**FRUIT DISEASE DETECTION USING COLOR, TEXTURE ANALYSIS AND ANN**

**PROBLEM STATEMENT:**

Our goal is to detect the fruit diseases by using ANN model andK-means clustering method. Such an application is immensely useful in the field of farming and it is inexpensive for the user. We will also use image processing to understand the images of fruits with diseases.

**ABSTRACT:**

Now-a-days as there is prohibitive demand for agricultural industry, effective growth and improved yield of fruit is necessary and important. For this purpose farmers need manual monitoring of fruits from harvest till its progress period. But manual monitoring will not give satisfactory result all the times and they always need satisfactory advice from expert. So it requires proposing an efficient smart farming technique which will help for better yield and growth with less human efforts. We introduce a technique which will diagnose and classify external disease within fruits. Traditional system uses thousands of words which lead to boundary of language. Whereas system that we have come up with, uses image processing techniques for implementation as image is easy way for conveying. In the proposed work, OpenCV library is applied for implementation. K-means clustering method is applied for image segmentation, the images are catalogue and mapped to their respective disease categories on basis of four feature vectors color, morphology, texture and structure of hole on the fruit. The system uses two image databases, one for implementation of query images and the other for training of already stored disease images. Artificial Neural Network (ANN) concept is used for pattern matching and classification of diseases.

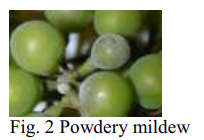
**INTRODUCTION**

The studies of fruit or plant can be determined by observable patterns of specific plant and it is critical to monitor health and detect disease within a plant. Through proper management strategies such as pesticides, fungicides and chemical applications one can facilitates control of diseases which interns improve quality. There are various techniques available such as spectroscopic and imaging technology, applied to achieve superior plant disease control and management. [1] With smart farming today’s farmer can use decision tools and automation techniques which seamlessly integrate product, knowledge and services for better productivity, grading and surplus yield. The purpose of this paper is to monitor diseases on fruits and suggest better solution for healthy yield and productivity with the help of Artificial Neural Network concept. System uses two image databases, one for training of already stored infected area image and other for execution of query images.

Three fruits namely grapes, apple and pomegranate have been used for research in this paper. Types of fruits and their respective diseases are as follows: [1] 1. Grapes: a) Black Rot: For grapes, black rot is most widely occurring and severe disease. In this disease, fungus attacks canes, tendrils, leaves and fruit and it is most devastating in hot and moist areas. Now-a-days using combination of sound cultural practices, fungicides and resistant varieties it is possible to control black rot. First sign of black rot is visible on leaves as black border forms around the edge and small yellowish spot is formed, after that spots enlarges. Also center of lesions become reddish-brown. Spore furnish structure of the fungus appear as a minute black dots and inside margin of lesion they are organize in a ring pattern. Outline of lesion is oval having purple to black color. Mostly symptoms appear after half growth of grapes. [1]



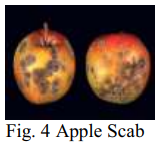
b) Powdery Mildew: It is originated by fungus Uncinulanecator; it is also called as Oidium. Only grapes and other few related species are affected by this fungus. It is most frequently occurring disease on grapes. Primary symptom of powdery mildew is whitish or greenish powdery patches appearing on the underneath of basal leaves. It also causes leaf curling, withering along with blotched or deformation of badly infected leaves. Old infections appear on dormant canes as reddish brown. In case of premature powdery mildew inflammation can cause less sugar content and small berry size. Cracking and scarring of berries can make fruit unsuitable for any purpose. One should be aware that many wine makers have low tolerance for powdery mildew on the grapes.



c) Downy Mildew: It is deeply devastating infection on grapevine. During bloom, summer, rainfall and if the temperature goes to 10˚C (50˚F) in grape-growing areas across the world downy mildew occurs. First symptom of downy mildew can be seen on leaves after 6 to 7 days once the plant is infected. It is caused by Plasmoparaviticola. Almost all green portion of the grape are susplectible. Oil spots (yellow circular spots) along with an oily appearance occur on the leaves. Greater number of oil spots may be found under favourable weather condition. Young oil spots on young foliar are surrounded by brownish-yellow halo. Because of its downy growth it is named as Downy mildew.



2. Apple: a) Apple Scab: It causes most devastating apple infection. It occurs throughout in the apple-growing areas. During the bloom in cool and wet weather apple scab is more severe but it is not reasonably significant in dry or warm climates. Signs of apple scab are visible on leaves, petals, flowers, husk, fruit, young shoots and bud scales of apple tree. Mostly infection on the fruit and leaves are common and obvious.

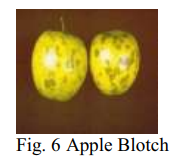


b) Apple Rot: It is a fungus caused due to Botryosphaeriaobtusa. It attacks on leaves, bark and fruits of apple tree. The first symptoms of apple rot appears on outer surfaces of leaves 1 to 3 weeks after petiole fall as small, purple blotch after which centre turn brown tan and yellowish brown. Second stage of apple rot occur after few weeks. In this stage secondary enlargement of leaf spots occurs. Leaf that are highly infected drop from the tree.

Apple rot disease occur in three forms: 1) leaf blotch on apple trees 2) fruit rot on apple trees 3) limb canker on apple tree.



c) Apple Blotch: It is most common “summer disease ” of apples in the northwest. Apple blotch is caused due to two different organisms. Economical loss and commercial quality damage are some of the harsh outcomes of this disease. Signs of disease appear as dark greenish-blue spots on the surface of contaminated fruit. One to many nearly circular colonies are develop individually. These symptoms occur 3-4 weeks later once leaflet falls. Large and unshaped colonies widens over the fruit.



3. Pomegranate: a) Bacterial Blight: This disease was first recorded in Delhi(India) in the year 1952. Until 1998 Bacterial Blight was considered a lower economic threat. However now-a-days this disease occurs widely and has been recorded in all states.This disease occurs in all pomegranate-growing states like Maharashtra, Karnataka, and Andhra Pradesh. Sepals, twigs and pomegranate are affected by Bacterial blight. Preliminary symptoms for the disease can be black colored spots surrounded by bacterial slime. 90% yield of pomegranate depletes due to bacterial blight. Fruits crack due to this disease.



b) Aspergillus Fruit Rot: Alias of alternaria fruit rot. It appear when flower begins to open after the rainfall and it infect the internal portion of pomegranate. Tiny off-color in the skin and less weight due to internal decay are some of the exterior signs of disease. But this problem usually is not apparent until harvesting or during fruit sorting. Without any external symptoms fungus may grow within the fruit. Mostly, infected fruit show some yellowish to brownish-red discoloration and are slightly off-color such as a pale red.



Fig. 8 Aspergillus Fruit Rot

c) Gray Mold: Gray mold is also called as Botrytis cinetea. This disease is more active and commonly occured during post-harvest wash and spread when kept at room temperature. Gray Mold damage flower part of pomegranate and affects the fruit until its ripening. Once the fruit is washed or stored at high humidity, condensation or water on the blossom tissues activates the fungal mycelium to begin growing. The typical grayish coating of spores and pathogen sporulats on the flower parts are developed. Eventually the fungus will outspread inside fruit tissue the crown tissue will be colonized. Infected fruit stored at high moisture.



**LITERATURE SURVEY**

**[1] Image Processing for Smart Farming: Detection of Disease and Fruit Grading, Authors (Monica Jhuria, Ashwani Kumar, Rushikesh Borse), 2013:**

As there is a need of high yield in agricultural industries improved yield of fruit is important, for this there is a need of automated technique which will find disease on fruits. For this artificial neural network methodology is suggested which can be helpful to categories fruit infection. K-Means clustering is applied to find diseased area on the fruit but it has disadvantage of sizable estimation load. It will encourage agronomist to build better production and make correct time to time judgment.

**[2] A Review of Image Processing For Pomegranate Disease Detection, Authors (Manisha A. Bhange, Prof. H. A. Hingoliwala), 2015:**

The process suggests a solution for the recognition of pomegranate fruit disease. In this process, web based technique is applied to help non experts in identifying fruit diseases which depends on the picture representing the symptoms of the fruit. Farmers can take image of fruit disease and upload it on the system. After this farmers would be able to see if the fruit is affected by bacterial blight or not.

**[3] A Cost Effective Tomato Maturity Grading System using Image Processing for Farmers, Authors (Sudhir Rao Rupangadi, Ranjani B.S., Prathik Nagaraj, Varsha G Bhat), 2014:**

This system classifies ripeness of fruit based on its color or texture. It involves current techniques mainly manual inspection which leads to errorious classification; it results in economic losses due to inferior produce in the market chain. The short comings are several methodologies which require highly expensive setups and complicated procedures; overall accuracy is achieved up to 98%.

**[4] Adapted Approach for Fruit Disease Identification using Images, Authors (Shiv Ram Dubey, Anand Singh Jalal):**

This adaptive approach is validated on the basis of experiments. The approach consist of steps and that are stated as; first step is k-means clustering technique which is applied for defect segmentation and second step involves some state of art features that are extracted from segmented image and then segmented image are classified into one of classes with the help of multi-class support vector machine. It achieves precision up to 93%.

**[5] Fruit Detection using Improved Multiple Features based Algorithm, Authors (Hetal N. Patel, Dr. R. K. Jain, and Dr. M. V. Joshi), 2011:**

This gives improved solution for locating the fruits on the plant based on multiple features. Multiple feature extortion technique can include steps like extraction of color and intensity feature, extraction of orientation feature, extraction of edge feature, extraction of area from feature maps. The process is entirely automatic and it can work without user involvement. To improve output it considers numerous features.

**[6] Tomato quality evaluation with image processing: A review, Authors(Abraham Gastélum-Barrios, Rafael A. BórquezLópez, Enrique Rico-García, Manuel Toledano-Ayala and Genaro M. Soto-Zarazúa), 2011:**

All over the world there is excessive requirement for tomato. Therefore grade assessment of tomato is prime task using image processing it can be acquire. Worldwide study of tomato production is done to accomplish the target. It is useful to obtain tomato quality, good color, pattern, size and composition. Instead of manual testing we can achieve fast and accurate testing in laboratories for tomato grading.

**[7] Manisha A. Bhange, Prof. H. A. Hingoliwala “A Review of Image Processing for Pomegranate Disease Detection” International Journal of Computer Science and Information Technologies, Vol. 6 (1), 2015, 92-94.**

In this paper, we suggest a solution for the detection of pomegranate fruit disease (bacterial blight) and also the solution for that disease after detection is proposed. Bacterial Blight need to control at primary stages otherwise it will lead to economic loss. Web-based system used to help non experts in identifying fruit diseases, based on the picture representing the symptoms of the fruit. Farmers can take the photo of the fruit disease and upload it to the system. Then system will show to the farmer is the fruit is infected by the bacterial blight or not. We have added new approach of IntentSearch in this system that is useful when quality of input image is poor. The image processing based proposed system uses two image databases, one for training and other for testing. The images are classified and mapped to their respective disease categories on basis of three feature vectors namely, color, texture and morphology.

**[8] Savita N. Ghaiwat, Parul Arora “Detection and Classification of Plant Leaf Diseases Using Image processing Techniques: A Review” International Journal of Recent Advances in Engineering & Technology (IJRAET) ISSN (Online): 2347 - 2812, Volume-2, Issue - 3, 2014.**

This paper present survey on different classification techniques that can be used for plant leaf disease classification. A classification technique deals with classifying each pattern in one of the distinct classes. A classification is a technique where leaf is classified based on its different morphological features. There are so many classification techniques such as k-Nearest Neighbour Classifier, Probabilistic Neural Network, Genetic Algorithm, Support Vector Machine, and Principal Component Analysis, Artificial neural network, Fuzzy logic. Selecting a classification method is always a difficult task because the quality of result can vary for different input data. Plant leaf disease classifications have wide applications in various fields such as in biological research, in Agriculture etc. This paper provides an overview of different classification techniques used for plant leaf disease classification.

**SCOPE:**

Farming and agricultural industries can use this system to harvest good fruits.

Food safety agencies can check for quality at very low cost.

People can better select their fruits based on the quality, especially when they order fruits online.

**EXISTING METHOD**

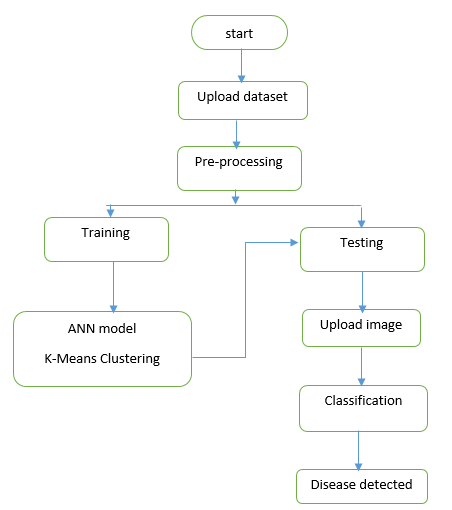
Now-a-days as there is prohibitive demand for agricultural industry, effective growth and improved yield of fruit is necessary and important. For this purpose farmers need manual monitoring of fruits from harvest till its progress period. But manual monitoring will not give satisfactory result all the times and they always need satisfactory advice from expert. So it requires proposing an efficient smart farming technique which will help for better yield and growth with less human efforts. We introduce a technique which will diagnose and classify external disease within fruits. Traditional system uses thousands of words which lead to boundary of language. Whereas system that we have come up with, uses image processing techniques for implementation as image is easy way for conveying.

**Disadvantages**

1. More Expensive.
2. Difficult to scale up.
3. Time consuming.

**PROPOSED METHOD**

In the proposed work, OpenCV library is applied for implementation. K-means clustering method is applied for image segmentation, the images are catalogue and mapped to their respective disease categories on basis of four feature vectors color, morphology, texture and structure of hole on the fruit. The system uses two image databases, one for implementation of query images and the other for training of already stored disease images. Artificial Neural Network (ANN) concept is used for pattern matching and classification of diseases.



**Advantages:**

* Cheaper to operate.
* It can be scaled up quickly.
* Time minimising.

**APPLICATIONS:**

* Formers.
* Agricultural Agencies.
* Fruit sellers.
* Customers/consumers.

**HARDWARE & SOFTWARE REQUIREMENTS**

# H/W Configuration:

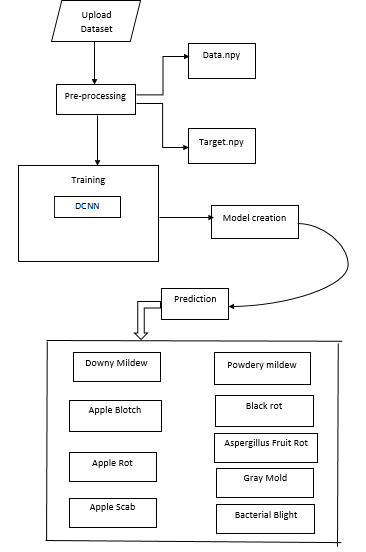
# Processor - I3/Intel Processor

* Hard Disk -160GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA
* RAM - 8Gb

**S/W Configuration:**

* Operating System : Windows 7/8/10
* Server side Script : Python, Anaconda
* IDE : Pycharm
* Libraries Used : Numpy, IO, OS, Flask, keras,
* Technology : Python 3.6+

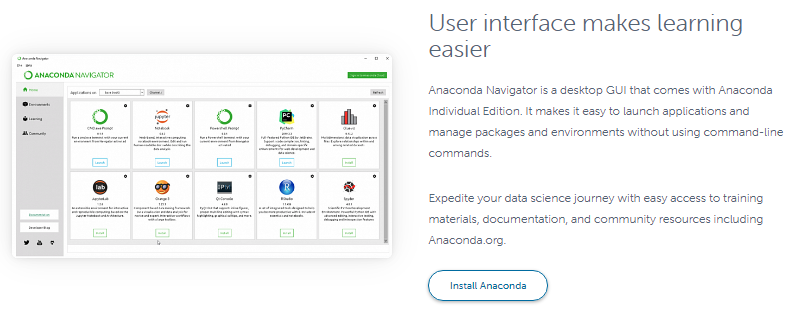
**ARCHITECTURE**



**SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECTS:**

#### 1. Visit the Anaconda downloads page.

Go to the following link: [Anaconda.com/downloads](https://www.anaconda.com/download/)



#### 2. Select Windows.

Select Windows where the three operating systems are listed.

#### 3. Download.

Choose Python 3.6 version, 64 bit graphical installer.

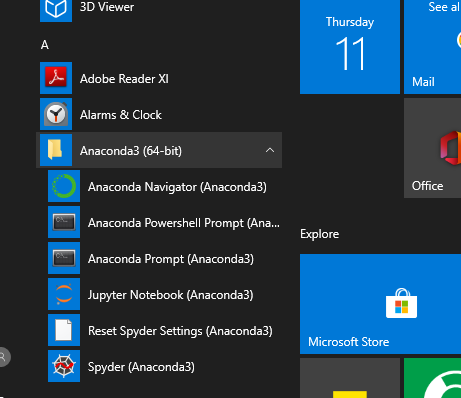
4. Let it download in an .exe format.

#### 5. Open and run the installer.

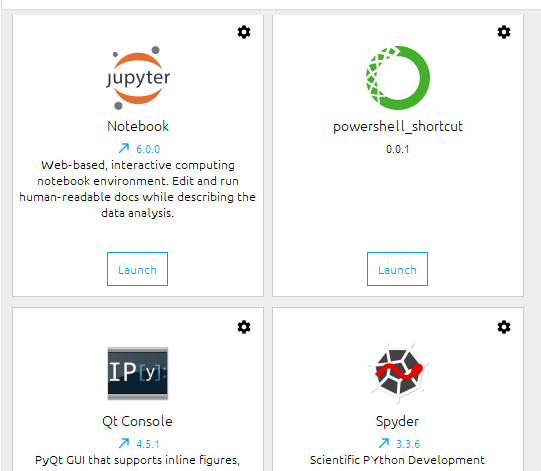
#### Once the download completes, open and run the .exe installer.

6. Click on next, I agree, install.

7. Completion of the installation, open your windows start menu and select the Anaconda navigator.

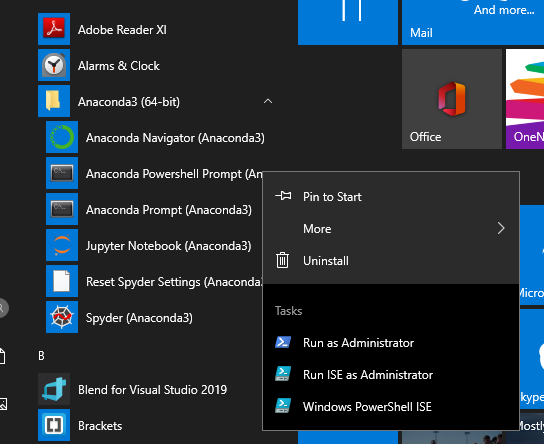


8. Click on Jupiter launch button, your Jupiter notebook will get start.



9. You need to install some packages to execute your project in a proper way.

10. Select windows start menu, right click on anaconda prompt, choose run as administrator.



11. Anaconda prompt will get open, with specified path, type “Pip install package name” which you want to install (like Jumpy, Pandas, seaborne, sickie learn, matplotlib, pilot)

Ex: pip install Numpy



12. You can also install required packages in Jupiter notebook directly by using the syntax as “! Pip install package name.

Ex: pip install Jumpy

**MODULES**

**System**

**User**

**1. System:**

**1.1 Create Dataset:**

The dataset containing fruit images of the desired objects to be recognize is split into training and testing dataset with the test size of 20-30%.

**1.2 Pre-processing:**

Resizing, gray scaling and reshaping the images into appropriate format to train our model. The final dataset is split into training and testing dataset with test size of 10%.

**1.3Training:**

Use the pre-processed training dataset to train our model using ANN algorithm.

**2.User:**

**2.1 Register**

The user needs to register and the data stored in MySQL database.

**2.2 About-Project**

In this application, we have successfully created an application which takes a fruit image for classify the fruit diseases.

**2.2 Login**

A registered user can login using the valid credentials to the website to use a application.

**2.4 Upload Image**

The user has to upload an image which needs to be tested for fruit diseases.

**2.5 Prediction**

The results of our model is displayed as either apple blotch, apple scab, aspergillums fruit rot, bacterial blight, black rot, downy mildew, gray mold, powdery mildew and rotten apples.

**2.6 Logout**

Once the prediction is over, the user can logout of the application.

**ALGORITHM:**

**ARTIFICIAL NEURAL NETWORK:**

Neural Network Algorithms – Artificial Neural Networks arguably works close enough to the human brain. Conceptually artificial neural networks are inspired by neural networks in the brain but the actual implementation in machine learning is way far from reality. ANN take in multiple inputs and produce a single output. Point to note ANN’s are inspired by the animal brain, but nowhere close to biological neural networks.

In this post, we will explore some basic understanding of artificial neural networks and Neural Network Algorithms, behind the scenes working and a quick glance to its working (Algorithms). This is part -2 of the previous post – Deep Learning – Introduction to Artificial Neural Networks



Artificial neural networks (ANNs) as “Biologically inspired computing code with the number of simple, highly interconnected processing elements for simulating (only an attempt) human brain working & to process information model”. It’s way different than computer program though. There are several kinds of Neural Networks in deep learning. Neural networks consist of input and output layers and at least one hidden layer.

Multi-Layer Perceptron

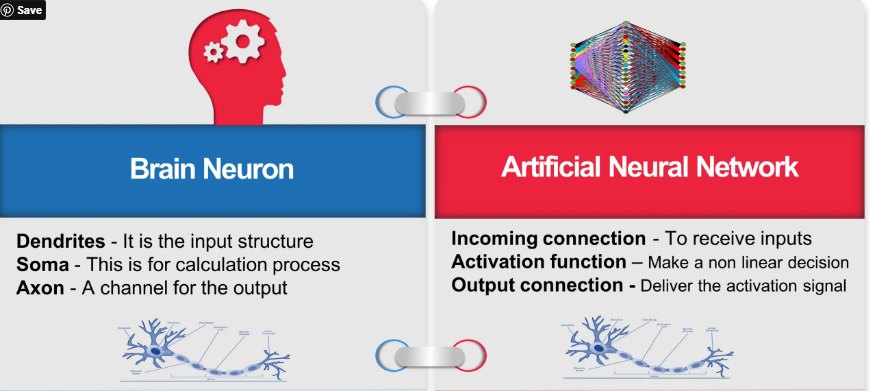
Radial Basis Network

Recurrent Neural Networks

Generative Adversarial Networks

Convolutional Neural Networks.

Neural Network Algorithms are based on radial basis function with can be used for strategic reasons. There are several other models of the neural network including what we have mentioned above. For an introduction to the neural network and their working model continue reading this post. You will get a sense of how they work and used for real mathematical problems.

 ANN’s learns, get trained and adjust automatically like we humans do. Though ANN’s are inspired by the human brain but for a fact they run on a far simpler plane. The structure of neurons is now used for machine learning thus called as artificial learning. This development has helped various problems to come to an end especially where layering is needed for refinement and granular details are needed.

**Neural Network Architecture**

Neural networks consist of input, output layers hidden layers. Transformation of input into valuable output unit is the main job. They are excellent examples of mathematical constructs.  Information flows in neural network happens in two ways.

**Feedforward Networks** – In these signals only travel in one direction without any loop i.e. towards the output layer. Extensively used in pattern recognition. This network with a single input layer and a single output layer can have zero or multiple hidden layers though. This method has two common designs as below

At the time of it’s learning or “being trained”

At the time of operating normally or “after being trained”

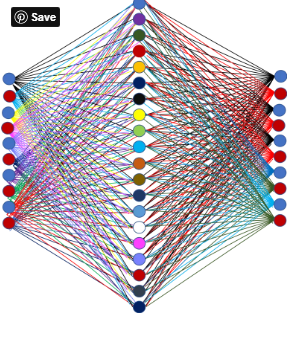
Feedback Networks – In this recurrent or interactive networks can use their internal state (memory) to process sequences of inputs.  Signals can travel in both directions with loops in the network. As of now limited to time-series/sequential tasks. Typical human brain model.

Neural Network Algorithms works on three main layers of its architecture i.e input layer, hidden layer (though there can be many hidden layers) and output layer.

Architectural Components

**Input Layers, Neurons, and Weights**  –  The basic unit in a neural network is called as the neuron or node. These units receive input from the external source or some other nodes. The idea here is to compute an output based associated weight. Weights to the neuron are assigned based on its relative importance compared with other inputs. Now finally function is applied to this for computations.

Let’s assume our task to it to make tea so our ingredients will represent the “neurons” or input neurons as these are building blocks or starting points. The amount of each ingredient is called a “weight.” After dumping tea, sugar, species, milk and water in a pan and then mixing will transform it another state and colour. This process of transformation can be called an “activation function”.



Hidden Layers and Output Layers – The hidden layer is always isolated from the external world hence its called as hidden. The main job of the hidden layer to take inputs from the input layer and perform its job i.e calculation and transform the result to output nodes. Bunch of hidden nodes can be called a hidden layer.

Continuing the same example above – In our tea making task, now using the mixture of our ingredients coming out of the input layer, the solution upon heating (computation process) starts changing colour. The layers made up by the intermediate products are called “hidden layers”. Heating can be compared with the activation process at the end we get our final tea as output.

The network described here is much simpler for ease of understanding compared to the one you will find in real life. All computations in the forward propagation step and backpropagation step are done in the same way (at each node) as discussed before. Neural Network Algorithms

**Neural Network Work Flow – Layers of Learning**

Neural networks learning process is not very different from humans, humans learn from experience in lives while neural networks require data to gain experience and learn. Accuracy increases with the amount of data over time. Similarly, humans also perform the same task better and better by doing any task you do over and over.

Neural Network Algorithms’ underlying foundation of neural networks is a layer and layers of connections. The entire neural network model is based on a layered architecture. Each layer has its own responsibility. These networks are designed to make use of layers of “neurons” to process raw data, find patterns into it and objects which are usually hidden to naked eyes. To train a neural network, data scientist put their data in three different baskets.

**Training data set** – This helps networks to understand and know the various weights between nodes.

**Validation data set** – To fine-tune the data sets.

Test data set – To evaluate the accuracy and records margin of error.

Layer takes input, extract feature and feed into the next layer i.e. each layer work as an input layer to another layer. This is to receive information and last layer job is to throw output of the required information. Hidden layers or core layers process all the information in between.

**K-NEAREST NEIGHBORS (KNN):**

**Introduction:**

K-nearest neighbors (KNN) algorithm is a type of supervised ML algorithm which can be used for both classification as well as regression predictive problems. However, it is mainly used for classification predictive problems in industry. The following two properties would define KNN well −

Lazy learning algorithm − KNN is a lazy learning algorithm because it does not have a specialized training phase and uses all the data for training while classification.

Non-parametric learning algorithm − KNN is also a non-parametric learning algorithm because it doesn’t assume anything about the underlying data.

**Working of KNN Algorithm:**

K-nearest neighbors (KNN) algorithm uses ‘feature similarity’ to predict the values of new datapoints which further means that the new data point will be assigned a value based on how closely it matches the points in the training set. We can understand its working with the help of following steps −

Step 1 − For implementing any algorithm, we need dataset. So during the first step of KNN, we must load the training as well as test data.

Step 2 − Next, we need to choose the value of K i.e. the nearest data points. K can be any integer.

Step 3 − For each point in the test data do the following −

3.1 − Calculate the distance between test data and each row of training data with the help of any of the method namely: Euclidean, Manhattan or Hamming distance. The most commonly used method to calculate distance is Euclidean.

3.2 − Now, based on the distance value, sort them in ascending order.

3.3 − Next, it will choose the top K rows from the sorted array.

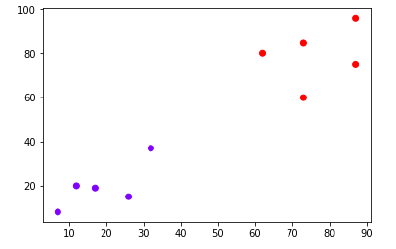
3.4 − Now, it will assign a class to the test point based on most frequent class of these rows.

Step 4 − End

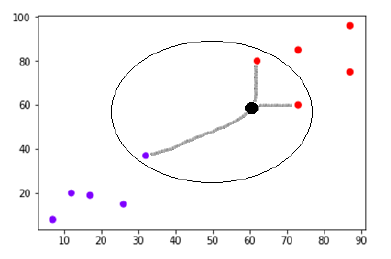
Example

The following is an example to understand the concept of K and working of KNN algorithm −

Suppose we have a dataset which can be plotted as follows –



Now, we need to classify new data point with black dot (at point 60,60) into blue or red class. We are assuming K = 3 i.e. it would find three nearest data points. It is shown in the next diagram –



We can see in the above diagram the three nearest neighbors of the data point with black dot. Among those three, two of them lies in Red class hence the black dot will also be assigned in red class.

**STEPS FOR EXECUTING THE PROJECTS**

1. Install the required packages

2. Load the datasets.

3. Combining the dataset into one dataset.

4. Pre-process the data.

5. Split the dataset into train and test.

6. Use the train dataset to train the ANN model.

7. Use the test data to test the model for prediction and accuracy generation.

**SYSTEM DESIGN**

**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 inssss 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.

**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.



**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.



**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.



**Collaboration Diagram:**

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

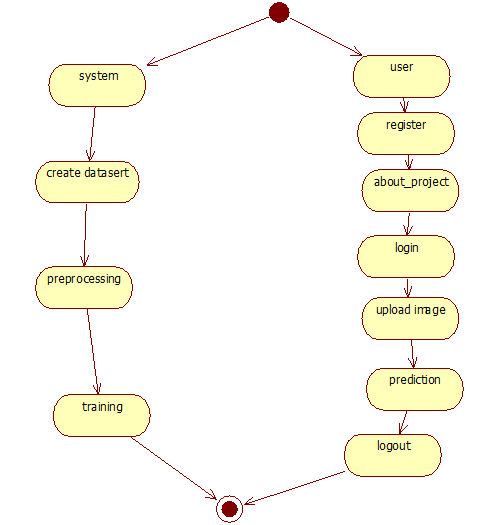
 **DEPLOYMENT DIAGRAM**

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware’s used to deploy the application.



**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.



**Component diagram**,

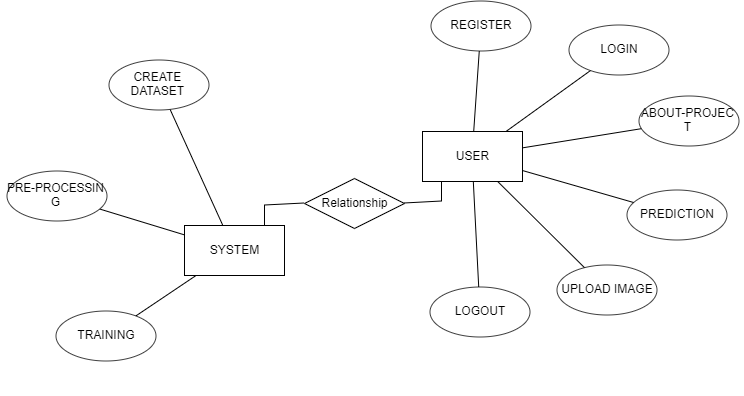
A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical **c**omponents in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required functions is covered by planned development.



**ER Diagram:**

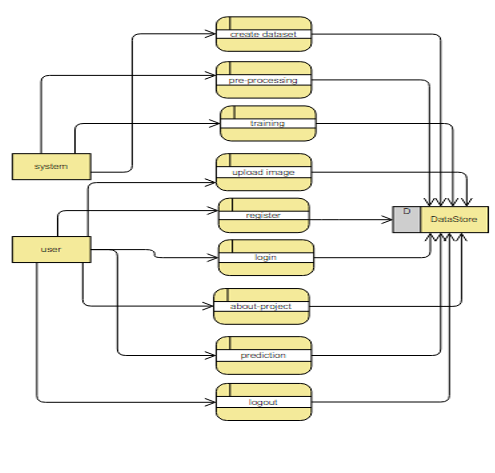
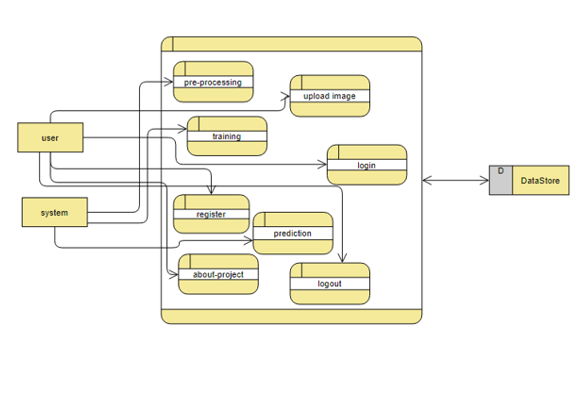
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let’s have a look at a simple ER diagram to understand this concept.



**DFD Diagram:**

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

# **INTRODUCTION TO PYTHON**

* **Python**

### What Is A Script?

Up to this point, I have concentrated on the interactive programming capability of Python.  This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

**Scripts are reusable**

Basically, a script is a text file containing the statements that comprise a Python program.  Once you have created the script, you can execute it over and over without having to retype it each time.

**Scripts are editable**

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor.  Then you can execute each of the individual versions.  In this way, it is easy to create different programs with a minimum amount of typing.

**You will need a text editor**

Just about any text editor will suffice for creating Python script files.

You can use *Microsoft Notepad, Microsoft WordPad, Microsoft Word,*or just about any word processor if you want to.

**Difference between a script and a program**

**Script:**

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

**Program:**

The program has an executable form that the computer can use directly to execute the instructions.

The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

**Python**

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don’t know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to “big name” languages. Hopefully I can explain it for you.

**Python concepts**

If you’re not interested in the the hows and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it’s a great one to start programming with.

• Open source general-purpose language.

• Object Oriented, Procedural, Functional

• Easy to interface with C/ObjC/Java/Fortran

• Easy-is to interface with C++ (via SWIG)

• Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

* **Python is 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 is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**History of Python**

Python was developed by Guido van Possum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, Smalltalk, and UNIX shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Possum still holds a vital role in directing its progress.

**Python Features**

Python's features include −

* **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
* **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
* **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
* **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
* **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
* **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
* **Extendable** − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
* **Databases** − Python provides interfaces to all major commercial databases.
* **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
* **Scalable** − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

* It supports functional and structured programming methods as well as OOP.
* It can be used as a scripting language or can be compiled to byte-code for building large applications.
* It provides very high-level dynamic data types and supports dynamic type checking.
* IT supports automatic garbage collection.
* It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

**Dynamic vs. Static**

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is.

For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type.

This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point.

If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn’t require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program.

With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn’t matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it’s more accurate to keep track of decimals so it automatically calculates the result as a floating point number

**Variables**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

**Standard Data Types**

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types −

* Numbers
* String
* List
* Tuple
* Dictionary

## Python Numbers

Number data types store numeric values. Number objects are created when you assign a value to them

## Python Strings

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

## Python Lists

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (\*) is the repetition operator.

## Python Tuples

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated. Tuples can be thought of as **read-only** lists.

## Python Dictionary

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.

Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

**Different modes in python**

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished .pie files are run in the Python interpreter.

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

# 20 Python libraries

**1.** Requests. The most famous http library written by Kenneth remits. It’s a must have for every python developer.

**2.** Scrappy. If you are involved in web scraping then this is a must have library for you. After using this library you won’t use any other.

**3.** Python. A guy toolkit for python. I have primarily used it in place of tinder. You will really love it.

**4.** Pillow. A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.

**5.** SQLAlchemy. A database library. Many love it and many hate it. The choice is yours.

**6.** Beautiful Soup. I know it’s slow but this xml and html parsing library is very useful for beginners.

**7.** Twisted. The most important tool for any network application developer. It has a very beautiful ape and is used by a lot of famous python developers.

**8.** Numbly. How can we leave this very important library? It provides some advance math functionalities to python.

**9.** Skippy. When we talk about numbly then we have to talk about spicy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.

**10.** Matplotlib. A numerical plotting library. It is very useful for any data scientist or any data analyzer.

**11.** Pygmy. Which developer does not like to play games and develop them? This library will help you achieve your goal of 2d game development.

**12.** Piglet. A 3d animation and game creation engine. This is the engine in which the famous [python port](https://github.com/fogleman/Minecraft) of mine craft was made

**13.** Pit. A GUI toolkit for python. It is my second choice after python for developing GUI’s for my python scripts.

**14.** Pit. Another python GUI library. It is the same library in which the famous Bit torrent client is created.

**15.** Scaly. A packet sniffer and analyzer for python made in python.

**16.** Pywin32. A python library which provides some useful methods and classes for interacting with windows.

**17.** Notch. Natural Language Toolkit – I realize most people won’t be using this one, but it’s generic enough. It is a very useful library if you want to manipulate strings. But its capacity is beyond that. Do check it out.

**18.** Nose. A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.

**19.** Simply. Simply can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.

**20.** I Python. I just can’t stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

**Numpy**

Humpy’s main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In numbly dimensions are called axes. The number of axes is rank.

• Offers Matlab-ish capabilities within Python

• Fast array operations

• 2D arrays, multi-D arrays, linear algebra etc.

**Matplotlib**

• High quality plotting library.

**Python class and objects**

These are the building blocks of OOP. Class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently “the big thing” in most programming languages. Hence, there are several chapters dedicated to OOP later in the book.

The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something.

Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects.

When I first learned C and C++, I did great; functions just made sense for me.

Having messed around with BASIC in the early ’90s, I realized functions were just like subroutines so there wasn’t much new to learn.

However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered.

Once you learn OOP, you’ll realize that it’s actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it’s not mandatory to use objects in your code in a way that works best; maybe you don’t need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want.

As you’ve already seen, Python can do just fine with functions. Unlike languages such as Java, you aren’t tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here’s a brief list of Python OOP ideas:

• The class statement creates a class object and gives it a name. This creates a new namespace.

• Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.

• Class attributes export the state of an object and its associated behavior. These attributes are shared by all instances of a class.

• Calling a class (just like a function) creates a new instance of the class.

This is where the multiple copies part comes in.

• Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.

• Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

**Inheritance**

First off, classes allow you to modify a program without really making changes to it.

To elaborate, by sub classing a class, you can change the behavior of the program by simply adding new components to it rather than rewriting the existing components.

As we’ve seen, an instance of a class inherits the attributes of that class.

However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.

The subclasses can override the logic in a superclass, allowing you to change the behavior of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python’s object model and the object interfaces are more consistent to Python’s built-in objects, hence overloading is easier to learn and use.

User-made classes can override nearly all of Python’s built-in operation methods

**Exceptions**

I’ve talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program’s flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn’t exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don’t occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

• Accessing a non−existent dictionary key will raise a Key Error exception.

• Searching a list for a non−existent value will raise a Value Error exception

. • Calling a non−existent method will raise an Attribute Error exception.

• Referencing a non−existent variable will raise a Name Error exception.

• Mixing data types without coercion will raise a Type Error exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files.

This is the most common way to use exceptions. When programming with the Python command line interpreter, you don’t need to worry about catching exceptions.

Your program is usually short enough to not be hurt too much if an exception occurs.

Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem.

However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception.

It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don’t put any overhead on the code processing.

Because exceptions aren’t supposed to happen very often, they aren’t processed until they occur.

Exceptions can be thought of as a special form of the if/elf statements. You can realistically do the same thing with if blocks as you can with exceptions.

However, as already mentioned, exceptions aren’t processed until they occur; if blocks are processed all the time.

Proper use of exceptions can help the performance of your program.

The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing.

Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

I won’t spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won’t have to do this very often but it’s nice to have the option when necessary.

However, before making your own exceptions, make sure there isn’t one of the built-in exceptions that will work for you.

They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter

. To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.

To make a custom exception, simply inherit the base exception and define what it will do.

**Python modules**

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

**Testing code**

As indicated above, code is usually developed in a file using an editor.

To test the code, import it into a Python session and try to run it.

Usually there is an error, so you go back to the file, make a correction, and test again.

This process is repeated until you are satisfied that the code works. T

His entire process is known as the development cycle.

There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.

This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

**Functions in Python**

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function.

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task.

## To carry out that specific task, the function might or might not need multiple inputs. When the task is carved out, the function can or cannot return one or more values.

## There are three types of functions in python:

## Help (), min (), print ().

## Python Namespace

Generally speaking, a **namespace** (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a name spacing system from daily life, i.e. the naming of people in first name and family name (surname).

An example is a network: each network device (workstation, server, printer,) needs a unique name and address. Yet another example is the directory structure of file systems.

The same file name can be used in different directories, the files can be uniquely accessed via the pathnames.   
Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace.

This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories) Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

* **global names** of a module
* **local names** in a function or method invocation
* **built-in names**: this namespace contains built-in functions (e.g. abs(), camp(), ...) and built-in exception names

**Garbage Collection**

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

**Python XML Parser**

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs the Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX: Here, you register callbacks for events of interest and then let the parser proceed through the document.

This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API : This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree − based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

**Python Web Frameworks**

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

## Why are web frameworks useful?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing
2. HTML, XML, JSON, and other output format tinplating
3. Database manipulation
4. Security against Cross-site request forgery (CSRF) and other attacks
5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possible comes bundled with the framework while others have a minimal core package that is amenable to extensions provided by other packages.

## Comparing web frameworks

There is also a repository called [compare-python-web-frameworks](https://github.com/mattmakai/compare-python-web-frameworks) where the same web application is being coded with varying Python web frameworks, tinplating engines and object.

## Web framework resources

* When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
* Frameworks is a really well done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found sessions and database ORMs to be a helpful part of a framework when done well.
* What is a web framework? Is an in-depth explanation of what web frameworks are and their relation to web servers?
* Jingo vs. Flash vs. Pyramid: Choosing a Python web framework contains background information and code comparisons for similar web applications built in these three big Python frameworks.
* This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
* Python’s web frameworks benchmarks  is a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.
* What web frameworks do you use and why are they awesome? Is a language agnostic Reedit discussion on web frameworks? It's interesting to see what programmers in other languages like and dislike about their suite of web frameworks compared to the main Python frameworks.
* This user-voted question & answer site asked "What are the best general purpose Python web frameworks usable in production?” The votes aren't as important as the list of the many frameworks that are available to Python developers.

## Web frameworks learning checklist

1. Choose a major Python web framework (Jingo or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.
2. Work through a detailed tutorial found within the resources links on the framework's page.
3. Study open source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.
4. Build the first simple iteration of your web application then go to the [deployment](https://www.fullstackpython.com/deployment.html) section to make it accessible on the web.

**2. SYSTEM STUDY**

**2.1 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* ECONOMICAL FEASIBILITY
* TECHNICAL FEASIBILITY
* SOCIAL FEASIBILITY

**ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

**SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### **6. SYSTEM TESTING**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

**TYPES OF TESTS**

**Unit testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

**Integration testing**

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

**Functional test**

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

**SYSTEM TEST**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

**White Box Testing**

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

**Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

**6.1 Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

**Test strategy and approach**

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# 6.2 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**6.3 Acceptance Testing**

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:** All the test cases mentioned above passed successfully. No defects encountered.

**CONCLUSION:**

The innovative outcome suggests that the advanced approach is a worth, which can distinctly support an accurate diagnosis of fruit diseases in a minor computational effort. It also dedicates future study on automatically estimating the severity of the disease.

An image processing derived solution is proposed for detection of grape, apple and pomegranate fruit disease. For Grape -Black Rot, Powdery Mildew, Downy Mildew; For Apple -Apple Scab, Apple Rot, Apple Blotch; For Pomegranate -Bacterial Blight, Aspergillus Fruit Rot, Gray Mold diseases are detected and classified. Once diseases are detected proper treatments are suggested accordingly.

It would also promote Indian 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 paper is to enhance the value of automatic fruit disease detection.

**FUTURE SCOPE:**

This application can be embedded in IOT and other autonomous farming system. Such a system allows for faster harvesting and better fruit safety.

**REFERENCES**

[1] Monica Jhuria, Ashwini Kumar, Rushikesh Borse “Image Processing for Smart Farming: Detection of Disease and Fruit Grading” Proceeding of the 2013 IEEE Second International Conference on Image Processing.

[2] Sudhir Rao Rupanagudi, Ranjani B.S., Prathik Nagaraj ,Varsha G. Bhat “A Cost Effective Tomato Maturity Grading System using Image Processing for 974 2015 International Conference on Green Computing and Internet of Things (ICGCIoT) Farmers” International Conference on Contemporary Computing and Information ,2014.

[3] Shiv Ram Dubey, Anand Singh Jalal “Adapted Approach for Fruit Disease Identification using Images”.

[4] Manisha A. Bhange, Prof. H. A. Hingoliwala “A Review of Image Processing for Pomegranate Disease Detection” International Journal of Computer Science and Information Technologies, Vol. 6 (1), 2015, 92-94.

[5] Hetal N. Patel, Dr. M. V. Joshi “Fruit Detection using Improved Multiple Features based Algorithm” International Journal of Computer Applications (0975 – 8887), Volume 13– No.2, January 2011.

[6] Abraham Gastélum-Barrios, Rafael A. BorquezLópez, Enrique Rico-García, Manuel ToledanoAyala and Genaro M. Soto-Zarazúa\* “Tomato Quality Evaluation with Image processing: A review” African Journal of Agricultural Research Vol. 6(14), pp. 3333-3339, 18 July, 2011.

[7] H. Al-Hiary, S. Bani-Ahmad, M. Reyalat, M. Braik and Z. ALRahamneh “Fast and Accurate Detection and Classification of Plant Diseases” International Journal of Computer Applications (0975 – 8887) Volume 17– No.1, March 2011.

[8] P. Vimala Devi and K. Vijayarekha “Machine Vision Application to Locate Fruits, Detect Defects and Remove Noise: A Review” Vol.7 | No.1 | 104-113| January – March | 2014.

[9] Shiv Ram Dubey, A. S. Jalal “Detection and Classification of Apple Fruit Diseases using Complete Local Binary Patterns”.

[10] Rashmi Pandey, Sapan Naik, Roma Marfatia “Image Processing and Machine Learning for Automated Fruit Grading System: A Technical Review” International Journal of Computer Applications (0975 – 8887) Volume 81 – No 16, November 2013.

[11] Anshuka Srivastava, Swapnil Kumar Sharma “Development of a Robotic Navigator to Assist the Farmer in Field” Proceeding of the International Multi Conference of Engineers and Computer Scientists 2010 Vol. (2) IMECS 2010 March 17-19, Hong Kong.

[12] Savita N. Ghaiwat, Parul Arora “Detection and Classification of Plant Leaf Diseases Using Image processing Techniques: A Review” International Journal of Recent Advances in Engineering & Technology (IJRAET) ISSN (Online): 2347 - 2812, Volume-2, Issue - 3, 2014.

[13]Anand H. Kulkarni, Ashwin Patil R. K. “Applying image processing technique to detect plant diseases” International Journal of Modern Engineering Research (IJMER) Vol.2, Issue.5, Sep-Oct. 2012 pp3661-3664 .

[14]Pradnya Ravindra Narvekar, Mahesh Manik Kumbhar2, S. N. Patil “Grape Leaf Diseases Detection & Analysis using SGDM Matrix Method” International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3237:2007 certified organization ) Vol.2,Issue 3, March 2014.

[15] Tejal Deshpande, Sharmila Sengupta, K. S. Raghuvanshi “Grading & Identification of Disease in Pomegranate Leaf and Fruit” International Journal of Computer Science and Information Technologies, Vol. 5 (3), 2014, 4638-4645.

[16] Vinita Tajane, Prof. N.J. Janwe “Medicinal Plants Disease Identification Using Canny Edge Detection Algorithm, Histogram Analysis and CBIR” International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 6, June 2014.

[17]Jayamala K. Patil , Raj Kumar “Advances in Image Processing for Detection of Plant Disease” Journal of Advanced Bioinformatics Applications and Research ISSN 0976-2604 Vol. 2, Issue 2, June-2011, pp 135- 141.

[18]Shiv Ram Dubey, Pushkar Dixit, Nishant Singh, Jay Prakash Gupta “Infected Fruit Part Detection using K-Means Clustering Segmentation Technique” International Journal of Artificial Intelligence and Interactive Multimedia, Vol. 2, 2013.