Table of Contents

1 Abstract

This document proposes an automated system for plant recognition which is used to classify a leaf into appropriate taxonomies. Such information can be useful for botanists, industrialists, food engineers and physicians. In this work leaf images belonging to seven different plants were taken. 50 images of each plant were borrowed from the Flavia dataset [1]. In order to get a proper result it is necessary for the user to use a leaf image with a white background. Pre-processing and feature extraction techniques were used before using a pattern matcher to compare the information from this image with the ones in the database to get potential matches. For pattern matching, we used three classifiers, Support Vector Machine, K-Nearest Neighbours and a neural net namely Multi Layer Perceptron. Accuracies ranging from 90%-100% were obtained.

2 Introduction

Plants play an important role in our environment. Without plants there will be no existence of the earth's ecology. But in recent days, many types of plants are at the risk of extinction. Biodiversity is declining steadily throughout the world [2]. The current rate of extinction is largely the result of direct and indirect human activities [3]. Building accurate knowledge of the identity and the geographic distribution of plants is essential for future biodiversity conservation [4]. To protect plants and to catalogue various types of flora diversities, so to conserve plant species it is necessary to set up a database. There are a huge number of plant species worldwide.

In a manual identification process, botanist use different plant characteristics as identification keys, which are examined sequentially and adaptively to identify plant species. In essence, a user of an identification key is answering a series of questions about one or more attributes of an unknown plant (e.g., shape, colour, number of petals, existence of thorns or hairs) continuously focusing on the most discriminating characteristics and narrowing down the set of candidate species. This series of answered questions leads eventually to the desired species. However, the determination of plant species from field observation requires a substantial botanical expertise. Traditional plant species identification is almost impossible for the general public and challenging even for professionals that deal with botanical problems daily, such as, conservationists, farmers, foresters, and landscape architects.

In recent times computer vision methodologies and pattern recognition techniques have been applied towards automated procedures of plant recognition to ease this process. By using computer aided methods even non-professionals can take part in it. The information generated form this system i.e. the taxonomy of the plant it classifies the leaf into can be used in various scenarios like

- One can search along of the details of the species to know more about it whether it is harmful or not.
- By knowing the plant botanists, farmers and other conservatives can know the environment needed for better nourishment of the plant.

This document talks about our implementation on such a plant recogniser. The project takes in an image and two click inputs form the user on image of the main stem of the leaf to ease the recognition process and get a better result.

For this project we narrowed our dataset to a list of 7 plants namely,

1. Acer Palmatum
2. Cedrus Deodara
3. Cercis Chinesis
4. Citrus Reticulata Blanco
5. Gingko Biloba
6. Liriodendron Chinense
7. Nerium Oleander

3 Software Requirement Specification

3.1 Purpose

The purpose of a Leaf Classification software is to reduce the manual labour in detecting a plant. Some systems employ descriptions used by botanists [5] - [8] but it is not easy to extract and transfer those features to a computer automatically. This project tries to have human interference in feature extraction as little as possible.

3.2 Product Scope

The scope of this project is to provide a classification of plant with the help of a picture of the leaf. The system can recognize and classify the pictures of leaves into 7 different types of plants. The classification of leaves is based on three machine learning models, Support Vector Machine, K-Nearest Neighbours and Multi Layer Perceptron.

3.3 Overall Description

3.3.1 Product Perspective

The main aim of the product is to recognize the leaf. This leaf picture can be clicked through any machine, the constraint being that this picture of leaf should be transferred to the appropriate folder in the desktop application. This Product will be helpful to farmer, agriculture related professionals and botanists. Rather than classifying leaf manually this product provides an accurate and efficient way to classify a plant with a simple picture of the leaf.

3.3.2 User Classes and Characteristics

A user with prior knowledge to Python environment and the rules to run the software will be able to use this product.

3.3.3 Operating Environment

The system is only a desktop application which works on any operating system configured with python development environment.

3.3.4 Design and Implementation Constraints

- The speed of the software is limited to the hardware of the system it is being run on.
- The system on the other hand requires a storage of around 200MB to hold the dataset of images as well as the software code.
- System is not compatible with the mobile interface.
- Only those well versed with Python programming environment would be able to understand and run the system.

- The Python environment has to be have other modules like OpenCV, Numpy, Scipy, Skimage, Sklearn and their dependency as the system uses these libraries to make the development process easier, faster and more efficient.
- The database used for storing feature vectors of images is Sqlite. Thus the user also has to have a general information of this platform.

3.3.5 Assumptions and Dependencies

The following assumptions are made, these assumptions could be considered as basic rules to follow so as to get the best possible classification

- User needs to give good resolution picture of the leaf.
- The picture resides in the appropriate folder of the desktop application.
- The picture should contain exactly one leaf.
- The background of leaf in picture should be white.
- The single image should be bounded inside image frame.

3.4 Interface Requirements

3.4.1 User Interfaces

The only user interface the system has is showing the user an image of the leaf that the user uploaded so as to get clicks on the main stem of the leaf.

3.4.2 Software Interfaces

The project was implemented in python language and various modules/libraries were used namely,

1. OpenCV: OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. In simple language it is library used for Image Processing. It is mainly used to do all the operation related to Images.
2. Numpy: It is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
3. SciPy: SciPy is an open-source Python library used for scientific computing and technical computing. It contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering.
4. Skimage: It is an open-source Python library which is a collection of algorithms for image processing.
5. Sklearn: It is an open-source, commercially usable Python library with simple and efficient tools for data mining and analysis.

3.5 System Requirements

ID	R 1
Title	The system recognizes and classifies the plant
Input	A picture of the leaf
Output	Taxonomy of the leaf.
Description	The system takes an image of leaf from user which he/she wants to know the taxonomy of, identifies leaf and provide necessary details of the plant as output.

ID	R 1.1
Title	The system pre-processes the leaf
Input	A picture of the leaf
Output	A processed leaf.
Description	The system takes an image of leaf from user which he/she wants to know the taxonomy of, it then performs various alterations to the copy of these image. These alterations include grey scaling, blurring, rotating, thresholding, cropping, etc. The reason to do so is to obtain an accurate feature vector.

ID	R 1.2
Title	The system extracts feature of an image
Input	An original picture of the leaf. A Processed image of the leaf.
Output	A feature vector.
Description	The system takes two input images, one the original image and other a pre-processed image. Using these two images and various methods of feature extraction explained in detail in the following sections it outputs a feature vector pertaining to the image.

ID	R 1.3
Title	The system trains the model
Input	A dataset of feature vector
Output	Trained model dump
Description	The system trains the model if an already trained model does not exist or if the admin explicitly ran the training program. It takes an input of feature vectors and their respective output labels and creates a trained model. A pickle dump of these model is taken and given as output.

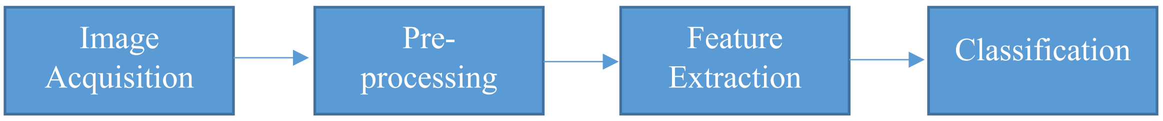
ID	R 1.4
Title	The system recognizes the leaf
Input	A feature vector. A trained model.
Output	Taxonomy of the leaf.
Description	The system takes a feature vector pertaining to the leaf it wants to classifies and inputs it into a trained model. The model them outputs its predicted taxonomy of the leaf.

3.6 Glossary

Name	**Description**
Dataset	It is a grouping for the different images of leafs that the system had to be pre-fed to train our machine learning model.
Pre-Processing	This consists of processing the image to obtain a more enhanced version so as to obtain better features.
Feature Vector	It consists of an array of values that were extracted using the various schemes described in the below sections
Database	It is a grouping of all the feature vectors pertaining to all the leaf images in our dataset.
Software	It is a combination of the Dataset, Database and the code/algorithm used to satisfy our project goal.
Taxonomy	The scientific name of the plant
Support Vector Machine	It is a classifier machine learning model.
K-Nearest Neighbours	It is a classifier machine learning model.
Multi Layer Perceptron	It is a neural network machine learning model.

4 Implementation

An image classification process can be broken down into the following steps:



4.1 Image Acquisition

The main goal of this step is to acquire the image that would be used to train the machine learning model. The images that we used in our project were supplied by the Flavia dataset [1]. These dataset contains images of 33 different plants at the time of when this document was written. For each plant there are about 40 to 70 images of leafs placed on a white paper.

4.2 Pre-processing

The aim of this step is to enhance the image data so as to be able to get better features and description vectors. The pre-processing process takes in an image as input and outputs a modified image. These modifications may be done repeatedly until satisfactory results aren't obtained.

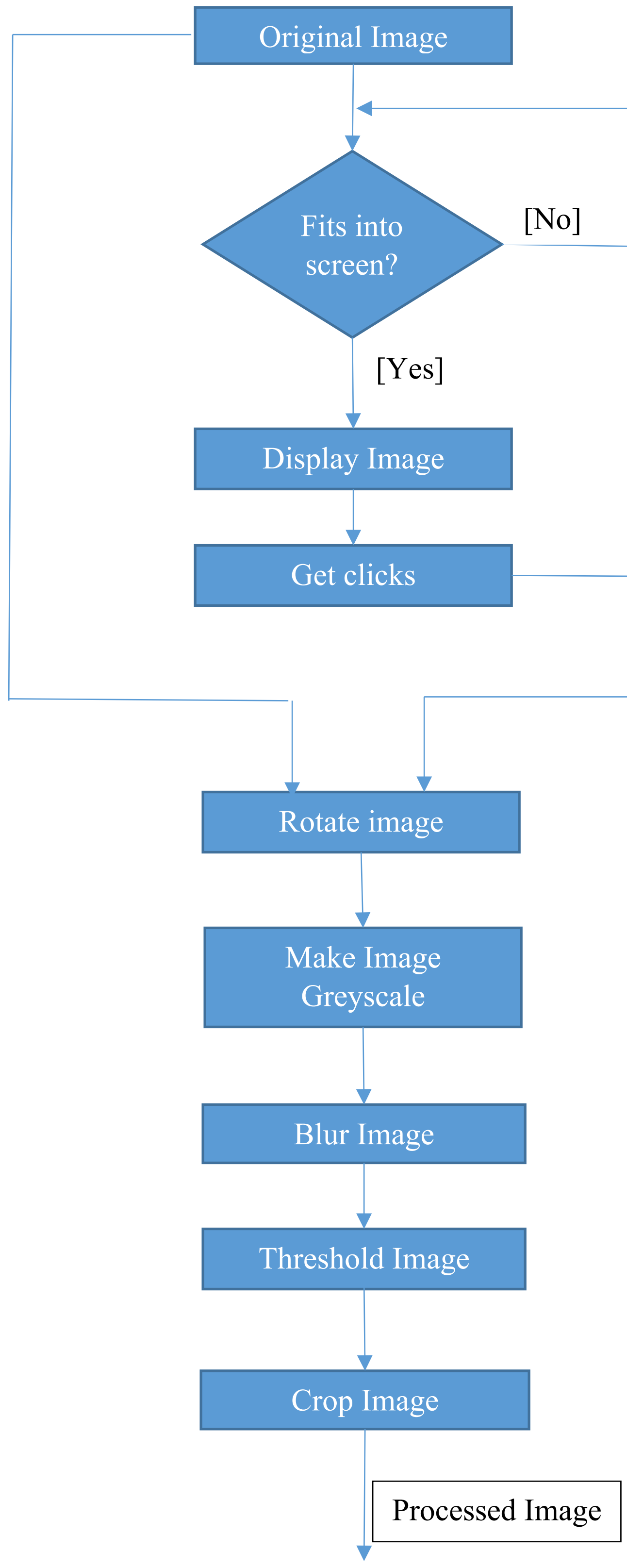
i. First we obtained an image from the user which he/she needs to classify.
ii. Then we repeatedly changed its size while maintaining its aspect ratio so as to fit the image onto the screen for user to witness.
iii. Next we ask the user for two clicks on the main stem of the leaf.
iv. From these two click, we obtain their respective coordinates and use these coordinates to calculate the slope of the main stem. From this slope we get the rotation angle. The following mathematical formula was used.

$$slope = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)$$

$$rotation\ angle = arctan(slope) * 180 * \frac{7}{22}$$

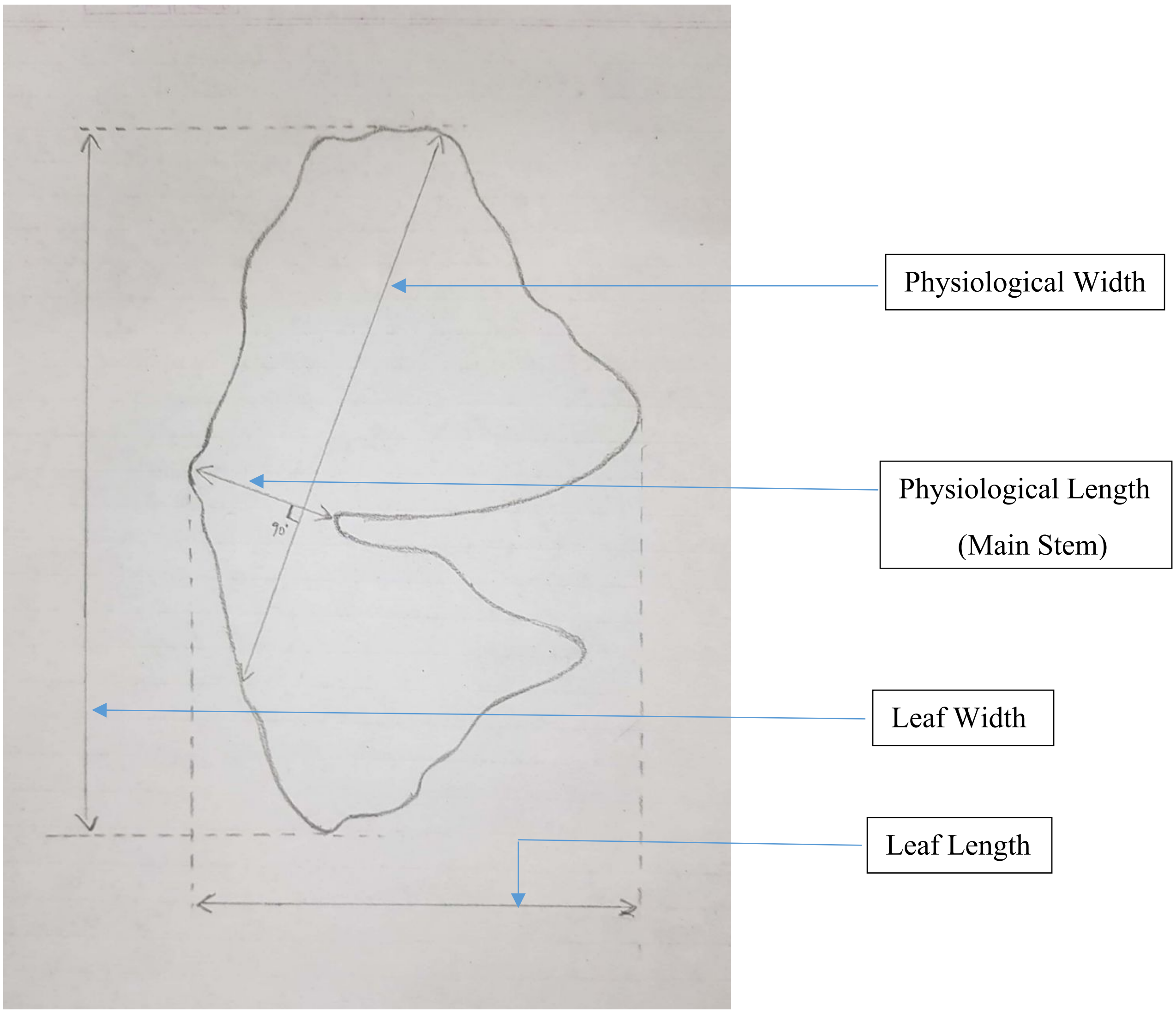
v. We rotate the original image using the rotation angle. We choose to use the original image so as to preserve as much as data of the leaf we can. By rotating the image we obtain the main stem of the leaf parallel to the horizontal axis.
vi. This rotated image of ours is actually a coloured image with RGB components to make computation faster we convert the image into greyscale where each pixel has a value ranging from 0 to 255.
vii. Then we blur the image so as to make it easier to extract the entire leaf rather than a part of it.
viii. We then pass this image into a threshold function which takes in a threshold value and changes the image into pixels of black and white.
ix. We then crop the image to only hold the black pixels i.e. the leaf.
x. This final image is our processed image which we would use to extract features.

The flow chart for the above steps is shown below



4.3 Feature Extraction

Feature extraction refers to taking measurements, geometric or otherwise, of possibly segmented, meaningful regions in the image. Features are described by a set of numbers that characterize some property of the plant or the plant's organs captured in the images (aka descriptors)



This leaf silhouette belongs to one 'gingko biloba' one of the seven plants of our dataset. This leaf was specifically chosen to highlight the differences between physiological and leaf measurements.

To obtain a descriptive vector for our image we calculate many such measurements.

i. *Physiological Length:* This is the length of the main stem of the leaf and it is denoted by P_l
ii. *Physiological Width:* This is the width of the leaf which is orthogonal to the physiological length. It is denoted by P_w

iii. *Leaf Length:* The distance between the farthest two points on the leaf silhouette which is parallel to the main stem. It is denoted by L_l.

iv. *Leaf Width:* The distance between the farthest two points on the leaf silhouette which is orthogonal to the main stem. It is denoted by L_w.

v. *Diameter:* It is the greater of the 4 values, Physiological length, physiological width, leaf length and leaf width. It is denoted by D.

vi. *Leaf Area:* Leaf area is calculated by the number of continuous black pixels. This can be calculated easily using the contour function of OpenCV. It is denoted as A

vii. *Leaf Perimeter:* Leaf perimeter can be calculated by counting the number of black pixels on the leaf margin. This can too be calculated using the contour function of OpenCV. It is denoted as P.

viii. *Aspect Ratio*: It is denoted by the ratio of physiological length P_l to physiological width P_w i.e. $\frac{P_l}{P_w}$.

ix. *Form Factor*: This feature is used to describe the difference between a leaf and a circle. It is calculated using the formula $\frac{4\pi A}{P^2}$, where A is the leaf area and P is the perimeter of the leaf margin.

x. *Rectangularity:* Rectangularity describes the similarity between a leaf and a rectangle. It is calculated using the formula$(P_l * P_w)/A$, where P_l is the physiological length, P_w is the physiological width and A is the leaf area.

xi. *Narrow Factor:* Narrow factor is defined as the ratio of the diameter D and physiological length L_p, thus D/L_p.

xii. *Perimeter Ratio of Diameter:* Ratio of perimeter to diameter, representing the ratio of leaf perimeter P and leaf diameter D, is calculated by P/D.

xiii. *Perimeter Ratio of Physiological Length and Physiological Width:* This feature is defined as the ratio of leaf perimeter P and the sum of physiological length P_l and physiological width P_w, thus $P/(P_l + P_w)$.

xiv. *Hu Moments*: Hu Moments are normally extracted from the silhouette or outline of an object in an image. By describing the silhouette or outline of an object, we are able to extract a shape feature vector (i.e. a list of numbers) to represent the shape of the object. Thus we take the processed image from pre-processing step and inverse its pixels and then calculate it hu moments.

xv. *Local Binary Patterns:* This field is used to get a vector which describes the texture of the leaf. The LBP feature vector, in its simplest form, is created in the following manner:

a. Divide the examined window into cells (e.g. 16x16 pixels for each cell).
b. For each pixel in a cell, compare the pixel to each of its 8 neighbours (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
c. Where the centre pixel's value is greater than the neighbour's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
d. Compute the histogram, over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are

greater than the centre). This histogram can be seen as a 256-dimensional feature vector.

e. Optionally normalize the histogram.

f. Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

Thus to calculate the texture i.e. the LBP feature vector we need the greyscale of the original coloured image and not the processed image which was the output of pre-processing step.

These 14 features that we extracted and/or calculated would be used to train the classifier in the next and final step.

4.4 Classification

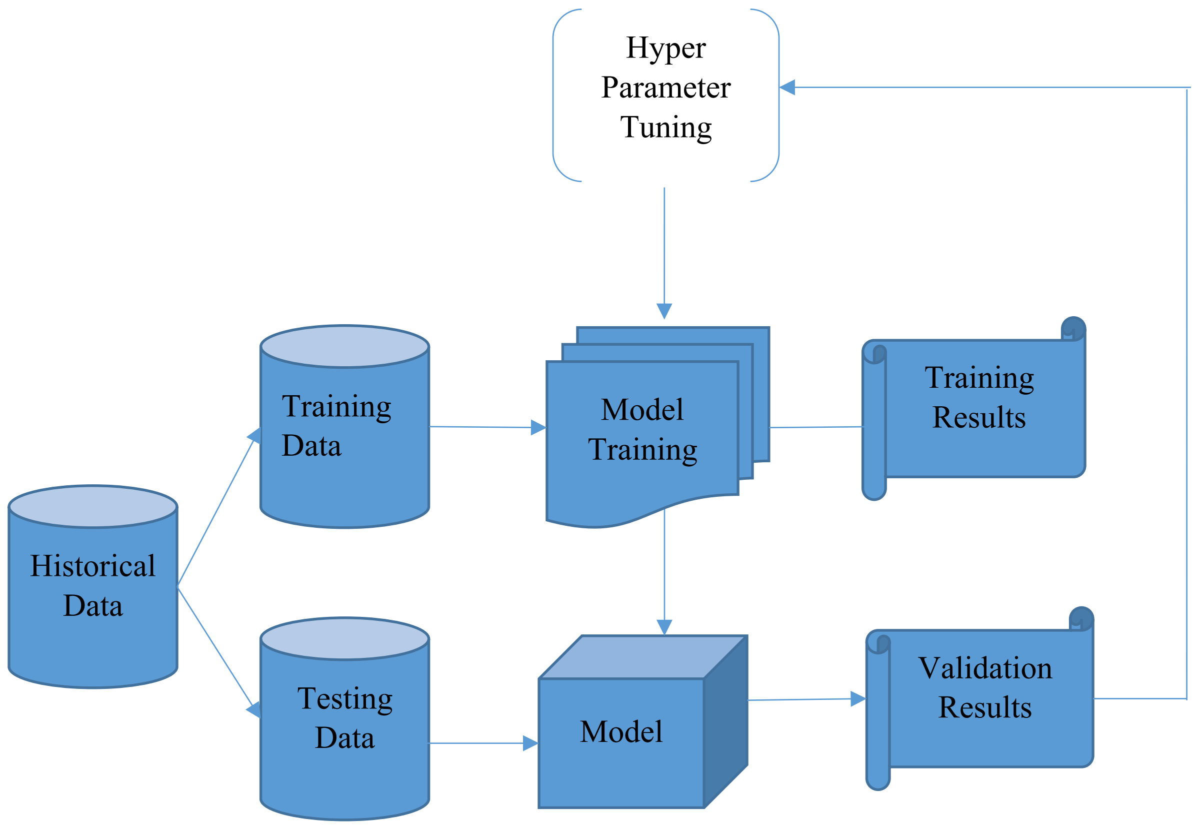
For our pattern matching component we have used three classifiers: Support Vector Machines (SVM) and K-Nearest Neighbours (KNN) and a Neural Network Multi Layer Perceptron.

The model is trained by passing the feature vector of each of the 50 images of the 7 plants along with an output label which tells the model which leaf it is. A dump of these 3 trained models is taken into a pickle format. By doing so, we don’t have to train the model every time before we classify an unknown leaf.

The feature vector of the unknown leaf is given to each of the 3 trained models. These trained models then output their predicted plant label. From these the plant label which was classified by maximum number of classifiers is taken as the predicted label.

5 Testing

The testing of a machine learning model can be done shown diagrammatically as follows:



To Test the accuracy of our three machine learning models. We followed the following steps

i. First we split our dataset of feature vectors into 80 – 20 percent ratio. This ratio was arbitrarily decided based on normalized accuracy testing rules of machine learning.
ii. The 80% given to each of the models along with its output label to train it.
iii. Then the 20% part was given to each of the models without the output label, to obtain the predicted output label of that model.
iv. The deviation from the actual and the predicted labels was calculated and the parameters which were passed to the model during the initial stage were tuned and the process was repeated till a suitable outcome was obtained. The parameters for the SVM model include: Gamma and cost. The parameters for KNN model include: Number of neighbours, Function used to weight points and the algorithm used to compute the nearest neighbours.
v. Once a suitable outcome was produced a confusion matrix was plotted to understand which labels were predicted wrong and how many.

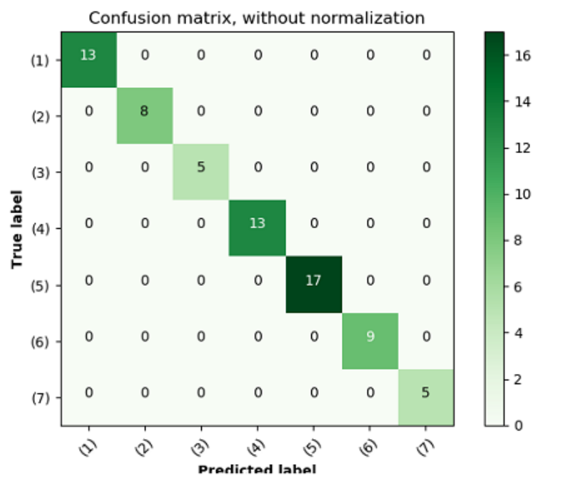
The testing phase resulted into the following parameters and accuracy

i. Support Vector Machine (SVM)
Cost = 1
Gamma = 0.03
Accuracy = 100%

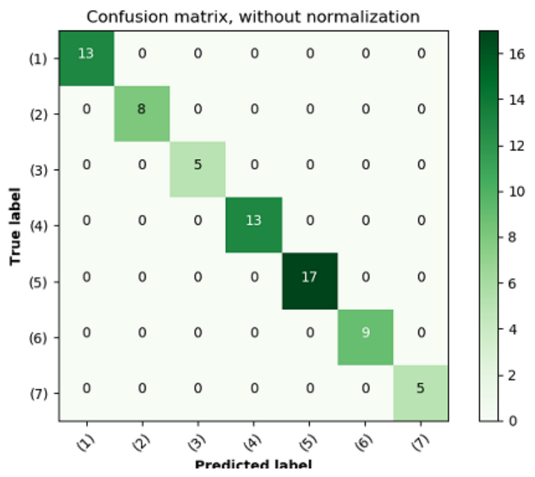
ii. K-Nearest Neighbour (KNN)
Number of Neighbours = 3
Function used to weigh = distance
Algorithm used to compute nearest neighbours = ball_tree
Accuracy = 98.5 %

iii. Multi Layer Perceptron (MLP)
Accuracy = 100%

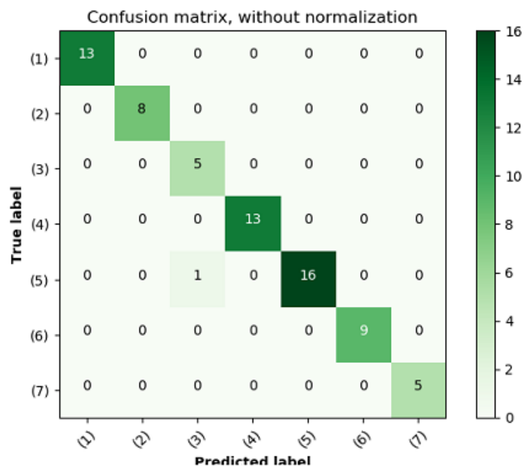
These are the various confusion matrices plotted for our models.



SVM Model



KNN Model

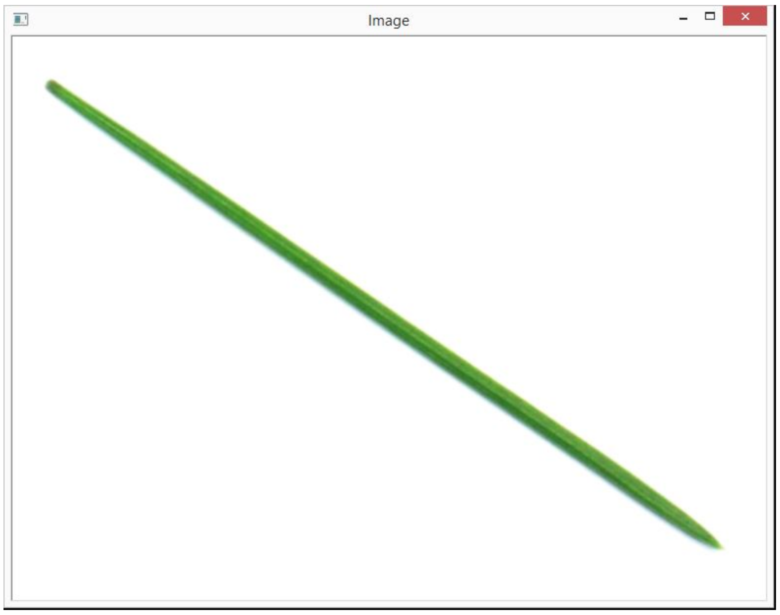


MLP Model

6 Screenshots

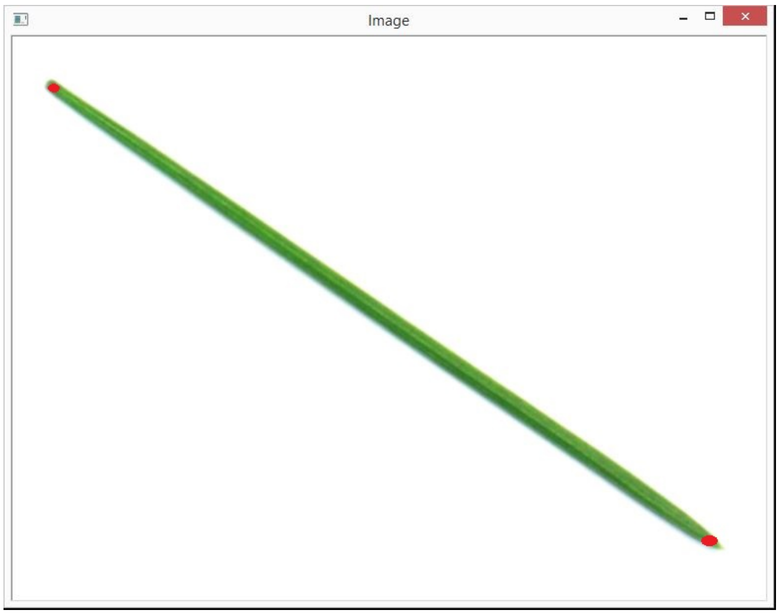
6.1 Original Image

This image is shown to the user to obtain clicks on the main stem of the leaf.



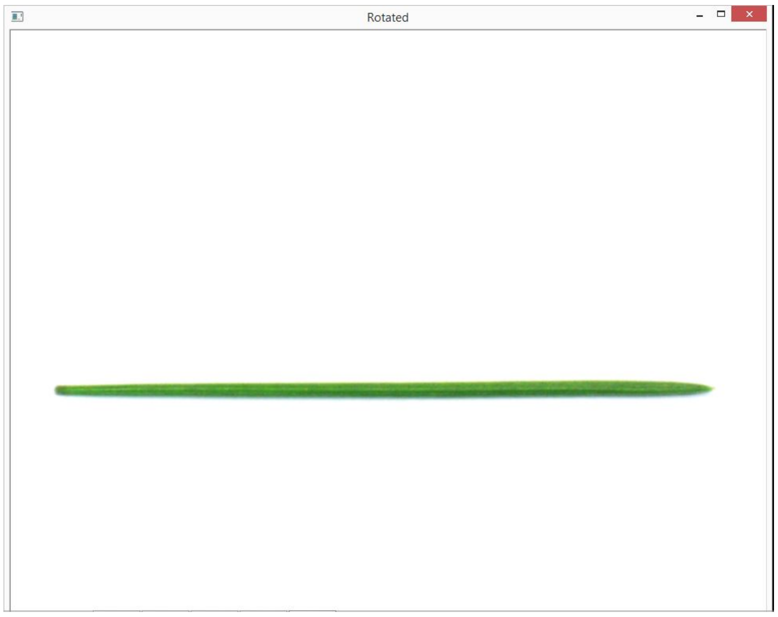
6.2 User clicks

The image produced depicting the clicks of the user. The red dots are the places where the user clicked.



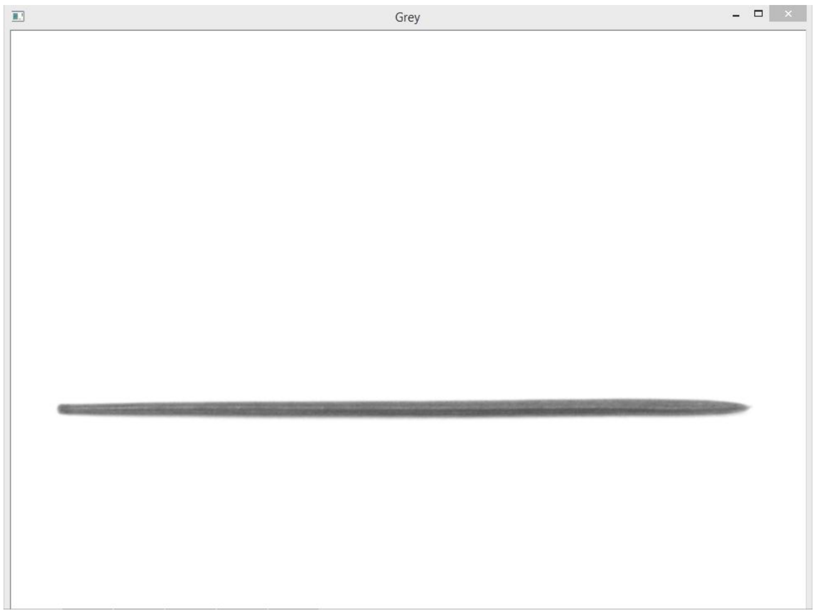
6.3 Rotated Image

The original image is rotated so as to have its main stem parallel to the horizontal axis



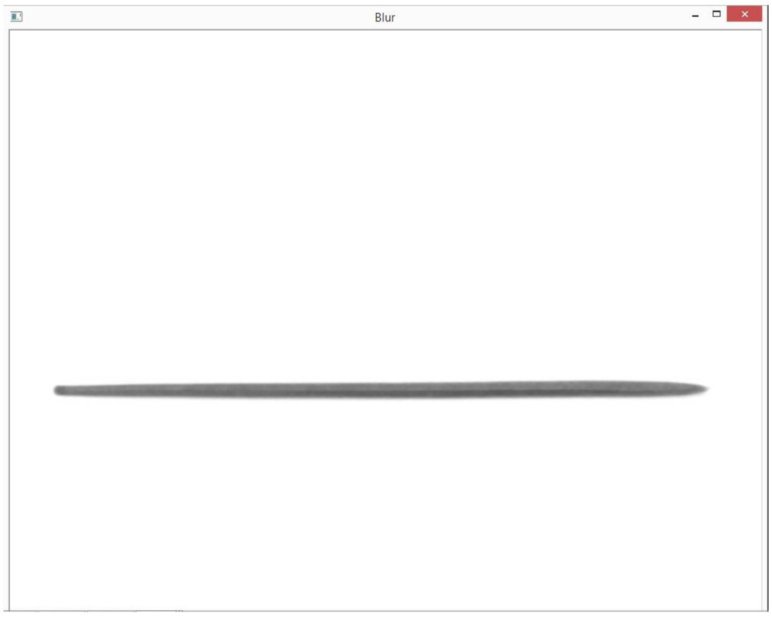
6.4 Grey scaled Image

The grey scale of the image is obtained as below



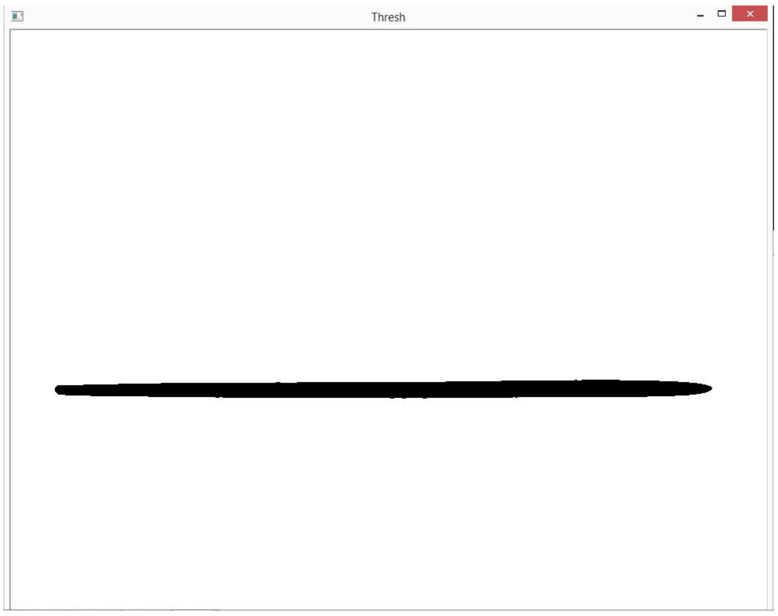
6.5 Blurred Image

The blurred version of the grey scaled image is obtained.



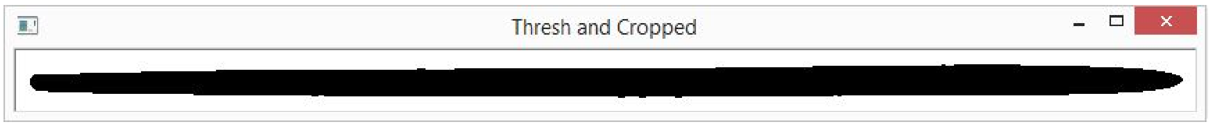
6.6 Binary Image

The threshold value is passed along with the blurred image to obtain a binary image with leaf as black pixels and the rest as white.



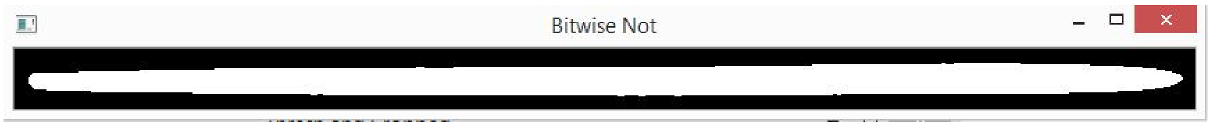
6.7 Cropped Image

The thresholded image is cropped so as to remove extra white spaces and other objects in the image. By doing so only the leaf is left in the image.



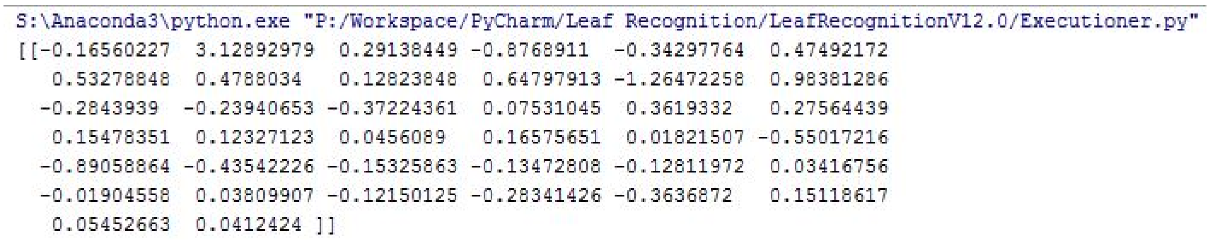
6.8 Bitwise Not Image

The bitwise not image is obtained by inverting pixels of cropped image above.



6.9 Feature Vector

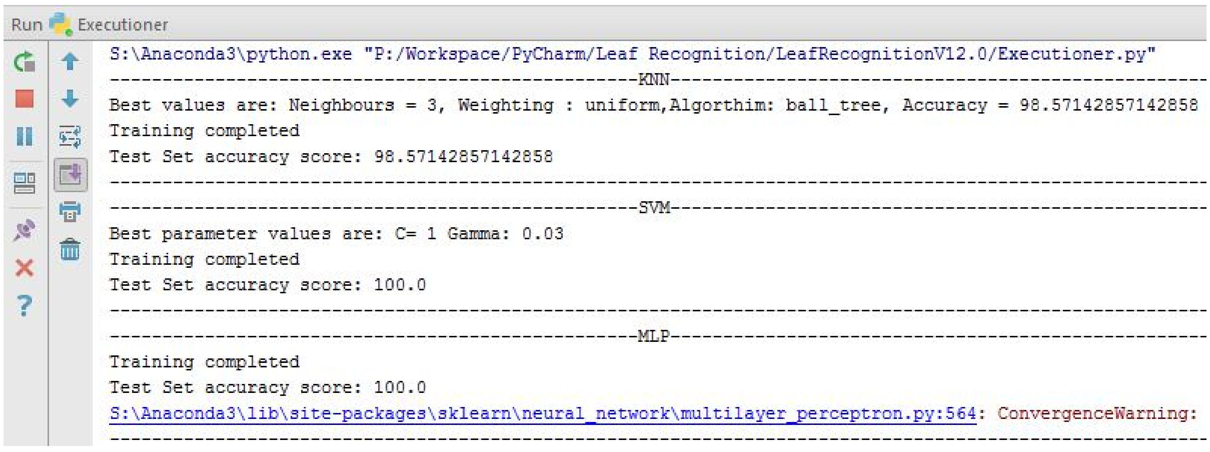
A sample feature vector is shown in the below image. This feature vector is of the above leaf image selected.



```
S:\Anaconda3\python.exe "P:/Workspace/PyCharm/Leaf Recognition/LeafRecognitionV12.0/Executioner.py"
[[-0.16560227  3.12892979  0.29138449 -0.8768911  -0.34297764  0.47492172
   0.53278848  0.4788034   0.12823848  0.64797913 -1.26472258  0.98381286
  -0.2843939  -0.23940653 -0.37224361  0.07531045  0.3619332   0.27564439
   0.15478351  0.12327123  0.0456089   0.16575651  0.01821507 -0.55017216
  -0.89058864 -0.43542226 -0.15325863 -0.13472808 -0.12811972  0.03416756
  -0.01904558  0.03809907 -0.12150125 -0.28341426 -0.3636872   0.15118617
   0.05452663  0.0412424 ]]
```

6.10 Model Trained

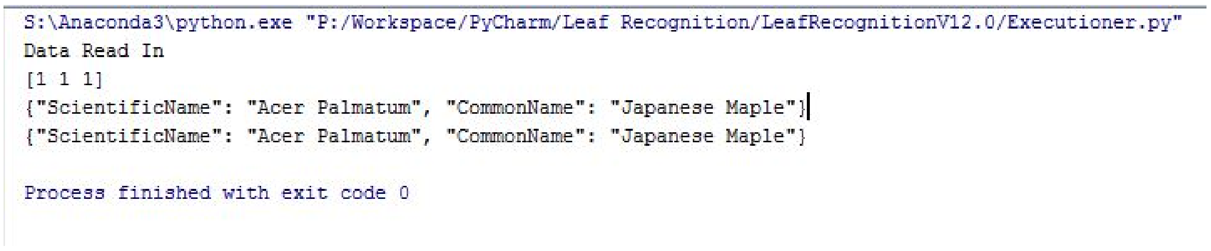
The output of training the model. It consists of the tuned parameters selected and its accuracy.



```
S:\Anaconda3\python.exe "P:/Workspace/PyCharm/Leaf Recognition/LeafRecognitionV12.0/Executioner.py"
-----------------------------------------------------KNN-----------------------------------------------------
Best values are: Neighbours = 3, Weighting : uniform,Algorthim: ball_tree, Accuracy = 98.57142857142858
Training completed
Test Set accuracy score: 98.57142857142858
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------SVM-----------------------------------------------------
Best parameter values are: C= 1 Gamma: 0.03
Training completed
Test Set accuracy score: 100.0
-------------------------------------------------------------------------------------------------------------
-----------------------------------------------------MLP-----------------------------------------------------
Training completed
Test Set accuracy score: 100.0
S:\Anaconda3\lib\site-packages\sklearn\neural_network\multilayer_perceptron.py:564: ConvergenceWarning:
-------------------------------------------------------------------------------------------------------------
```

6.11 Taxonomy Prediction

It consists the output labels predicted by the three trained models. In this case the value [1, 1, 1]. Then from this label the taxonomy of the leaf is obtained and displayed to the user.



```
S:\Anaconda3\python.exe "P:/Workspace/PyCharm/Leaf Recognition/LeafRecognitionV12.0/Executioner.py"
Data Read In
[1 1 1]
{"ScientificName": "Acer Palmatum", "CommonName": "Japanese Maple"}
{"ScientificName": "Acer Palmatum", "CommonName": "Japanese Maple"}

Process finished with exit code 0
```

7 Conclusion

The software does indeed detect leafs belonging to the 7 chosen dataset leafs. All the afar mentioned steps in the implementation part were converted into corresponding functions and code. Our initial assumption that a computer aided plant recognition would drastically reduce the complexity of detecting leafs does indeed hold true. Using our three trained models, Support Vector Machine, K-Nearest Neighbour and Mutli layer Perceptron with 90 – 100% accuracy we were able to properly classify the leafs belonging to the 7 plants we chose.

8 Limitation and Future Extension

Our software works only on python configured systems and thus making it inaccessible to common systems which in turn renders it useless for the common man. So our first future extension would be to shift the projects trained model onto a server and use a platform like website, mobile apps with a simple user interface so that any one from anywhere could simply access the trained models data to predict a plants name.

The software's next limitation is that it only classifies leafs into 7 plants as it was trained with the dataset of these 7 plants. In reality there are thousands of plant species, so once we make the software accessible to the common man we would be faced with a challenge to increase our dataset while keeping the accuracy constant. So out next future extension would be to increase the number of plants into which we could classify a leaf and also to increase the image dataset of each of these plants.

The software only gives the output as the taxonomy of the plant it predicted, thus making the user go to other websites to obtain information on that plant. Once the software is being used by people it becomes extremely important that we not only give the user the taxonomy of the plant into which the model classified his/her leaf but also with further information on the leaf and the plant. These further information may include the environmental conditions needed for the plants survival, places to which this plant is indigenous too, the common diseases that these plants have and their prevention and remedies, etc.

9 Bibliography

[1] W. S. Gang, B. F. Sheng, X. E. You, W. Yu-Xuan, C. Yi-Fan and S. Chiao-Liang, “A Leaf Recognition Algorithm for Plant classification Using Probabilistic Neural Network,” IEEE 7th International Symposium on Signal Processing and Information Technology, Cario, Egypt, December, 2017.

[2] S. Pimm, C. Jenkins, R. Abell, T. Brooks, J. Gittleman, L. Joppa, P. Raven, C. Roberts and J. Sexton, “The biodiversity of species and their rates of extinction, distribution, and protection,” American Association for the Advancement of Science, 2014.

[3] G. Murphy and T. Romanuk, “A meta-analysis of declines in local species richness from human disturbances.,” Ecology and Evolution, 2013.

[4] A. Joly, H. Müller, H. Goëau, H. Glotin, C. Spampinato, A. Rauber, P. Bonnet, W. Vellinga and B. Fisher, “International workshop on environmental multimedia retrieval 2014.,” Glasglow, 2014.

[5] M. J. Dallwitz, “A general system for coding taxonomic descriptions,” International Association for Plant Taxonomy (IAPT), 1980.

[6] T. Brendel, J. Schwanke, P. Jensch and R. Megnet, “Knowledge based object recognition for different morphological classes of plants,” in *Proceedings of SPIE*, 1995.

[7] D. Warren, “Automated leaf shape description for variety testing in chrysanthemums,” in *Proceedings of IEE 6th International Conference Image Processing and Its Applications*, 1997.

[8] H. Fu, Z. Chi, D. Feng and J. Song, “Machine learning techniques for ontology based leaf classification,” in *IEEE 8th International Conference on Control, Automation, Robotics and Vision*, Kumming, China, 2004.