (SYSTEM ANALYSIS)

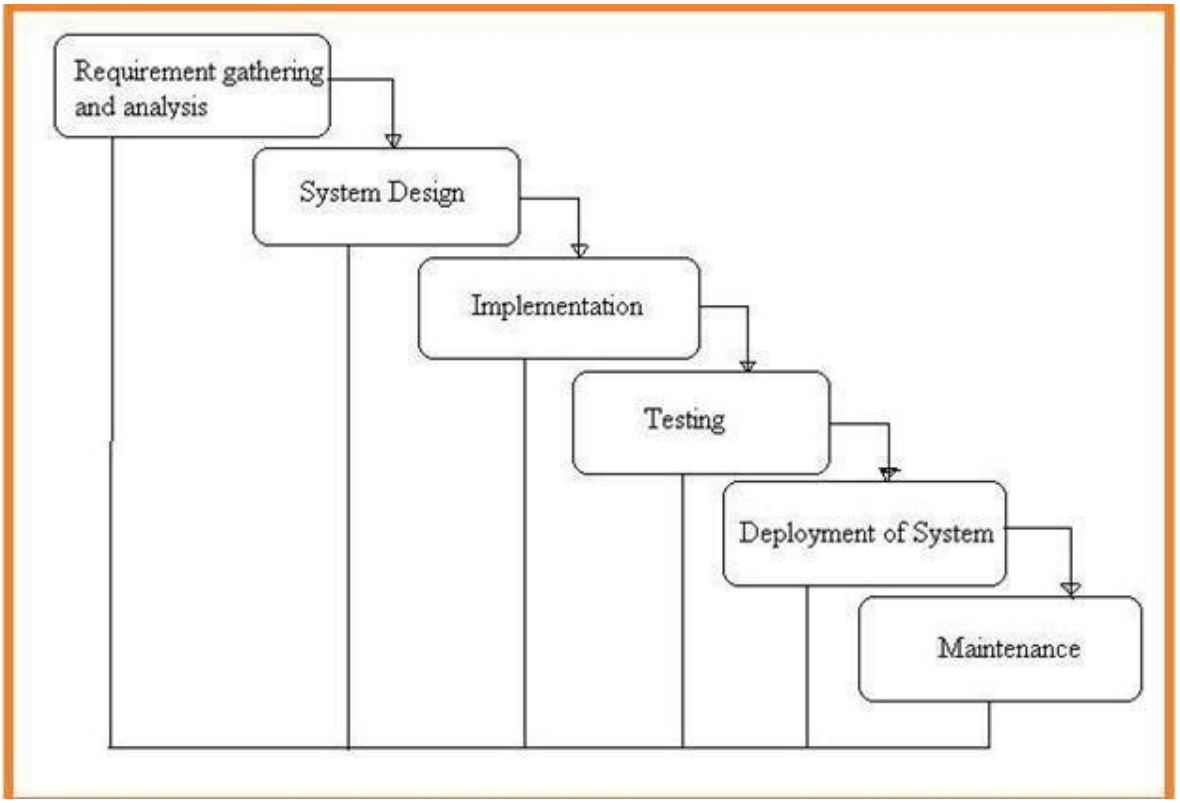


Fig: 1.6 Project SDLC

- Project Requisites Accumulating and Analysis
- Application System Design
- Practical Implementation
- Manual Testing of My Application
- Application Deployment of System
- Maintenance of the Project

1.7 REQUISITES ACCUMULATING ANDANALYSIS

It‟s the first and foremost stage of the any project as our is a an academic leave for requisites amassing we followed of IEEE Journals and Amassed so many IEEE Relegated papers and final culled a Paper designated “Individual web revisitation by setting and substance importance input andfor analysis stage we took referees from the paper and did literature surveyof some papers and amassed all the Requisites of the project in this stage.

CHAPTER-2

LITERATURE SURVEY

Web enabled disease detection system have been proposed in [8]. The system proposed a segmentation method which has used mean based strategy for computing threshold and textual features were extracted and classification was done by SVM. The survey proposed by Vijai et al. in [12], discuses about different disease classification techniques used for plant leaf disease and used genetic algorithm for image segmentation. An integrated approach of particle swarm optimization and SVM for plant leaf disease detection and classification was proposed in [10]. Disease detection system for pomegranate leaves was proposed in [5] which used colour-based segmentation and features like color, morphology and texturefor classifying the leaves. Agrawal et al. proposed a leaf detection and climatic parametric monitoring of plants using IOT in [1]. Neural Network based classification was proposed in [9] for detecting plant leaf diseases based on the texture features extracted using GLCM matrix. Mokhtar et al. proposed SVM based classification by extracting the texture based features in [6]. SVM classifier with different kernel functions including Cauchy kernel, Invmult Kernel and Laplacian Kernel were employed to evaluate the ability of the approach to detect and identify the infected tomato leaf. Leaf detection system for pomegranate leaves was proposed in [13] which uses K-means for segmentation and statistical features for classification using SVM. Sabrol et al. have proposed a system for leaf disease classification using decision tree by extracting different features after segmenting the leaf using ostuthresholding [4]. A system for two type of disease classification such as Downy mildew and Powdery mildew ingrape leaves was proposed in [11] using Back propogation Neural Network. A fast system was proposed for disease detection and classification using Neural Network after extracting the texture features using gray level co-occurrence methodology in [2]. A smartphone based system was developed by Mwebaze et al. in [7] using machinelearingtechnique to detect the state of the disease of the plant and also the severity levels of each diseases. Machine learning based techniques suchas decision tree, Navie Bayes theorem, Neural Network, K-Means and Random forest algorithms were proposed for leaf disease classification in using the features such as size, shape, dryness, wilting. Most of the work in the 978-1-5386-9471-8/19/$31.00 2019 IEEE Second International Conference on Computational Intelligence in Data Science (ICCIDS-2019) literature uses K-means segmentation for segmenting the leaf and extract low level features of the image to classify the plant leaf diseases. We have

proposed a system which uses global features to classifythe plant diseases and segmented the region of interest using graph cut method. We have also compared the results obtained using different machine learning techniques.

H.Wagner, T. Finkenzeller, S.Wurth, and S. P. Von Duvillard,``Individual and team performance in team-handball: A review,'' J. Sports Sci. Med., vol. 13, no. 4, p. 808, 2014.

Team handball is a complex sport game that is determined by the individual performance of each player as well as tactical components and interaction of the team. The aim of this review was to specify the elements of team-handball performance based on scientific studies and practical experience, and to convey perspectives for practical implication. Scientific studies were identified via data bases of PubMed, Web of Knowledge, SPORT Discus, Google Scholar, and Hercules. A total of 56 articles met the inclusion criteria. In addition, we supplemented the review with 13 additional articles, proceedings and book sections. It was found that the specific characteristics of team-handball with frequent intensity changes, team-handball techniques, hard body confrontations, mental skills and social factors specify the determinants of coordination, endurance, strength and cognition. Although we found comprehensive studies examining individual performance in team-handball players of different experience level, sex or age, there is a lack of studies, particularly for team-handball specific training, as well as cognition and social factors. Key PointsThe specific characteristics of team-handball with frequent intensity changes, specific skills, hard body confrontations, mental skills and social factors define the determinants of coordination, endurance, strength and cognition.To increase individual and team performance in team-handball specific training based on these determinants have been suggested.Although there are comprehensive studies examining individual performance in team- handball players of different experience level, sex, or age are published, there is a lack of training studies, particularly for team- handball specific techniques and endurance, as well as cognition and social factors.

B. Bideau, R. Kulpa, N. Vignais, S. Brault, F. Multon, and C. Craig, ``Using virtual reality to analyze sports performance,'' IEEE Comput. Graph. Appl., vol. 30, no. 2, pp. 14 21, Mar./Dec. 2010.

Improving performance in sports can be difficult because many biomechanical, physiological, and psychological factors come into play during competition. A better understanding of the perception- action loop employed by athletes is necessary. This requires isolating contributing factors to determine their role in player performance. Because of its inherent limitations, video playback doesn't permit such in-depth analysis. Interactive, immersive virtual reality (VR) can overcome these limitations and foster a better understanding of sports performance from a behavioral-neuroscience perspective. Two case studies using VRIn this paper an automated system has been developed to determine whether the plant is normal or diseased. The normal growth of the plants, yield and quality of agricultural products is seriously affected by plant disease. This paper attempts to develop an automated system that detects the presence of disease in the plants. An automated disease detection system is developed using sensors like temperature, humidity and colour based on variation in plant leaf health condition. The values based on temperature, humidity and colour parameters are used to identify presence ofplant disease. 1. INTRODUCTION India is a land of agriculture. Two-third of population relies upon agriculture for their livelihood. It is the basic foundation of economic development of the country. The agriculture also provides employment opportunities to very large percentage of population. Plant health condition plays a vital role to earn good profit for the farmers. Proper monitoring of plant health is required at different stages of plant growth in order to prevent disease affecting plants. Existence of pests and disease affect the estimation of crop cultivation and minimizes crop yield substantially. Present day system depends on naked eye observation which is a time consuming process. Automatic detection of plant disease can be adopted to detect plant disease at early stages. Various disease management strategies have been used by farmers at regular intervals in order to prevent plant diseases. Some of the sample images of disease affecting plant leaves are shown in the Figure.1. Figure.1: Disease affecting on plant leaves The Internet of Things (IoT) is the connectivity of physical devices, vehicles, home appliances, and other items embedded with electronics, software, sensors, actuators, and network which allow these things to attach and interchange data, generating opportunities for more direct merging of the physical world into computer-based systems, resulting in reduced human intervention. By the year 2050 the global population is set to reach 9.6 billion. So, to nourish this much population, the farming industry must adopt IoT technology. The demand for more food has to be met against the challenges such as intense weather conditions and exhaustive farming practices. Smart farming based on IoT technologies enhances crop production in

farming industry. Detection of diseases in the plant is utmost need for farmers and agricultural experts. The main aim of the proposed system is to detect plant diseases using IoT. In most of the plants the disease inception takes place on plant leaves. Hence, in the proposed work we have considered detection of plant disease present on leaves. The discrimination of normal and affected plant leaf can be measured based on variation in temperature, humidity and colour. The following papers have been cited during the literature survey to understand the different applications of computer systems in allied areas of the present work carried out. (Mark Seelye et al., 2011) have presented low cost colour sensors for monitoring plant growth in a laboratory. An automated system for measuring plant leaf colour is developed to check plant health status. (Sushma R. Huddar et al., 2012) have presented novel algorithm for segmentation and automatic identification of pests on plants using image processing. The proposed methodology involves reduced computational complexity and aims at pest detection not only in a greenhouse environment but also in a farm environment as well. The whitefly, a bio-aggressor which poses a threat to a multitude of crops, was chosen as the pest of interest in this paper. The algorithm was tested for several whiteflies affecting different leaves and an accuracy of 96% of whitefly detection was achieved. (Murali Krishnan and Jabert.G, 2013) have presented pest control in agricultural plantations using image processing techniques in MATLAB. Images are then subjected to pre-processing, transformation and clustering. (Prof. S. G. Galande, et al., 2015) have presented IoT Implementation for wireless monitoring of agricultural parameters. Wireless system is developed to monitor environmental conditions in agriculture field like temperature, soil pH, soil wet level and humidness beside leaf diseases detection. (Yun Shi et al., 2015) have presented IoT application to monitoring plant diseases and insect pests. IoT technology to percept information, and the role of the IOT technology in agricultural disease and insect pest control, which includes agricultural disease and insect pest monitoring system, collecting disease and insect pest information using sensor nodes, data processing and mining, etc have been described in this paper. (NimishGopal, 2016) have presented micro- controller based auto-irrigation and pest detection using Research Article Volume 8 Issue No.9 International Journal of Engineering Science and Computing, September 2018 18903 http://ijesc.org/ image processing. The technique of image analysis is extensively applied to agriculture science to provide maximum protection to crops which can ultimately lead to better crop management and production. (S. Gavaskar and A. Sumithra, 2017) have presented design and development of pest monitoring system for implementing precision agriculture using IoT. India‟s most

of the farmer grow sugarcane but did not get yielding due to bugs and larvae in sugarcane. In this proposed design system used arduino for monitoring the noise and temperature. (SaiVivek et al., 2017) have presented arduino based pest control using real time environmental monitoring sensors. This paper strives to develop a robot capable of performing operation of dispensing pest control agents, obstacle avoidance for self-guidance on the field without any user interference and create a sterile environment for the optimum growth of the crops in a real time monitored closed environment. (Oliver Schmittmann et al., 2017) have presented a true-color sensor and suitable evaluation algorithm for plant recognition. The system developed is based on free cascadable and programmable true-colour sensors for realtime recognition and identification of individual weed and crop plants using mathematical algorithms and decision models. (Zhang Chuanlei et al., 2017) have presented apple leaf disease identification using genetic algorithm and correlation based feature selection method. A color transformation structure for the input RGB (Red, Green and Blue) image was designed firstly and then RGB model was converted to HSI (Hue, Saturation and Intensity), YUV and gray models. The background was removed and then the disease spot image was segmented with region growing algorithm (RGA). Finally, the diseases were recognized by SVM classifier. (K.Lakshmi and S.Gayatri, 2017) have presented implementation of IoT with image processing in plant growth monitoring system. This work combines image processing and IoTto monitor the plant and to collect the environmental factors such as humidity and temperature. Plant diseases seriously affect the normal growth of plants, the yield and quality of agricultural products. In recent years, with the dramatic changes in climate, the natural environment of the plant growth has been damaged by pollution, frequent natural disasters, as well as the development of agricultural production. From the literature survey presentedabove, it is observed that the work on plant disease detection using IoT reported in the literature is scarce. In the present work, this issue is addressed using sensor based technology. This being the motivation, the problem entitled “Leaf Disease Detection using IoT is proposed to assist the farmers technologically. In the proposed work, focus has been on early detection of disease infection on plant leaves. The paper is organized into four sections. Section.1 gives the proposed methodology. Section 3 describes results and discussions. Section 4 gives conclusions of the work. 2. PROPOSED METHODOLOGY The proposed system consists of temperature, humidity, and color sensors for collecting data from plant leaves based on variation in temperature, humidity and color of plant leaves. The data collected from the leaves consists of current environmental factors like

temperature, humidity and color. The changes that a plant undergoes are captured by the temperature humidity and color sensors and analyzed with the Arduino software. The data collected from temperature, humidity and color sensors are given to Arduino UNO kit from which the information is communicated to the farmers. The system makes use of WiFi shield in order to send the data from the host system to the cloud platform for analysis. The cloud platform that we have used is the www.thingspeak.com. The collected data in the cloud platform is then compared with the dataset in order to detect whether the leaf under consideration is normal or affected. The Figure.2 shows schematic diagram of the proposed work. Data acquisition: Here we take samples of different leaves as the input. These leaves are then sensed by the sensors to determine different parameters based on which it is recognized to be healthy or diseased. Temperature sensors: The DHT11 isa basic, ultra lowcost digital temperature sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). We use the DHT11 to sense the temperature on the surface of leaf to determine whether it is healthy or diseased. Humidity sensor: The DHT11 is a basic, ultra low-cost digital humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal onthe data pin (no analog input pins needed). We use the DHT11 to sense the humidity on the surface of leaf to determine whether it is healthy or diseased. Colour Sensor: The TCS3200 is a programmable color light-to- frequency converter/sensor. The sensor is a single monolithic CMOS integrated circuit that combines a configurable silicon photodiode and a current-tofrequency converter. The output is a square wave (50% duty cycle) with frequency directly proportional to light intensity (irradiance). We use the DHT11 to sense the colour of leaf to determine whether it is healthy or diseased. Aurdino: The Arduino UNO is a widely used opensource microcontroller board based on the ATmega328P microcontroller and developed by Arduino.cc. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits.] The board features 14 Digital pins and 6 Analog pins. It is programmable with the Arduino IDE (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts.Here we are using the Arduino in order to processthe data that is being collected from the sensor through the Arduino IDE.

Stages to predict VO2max in adults,'' *Can. J. Sport Sci. J. Canadien Sci. Sport*, vol. 14, no. 1, pp. 21_26, 1989..

Direct estimation of cardiorespiratory fitness in terms of VO2max is restricted within well equipped laboratory. Eighty four (84) sedentary male university students (Age 22.77±1.73 years, Body height 167.73±4.07 cm and Body mass 58.25±4.02 kg) of same socio-economic background were recruited from students of University of Calcutta, Kolkata, India to validate the applicability of 20 meter shuttle run test (SRT) for indirect estimation of VO2max in young male sedentary university students of Kolkata, India. They were further asigned to “study group” (N=54) on which the existing experimental protocol of SRT was tested and “confirmatory group” (N=30) on which the modified equations were validated. VO2max of each participant was determined by direct procedure and indirect SRT method with a gap of four days in between the tests. The difference between the mean values of directly measured VO2max and indirectlypredicted VO2max (PVO2max) in the “study group” was statistically significant (P3. PROBLEM IDENTIFICATION &OBJECTIVE

CHAPTER-3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

The system proposed a segmentation method which has used mean based strategy for computing threshold and textual features were extracted and classification was done by SVM. The survey proposed by Vijaietal. in, discuses about different disease classification techniques used for plant leaf disease and used genetic algorithm for image segmentation. An integrated approach of particle swarm optimization and SVM for plant leaf disease detection and classification was proposed in. Disease detection system for pomegranate leaves was proposed in whichused colour-based segmentation and features like color, morphology and texture for classifying the leaves.

Drawbacks

Used for plant leaf disease and used genetic algorithm for imagesegmentation.

3.2 PROPOSED SYSTEM

We have proposed an automated disease detection and classification system for grape leaves using traditional image processing and machine learning techniques. The proposed system first segments the ROI from the back ground using grab cut algorithm and classify the segmented leaves as healthy, balck-rot, esca and leaf blight. Figure. 1 depicts different types of disease in grape leaves. These diseases are caused due to fungi infection on the leaves. Each disease have differentcharacteristics where black rot appears to be circular in shape and has dark margins, esca appears as dark red stripes and leaf blight appears to be solid reddish-purple spots. The proposed system consists of five different process such as image preprocessing, image segmentation, feature extraction, disease detection and identification Image Preprocessing, Image Segmentation.

Advantages

System consists of five different process such as image preprocessing, image segmentation, feature extraction, disease detection and identification.

3.3 FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Economical Feasibility
- Operational Feasibility
- Technical Feasibility

ECONOMIC FEASIBILITY

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

OPERATIONAL FEASIBILITY

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization‟s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?
- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of „Secure Infrastructure Implementation System". The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database"s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source. The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing a fast feedback to the users irrespective of the number of users using the system.

CHAPTER-4

SYSTEM REQUIREMENTS SPECIFICATION

4.1 FUNCTIONAL REQUIREMENTS

In software engineering and systems engineering, a **functional requirement** defines a function of a system or its component, where a function is described as a specification of behavior between outputs and inputs.[1]

Functional requirements may involve calculations, technical details, data manipulation and processing, and other specific functionality that define what a system is supposed to accomplish.[2] Behavioral requirements describe all the cases where the system uses the functional requirements, these are captured in use cases. Functional requirements are supported by non-functional requirements (also known as "quality requirements"), which impose constraints on the design or implementation (such as performance requirements, security, or reliability). Generally, functional requirements are expressed in the form "system must do <requirement>," while non-functional requirements take the form "system shall be <requirement>." The plan for implementing functional requirements is detailed in the system design, whereas *non-functional* requirements are detailed in the system architecture.[4][5]

As defined in requirements engineering, functional requirements specify particular results of a system. This should be contrasted with non-functional requirements, which specify overall characteristics such as cost and reliability. Functional requirements drive the application architecture of a system, while non-functional requirements drive the technical architecture of a system.[4]

In some cases a requirements analyst generates use cases after gathering and validating a set of functional requirements. The hierarchy of functional requirements collection and change, broadly speaking, is: user/stakeholder request → analyze → use case → incorporate. Stakeholders make a request; systems engineers attempt to discuss, observe, and understand the aspects of the requirement; use cases, entity relationship diagrams, and other models are built to validate the requirement; and, if documented and approved, the requirement is implemented/incorporated.[6] Each use case illustrates behavioral scenarios through one or more functional requirements. Often, though, an analyst will begin by eliciting a set of use

cases, from which the analyst can derive the functional requirements that must be implemented to allow a user to perform each use case.

4.2 NON-FUNCTIONAL REQUIREMENT

NON-FUNCTIONAL REQUIREMENT (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, *"how fast does the website load?"* Failing to meet non-functional requirements can result in systems that fail to satisfy user needs.

Non-functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are > 10000. Description of non-functional requirements is just as critical as a functional requirement.

- Usability requirement
- Serviceability requirement
- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement

- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

Advantages of Non-Functional Requirement

Benefits/pros of Non-functional testing are:

- The nonfunctional requirements ensure the software system follow legal and compliance rules.
- They ensure the reliability, availability, and performance of the software system
- They ensure good user experience and ease of operating the software.
- They help in formulating security policy of the software system.

Disadvantages of Non-functional requirement

Cons/drawbacks of Non-function requirement are:

- None functional requirement may affect the various high-level software subsystem
- They require special consideration during the software architecture/high-level design phase which increases costs.
- Their implementation does not usually map to the specific software sub-system,
- It is tough to modify non-functional once you pass the architecture phase.

KEY LEARNING

- A non-functional requirement defines the performance attribute of a software system.
- Types of Non-functional requirement are Scalability Capacity, Availability, Reliability, Recoverability, Data Integrity, etc.
- Example of Non Functional Requirement is Employees never allowed to update their salary information. Such attempt should be reported to the security administrator.

- Functional Requirement is a verb while Non-Functional Requirement is an attribute
- The advantage of Non-functional requirement is that it helps you to ensure good user experience and ease of operating the software

The biggest disadvantage of Non-functional requirement is that it may affect the various high-level software subsystems.

CHAPTER-5

SYSTEM DESIGN

5.1 SYSTEM SPECIFICATIONS

Hardware Requirements:

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python / Canopy / VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, whereas applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

- System : Pentium i3
- Hard Disk : 500 GB.
- Monitor : 14‟ Colour Monitor.
- Mouse : Optical Mouse.
- Ram : 4 GB.

Software Requirements:

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation. The appropriation of requirements and implementation constraints gives the general overview of the project in regards to what the areas of strength and deficit are and how to tackle them.

- Operating system : Windows 8/10.
- Coding Language : PYTHON
- Software : Jupyter

5.2 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

5.2.1 USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

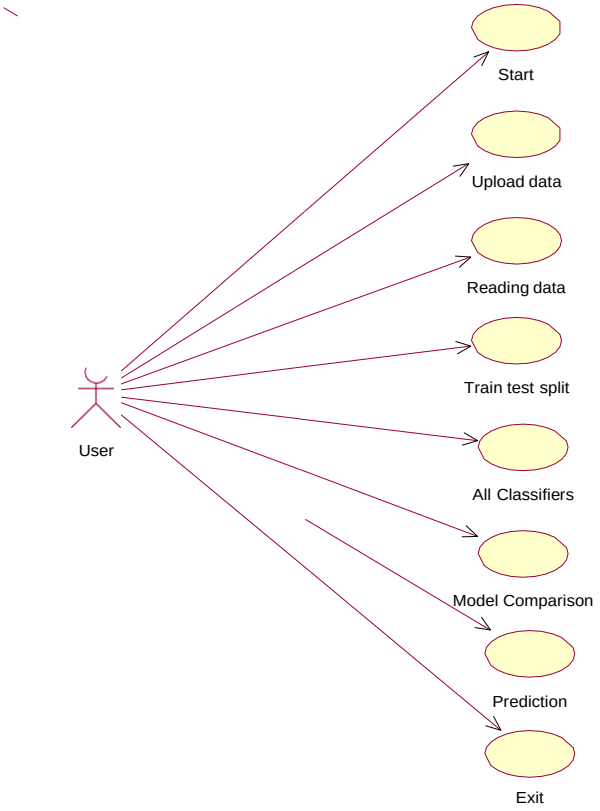


Fig 5.2.1 Use Case Diagram

5.2.2 CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

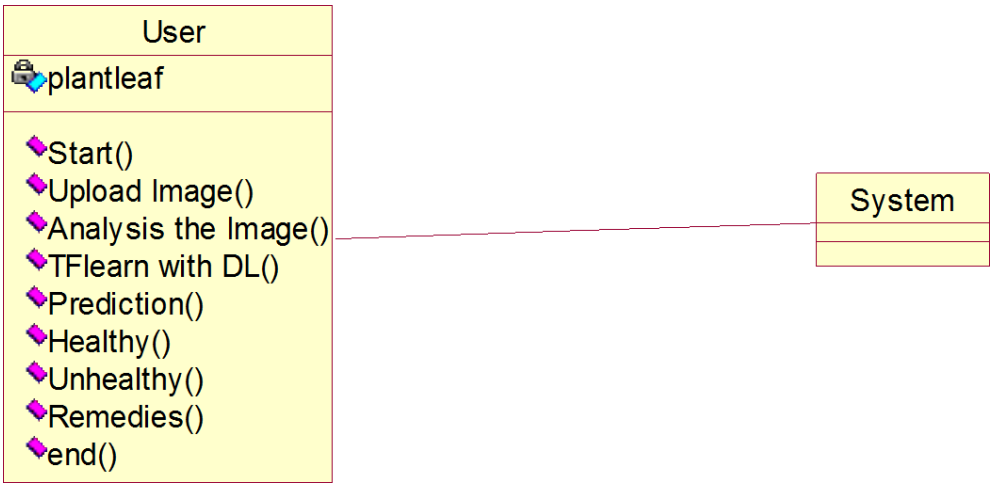


Fig 5.2.2 Class Diagram

5.2.3 SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

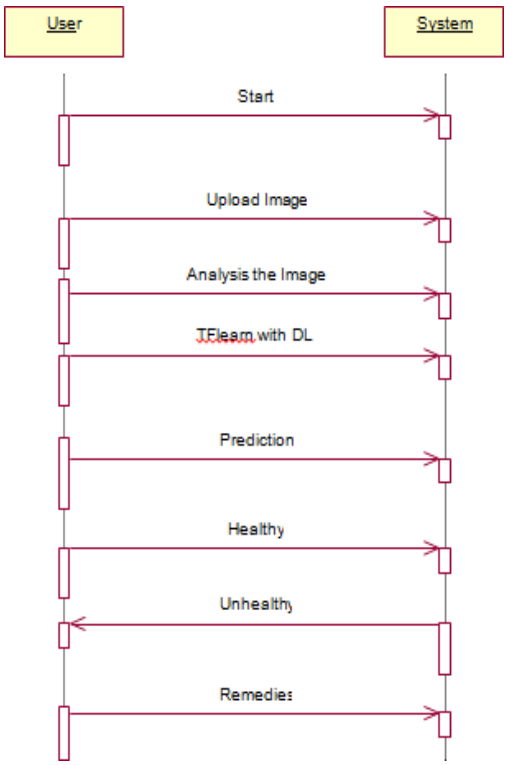


Fig 5.2.3 Sequence Diagram

5.2.4 FLOW CHART

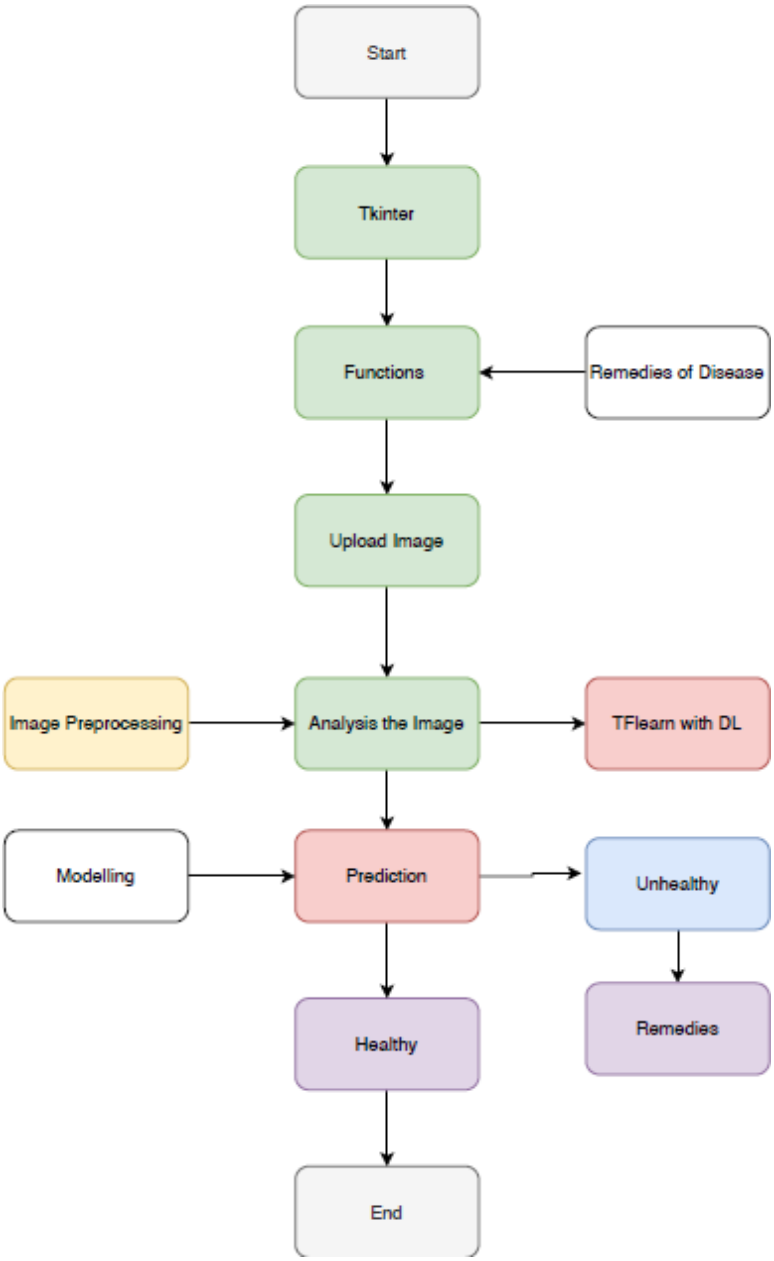


Fig 5.2.4 Flow chart

CHAPTER-6

SYSTEM IMPLEMENTATION

6.1 MODULES

1) **Data Collection:** Collect sufficient data samples and legitimate software samples.

2) **Feature Extraction**: For each image extract the features ubsing image processing and save in „.csv‟ extension

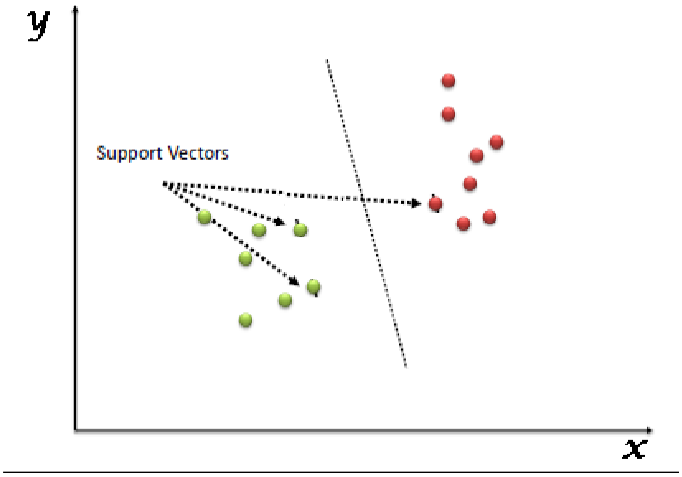
3) **Train and Test Modelling:** Split the data into train and test data Train will be used for trainging the model and Test data to check the performace.

4) **Modelling:** SVM Navie bayes, Random FOrest, KNN, Ada boost, Decision tree, Ada boost with random forest. Combine the training using machine learning algorithms and establish a classification model.

6.2 ALGORITHMS

Support Vector Machine

“Support Vector Machine” (SVM) is a supervised machine learning algorithm which can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well (look at the below snapshot).



Support Vectors are simply the co-ordinates of individual observation. The SVM classifier is a frontier which best segregates the two classes (hyper-plane/ line).

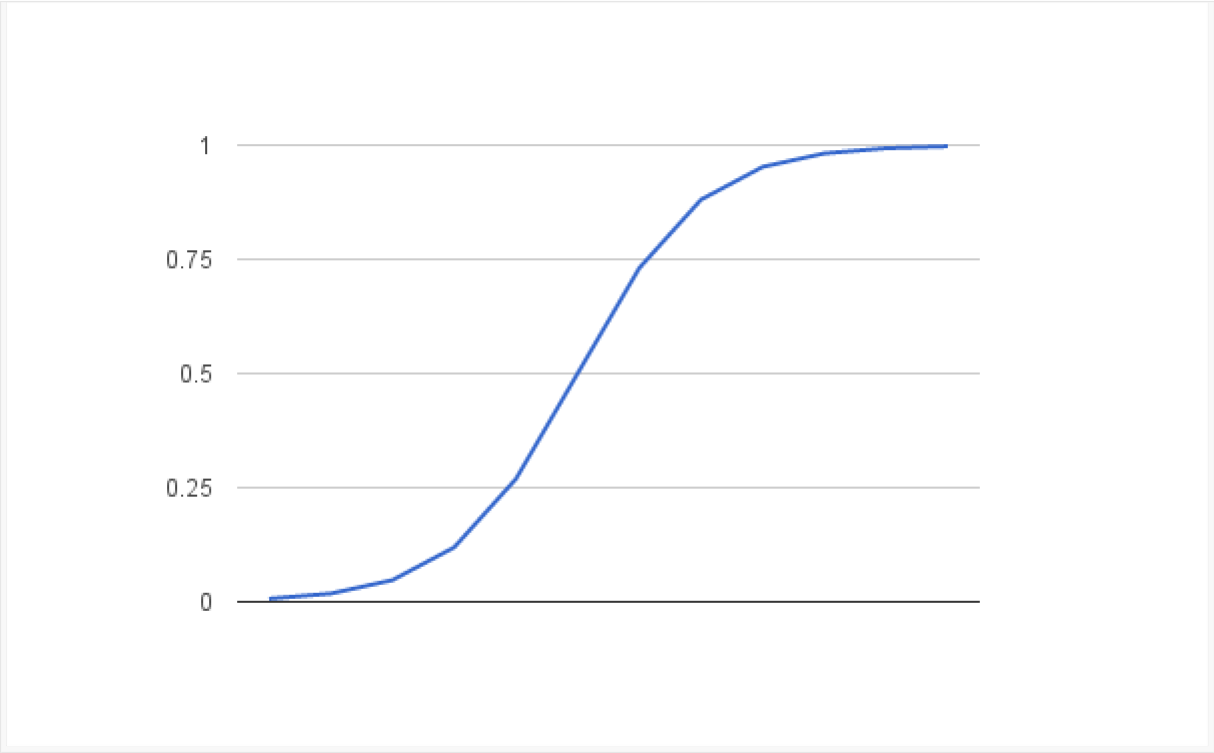
Logistic regression

Logistic regression is named for the function used at the core of the method, the logistic function.

The logistic function, also called the sigmoid function was developed by statisticians to describe properties of population growth in ecology, rising quickly and maxing out at the carrying capacity of the environment. It‟s an S-shaped curve that can take any real-valued number and map it into a value between 0 and 1, but never exactly at those limits.

1 / (1 + e^-value)

Where e is the base of the natural logarithms (Euler‟s number or the EXP() function in your spreadsheet) and value is the actual numerical value that you want to transform. Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.



Logistic Function

Now that we know what the logistic function is, let‟s see how it is used in logistic regression.

Naive Bayes Classifiers

Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or categorical input values.

It is called *naive Bayes* or *idiot Bayes* because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable. Rather than attempting to calculate the values of each attribute value P(d1, d2, d3|h), they are assumed to be conditionally independent given the target value and calculated as P(d1|h) * P(d2|H) and so on.

This is a very strong assumption that is most unlikely in real data, i.e. that the attributes do not interact. Nevertheless, the approach performs surprisingly well on data where this assumption does not hold.

Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning,** which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model.*

As the name suggests, **"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

Decision tree Algorithm:

Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute as shown in the above figure. This process is then repeated for the subtree rooted at the new node.

AdaBoost Classifier

Ada-boost or Adaptive Boosting is one of ensemble boosting classifier proposed by Yoav Freund and Robert Schapire in 1996. It combines multiple classifiers to increase the accuracy of classifiers. AdaBoost is an iterative ensemble method. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set. Adaboost should meet two conditions:

1. The classifier should be trained interactively on various weighed training examples.
2. In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

6.3 SAMPLE CODE

```
from tkinter import messagebox

from tkinter import *

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

from tkinter.filedialog import  askopenfilename

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

import numpy as np

import pandas as pd

from genetic_selection import GeneticSelectionCV

from sklearn.metrics import classification_report
```

```
from sklearn.metrics import confusion_matrix
from sklearn import svm
from keras.models import Sequential
from keras.layers import Dense
import time
main = tkinter.Tk()
main.title("Android Malware Detection")
main.geometry("1300x1200")
global filename
global train
global svm_acc, nn_acc, svmga_acc, annga_acc
global X_train, X_test, y_train, y_test
global svmga_classifier
global nnga_classifier
global svm_time,svmga_time,nn_time,nnga_time
def upload():
    global filename
    filename = filedialog.askopenfilename(initialdir="dataset")
    pathlabel.config(text=filename)
    text.delete('1.0', END)
    text.insert(END,filename+" loaded\n");
def generateModel():
    global X_train, X_test, y_train, y_test
```

```
    text.delete('1.0', END)
    train = pd.read_csv(filename)
    rows = train.shape[0] # gives number of row count
    cols = train.shape[1] # gives number of col count
    features = cols - 1
    print(features)
    X = train.values[:, 0:features]
    Y = train.values[:, features]
    print(Y)
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.2,
random_state = 0)
    text.insert(END,"Dataset Length : "+str(len(X))+"\n");
    text.insert(END,"Splitted Training Length : "+str(len(X_train))+"\n");
    text.insert(END,"Splitted Test Length : "+str(len(X_test))+"\n\n");
def prediction(X_test, cls): #prediction done here
    y_pred = cls.predict(X_test)
    for i in range(len(X_test)):
      print("X=%s, Predicted=%s" % (X_test[i], y_pred[i]))
    return y_pred
# Function to calculate accuracy
def cal_accuracy(y_test, y_pred, details):
    cm = confusion_matrix(y_test, y_pred)
    accuracy = accuracy_score(y_test,y_pred)*100
```

```
    text.insert(END,details+"\n\n")
    text.insert(END,"Accuracy : "+str(accuracy)+"\n\n")
    text.insert(END,"Report : "+str(classification_report(y_test, y_pred))+"\n")
    text.insert(END,"Confusion Matrix : "+str(cm)+"\n\n\n\n\n")
    return accuracy
def runSVM():
    global svm_acc
    global svm_time
    start_time = time.time()
    text.delete('1.0', END)
    cls = svm.SVC(C=2.0,gamma='scale',kernel = 'rbf', random_state = 2)
    cls.fit(X_train, y_train)
    prediction_data = prediction(X_test, cls)
    svm_acc = cal_accuracy(y_test, prediction_data,'SVM Accuracy')
    svm_time = (time.time() - start_time)
def runSVMGenetic():
    text.delete('1.0', END)
    global svmga_acc
    global svmga_classifier
    global svmga_time
    estimator = svm.SVC(C=2.0,gamma='scale',kernel = 'rbf', random_state = 2)
    svmga_classifier = GeneticSelectionCV(estimator,
                                          cv=5,
```

```
                    verbose=1,
                    scoring="accuracy",
                    max_features=5,
                    n_population=50,
                    crossover_proba=0.5,
                    mutation_proba=0.2,
                    n_generations=40,
                    crossover_independent_proba=0.5,
                    mutation_independent_proba=0.05,
                    tournament_size=3,
                    n_gen_no_change=10,
                    caching=True,
                    n_jobs=-1)
    start_time = time.time()
    svmga_classifier = svmga_classifier.fit(X_train, y_train)
    svmga_time = svm_time/2
    prediction_data = prediction(X_test, svmga_classifier)
    svmga_acc = cal_accuracy(y_test, prediction_data,'SVM with GA Algorithm Accuracy, Classification Report & Confusion Matrix')
def runNN():
    global nn_acc
    global nn_time
    text.delete('1.0', END)
```

```
    start_time = time.time()
    model = Sequential()
    model.add(Dense(4, input_dim=215, activation='relu'))
    model.add(Dense(215, activation='relu'))
    model.add(Dense(1, activation='sigmoid'))
    model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])
    model.fit(X_train, y_train, epochs=50, batch_size=64)
    _, ann_acc = model.evaluate(X_test, y_test)
    nn_acc = ann_acc*100
    text.insert(END,"ANN Accuracy : "+str(nn_acc)+"\n\n")
    nn_time = (time.time() - start_time)
def runNNGenetic():
    global annga_acc
    global nnga_time
    text.delete('1.0', END)
    train = pd.read_csv(filename)
    rows = train.shape[0] # gives number of row count
    cols = train.shape[1] # gives number of col count
    features = cols - 1
    print(features)
    X = train.values[:, 0:100]
    Y = train.values[:, features]
```

```
    print(Y)

    X_train1, X_test1, y_train1, y_test1 = train_test_split(X, Y, test_size = 0.2,
random_state = 0)

    model = Sequential()

    model.add(Dense(4, input_dim=100, activation='relu'))

    model.add(Dense(100, activation='relu'))

    model.add(Dense(1, activation='sigmoid'))

    model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])

    start_time = time.time()

    model.fit(X_train1, y_train1)

    nnga_time = (time.time() - start_time)

    _, ann_acc = model.evaluate(X_test1, y_test1)

    annga_acc = ann_acc*100

    text.insert(END,"ANN with Genetic Algorithm Accuracy :
"+str(annga_acc)+"\n\n")

def graph():

    height = [svm_acc, nn_acc, svmga_acc, annga_acc]

    bars = ('SVM Accuracy','NN Accuracy','SVM Genetic Acc','NN Genetic Acc')

    y_pos = np.arange(len(bars))

    plt.bar(y_pos,  height)

    plt.xticks(y_pos, bars)

    plt.show()

def timeGraph():
```

```
    height = [svm_time,svmga_time,nn_time,nnga_time]
    bars = ('SVM Time','SVM Genetic Time','NN Time','NN Genetic Time')
    y_pos = np.arange(len(bars))
    plt.bar(y_pos,  height)
    plt.xticks(y_pos, bars)
    plt.show()
font = ('times', 16, 'bold')
title = Label(main, text='Android Malware Detection Using Genetic Algorithm based Optimized Feature Selection and Machine Learning')
#title.config(bg='brown', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)
font1 = ('times', 14, 'bold')
uploadButton = Button(main, text="Upload Android Malware Dataset", command=upload)
uploadButton.place(x=50,y=100)
uploadButton.config(font=font1)
pathlabel = Label(main)
pathlabel.config(bg='brown', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=460,y=100)
generateButton = Button(main, text="Generate Train & Test Model", command=generateModel)
```

```
generateButton.place(x=50,y=150)

generateButton.config(font=font1)

svmButton = Button(main, text="Run SVM Algorithm", command=runSVM)

svmButton.place(x=330,y=150)

svmButton.config(font=font1)

svmgaButton = Button(main, text="Run SVM with Genetic Algorithm",
command=runSVMGenetic)

svmgaButton.place(x=540,y=150)

svmgaButton.config(font=font1)

nnButton = Button(main, text="Run Neural Network Algorithm",
command=runNN)

nnButton.place(x=870,y=150)

nnButton.config(font=font1)

nngaButton = Button(main, text="Run Neural Network with Genetic Algorithm",
command=runNNGenetic)

nngaButton.place(x=50,y=200)

nngaButton.config(font=font1)

graphButton = Button(main, text="Accuracy Graph", command=graph)

graphButton.place(x=460,y=200)

graphButton.config(font=font1)

exitButton = Button(main, text="Execution Time Graph", command=timeGraph)

exitButton.place(x=650,y=200)

exitButton.config(font=font1)

font1 = ('times', 12, 'bold')
```

```
        text=Text(main,height=20,width=150)
        scroll=Scrollbar(text)
        text.configure(yscrollcommand=scroll.set)
        text.place(x=10,y=250)
        text.config(font=font1)
        #main.config()
main.mainloop()
```

6.4 SOFTWARE TESTING

TESTING

Testing is a process of executing a program with the aim of finding error. To make our software perform well it should be error free. If testing is done successfully it will remove all the errors from the software.

6.5 TYPES OF TESTS

1. White Box Testing
2. Black Box Testing
3. Unit testing
4. Integration Testing
5. Alpha Testing
6. Beta Testing
7. Performance Testing and so on

White Box Testing

Testing technique based on knowledge of the internal logic of an application's code and includes tests like coverage of code statements, branches, paths, conditions. It is performed by software developers.

Black Box Testing

A method of software testing that verifies the functionality of an application without having specific knowledge of the application's code/internal structure. Tests are based on requirements and functionality.

Unit Testing

Software verification and validation method in which a programmer tests if individual unitsof source code are fit for use. It is usually conducted by the development team.

Integration Testing

The phase in software testing in which individual software modules are combined and tested as a group. It is usually conducted by testing teams.

Alpha Testing

Type of testing a software product or system conducted at the developer's site. Usually it is performed by the end users.

Beta Testing

Final testing before releasing application for commercial purpose. It is typically done by end-users or others.

Performance Testing

Functional testing conducted to evaluate the compliance of a system or component with specified performance requirements. It is usually conducted by the performance engineer.

Black Box Testing

Blackbox testing is testing the functionality of an application without knowing the details of its implementation including internal program structure, data structures etc. Test cases for black box testing are created based on the requirement specifications. Therefore, it is also called as specification-based testing. Fig.6.4 represents the black box testing:

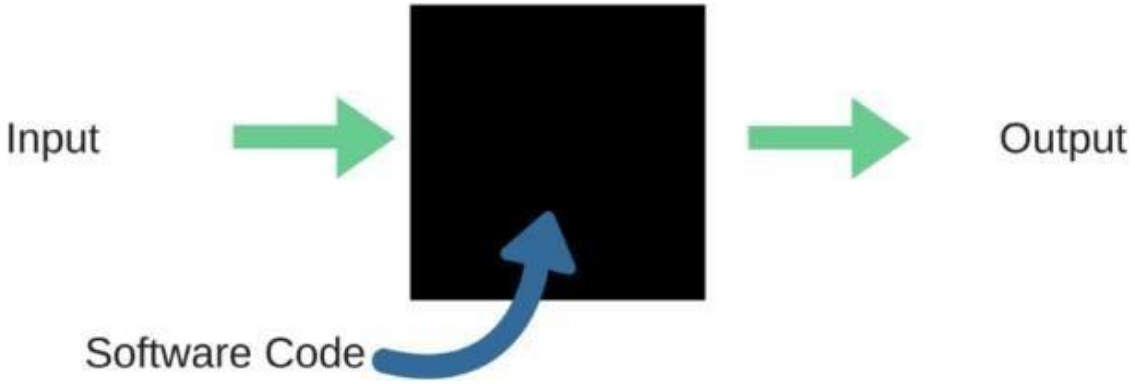


Fig 6.4 Black Box Testing

When applied to machine learning models, black box testing would mean testing machine learning models without knowing the internal details such as features of the machine learning model, the algorithm used to create the model etc. The challenge, however, is to verify the test outcome against the expected values that are known beforehand.

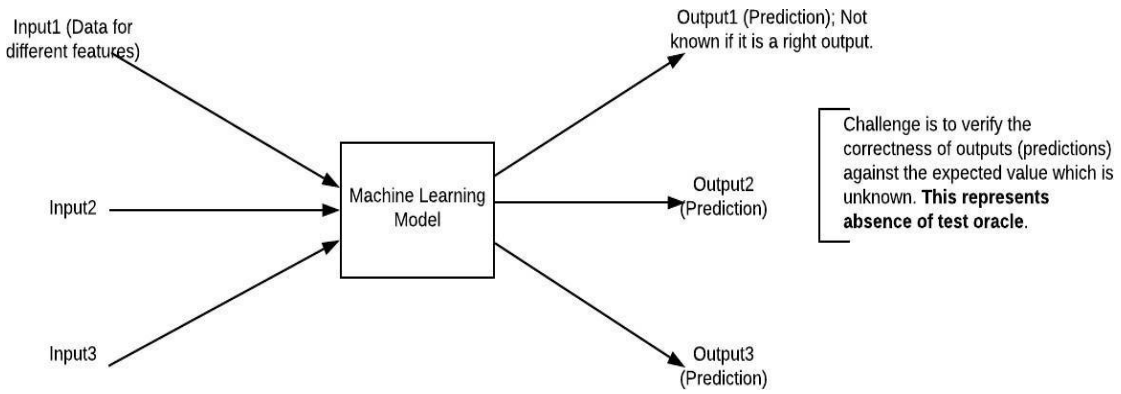


Fig. 6.5 Black Box Testing for Machine Learning algorithms

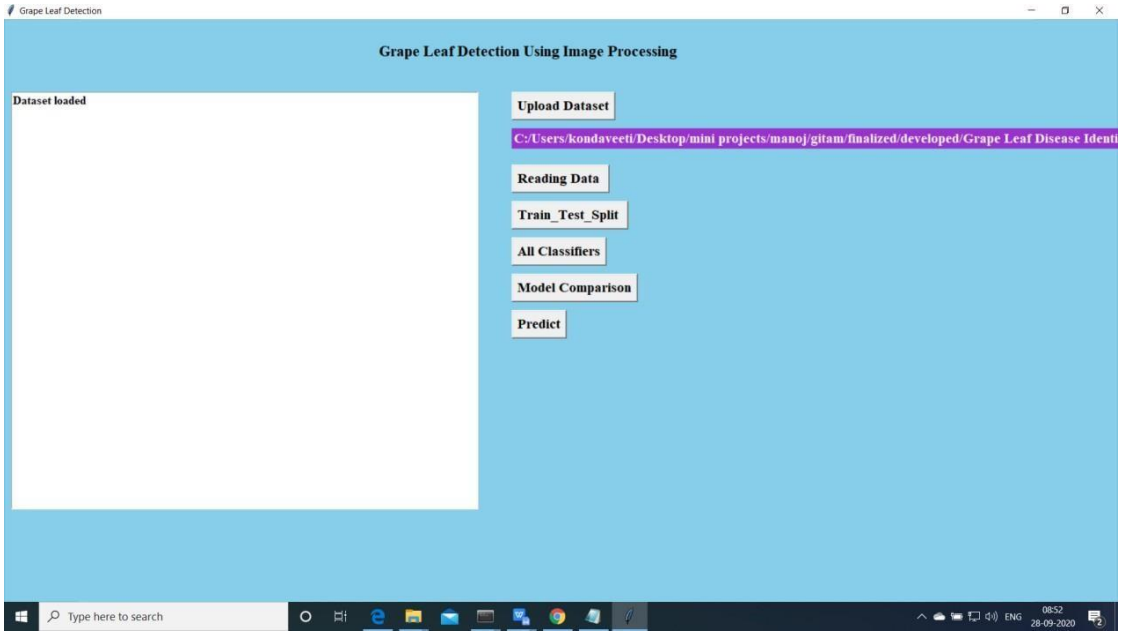
Input	**Actual Output**	**Predicted Output**
[16,6,324,0,0,0,22,0,0,0,0,0,0]	0	0
[16,7,263,7,0,2,700,9,10,1153,832,9,2]	1	1

The model gives out the correct output when different inputs are given which are mentioned in Table 6.4. Therefore the program is said to be executed as expected or correct program.

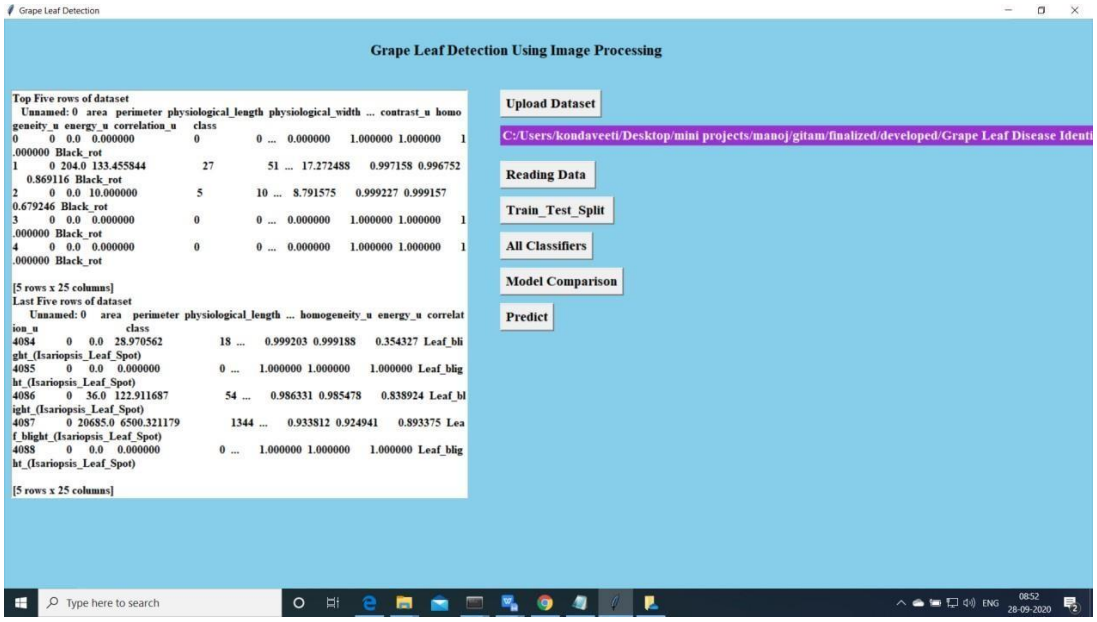
Test Case Id	Test Case Name	Test Case Description	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Start the Application	Host the Application and test if it Starts making sure the required software is Available	If it doesn't Start	We cannot run the application.	The application Hosts success.	High	High
02	Home Page	Check the Deployment Environment for Properly loading the application.	If it doesn‟t load.	We cannot access the application.	The application is running successfully.	High	High
03	User Mode	Verify the working of The Application in freestyle Mode	If it doesn‟t Respond	We cannot use the Freestyle mode.	The application displays the Freestyle Page	High	High
04	Data Input	Verify if the Application takes input and updates	If it fails to take the in put or store In the Database	We cannot Proceed Further	The application updates the input to application	High	High

CHAPTER-7

SCREEN SHOTS

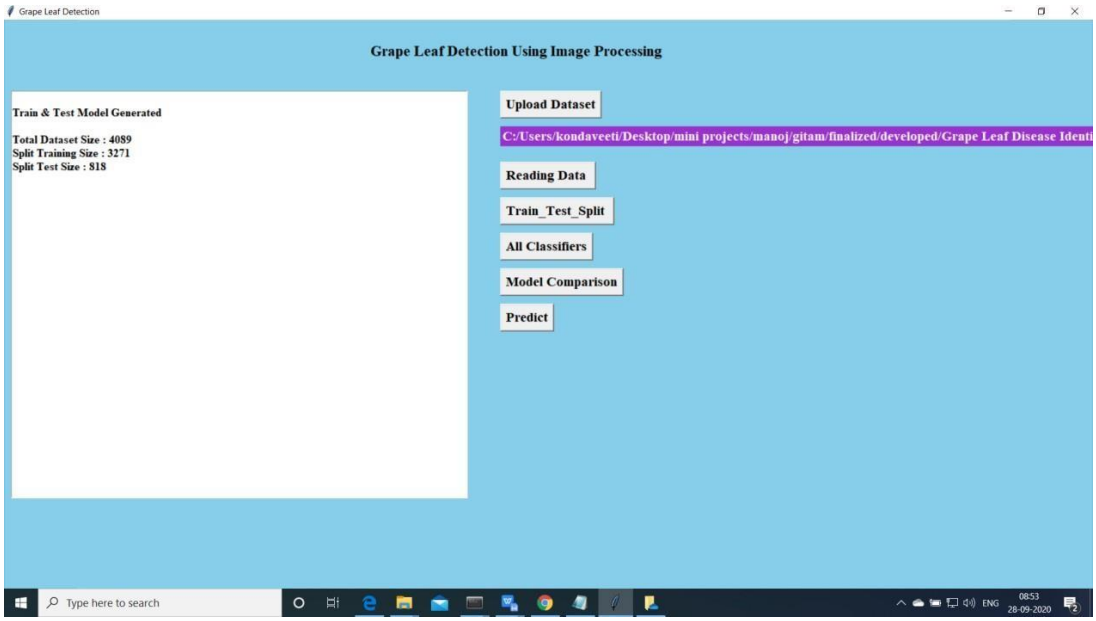


Import the data and Preprocess

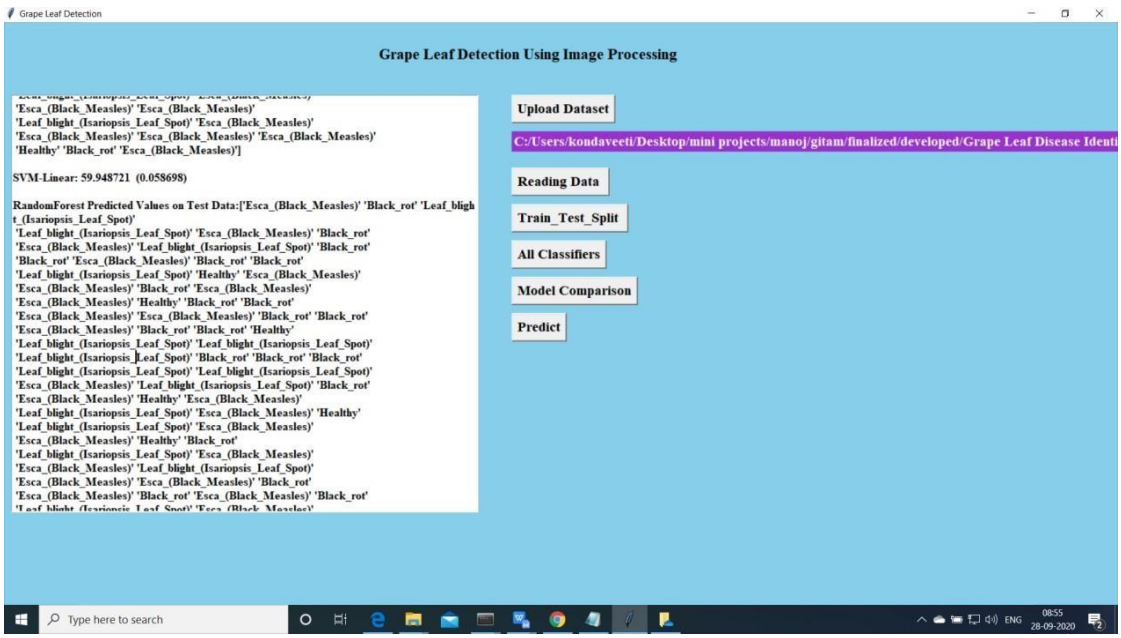


Upload the data and read the basic data information will be shown on the screen

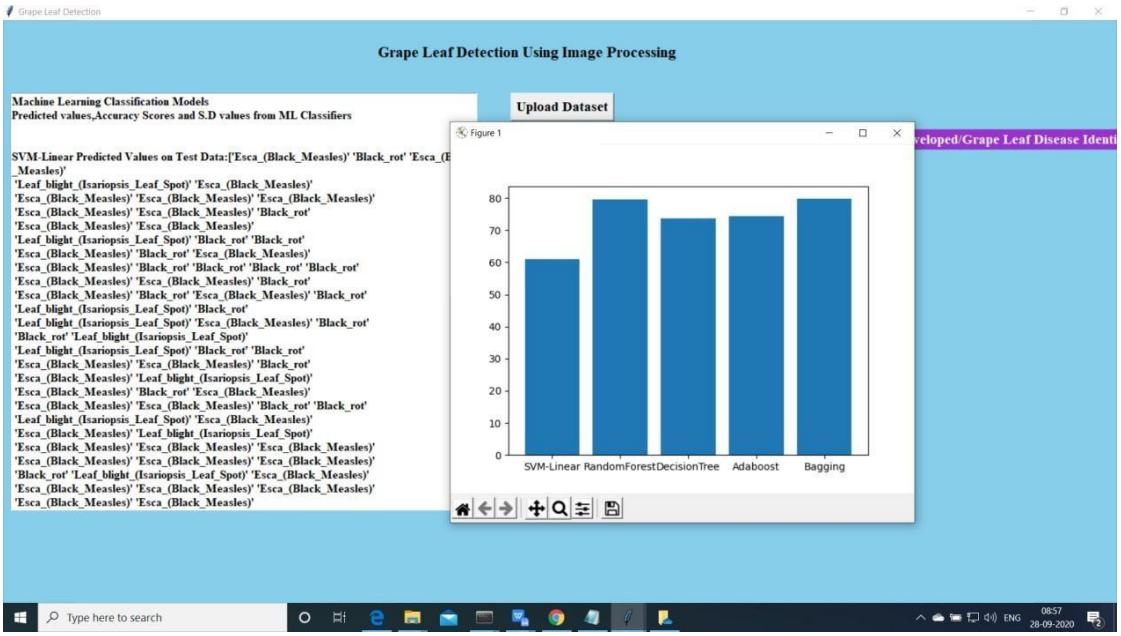
Now click on “Train and Test model”. split the data into train and test and traain will be used fortraining and to tets the performace we are using test data



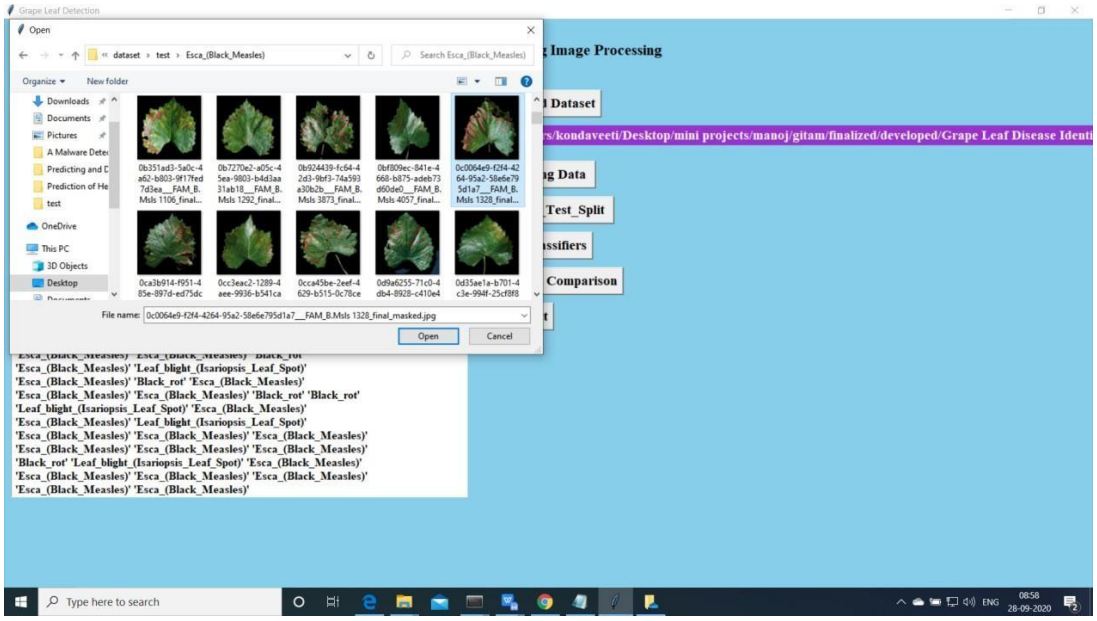
Now click on “Run Algoruimns”. Mentioned algorithms will be run on the data

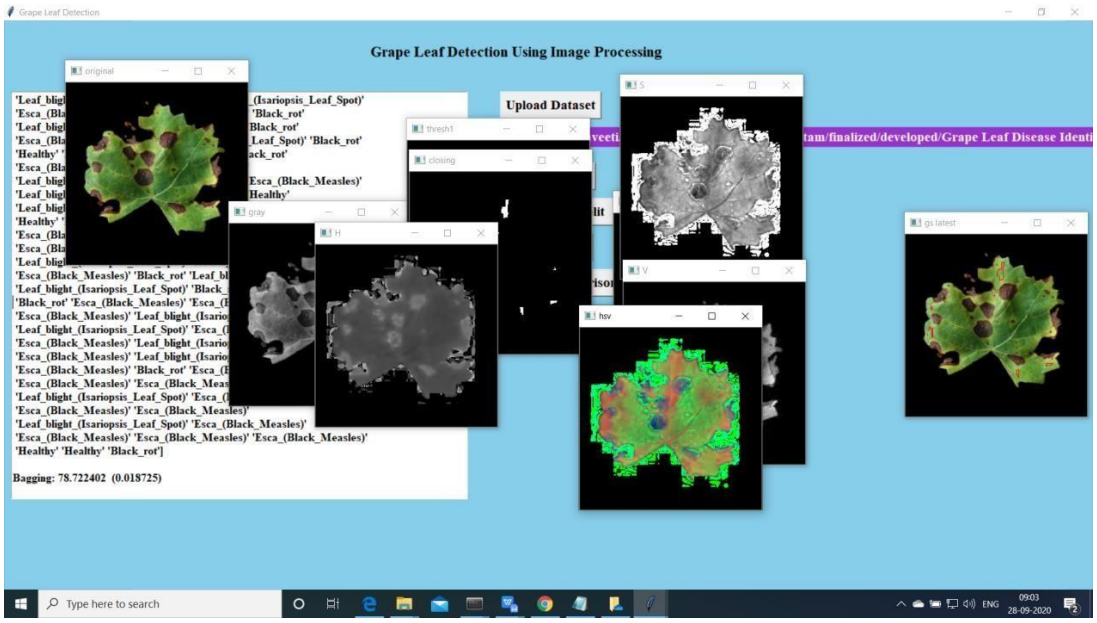


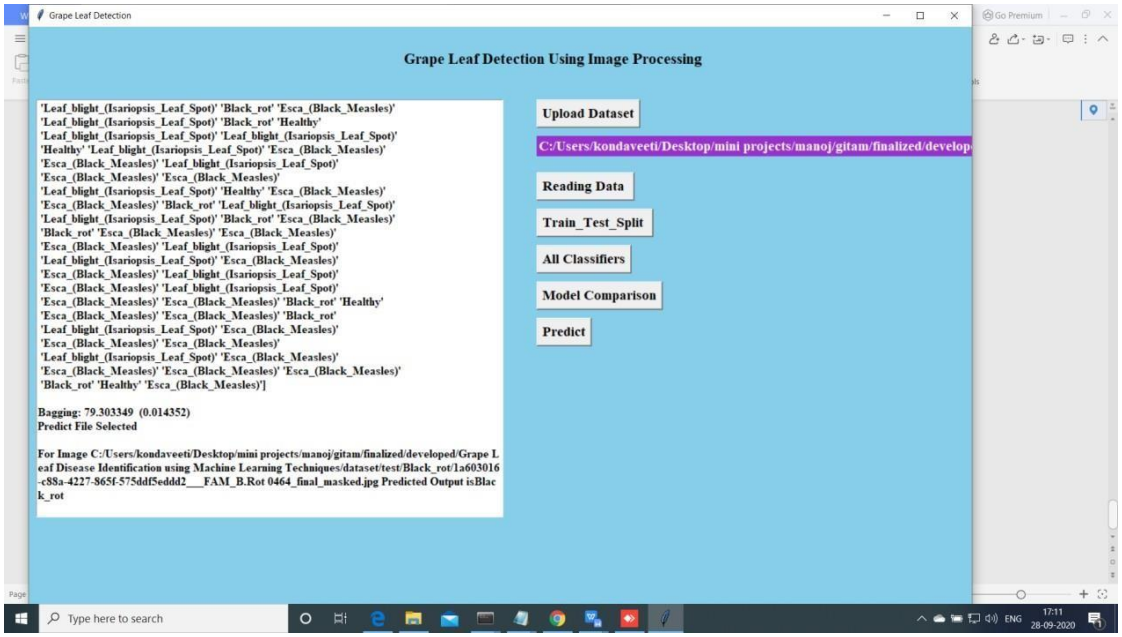
Accuracy Comparision for all the model



Predict:







Extension Bagging algorithm is performed well compared other ml algortithms

CHAPTER-8

CONCLUSION

In this paper, we propose an automatic leaf recognition system that identifydiseases in grape leaves using machine learning technique. The proposed system first segments the leaf part from the background using grab cut segmentation technique. From the segmented leaves diseased region are identified using two different methods. The first method uses globlal thresholding technique whereas the second method using semi supervised learning technique. From the identified diseased part texture and color features are extracted and trained using different classifiers and the results are compared. We have used SVM, random forest and Ada boost algorithms for classification. We have achieved a better result of 93.035% as testing accuracy by using globlal thresholding and SVM.

FUTURE SCOPE:

Future work can be developing the algorithm better segmented techniques. So there is a scope of improvement in the techniques.

APPENDIX-A

BIBLIOGRAPHY

1. Al-hiary, H., Bani-ahmad, S., Reyalat, M., Braik, M., Alrahamneh, Z.: Fast and accurate detection and classification of plant diseases. International Journal of Computer Applications 17(1) (2011)
2. G.PremRishiKranth, HemaLalitha, LaharikaBasava, AnjaliMathurh: Plant disease prediction using machine learning algorithms. International Journal of Computer Applications 18(2) (2018)
3. H.Sabrol, K.Satish: Tomato plant disease classification in digital images using classification tree. In: 2016 International Conference on Communication and Signal Processing (ICCSP), pp. 1242–1246 (2016)
4. Khot.S.T, Supriya, P., Gitanjali, M., Vidya, L.: Pomegranate disease detection using image processing techniques 5(1), 2248 – 2251 (2016)
5. Mokhtar, U., Ali, M.A.S., Hassenian, A.E., Hefny, H.: Tomato leaves diseases detection approach based on support vector machines. In: 2015 11th International Computer Engineering Conference (ICENCO), pp. 246–250 (2015)
6. Mwebaze, E., Owomugisha, G.: Machine learning for plant disease incidence and severity measurements from leaf images. In: 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 158–163 (2016) 978-1-5386-9471-8/19/$31.00 2019 IEEE Second International Conference on Computational Intelligence in Data Science (ICCIDS-2019)
7. Nandhini, A., Hemalatha, Radha, Indumathi: Web enabled plant disease detection system for agricultural applications using wmsn. Wireless Personal Communications 102(2), 725–740 (2018)
8. Nivedita.R.Kakade, Dnyaneswar.D.Ahire: Real time grape leaf disease detection 1, 598 – 610 (2015)
9. P, K., S, S., S, S.: Detection and classification of leaf diseases using integrated approach of support vector machine and particle swarm optimization. 4(1), 79 – 83 (2017)

TECHNOLOGY USED

Python Introduction

Python is a general purpose, dynamic, high level and interpreted programming language. It supports Object Oriented programming approach to develop applications. It is simple and easy to learn and provides lots of high-level data structures.

Python is easy to learn yet powerful and versatile scripting language which makes it attractive for Application Development.

Python's syntax and dynamic typing with its interpreted nature, makes it an ideal language for scripting and rapid application development.

Python supports multiple programming pattern, including object oriented, imperative and functional or procedural programming styles.

Python is not intended to work on special area such as web programming. That is why it is known as multipurpose because it can be used with web, enterprise, 3D CAD etc.

We don't need to use data types to declare variable because it is dynamically typed so we can write a=10 to assign an integer value in an integer variable.

Python makes the development and debugging fast because there is no compilation step included in python development and edit-test-debug cycle is very fast.

Python History

- Python laid its foundation in the late 1980s.
- The implementation of Python was started in the December 1989 by **Guido Van Rossum** at CWI in Netherland.
- In February 1991, van Rossum published the code (labeled version 0.9.0) to alt.sources.
- In 1994, Python 1.0 was released with new features like: lambda, map, filter, and reduce.
- Python 2.0 added new features like: list comprehensions, garbage collection system.

- On December 3, 2008, Python 3.0 (also called "Py3K") was released. It was designed to rectify fundamental flaw of the language.
- ABC programming language is said to be the predecessor of Python language which was capable of Exception Handling and interfacing with Amoeba Operating System.
- Python is influenced by following programming languages:
    - ABC language.
    - Modula-3

Python Features

Python provides lots of features that are listed below.

1) Easy to Learn and Use

Python is easy to learn and use. It is developer-friendly and high level programming language.

2) Expressive Language

Python language is more expressive means that it is more understandable and readable.

3) Interpreted Language

Python is an interpreted language i.e. interpreter executes the code line by line at a time. This makes debugging easy and thus suitable for beginners.

4) Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, Unix and Macintosh etc. So, we can say that Python is a portable language.

5) Free and Open Source

Python language is freely available at offical web address. The source-code is also available. Therefore it is open source.

6) Object-Oriented Language

Python supports object oriented language and concepts of classes and objects come into existence.

7) Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our python code.

8) Large Standard Library

Python has a large and broad library and prvides rich set of module and functions for rapid application development.

9) GUI Programming Support

Graphical user interfaces can be developed using Python.

10) Integrated

It can be easily integrated with languages like C, C++, JAVA etc.

Python Applications

Python is known for its general purpose nature that makes it applicable in almost each domain of software development. Python as a whole can be used in any sphere of development.

Here, we are specifing applications areas where python can be applied.

1) Web Applications

We can use Python to develop web applications. It provides libraries to handle internet protocols such as HTML and XML, JSON, Email processing, request, beautifulSoup, Feedparser etc. It also provides Frameworks such as Django, Pyramid, Flask etc to design and delelop web based applications. Some important developments are: PythonWikiEngines, Pocoo, PythonBlogSoftware etc.

2) Desktop GUI Applications

Python provides Tk GUI library to develop user interface in python based application. Some other useful toolkits wxWidgets, Kivy, pyqt that are useable on several platforms. The Kivy is popular for writing multitouch applications.

3) Software Development

Python is helpful for software development process. It works as a support language and can be used for build control and management, testing etc.

4) Scientific and Numeric

Python is popular and widely used in scientific and numeric computing. Some useful library and package are SciPy, Pandas, IPython etc. SciPy is group of packages of engineering, science and mathematics.

5) Business Applications

Python is used to build Bussiness applications like ERP and e-commerce systems. Tryton is a high level application platform.

6) Console Based Application

We can use Python to develop console based applications. For example: **IPython**.

7) Audio or Video based Applications

Python is awesome to perform multiple tasks and can be used to develop multimedia applications. Some of real applications are: TimPlayer, cplay etc.

8) 3D CAD Applications

To create CAD application Fandango is a real application which provides full features of CAD.

9) Enterprise Applications

Python can be used to create applications which can be used within an Enterprise or an Organization. Some real time applications are: OpenErp, Tryton, Picalo etc.

10) Applications for Images

Using Python several application can be developed for image. Applications developed are: VPython, Gogh, imgSeek etc.

There are several such applications which can be developed using Python

How to Install Python (Environment Set-up)

In this section of the tutorial, we will discuss the installation of python on various operating systems.

Why Python

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, Pycharm, Netbeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

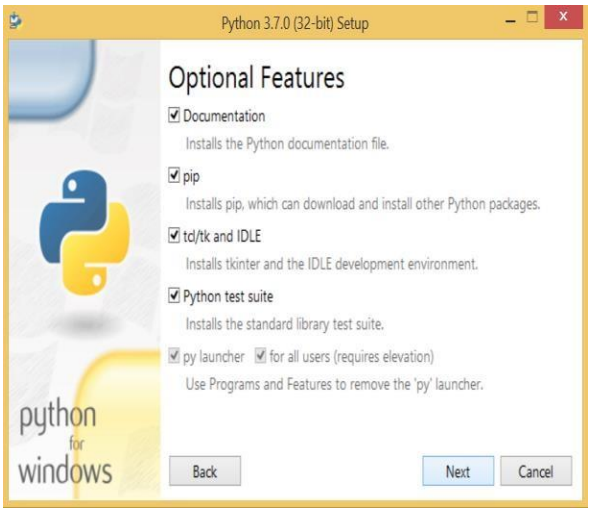
- Python was designed for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Installation on Windows

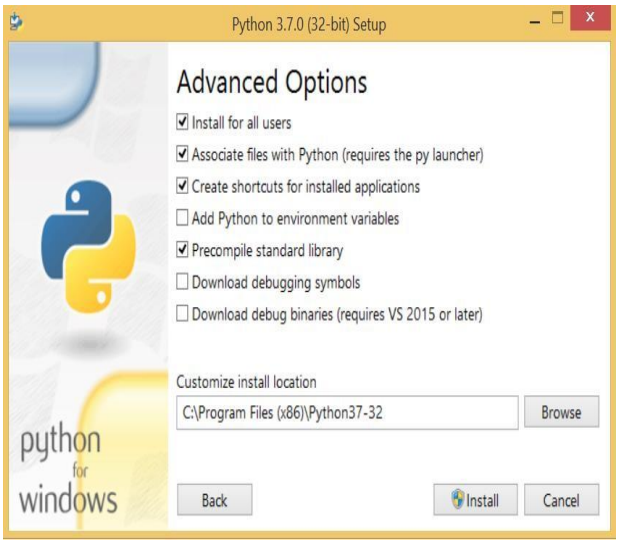
Visit the link *https://www.python.org/downloads/* to download the latest release of Python. In this process, we will install Python 3.6.7 on our Windows operating system.

Double-click the executable file which is downloaded; the following window will open. Select Customize installation and proceed.

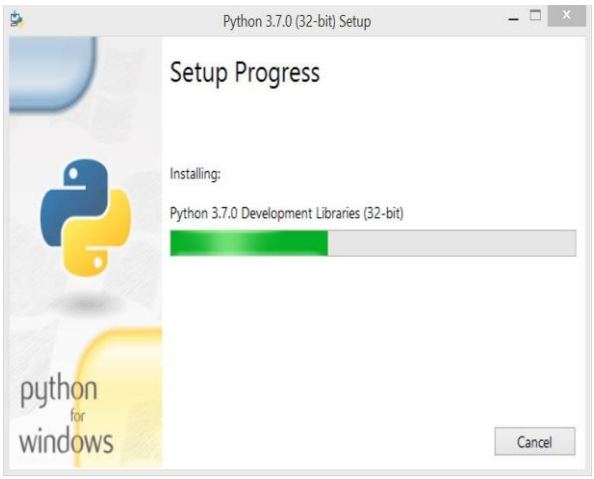
The following window shows all the optional features. All the features need to be installed and are checked by default; we need to click next to continue.



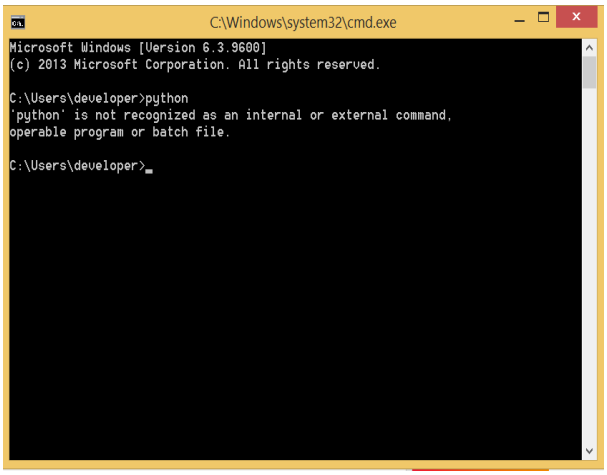
The following window shows a list of advanced options. Check all the options which you want to install and click next. Here, we must notice that the first check-box (install for all users) must be checked.



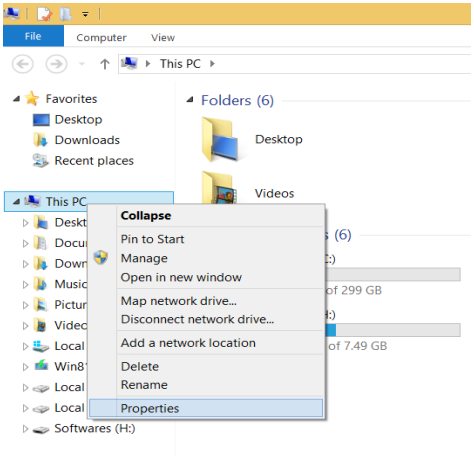
Now, we are ready to install python-3.6.6. Lets install it.

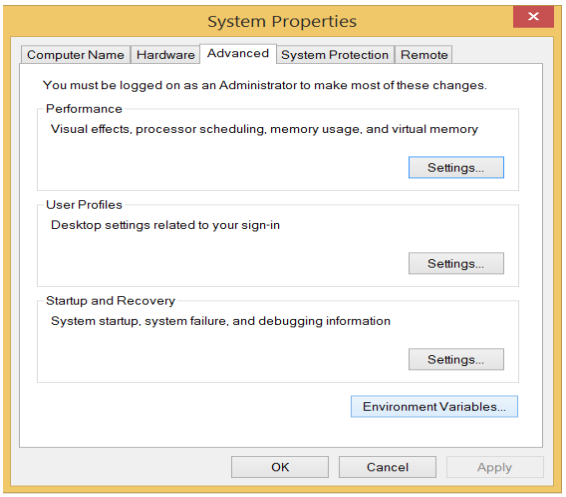


Now, try to run python on the command prompt. Type the command python in case of python2 or python3 in case of python3. It will show an error as given in the below image. It is because we haven't set the path.

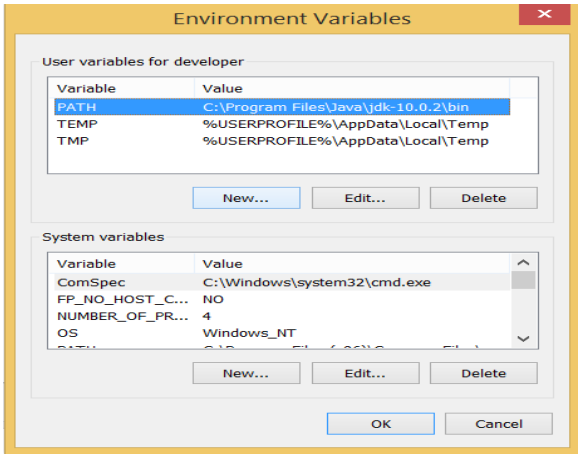


To set the path of python, we need to the right click on "my computer" and go to Properties → Advanced → Environment Variables.

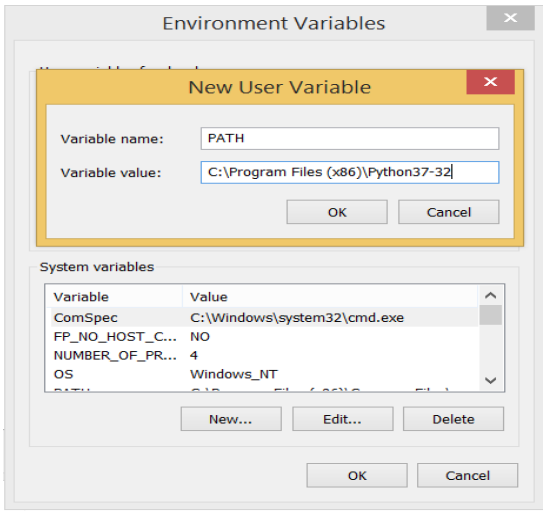




Add the new path variable in the user variable section.



Type PATH as the variable name and set the path to the installation directory of the python shown in the below image.



Now, the path is set, we are ready to run python on our local system. Restart CMD, and type python again. It will open the python interpreter shell where we can execute the python statements.

Virtual Environments and Packages

Introduction

Python applications will often use packages and modules that don‟t come as part of the standard library. Applications will sometimes need a specific version of a library, because the application may require that a particular bug has been fixed or the application may be written using an obsolete version of the library‟s interface.

This means it may not be possible for one Python installation to meet the requirements of every application. If application A needs version 1.0 of a particular module but application B needs version 2.0, then the requirements are in conflict and installing either version 1.0 or 2.0 will leave one application unable to run.

The solution for this problem is to create a virtual environment, a self-contained directory tree that contains a Python installation for a particular version of Python, plus a number of additional packages.

Different applications can then use different virtual environments. To resolve the earlier example of conflicting requirements, application A can have its own virtual environment with version 1.0 installed while application B has another virtual environment with version 2.0. If application B requires a library be upgraded to version 3.0, this will not affect application A‟s environment.

Creating Virtual Environments

The module used to create and manage virtual environments is called venv. venv will usually install the most recent version of Python that you have available. If you have multiple versions of Python on your system, you can select a specific Python version by running python3 or whichever version you want.

To create a virtual environment, decide upon a directory where you want to place it, and run the venv module as a script with the directory path:

python3 -m venv tutorial-env

This will create the tutorial-env directory if it doesn‟t exist, and also create directories inside it containing a copy of the Python interpreter, the standard library, and various supporting files.

A common directory location for a virtual environment is .venv. This name keeps the directory typically hidden in your shell and thus out of the way while giving it a name that explains why the directory exists. It also prevents clashing with .env environment variable definition files that some tooling supports.

Once you‟ve created a virtual environment, you may activate it.

On Windows, run:

tutorial-env\Scripts\activate.bat

On Unix or MacOS, run:

source tutorial-env/bin/activate

(This script is written for the bash shell. If you use the csh or fish shells, there are alternate activate.csh and activate.fish scripts you should use instead.)

Activating the virtual environment will change your shell‟s prompt to show what virtual environment you‟re using, and modify the environment so that running python will get you that particular version and installation of Python. For example:

$ source ~/envs/tutorial-env/bin/activate

(tutorial-env) $ python

Python 3.5.1 (default, May 6 2016, 10:59:36)

...

```
>>> import sys

>>>sys.path

['', '/usr/local/lib/python35.zip', ...,

'~/envs/tutorial-env/lib/python3.5/site-packages']

>>>
```

Managing Packages with pip

You can install, upgrade, and remove packages using a program called pip. By default pip will install packages from the Python Package Index, <https://pypi.org>. You can browse the Python Package Index by going to it in your web browser, or you can use pip‟s limited search feature:

```
(tutorial-env) $ pip search astronomy
skyfield            - Elegant astronomy for Python
gary                - Galactic astronomy and gravitational dynamics.
novas               - The United States Naval Observatory NOVAS astronomy library
astroobs            - Provides astronomy ephemeris to plan telescope observations
PyAstronomy         - A collection of astronomy related tools for Python.
...
```

pip has a number of subcommands: “search”, “install”, “uninstall”, “freeze”, etc. (Consult the Installing Python Modules guide for complete documentation for pip.)

You can install the latest version of a package by specifying a package‟s name:

```
(tutorial-env) $ pip install novas
Collecting novas
Downloading novas-3.1.1.3.tar.gz (136kB)
Installing collected packages: novas
Running setup.py install for novas
Successfully installed novas-3.1.1.3
```

You can also install a specific version of a package by giving the package name followed by == and the version number:

```
(tutorial-env) $ pip install requests==2.6.0
```

Collecting requests==2.6.0
Using cached requests-2.6.0-py2.py3-none-any.whl
Installing collected packages: requests
Successfully installed requests-2.6.0
If you re-run this command, pip will notice that the requested version is already installed and do nothing. You can supply a different version number to get that version, or you can run pip install --upgrade to upgrade the package to the latest version:
(tutorial-env) $ pip install --upgrade requests
Collecting requests
Installing collected packages: requests
Found existing installation: requests 2.6.0
Uninstalling requests-2.6.0:
Successfully uninstalled requests-2.6.0
Successfully installed requests-2.7.0
pip uninstall followed by one or more package names will remove the packages from the virtual environment.
pip show will display information about a particular package:
(tutorial-env) $ pip show requests

Metadata-Version: 2.0
Name: requests
Version: 2.7.0
Summary: Python HTTP for Humans.
Home-page: http://python-requests.org
Author: Kenneth Reitz
Author-email: me@kennethreitz.com
License: Apache 2.0
Location: /Users/akuchling/envs/tutorial-env/lib/python3.4/site-packages
Requires:
pip list will display all of the packages installed in the virtual environment:
(tutorial-env) $ pip list
novas (3.1.1.3)
numpy (1.9.2)
pip (7.0.3)

requests (2.7.0)

setuptools (16.0)

pip freeze will produce a similar list of the installed packages, but the output uses the format that pip install expects. A common convention is to put this list in a requirements.txt file:

(tutorial-env) $ pip freeze > requirements.txt

(tutorial-env) $ cat requirements.txt

novas==3.1.1.3

numpy==1.9.2

requests==2.7.0

The requirements.txt can then be committed to version control and shipped as part of an application. Users can then install all the necessary packages with install -r:

(tutorial-env) $ pip install -r requirements.txt

Collecting novas==3.1.1.3 (from -r requirements.txt (line 1))

...

Collecting numpy==1.9.2 (from -r requirements.txt (line 2))

...

Collecting requests==2.7.0 (from -r requirements.txt (line 3))

...

Installing collected packages: novas, numpy, requests

Running setup.py install for novas

Successfully installed novas-3.1.1.3 numpy-1.9.2 requests-2.7.0

pip has many more options. Consult the Installing Python Modules guide for complete documentation for pip. When you‟ve written a package and want to make it available on the Python Package Index, consult the Distributing Python Modules guide.