# Fertilizers Recommendation System for Disease Prediction

Submitted by

**Team ID : PNT2022TMID09294**

| | |
|---|---|
| **MANICK SRIRAM M** | **REG : 310619104065** |
| **RAJAVARMAN R** | **REG : 310619104099** |
| **RANJITH G** | **REG : 310619104101** |
| **SAMRISH R** | **REG : 310619104114** |

## 1 . INTRODUCTION :

Overview Two datasets—the fruit dataset and the vegetable dataset—are gathered for this research. Convolutional Neural Networks, a deep learning neural network, is used to train and test the datasets that has been collected (CNN). The fruit dataset is first trained, and then CNN is tested. There are 6 courses total, and each class is trained and tested. The vegetable dataset is then tested and trained.
Python is the language used to train and test datasets.

All of the Python code is initially created in the Jupyter notebook that comes with Anaconda Python, and it is then tested in the IBM cloud. Finally, Flask, a Python package, is used to construct a web-based framework. Along with their related files, two html files are created in the templates folder.

Purpose of this study is used to test samples of fruits and vegetables and find out which diseases they may have. Additionally, this project suggests fertilizer for certain ailments**.**

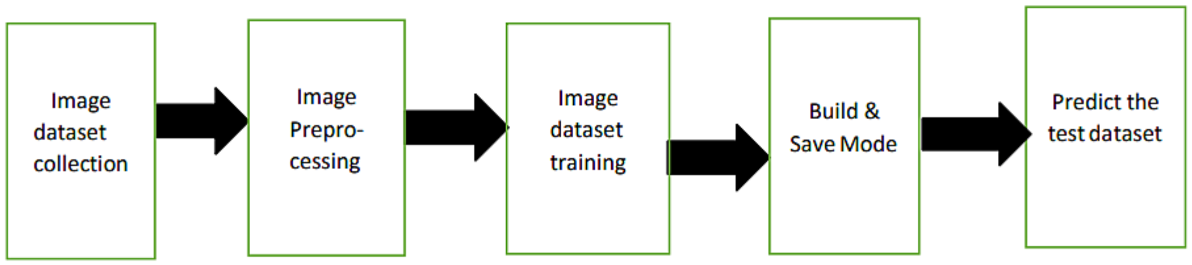
## 2 . LITERATURE SURVEY :

Existing issue our Team suggested a method for identifying leaf diseases and suggested fertilizer to treat them. However, the method's low number of train and test sets leads to subpar accuracy.
In order to recommend soil-based fertilizers for anticipated crop diseases, our team put up a straightforward prediction method. This approach offers less predictability and accuracy. An IOT-based system for leaf disease identification and fertilizer prescription that was proposed by us, which uses machine learning techniques and achieves accuracy levels of under 80%.

Proposed Remedy A deep learning-based neural network is implemented in this project's effort to train and test the datasets that were gathered. CNN, a deep learning-based neural network, provides classification accuracy rates of greater than 90%. By boosting the accuracy rate can be raised to 95% to 98% by adding more dense layers and changing hyperparameters like the number of epochs and batch size.

## 3 . THEORETICAL ANALYSIS :

BLOCK DIAGRAM :



Block Diagram of the project

Above Diagram represents the project's overall block diagram. The collecting of picture datasets comes first, then enters image preparation. The training of picture datasets using various hyperparameter initializations is the third phase. After that, create the model and save it in ".h5 format". The final step involves applying the trained model to test new or existing datasets.

**Hardware / Software Designing :**

Python is the language used to train and test the dataset. Python programming is done in a notebook tool called Jupyter, which also works with the IBM cloud.
Convolutional Neural Network is the neural network that was utilised to train and test the model (CNN).

**The CNN has following layers :**

- Convolutional layer (32x32 kernal (3x3)),
- Max-pool layer (kernel(2x2)),

➤ Flatten layer,

➤ Dense layer (different layers with different size),

➤ Drop out layer (optional),

➤ Final output dense layer(size 6x1 for fruit dataset and 9x1 for Vegetable dataset).

Images are normalised to 1 and then reduced to 128x128 in the preparation stage.Different batch sizes are used to arrange the photos. Then, using the gathered datasets, train set and test set are created. The following Python libraries need to be imported before beginning the process in order to perform the aforementioned actions in Python :

➤ NumPy,

➤ TensorFlow,

➤ Keras,

➤ Matplotlib (optional for data visualization).

The following activation functions used in the CNN training :

➤ RELU at the end of convolution layer and Max Pool layer,

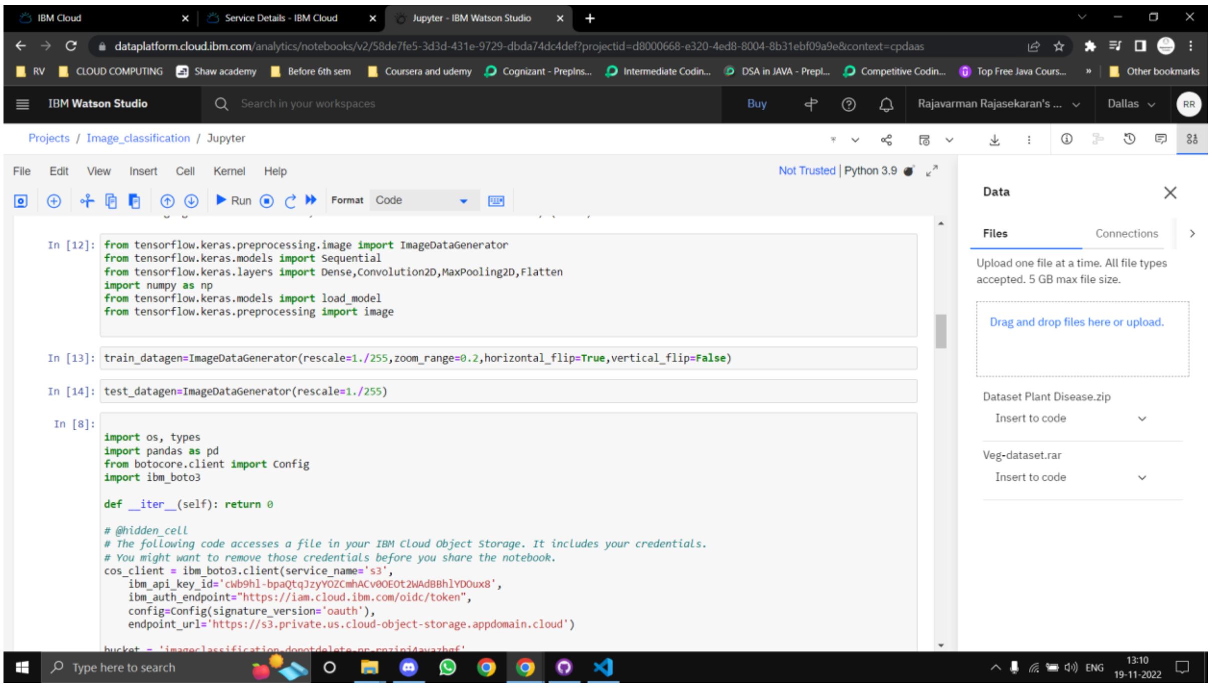
➤ SoftMax at the end of output dense layer,

➤ For testing the dataset argmax is used, its an optional.

## 4 . EXPERIMENTAL INVESTIGATIONS :

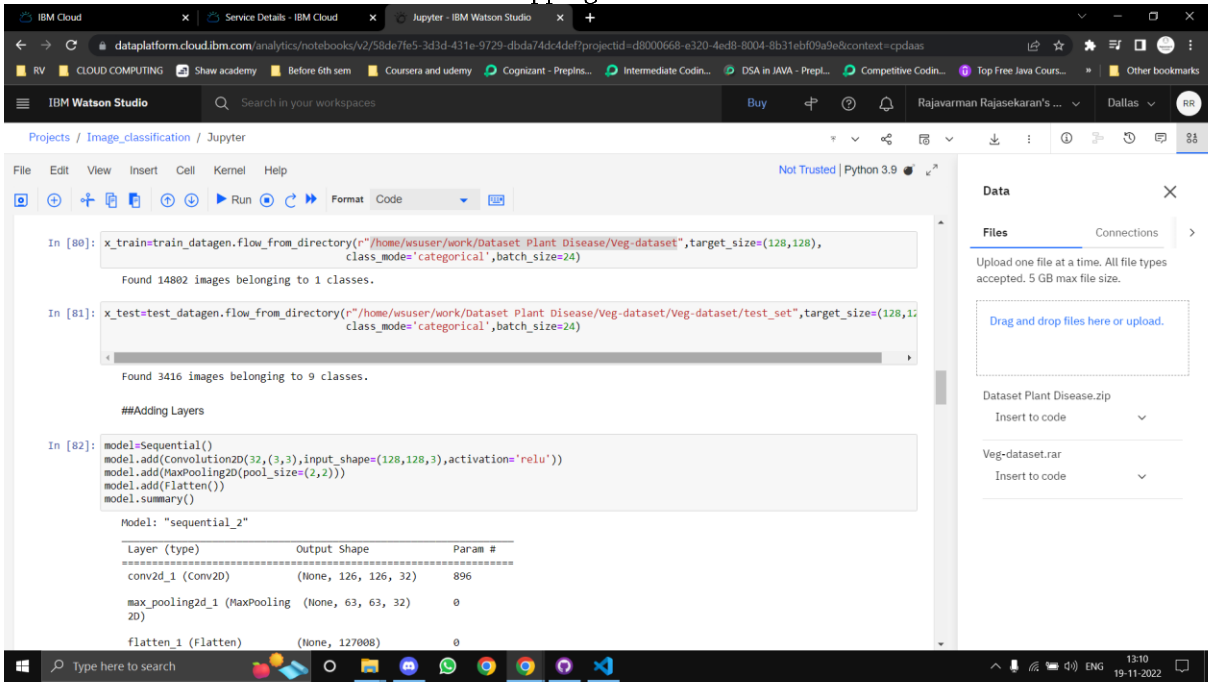
Analysis performed when developing the solution The batch sizes are varied and tested. The accuracy provided by CNN varies with batch size. The number of iterations each epoch is determined by the batch size. The quantity of epochs is a key hyper parameter. When compared to other hyper metrics, this affects accuracy and has a significant impact on accuracy. By increasing the number of epochs, the accuracy may be adjusted from 80% to 90% for the vegetable dataset and from 95% to 98% for the fruit dataset. The size of the training and test datasets also has a significant impact on accuracy. More photos can be added to the train dataset to boost accuracy. The size of the train dataset and the number of epochs both increase the computational time required to create a model. The train dataset and test dataset batch sizes are also very important in terms of processing time. When there are additional convolutional layers, the complexity of the neural network increases. A higher layer count will produce results with greater accuracy. The CNN algorithm requires more time to develop a model and spend more time training as the number of layers increases. The model .h5 size depends on the size of train datasets. However, the amount of the train dataset and the complexity of the CNN architecture determine the memory need.

## 5 . RESULTS :


IBM Watson Studio
Projects / Image_classification / Jupyter
Not Trusted | Python 3.9
Data
Files
Connections
Upload one file at a time. All file types accepted. 5 GB max file size.
Drag and drop files here or upload.
Dataset Plant Disease.zip
Insert to code
Veg-dataset.rar
Insert to code

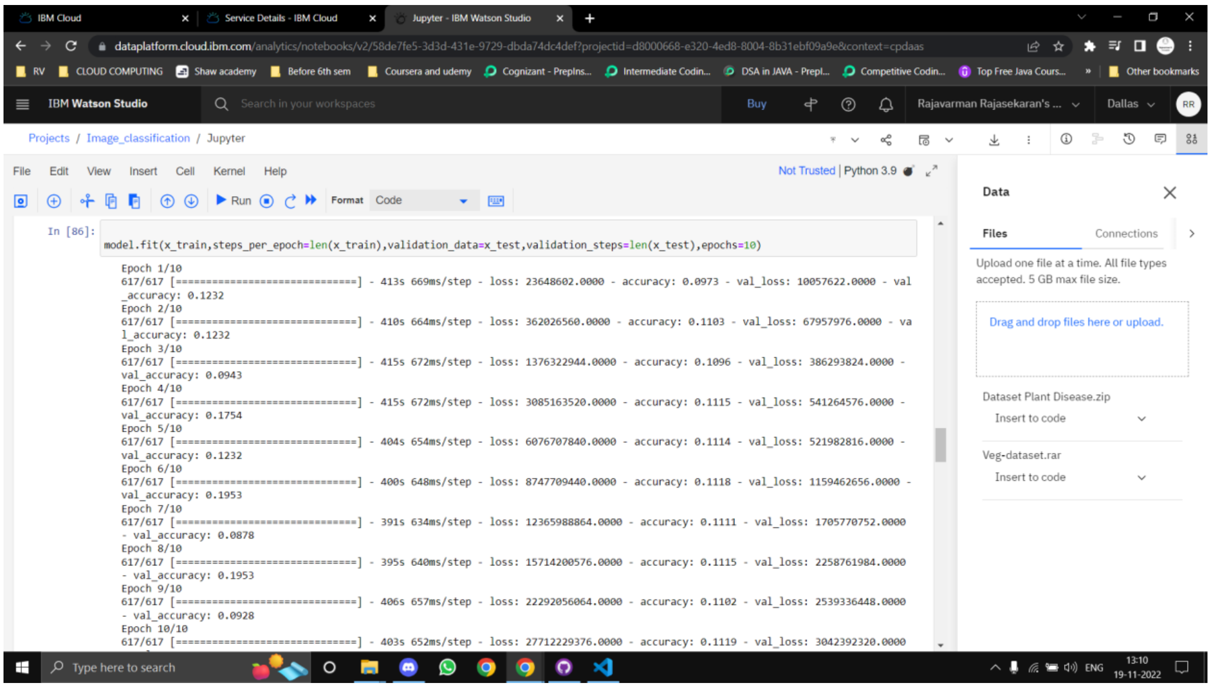
```
In [12]: from tensorflow.keras.preprocessing.image import ImageDataGenerator
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense,Convolution2D,MaxPooling2D,Flatten
         import numpy as np
         from tensorflow.keras.models import load_model
         from tensorflow.keras.preprocessing import image

In [13]: train_datagen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True,vertical_flip=False)

In [14]: test_datagen=ImageDataGenerator(rescale=1./255)

In [8]:

         import os, types
         import pandas as pd
         from botocore.client import Config
         import ibm_boto3

         def __iter__(self): return 0

         # @hidden_cell
         # The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
         # You might want to remove those credentials before you share the notebook.
         cos_client = ibm_boto3.client(service_name='s3',
             ibm_api_key_id='cWb9hl-bpaQtqJzyYOZCmhACv0OEOt2WAdBBhlYDOux8',
             ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
             config=Config(signature_version='oauth'),
             endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

         bucket = 'imageclassification-donotdelete-pr-rnzinj4ayazhgf'
```

Unzipping the Dataset


IBM Watson Studio
Projects / Image_classification / Jupyter
Not Trusted | Python 3.9
Data
Files
Connections
Upload one file at a time. All file types accepted. 5 GB max file size.
Drag and drop files here or upload.
Dataset Plant Disease.zip
Insert to code
Veg-dataset.rar
Insert to code

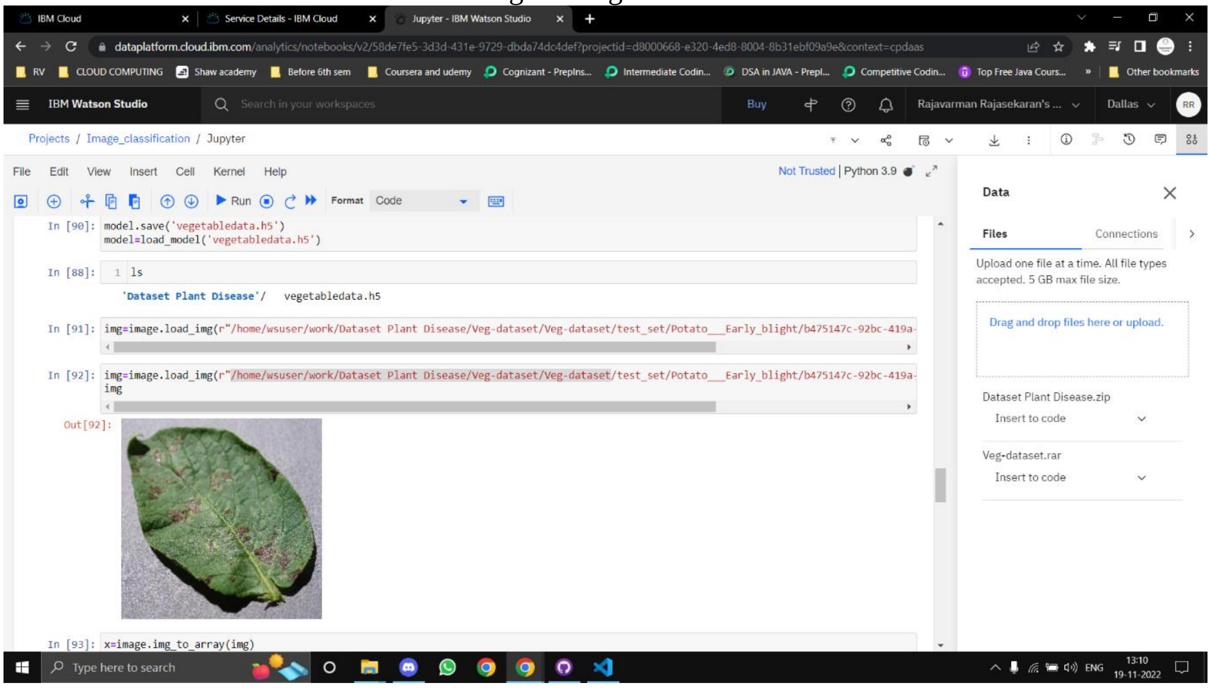
```
In [80]: x_train=train_datagen.flow_from_directory(r"/home/wsuser/work/Dataset Plant Disease/Veg-dataset",target_size=(128,128),
                                                   class_mode='categorical',batch_size=24)

    Found 14802 images belonging to 1 classes.

In [81]: x_test=test_datagen.flow_from_directory(r"/home/wsuser/work/Dataset Plant Disease/Veg-dataset/Veg-dataset/test_set",target_size=(128,1
                                                   class_mode='categorical',batch_size=24)

    Found 3416 images belonging to 9 classes.

    ##Adding Layers

In [82]: model=Sequential()
         model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
         model.add(MaxPooling2D(pool_size=(2,2)))
         model.add(Flatten())
         model.summary()

    Model: "sequential_2"
    _________________________________________________________________
     Layer (type)                Output Shape              Param #
    =================================================================
     conv2d_1 (Conv2D)           (None, 126, 126, 32)      896

     max_pooling2d_1 (MaxPooling  (None, 63, 63, 32)       0
     2D)

     flatten_1 (Flatten)         (None, 127008)            0
```

Training the Output Dataset



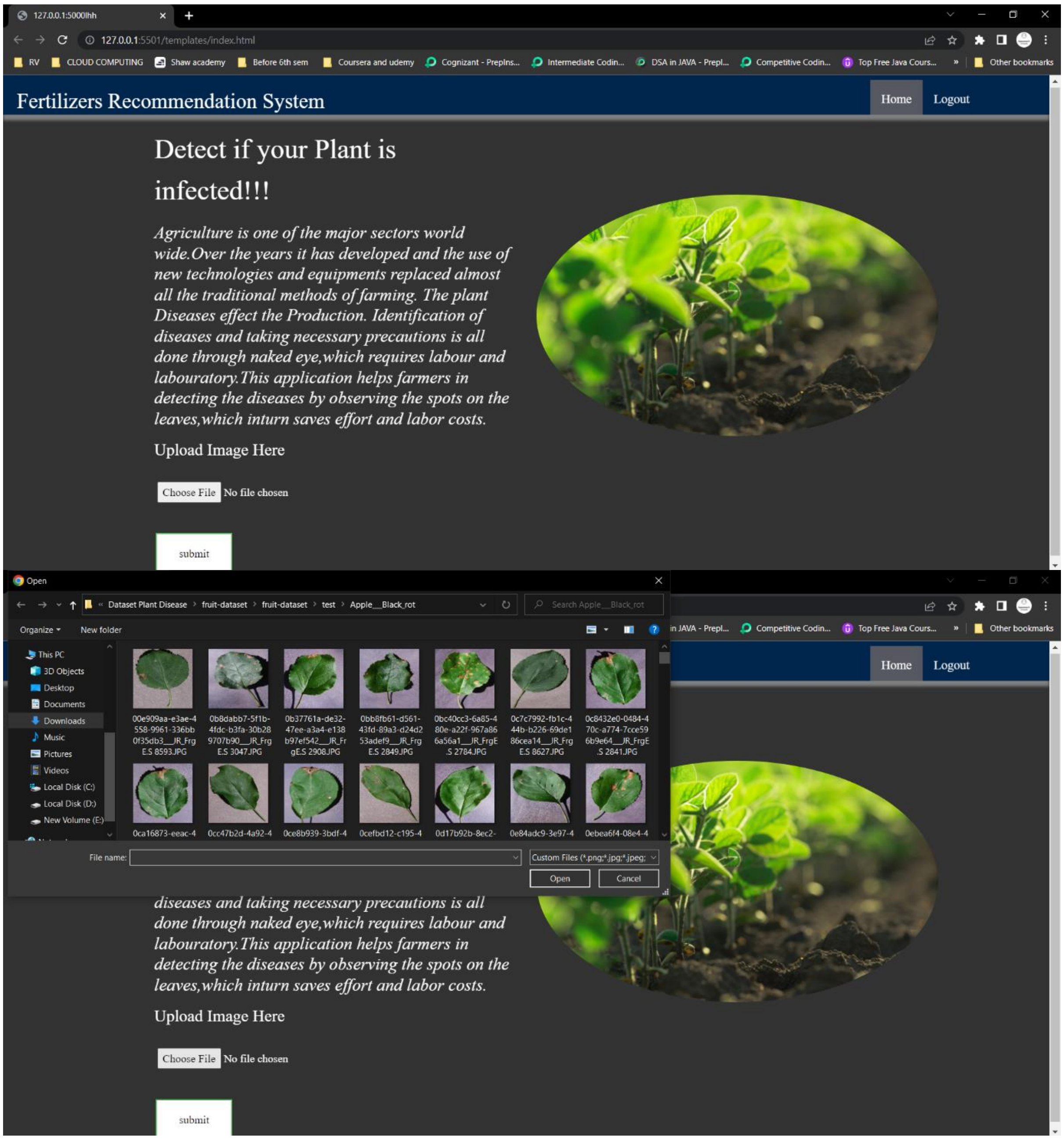
```
In [86]: model.fit(x_train,steps_per_epoch=len(x_train),validation_data=x_test,validation_steps=len(x_test),epochs=10)

         Epoch 1/10
         617/617 [==============================] - 413s 669ms/step - loss: 23648602.0000 - accuracy: 0.0973 - val_loss: 10057622.0000 - val
         _accuracy: 0.1232
         Epoch 2/10
         617/617 [==============================] - 410s 664ms/step - loss: 362026560.0000 - accuracy: 0.1103 - val_loss: 67957976.0000 - va
         l_accuracy: 0.1232
         Epoch 3/10
         617/617 [==============================] - 415s 672ms/step - loss: 1376322944.0000 - accuracy: 0.1096 - val_loss: 386293824.0000 -
         val_accuracy: 0.0943
         Epoch 4/10
         617/617 [==============================] - 415s 672ms/step - loss: 3085163520.0000 - accuracy: 0.1115 - val_loss: 541264576.0000 -
         val_accuracy: 0.1754
         Epoch 5/10
         617/617 [==============================] - 404s 654ms/step - loss: 6076707840.0000 - accuracy: 0.1114 - val_loss: 521982816.0000 -
         val_accuracy: 0.1232
         Epoch 6/10
         617/617 [==============================] - 400s 648ms/step - loss: 8747709440.0000 - accuracy: 0.1118 - val_loss: 1159462656.0000 -
         val_accuracy: 0.1953
         Epoch 7/10
         617/617 [==============================] - 391s 634ms/step - loss: 12365988864.0000 - accuracy: 0.1111 - val_loss: 1705770752.0000
         - val_accuracy: 0.0878
         Epoch 8/10
         617/617 [==============================] - 395s 640ms/step - loss: 15714200576.0000 - accuracy: 0.1115 - val_loss: 2258761984.0000
         - val_accuracy: 0.1953
         Epoch 9/10
         617/617 [==============================] - 406s 657ms/step - loss: 22292056064.0000 - accuracy: 0.1102 - val_loss: 2539336448.0000
         - val_accuracy: 0.0928
         Epoch 10/10
         617/617 [==============================] - 403s 652ms/step - loss: 27712229376.0000 - accuracy: 0.1119 - val_loss: 3042392320.0000
```

Training the Vegetable Dataset



```
In [90]: model.save('vegetabledata.h5')
         model=load_model('vegetabledata.h5')

In [88]: ls

         'Dataset Plant Disease'/    vegetabledata.h5

In [91]: img=image.load_img(r"/home/wsuser/work/Dataset Plant Disease/Veg-dataset/Veg-dataset/test_set/Potato___Early_blight/b475147c-92bc-419a-

In [92]: img=image.load_img(r"/home/wsuser/work/Dataset Plant Disease/Veg-dataset/Veg-dataset/test_set/Potato___Early_blight/b475147c-92bc-419a-
         img

Out[92]:

In [93]: x=image.img_to_array(img)
```

## 6 . OUTPUT :


Fertilizers Recommendation System

Home Logout

Detect if your Plant is infected!!!

*Agriculture is one of the major sectors world wide.Over the years it has developed and the use of new technologies and equipments replaced almost all the traditional methods of farming. The plant Diseases effect the Production. Identification of diseases and taking necessary precautions is all done through naked eye,which requires labour and labouratory.This application helps farmers in detecting the diseases by observing the spots on the leaves,which inturn saves effort and labor costs.*

Upload Image Here

Choose File No file chosen

submit

Open

Dataset Plant Disease › fruit-dataset › fruit-dataset › test › Apple___Black_rot

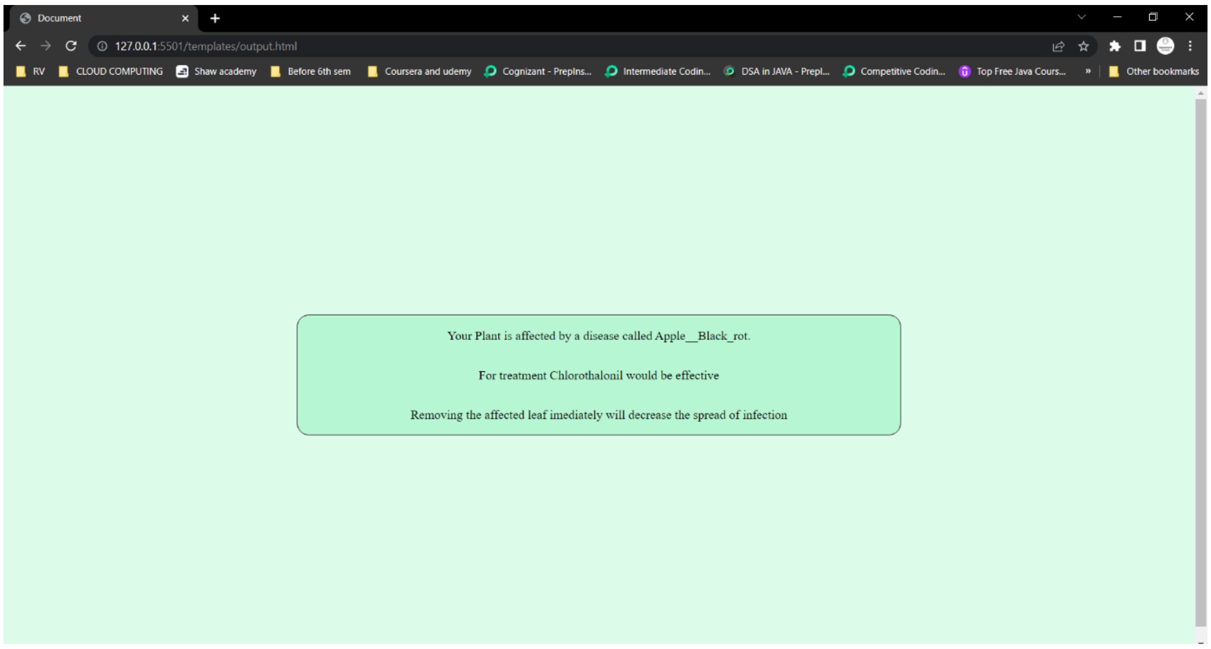
Search Apple___Black_rot

```
00e909aa-e3ae-4558-9961-336bb0f35db3___JR_FrgE.S 8593.JPG
0b8dabb7-5f1b-4fdc-b3fa-30b289707b90___JR_FrgE.S 3047.JPG
0b37761a-de32-47ee-a3a4-e138b97ef542___JR_FrgE.S 2908.JPG
0bb8fb61-d561-43fd-89a3-d24d253adef9___JR_FrgE.S 2849.JPG
0bc40cc3-6a85-480e-a22f-967a866a56a1___JR_FrgE.S 2784.JPG
0c7c7992-fb1c-444b-b226-69de186cea14___JR_FrgE.S 8627.JPG
0c8432e0-0484-470c-a774-7cce596b9e64___JR_FrgE.S 2841.JPG
0ca16873-eeac-4
0cc47b2d-4a92-4
0ce8b939-3bdf-4
0cefbd12-c195-4
0d17b92b-8ec2-
0e84adc9-3e97-4
0ebea6f4-08e4-4
```

File name:

Custom Files (*.png;*.jpg;*.jpeg;

Uploading the Image



```
Your Plant is affected by a disease called Apple__Black_rot.

For treatment Chlorothalonil would be effective

Removing the affected leaf imediately will decrease the spread of infection
```

Output

## 7 . 1 . ADVANTAGES :

➤ The model that is being suggested here achieves extremely high categorization accuracy.

➤ It is also possible to train and test very big datasets.

➤ Very high resolution images can be modified within the proposal itself.

## 7 . 2 . DIS - ADVANTAGES :

➤ The suggested model demands a significant amount of computational time for both training and testing.

➤ This project's neural network design is quite complicated.

## 8 . APPLICATIONS :

1. The trained network model used to classify the image patterns with high accuracy.
2. The proposed model not only used for plant disease classification but also for other image pattern classification such as animal classification.
3. This project work application involves not only image classification but also for pattern recognition.

## 9 . CONCLUSION :

The model proposed here involves classifying images from datasets of fruits and vegetables. Observations made during model testing and training include the following:

➤ The accuracy of classification increased by increasing the number of epochs.

➤ For different batch sizes, different classification accuracies are obtained.

➤ The accuracies are increased by increasing more convolution layers.

➤ The accuracy of classification also increased by varying dense layers.

➤ Different accuracies are obtained by varying the size of kernel used in the convolution layer output.

➤ Accuracies are different while varying the size of the train and test datasets.

## 10 . FUTURE SCOPE :

The proposed model in this project work can be extended to image recognition. The entire model can be converted to application software using python to exe software. The real time image classification, image recognition and video processing are possible with help OpenCV python library. This project work can be extended for security applications such as figure print recognition, iris recognition and face recognition.

## 11 . BIBILOGRAPHY :

[1]. H Shiva reddy, Ganesh hedge, Prof. DR Chinnaya3, "IoT based Leaf Disease Detection and Fertilizer Recommendation", International Research Journal of Engineering and Technology (IRJET), Volume: 06 Issue: 11, Nov 2019, e-ISSN: 2395- 0056.

[2]. R Indumathi Leaf Disease Detection and Fertilizer Suggestion", IEEE International Conference on System, Computation, Automation and Networking (ICSCAN), 29-30 March 2019, DOI:10.1109/ICSCAN.2019. 8878781.

[3]. P. Pandi Selvi, P. Poornima, "Soil Based Fertilizer Recommendation System for Crop Disease Prediction System", International Journal of Engineering Trends and Applications (IJETA) – Volume 8 Issue 2, Mar-Apr 2021.