1. **AIM: -** An algorithm has been written for detection of diseased and a healthy plant which works on classification of both the types of plant. A python code has been made using the reference paper attached in the assignment section. We have taken into mind the algorithm followed in the research paper and made it better and improved the accuracy.

2. **SOFTWARE USED: -**

a) Jupyter notebook
b) Various libraries of python used and imported in Jupyter notebook (eg. MatPlotLib etc.)
c) PlantVillage Dataset from Kaggle for training (around 740+ images).
d) A leaf image for testing.
e) Python Programming Software.
f) Anaconda Navigator.

3. **PROCEDURE FOLLOWED**

a) All the various types of machine learning and image processing modules are imported.
b) Firstly, the RGB image is extracted from BGR format.
c) Next, we convert the RGB image to HSV format
d) Next, we perform image segmentation and extract green and brown colour from the leaf.
e) Next, we find the Hu moments and find another feature known as haralick texture. The output of these functions is evaluated towards the end.
f) Next we start extracting the training labels and the feature vectors from the training dataset of the image.
g) In the next cell we first read the image and resize it according to the functions in the cv2 module. We call the global feature functions that we made in the above cells of the jupyter notebook and process the image. We keep updating the list of labels and feature vectors.
h) Next we extract the overall training label size and feature vector size.
i) Next, we start the training of the models using Logistic Regression, LDA, KNeighbours Classification, Decision Tree Classification, Random Forest Classification and Gaussian Naïve Bayes and the last algorithm of machine learning used in the training is Support Vector Machine.
j) We import the feature vector and the training labels.
k) Next, we plot a box plot which depicts the comparison between various types of algorithms used in the above step.
l) We start the prediction testing using the testing dataset and prepare a proper classification report along with the heatmap using MatPlotLib.
m) Next, we print the accuracy score which comes out to be **98.31649%**
n) Next, we take a testing image and perform the various image processing on it and predict the diseased areas in the image by the help of various image processing techniques described above.
o) We print the accuracy of the trained model once again.

4. COMPARISION BETWEEN OUR ALGORITHM AND THE ALGORITHM DESCRIBED IN THE PAPER.

With due respect to the content written in the paper and to the authors we perform a healthy comparison between both the algorithms used.

a) We find out the accuracy of the prediction of our model exceeds the accuracy of the model given in the paper by 1%.
b) We find out that more types of image processing techniques are used in this algorithm. We have used Threshold Binary and also have plotted a histogram showing intensity of pixels in the notebook.

COMPARISION BETWEEN ACCURACY.

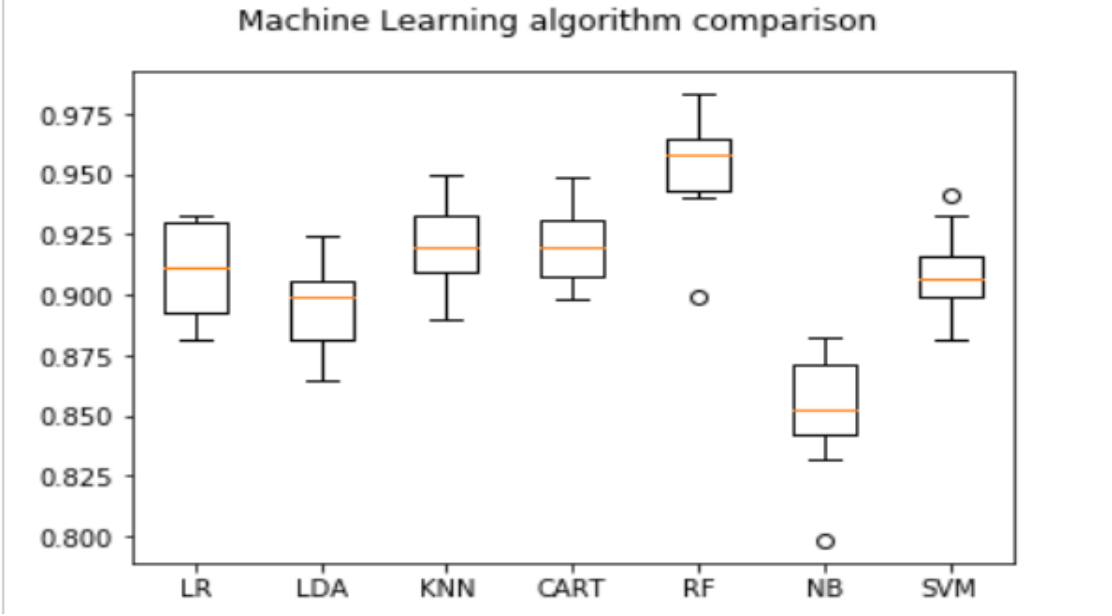
1. The accuracy of the paper given to us is shown in the image below.

Various Machine learning models	Accuracy(percent)
Logistic regression	65.33
Support vector machine	40.33
k- nearest neighbor	66.76
CART	64.66
Random Forests	70.14
Naïve Bayes	57.61

Accuracy according to our programming.

Various Machine Learning Models	Accuracy (in %)
Logistic Regression	91.0675
Support Vector Machine	90.8154
K – nearest neighbours	91.9947
CART	91.9954
Random Forest	95.2806
Naïve Bayes	85.2578

```
LR: 0.910675 (0.018653)
LDA: 0.893804 (0.019166)
KNN: 0.919947 (0.019009)
CART: 0.919954 (0.016633)
RF: 0.952806 (0.022024)
NB: 0.852578 (0.024386)
SVM: 0.908154 (0.018319)
```



5. OUTPUT IMAGES

```
train_labels = os.listdir(train_path)

# sort the training labels
train_labels.sort()
print(train_labels)

# empty lists to hold feature vectors an
global_features = []
labels          = []
```

```
['diseased', 'healthy']
```

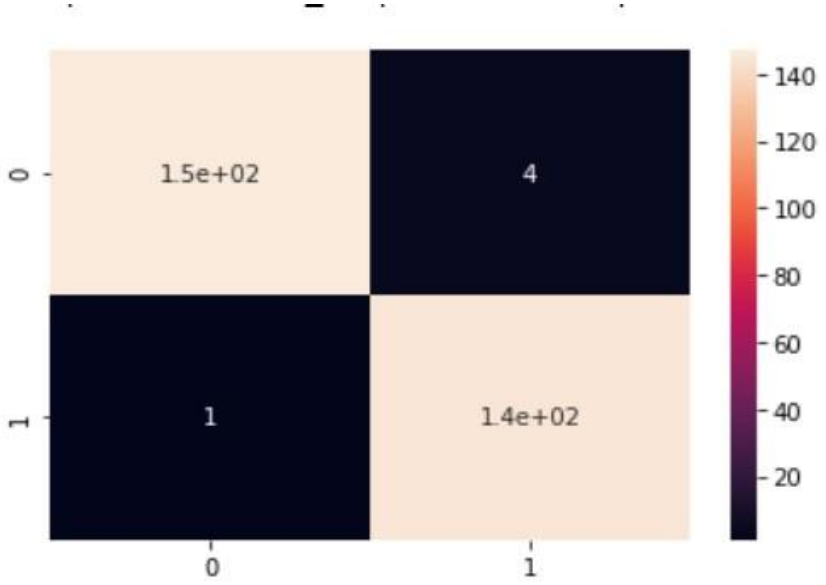
```
[STATUS] processed folder: diseased
[STATUS] processed folder: healthy
[STATUS] completed Global Feature Extraction...
```

```
: print("[STATUS] feature vector size {}".format(np.array(global_features).shape))
```

```
[STATUS] feature vector size (1484, 532)
```

```
: print("[STATUS] training Labels {}".format(np.array(labels).shape))
```

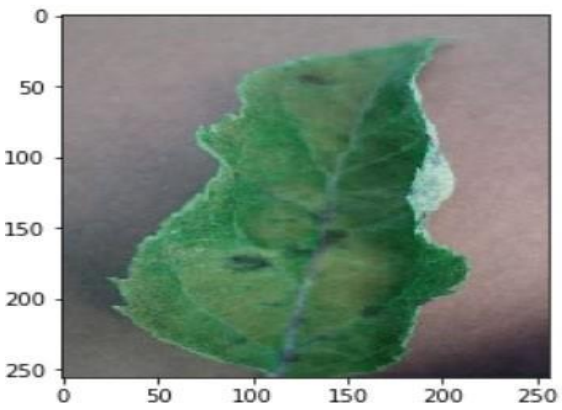
```
[STATUS] training Labels (1484,)
```



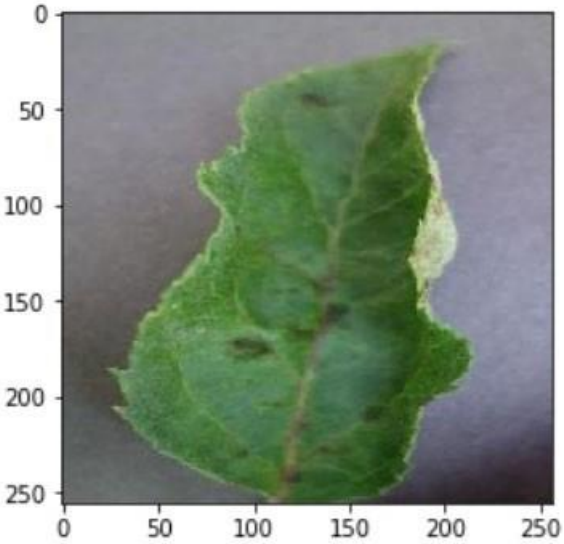

```
]: print(classification_report(testLabelsGlobal,y_predict))
```

	precision	recall	f1-score	support
0	0.99	0.97	0.98	151
1	0.97	0.99	0.98	146
accuracy			0.98	297
macro avg	0.98	0.98	0.98	297
weighted avg	0.98	0.98	0.98	297

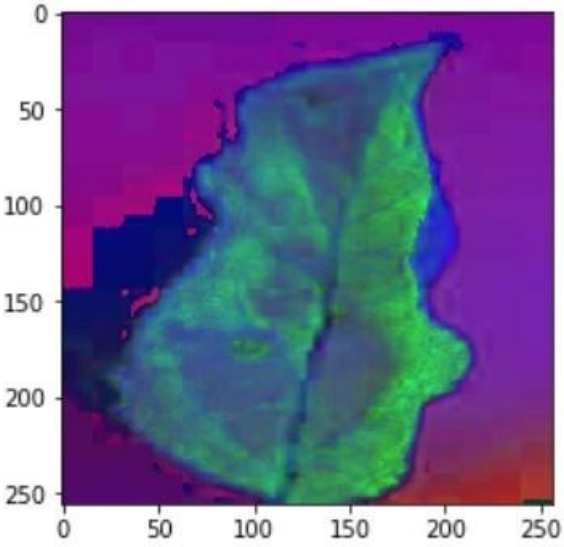
ORIGNAL IMAGE



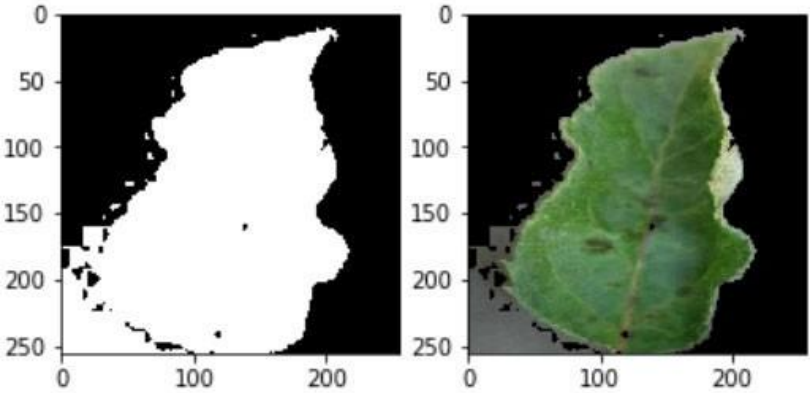
BGR TO RGB



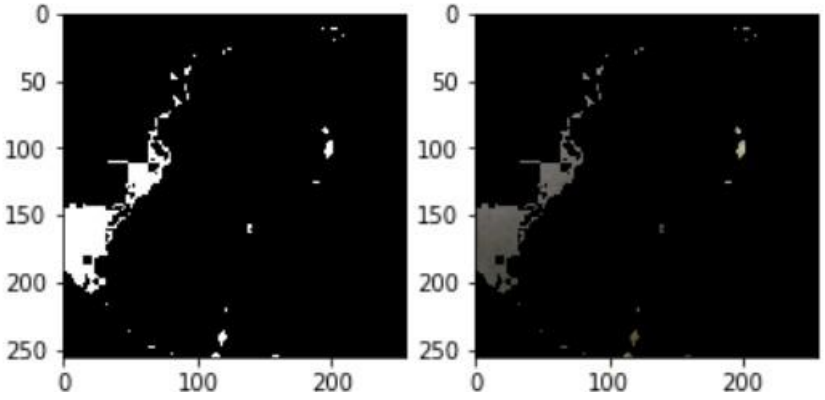
RGB TO HSV



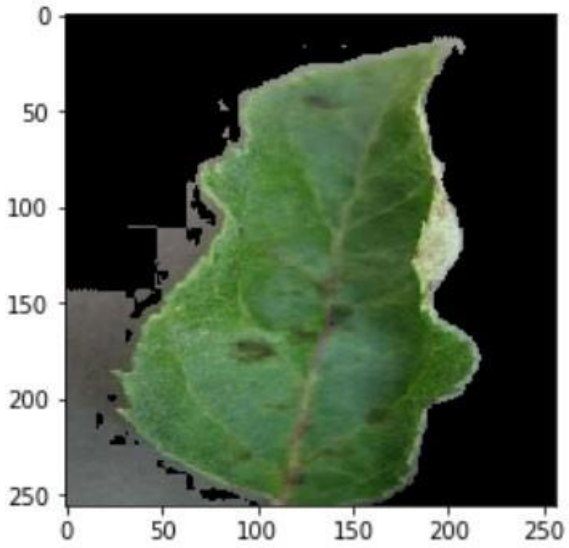
GRAYSCALE IMAGE



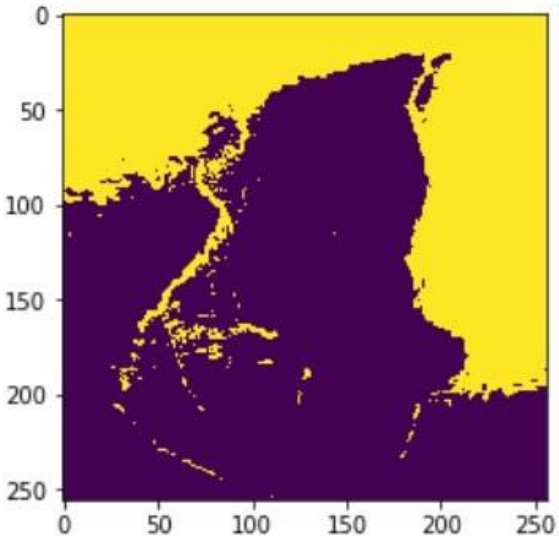
EXTRACTING PIXELS



FINAL RESULT



THRESH BINARY



Hu MOMENTS

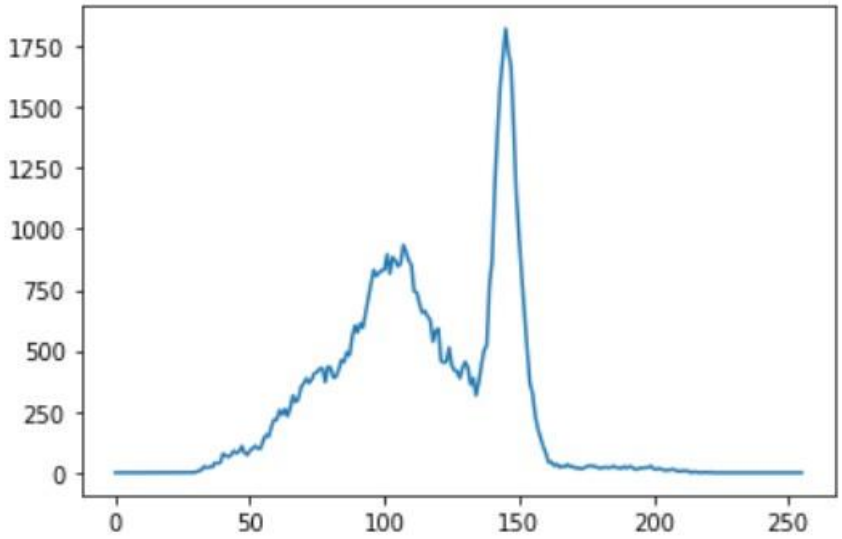
```
moments = cv2.moments(im)
huMoments = cv2.HuMoments(moments)
print(huMoments)
```

```
[[ 1.61198466e-03]
 [ 7.01185814e-07]
 [ 6.76590029e-10]
 [ 1.11789394e-10]
 [-2.44386158e-20]
 [-5.23014252e-14]
 [ 1.86537420e-20]]
```

HISTOGRAM

```
img = cv2.imread(r'C:\Users\aryan\Downloads\testing_image.jpg',0)
histr = cv2.calcHist([img],[0],None,[256],[0,256])

plt.plot(histr)
plt.show()
```



ACCURACY

```
In [61]: accuracy_score(testLabelsGlobal, y_predict)
```

```
Out[61]: 0.9831649831649831
```

```
In [66]: print("The accuracy of the prediction is: ")
         accuracy_score(testLabelsGlobal, y_predict)*100
```

```
         The accuracy of the prediction is:
```

```
Out[66]: 98.31649831649831
```

6. CONCLUSION

We conclude that the model can successfully classify the image between an healthy leaf and a d iseased leaf by the help of various image processing methods as well as machine learning algori thms specified above. **The accuracy of the model is 98.31649831649831 % as shown in the above image.**

THANK YOU

----------XXXXX----------XXXXXXX----------