A Report

On

Plant Disease detection using Machine Learning

BY

Aaryan Gupta	**2018B1A70775H**
Shubham Singla	**2019A3PS0392H**
Aditya Sharma	**2018A7PS0315H**
Doni Akhil Lohith	**2019A7PS0026H**
DSK Karthik	**2019A7PS0189H**

Under the guidance of

Dr. Paresh Saxena

SUBMITTED IN FULFILMENT OF THE REQUIREMENTS OF

BITS F464: Machine Learning

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI (RAJASTHAN)

HYDERABAD CAMPUS

(April 2022)

ACKNOWLEDGMENTS

We would like to thank Dr. Paresh Saxena, Assistant Professor, Department of Computer Sciences and Information Systems for his continuous and enthusiastic support, cooperation and help throughout the duration of this Assignment. We would also like to thank Prof. Sridhar Raju, Associate Dean, Academic and Undergraduate Studies Division and Prof. Runa Kumari, Associate Dean, Timetable Division, BITS Pilani, Hyderabad Campus for giving us the opportunity to register.

Table of Contents

Plant Disease detection using Machine Learning

Chapter 1: Introduction

About 70% of the rural population is dependent on Agriculture as a livelihood in India and the Plant Disease Detection is one of the major problems in agriculture that can be solved through Machine Learning technology nowadays. The main aim of Machine Learning is to get the training data and fit it into a model.. We train the machine learning model with a large amount of image dataset of both healthy and diseased images and then finally test it using the test dataset, which may assist the stakeholders in making good decisions that reduce their losses, are less expensive and takes less time than visual inspection by predicting the correct output. The colour of leaves, amount of damage to leaves, area of the leaf, texture parameters are used for prediction of leaves with labels as Diseased or Healthy. All these parameters or features are extracted from the images with the help of image processing techniques. New ways are being discovered still but the research community has already done a lot in this field using Machine Learning, Image processing, Deep Learning and Neural Networks.

A literature review on this topic is presented in the report with 5 selected papers, out of which implementation of a research paper is done, titled- 'Plant Disease detection using Machine Learning' by R. Shima et. al (2018) presented in International Conference on Design Innovations for 3Cs Compute Communicate Control.

After the implementation of the above mentioned paper, the results are presented, and then a proposed change is included in the original implementation. The results of proposed change and original implementation are compared to obtain the Main findings and conclusion. We propose a change in the form of a Hybrid KNN/Random-Forest and PSO (Particle Swarm Optimization), where we first perform feature subset selection through a metaheuristic approach i.e. PSO algorithm and afterwards we model the selected features with Random Forest Classifier, to obtain our predicted results.

Chapter 2: Literature Survey

A comparative study was done for the detection of disease in plants by using Different techniques of image processing and ML algorithms for the 5 submitted papers.

S. Ramesh et al. (2018) performed detection of diseased plants using Machine Learning techniques, like Random forest classifier to identify the diseased and healthy leaves of the plant. Some image processing techniques like image resizing, converting rgb to bgr and hsv, all this was done. They used three feature descriptors such as Hu Moments, Haralick Features and Colour Histogram to extract features. Training was done using random forest to classify the input images. It was modelled on 7 different algorithms and a comparison was made, and the best accuracy was given by Random Forest Classifier, where they got around 70% accuracy. The dataset used was a papaya leaf image dataset with 160 images, which is not in public domain.

P. Kulkarni et al. (2021) a public dataset for plant leaf disease detection called PlantVillage curated by S. Mohanty et Al which consists of 87000 RGB images of healthy and unhealthy plant leaves having 38 classes. Out of these, selected classes were chosen to do research upon. Background noise removal, Otsu's thresholding algorithm, foreground detection, image segmentation standard statistical calculations on perimeter, were some of the preprocessing techniques. They used feature selection based correlation and did the classification using random forest classifier. Following this, they developed a computer vision based system for plant disease detection with average 93% accuracy.

N. Paliwal et al. (2019) took inspiration from past work in the robotic field with respect to agriculture, to introduce the use of a common machine learning algorithm to support the execution of mixed cropping on a ground bot which could help the farmer in sowing the crops after soil analysis. They took 2000 images for classification with the help of K-means algorithm, feature extraction was done using Otsu's method, adaptive mean thresholding, image segmentation and field classification. An overall prediction accuracy of 69.89% was achieved.

A. Kalvakolanu (2020) talks about the detection of plant diseases using Deep Learning. The author considered three plants: Tomato, Potato and Bell pepper with diseases like late blight, mosaic virus etc. Transfer learning with input as plant leaves was done, with both ResNet 34 and ResNet 50 as the base model, which was trained with 4000 images with discriminative learning. The preprocessing steps included data augmentation, and also the images were rotated to make the dataset more generic. 99.44% accuracy was achieved with the proposed model.

K.K. Zaw et al. (2019) deal with the identification of diseases which affect leaves, stem and fruit of plants. For leaf disease identification and classification image processing techniques can be used. Bacterial Blight, Anthracnose, Alternaria Alternata are some of the common diseases. These diseases were identified and classified by using multiclass support vector machine and an accuracy of about 83.87% was obtained.

Chapter 3: Implementation of Paper

3.1.Dataset Details

The dataset used for this project has been taken from kaggle: Plant Village Dataset which can be found **here**. The Data fed for the modelling is of **Apple Leaves**. For training purposes, the Dataset comprises 2 folders named Diseased and Healthy which contains images of leaves with respective labels. The Diseased Folder contains 805 images of apple leaves affected by Apple Scab, Black Rot or Cedar Apple Rust and the healthy folder contains 904 leaf images of Healthy label. A total of 800 images for each class Diseased and Healthy is fed into our ML model.

3.1.1 Image Properties *(Type of File : JPG)*

Dimensions	256 x 256	**Horizontal Resolution**	96 dpi
Width	256 Pixels	**Vertical Resolution**	96 dpi
Height	256 Pixels	**Bit Depth**	24

3.2. Libraries Used

sklearn, numpy, pandas, h5py, os, cv2, matplotlib, mahotas

3.3. Data Pre-processing

1. **Resizing the Images:** Bringing all the images to a reduced uniform size of 500*500
2. **Conversion of image from RGB to BGR**. OpenCV library accepts the images in RGB colouring format so the conversion to original BGR format is required.
3. **Conversion of image from BGR to HSV.** Unlike RGB, HSV separates luma (image intensity), from chroma (colour information) which is useful for histogram equalisation of a colour image, which is done on the intensity component, and not on the colour components.
4. **Image Segmentation for extraction of Colours.** Used to separate the picture of the leaf from the background segmentation. The colour of the leaf is extracted from the image.
5. **Label Encoding:** According to the images situated in the folder the labels are encoded in numeric format for better understanding of the machine.
6. **Feature Scaling:** Normalisation of feature values so that they range from [0,1] using min-max normalisation to handle highly varying values.

3.4. Feature Extraction

3.4.1 Feature Descriptors

Global features are extracted from the image using three feature descriptors:

- **Colour Histogram:** It gives information on different types of colours and the number of pixels in each type of the colour.
- **Hu Moments:** It gives numerical representation of shape in the form of a vector.
- **Haralick Texture:** It gives information on the texture or feel of the surface in the image.

After extracting the features of images, the features are stacked together in an array.

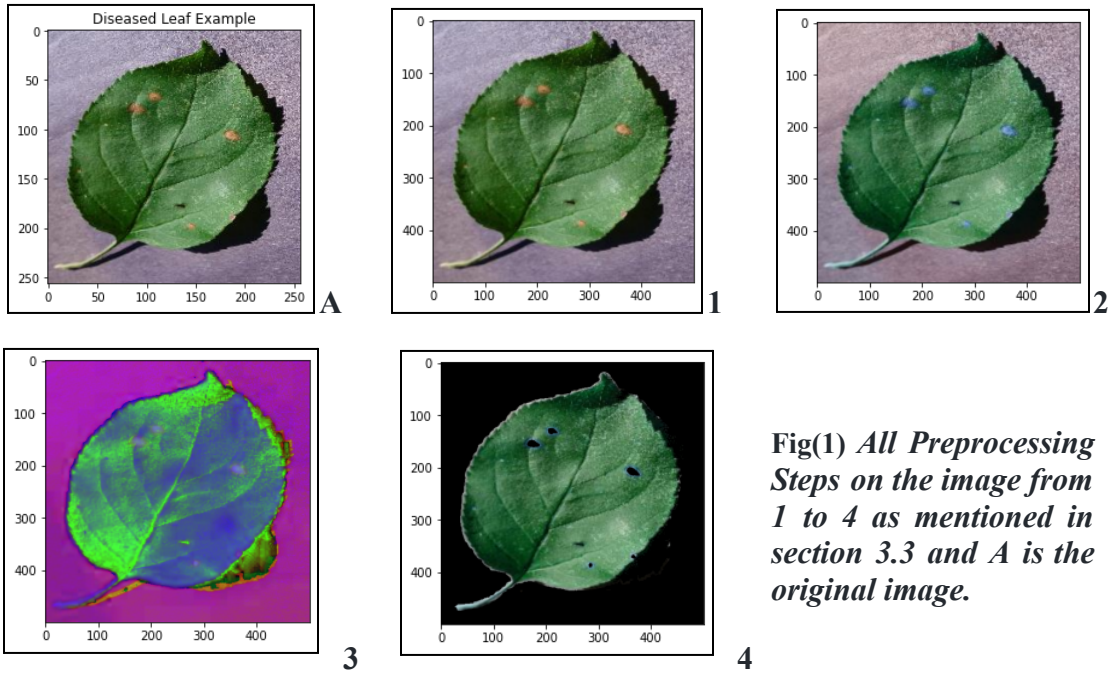
3.4.2 HDF5 for saving the features

We used HDF5 (Hierarchical Data Format version 5) for saving the extracted features as it supports easy retrieval of large and complex data.

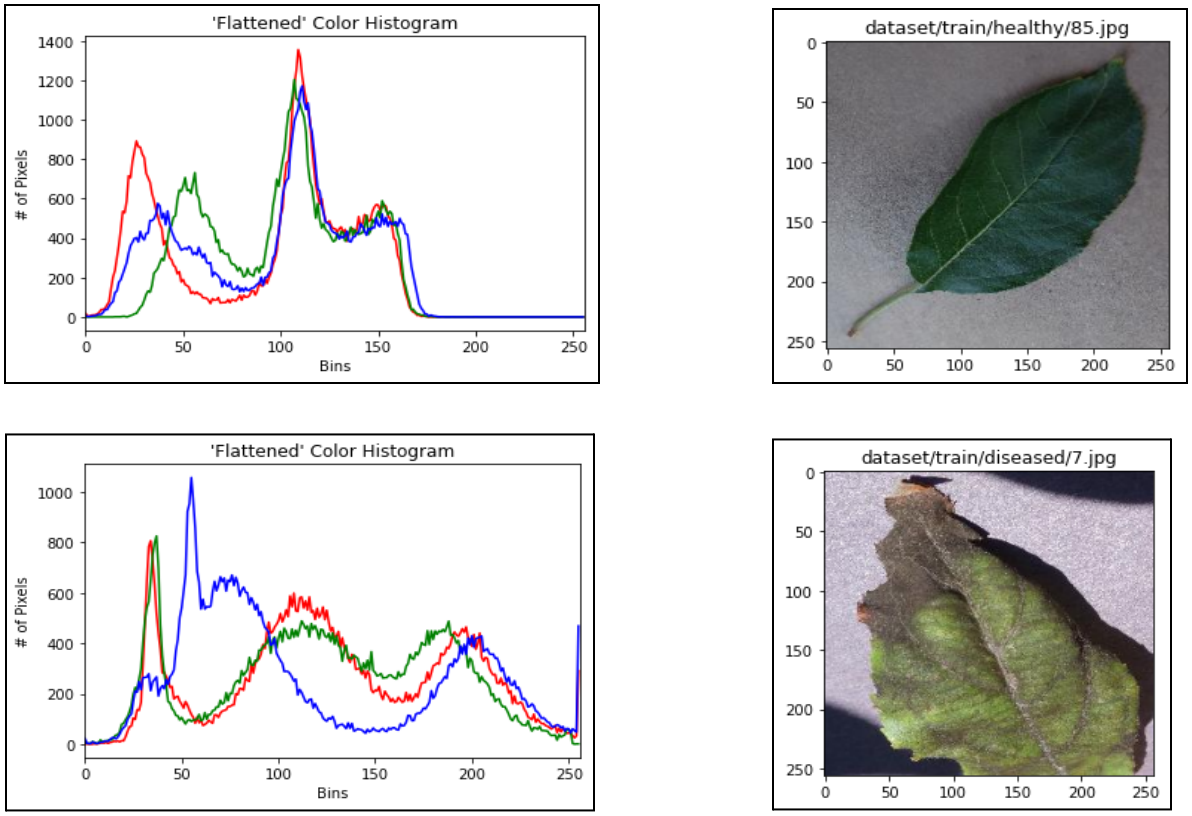
3.5. Model Training and Validation

The Dataset is splitted into training and testing sets with the ratio of 80:20 respectively. The Model is trained over 7 machine learning models i.e., Logistic Regression(LR), Linear Discriminant Analysis(LDA), K Nearest Neighbours(KNN), Decision Trees(DT), Random Forest(RF), Naïve Bayes(NB), Support Vector Machine(SVM). And the model is validated using 10 k fold cross validation technique.

3.6. Tests and Results

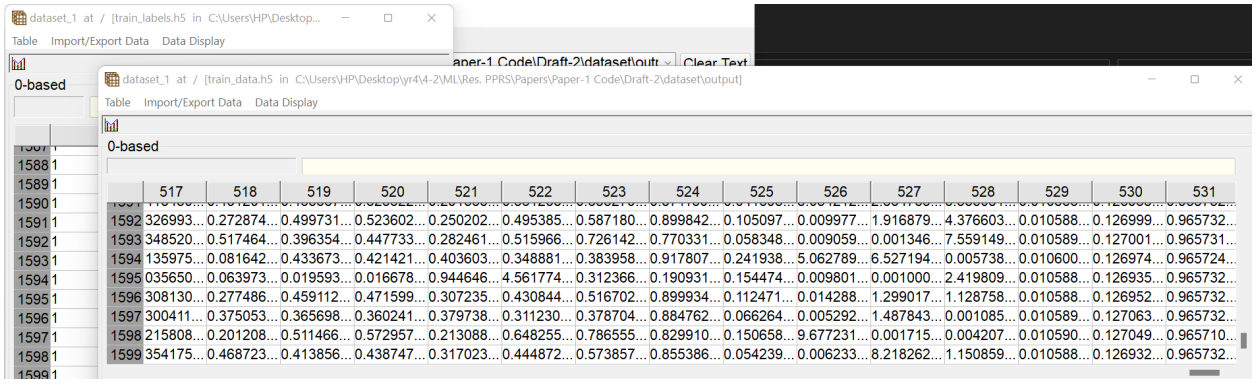


Fig(1) ***All Preprocessing Steps on the image from 1 to 4 as mentioned in section 3.3 and A is the original image.***



Fig(2) (i) Colour Histogram for a healthy leaf (ii) disease leaf

Originally 1600 images (800 from Diseased and 800 from Healthy Label) with 532 features. Each Image encoded with 0 (Diseased) or 1 (Healthy) label. After Splitting into 80:20 ratio, The train data contains 1280 images in Train data and 320 images into Test Data. All the features are in range of [0,1] normalized, and saved in h5 files.

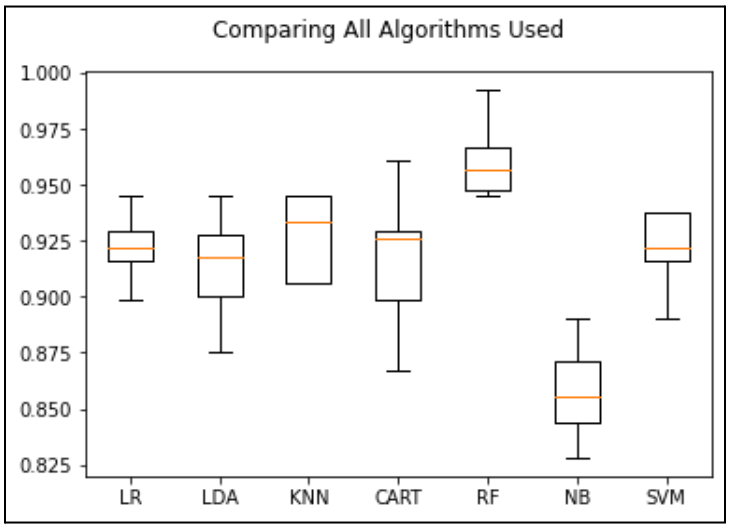


Fig(3) ***H5 files visualisation: train_labels.h5 showing all labels for 1600 images and train_data.h5 showing all 532 features for 1600 images used for training and testing.***

All the models created are then trained and cross validated using k-fold cross validation technique, k =10, with hyperparameters for Random Forest as Number of Trees=100,

Rounds	**LR**	**LDA**	**KNN**	**CART**	**RF**	**NB**	**SVM**
1	0.9296875	0.90625	0.9375	0.8984375	**0.9453125**	0.84375	0.921875
2	0.9296875	0.9453125	0.9453125	0.953125	**0.9921875**	0.890625	0.9375
3	0.921875	0.921875	0.9453125	0.9296875	**0.96875**	0.859375	0.921875
4	0.921875	0.9296875	0.9453125	0.8671875	**0.953125**	0.890625	0.921875
5	0.8984375	0.9375	0.90625	0.8984375	**0.9453125**	0.859375	0.890625
6	0.90625	0.8984375	0.90625	0.921875	**0.9609375**	0.84375	0.9140625
7	0.9453125	0.890625	0.90625	0.9609375	**0.984375**	0.875	0.9375
8	0.9453125	0.921875	0.9453125	0.9296875	**0.9609375**	0.828125	0.9375
9	0.921875	0.9140625	0.9296875	0.8671875	**0.953125**	0.8359375	0.9375
10	0.9140625	0.875	0.90625	0.9296875	**0.9453125**	0.8515625	0.90625
Mean:	0.923438	0.914062	0.927344	0.915625	**0.960938**	0.857812	0.922656
Std dev:	0.014321	0.020670	0.017832	0.030619	**0.015625**	0.020611	0.015007

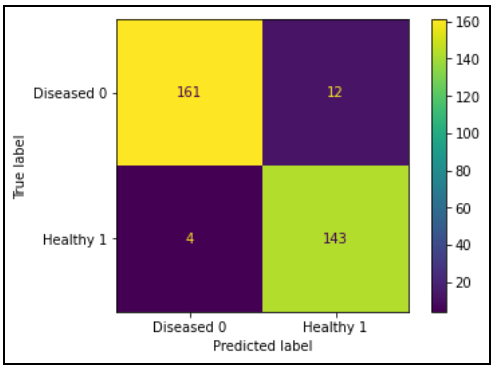
Fig(4) ***Comparison between different machine learning models, abbreviations used as in section 3.5***



Fig(5) A Box-Plot comparing all the 7 different algorithms, values of each iteration are shown in Fig(4), and abbreviations used as in section 3.5

We can see that the Random Forest Algorithm gave us the highest accuracy, i.e. 96.09%.

Now, we model Random Forest separately with the whole of the dataset using the training data of 1280 images and tested the 320 images.



	Test	Predicted
0	0	0
1	1	1
2	1	1
3	0	0
4	1	1
...	...	...
315	1	1
316	0	0
317	1	1
318	0	0
319	1	1

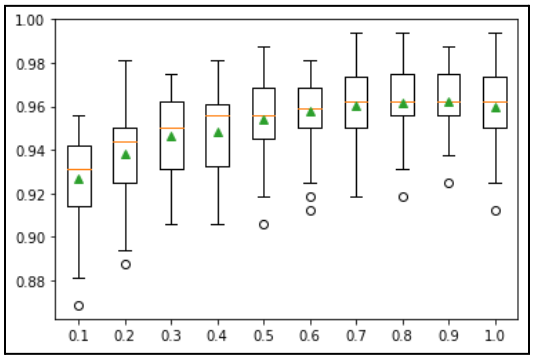
320 rows × 2 columns

Fig(6)*(i) Confusion Matrix (i) A table with the Testing and Predicted Labels*

The accuracy is **95%**, which is nearly the same as above.

3.6.1. Varying the bootstrap sample size

Sample Size	Accuracy	Std dev	Sample Size	Accuracy	Std dev
0.1	0.927	0.023	0.6	0.958	0.017
0.2	0.938	0.024	0.7	0.960	0.018
0.3	0.946	0.019	0.8	0.962	0.016
0.4	0.949	0.020	0.9	0.962	0.015
0.5	0.954	0.019	1.0	0.960	0.017

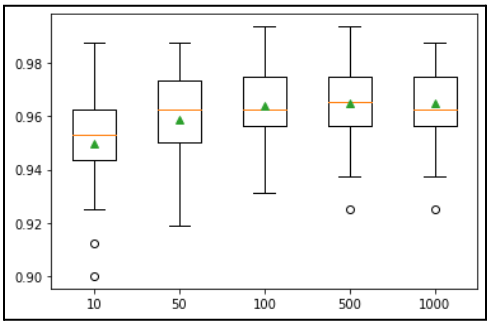


Fig(7) (i) ***Table showing different sample size and accuracy, (ii) Box-Plot for the same***

We observe the accuracy increases on increasing the sample size, but after some time, it is stable.

3.6.2. Varying the number of trees

Num_trees	Accuracy	Std dev
10	0.950	0.019
50	0.959	0.018
100	0.964	0.016
500	0.965	0.017
1000	0.965	0.016



Fig(8) (i) ***Table showing different number of trees and accuracy, (ii) Box-Plot for the same***

We observe that the accuracy increases on increasing the number of trees and after some time, it is stable.

Chapter 4: Proposed Changes

4.1. Optimization of Model

We also propose a change in the form of a Hybrid Random-Forest and PSO (Particle Swarm Optimization), where we first perform feature subset selection through a metaheuristic approach i.e. PSO algorithm and afterwards we model the selected features with Random Forest Classifier, to obtain our predicted results.

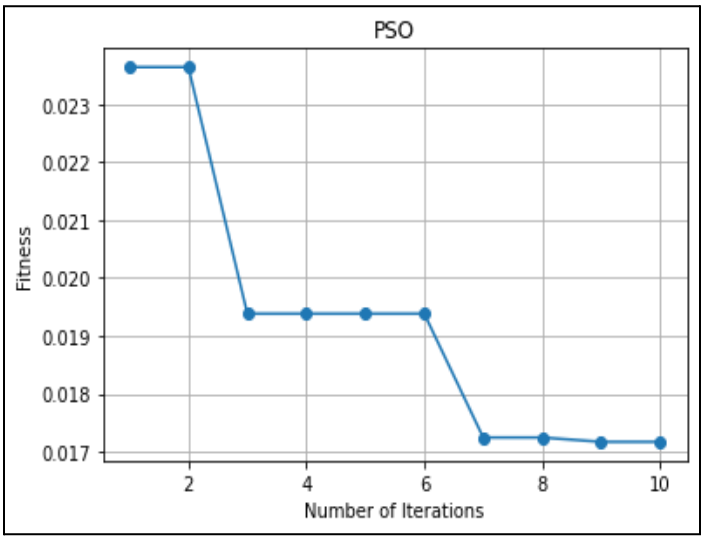
Tunable Parameters of PSO:-

PSO Hyperparameters	Symbols	Value
Number of particles	N	15
Maximum number of iterations	T	10
Inertia weight	w	0.5
Cognitive factor	c1	2
Social factor	c2	2

Fitness Value after Each iteration:-

Iteration	Fitness Value
1	0.02363768796992486
2	0.02363768796992486
3	0.01938110902255634
4	0.01938110902255634
5	0.01938110902255634
6	0.01938110902255634
7	0.01724342105263154
8	0.01724342105263154
9	0.01716823308270673
10	0.01716823308270673

Fitness vs Number of Iterations Curve (PSO)



Fig(9) ***PSO fitness value after each iteration, (ii) Convergence curve: Fitness vs iteration***

Final Number of Features Selected:	255

After Obtaining the Selected Features, the Random Forest model is built with the number of trees=100 and the accuracy is calculated.

Final Accuracy(%):	98.75

4.2. Multi-label Classification

We propose a multi-label classification, so instead of creating two folders and putting all the diseased images into one folder, we will create a separate folder for each disease type and a folder for the healthy leaves. This separation on the basis of diseased categories was already done in the kaggle Apple leaves dataset. We follow the same procedure. We took 250 images from each class, for easy processing. For the first part, from a total of 1000 images, we took 800 images for the training and 200 for testing, and the features after feature extraction were 532.

Here, the unique labels and the respective encodings were:-

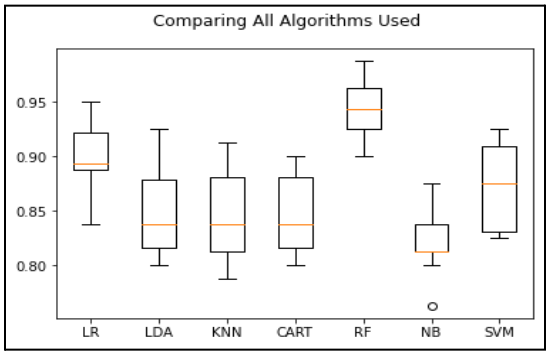
'Apple__Apple_scab': 0, 'Apple__Black_rot': 1, 'Apple__Cedar_apple_rust': 2, 'Apple_healthy': 3

All the models created are then trained and cross validated using k-fold cross validation technique, k =10, with hyperparameters for Random Forest as Number of Trees=100.

Rounds	**LR**	**LDA**	**KNN**	**CART**	**RF**	**NB**	**SVM**
1	0.8875	0.8375	0.85	0.825	**0.925**	0.8125	0.825
2	0.8875	0.8875	0.8625	0.8875	**0.925**	0.8	0.85
3	0.9	0.8125	0.8125	0.9	**0.9625**	0.7625	0.9
4	0.8875	0.8875	0.9	0.8	**0.9125**	0.8125	0.85
5	0.9125	0.8	0.825	0.8125	**0.9875**	0.8375	0.9
6	0.95	0.925	0.8875	0.8625	**0.9625**	0.875	0.925
7	0.925	0.85	0.9125	0.825	**0.9625**	0.8375	0.9125
8	0.875	0.8375	0.8125	0.8125	**0.9**	0.8125	0.825
9	0.9375	0.8	0.8125	0.8875	**0.9625**	0.8375	0.9125
10	0.8375	0.825	0.7875	0.85	**0.925**	0.8125	0.825
Mean:	0.900000	0.846250	0.846250	0.846250	**0.942500**	0.820000	0.825
Std dev:	0.031125	0.039548	0.040716	0.034483	**0.026926**	0.028062	0.039051

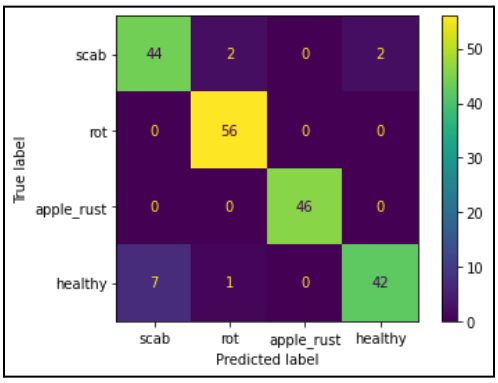
Fig(10) ***Comparison between different algorithms for the multi-label classification.***

Here we see that the accuracy for RF is again the highest, ie. **94.25%**



Fig(11) A Box-Plot comparing all the 7 different algorithms, values of each iteration are shown in the table above, for the multi-label classification and abbreviations used as in section 3.5

Now, we model Random Forest separately with the whole of the dataset using the training data of 800 images and tested the 200 images.



	Test	Predicted
0	1	1
1	2	2
2	0	0
3	1	1
4	3	3
...	...	...
195	3	3
196	2	2
197	3	3
198	0	0
199	1	1

200 rows × 2 columns

Fig(12)*(i) Confusion Matrix (i) A table with the Testing and Predicted Labels for multi-label classification*

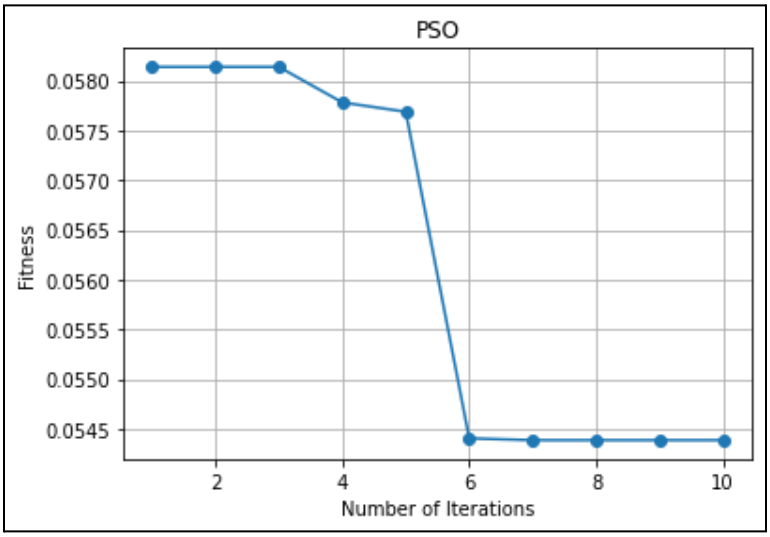
The accuracy in this case turned out to be **94%,**

Now, again, we try our hybrid PSO-RF model (with same hyperparameters for PSO, but particles=20) on multi-label classification, and the results are as follows:-

Fitness Value after Each iteration:-

Iteration	Fitness Value
1	0.05813834586466167
2	0.05813834586466167
3	0.05813834586466167
4	0.05813834586466167
5	0.057687218045112794
6	0.054406015037594034
7	0.05438721804511283
8	0.05438721804511283
9	0.05438721804511283
10	0.05438721804511283

Fitness vs Number of Iterations Curve (PSO)



Fig(13) *PSO fitness value after each iteration, (ii) Convergence curve: Fitness vs iteration*

Final Number of Features Selected:	260

After Obtaining the Selected Features, the Random Forest model is built with the number of trees=100 and the accuracy is calculated.

Final Accuracy(%):	95

Chapter 5: Main Findings and Conclusion

In our implementation, we implemented the paper 'Plant Disease detection using Machine Learning' by R. Shima et. al (2018), with Kaggle Apple dataset, the accuracy came out to be around 95%. We also saw how Random Forest Algorithm could fetch even better results by fine tuning its hyperparameters. With an increase in the number of samples and number of trees, its accuracy increased and after some point of time, it remained stable. A smaller sample size makes trees more different, and a larger sample size makes the trees more similar. The number of trees should be ideally increased until no further improvement in performance is seen on the dataset, here we saw that at around 500 and 1000 decision trees, our accuracy was coming out to be around 96.5%, similar in both cases. Intuition may suggest that adding additional trees will result in overfitting, however this is not the case.Given the stochastic character of the learning method, random forest algorithm appears to be somewhat immune to overfitting the training dataset.

Then we proposed a Hybrid PSO-RF for optimizing the results and the accuracy this time was around 98.75%. We can safely conclude that our proposed approach improved the results significantly, using the metaheuristics approach. PSO (Particle Swarm Optimization) is a computational algorithm that optimises a given problem by trying to improve candidate solutions with each iteration using some error function. It makes use of a biological concept of Swarms, and the movement of these swarm particles in a search space. After each iteration, the position and velocity of a particle is improved and global and local best positions are updated. After some point of time, a convergence is received, and we get our best solution. Even here, we used PSO to improve our accuracy by selecting only useful features and removing each feature from the subset after each iteration. We got 98.75% accuracy using 255 out of 532 features.

In the last part, we also proposed multi-label classification, where we try to label the Kaggle Apple dataset for scab, rot and rust disease and a healthy folder. Using the same steps, we achieved an accuracy of 94% and 95% using our hybrid algorithm.

Link to Github Repository: Here

Task Division

Name and ID of the Group Member	Tasks Performed
Aaryan Gupta **2018B1A70775H**	• Implementation of paper-code • Change Proposal-1-code • Report-graphs, results
Shubham Singla **2019A3PS0392H**	• Implementation of paper-code • Change Proposal-2-code • Report graphs, results
Aditya Sharma **2018A7PS0315H**	• Report- finalizing, formatting • Literature Survey • Fine tuning and experimentations
Doni Akhil Lohith **2019A7PS0026H**	• Report- finalizing, formatting • Literature Survey • Fine tuning and experimentations
DSK Karthik **2019A7PS0189H**	• Report- finalizing, formatting • Literature Survey • Fine tuning and experimentations

References

[1] Ramesh, S., Hebbar, R., M., N., R., P., N., P. B., N., S., & P.V., V. (2018). Plant Disease Detection Using Machine Learning. 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control (ICDI3C). https://doi.org/10.1109/icdi3c.2018.00017

[2] P. Kulkarni, A. Karwande, T. Kolhe, S. Kamble, A. Joshi, and M. Wyawahare, "Plant Disease Detection Using Image Processing and Machine Learning," arXiv:2106.10698 [cs], Nov. 2021. https://doi.org/10.48550/arXiv.2106.10698

[3] Paliwal, N., Vanjani, P., Liu, J. W., Saini, S., & Sharma, A. (2019). Image processing-based intelligent robotic system for assistance of agricultural crops. International Journal of Social and Humanistic Computing, 3(2), 191. https://doi.org/10.1504/ijshc.2019.101602

[4] Kalvakolanu, A.T. (2020). Plant Disease Detection using Deep Learning. International Journal of Recent Technology and Engineering.

[5] K. K. Zaw, Z. M. M. Myo and D. T. H. Thoung, "Support Vector Machine Based Classification of Leaf Diseases", International Journal of Science and Engineering Applications, 2019, vol.7, no.8.