Fertilizers Recommendation System for Disease Prediction

### PROJECT REPORT

*Submitted by*

# Team ID: PNT2022TMID03452

**RAVI KKUMAAR S V (212219220041)**

**VALLERI GNANA THEJA (212219220056)**

**MAHESH KUMAR REDDY(212219220028)**

**KANISHA B (21229220021)**

*In partial fulfilment for the award of the degree Of*

# BACHELOR OF TECHNOLOGY

In

# INFORMATION TECHNOLOGY

**SAVEETH A ENGINEERING COLLEGE , CHENNAI.**

# INTRODUCTION

Overview In this project, two datasets named fruits and vegetables are collected. The collected datasets are trained and tested using a deep learning neural network called Convolutional Neural Networks (CNN). First, the fruit set is trained and then tested using CNN. It has 6 classes and all classes are trained and tested. Second, a vegetable dataset is trained and tested. The software used for training and testing datasets is Python. All Python code is first written in the Jupyter notebook that comes with Anaconda Python, and then the code is tested in the IBM cloud. Finally, a web framework is designed with the help of Flask and the Python library. 2 html files are created in the templates folder along with their associated files in the static folder. The python program 'app.py' used to link to these two web pages is written in Spyder-Anaconda python and tested.

This project is used to test the fruits and vegetables samples and identify the different diseases. Also, this project recommends fertilizers for predicted diseases.

# LITERATURE SURVEY

* 1. **Existing problem**:

proposed a method to detect foliar diseases and designed fertilizers to treat foliar diseases. However, this method involves a smaller number of train and test sets, resulting in low accuracy. proposed a simple prediction method for a soil fertilizer recommendation system for predicted crop diseases. This method provides less accuracy and prediction. The other has designed an internet of things based system for leaf disease detection and fertilizer recommendation which is based on machine learning techniques and provides less than 80 percent accuracy.

* 1. **Proposed solution:**

In this project work, a deep learning based neural network is used to train the collected data sets and test them. The deep learning based neural network is CNN which provides more than 90% classification accuracy. By increasing more dense layers and adjusting hyperparameters such as number of epochs, batch size, the accuracy rate can be increased to 95% to 98%.

# THEORITICAL ANALYSIS

## Block diagram

Image Prepro- cessing

Image dataset collection

Image dataset training

Build & Save Mode

Predict the test dataset

Figure.3.1.

Block Diagram of the entire project is in Fig. 3.1. The first step is the collection of a dataset of images followed by pre-processing of the images. The third step is the training of image data sets with the initialization of various hyper parameters. Then build the model and save the model file in .h5 format. The final stage is testing existing or new datasets using the trained model.

# Hardware/Software designing

The software used for training and testing the dataset is Python. Jupyter notebook (also IBM cloud notebook) is used for python programming. The neural network used for training and testing the model is a convolutional neural network (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)

In the preprocessing step, the image is normalized to 1 and then scaled to 128x128. The images are arranged in different stack sizes. A training set and a test set are then formed from the collected datasets. To perform the above steps in Python, you need to import the following Python libraries before starting the process.

* + 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

# EXPERIMENTAL INVESTIGATIONS

Analyzing solutions while working Batch sizes have been varied and tested. For different batch sizes, CNNs specify different accuracies. The stack size determines the number of iterations per epoch. Another important hyperparameter is the number of epochs. This determines accuracy and has a large impact on accuracy compared to other hyperparameters. By increasing the number of epochs, we can change the accuracy of the vegetable dataset from 80% to 90% and the accuracy of the fruit dataset from 95% to 98%. The size of the test and training data sets also has a huge impact on accuracy. Accuracy can be improved by using more images in the train record. As the size of the training data set increases and the number of epochs also increases, the modeling computation time increases. The batch sizes of the training and test datasets also play an important role in computation time. As the number of convolutional layers increases, so does the complexity of the neural network. Increasing the number of layers will give more accurate results. At the same time, increasing the number of layers in the CNN increases the training time and also builds the model. The size of the .h5 model depends on the size of the train record.

# FLOWCHARTS

Start

Image Data collection

Norm factor=1./255

Batch size=32 for fruit dataset ,16 for vegetable dataset image resize=128\*128

Build the model and train the dataset using desired hyperparameters

If classified

correctly

No

Yes

Accuracy=Accuracy+1

Accuracy=Accuracy

Save the model with .h5 format

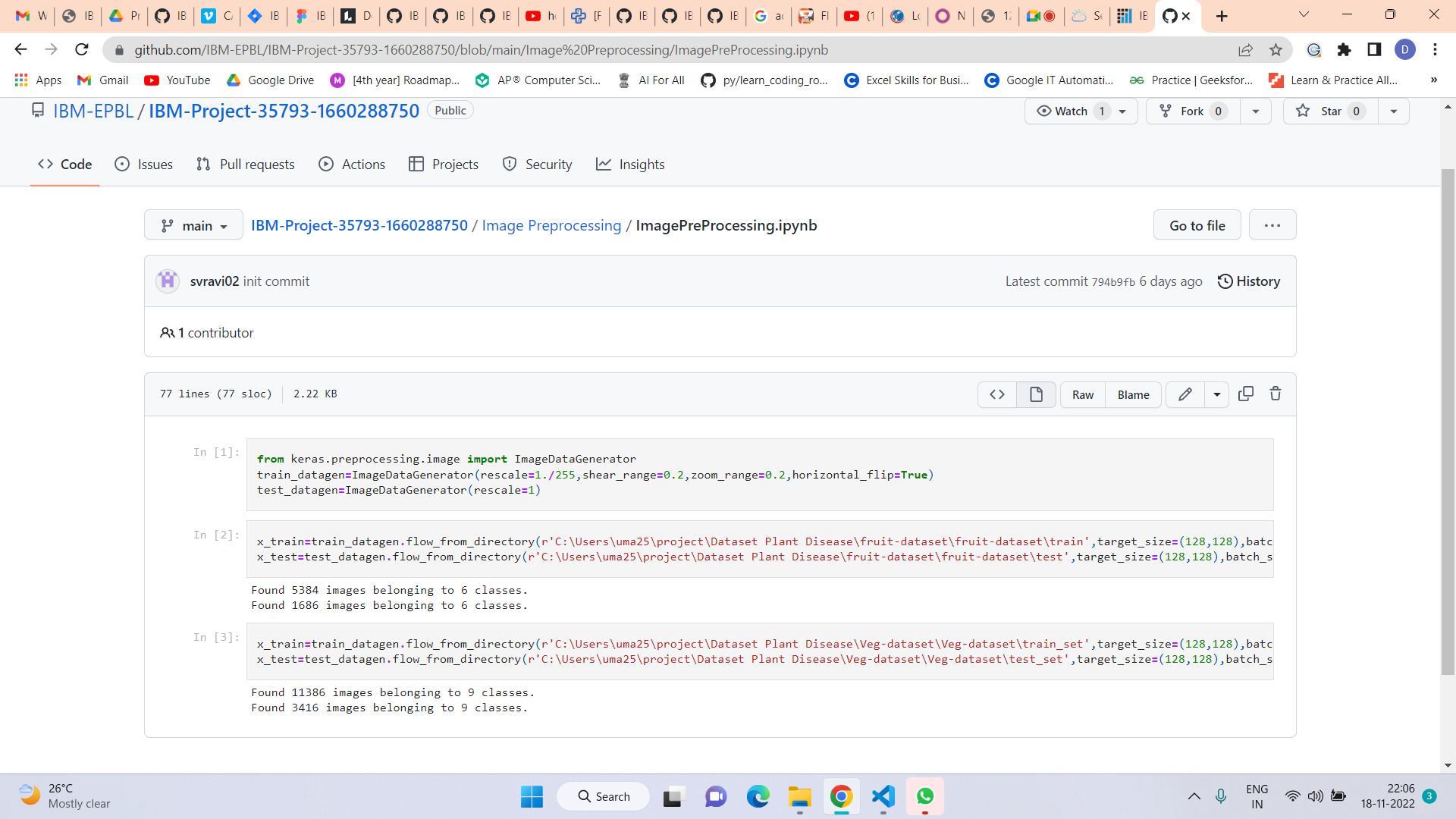
Test and predict either new dataset or existing testset using model.h5

stop

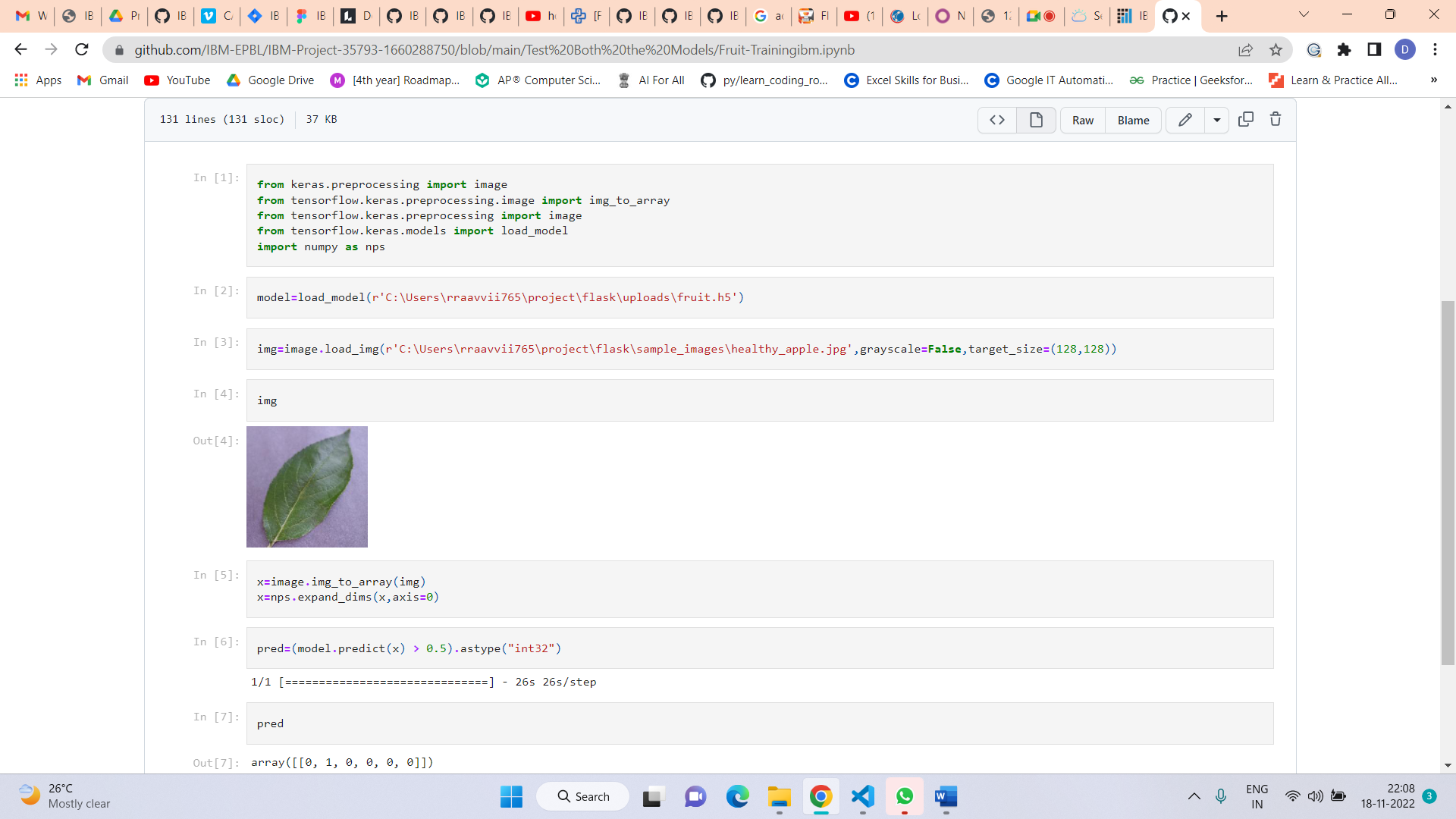
Display the classified image class and name of the class

1. **RESULTS**

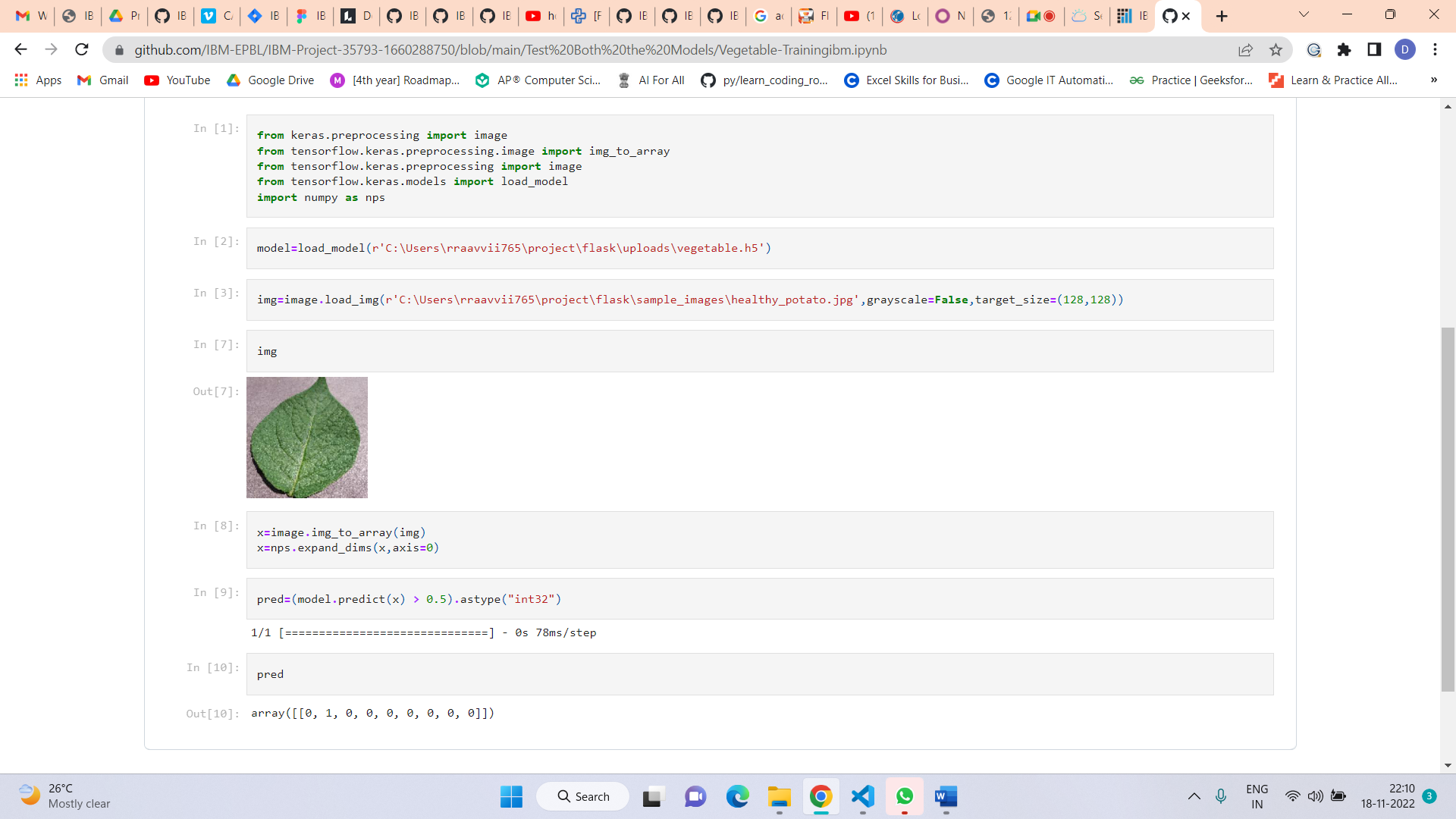
Final findings(output) of the project given below in the form of screenshot: Training and Testing of Fruit dataset



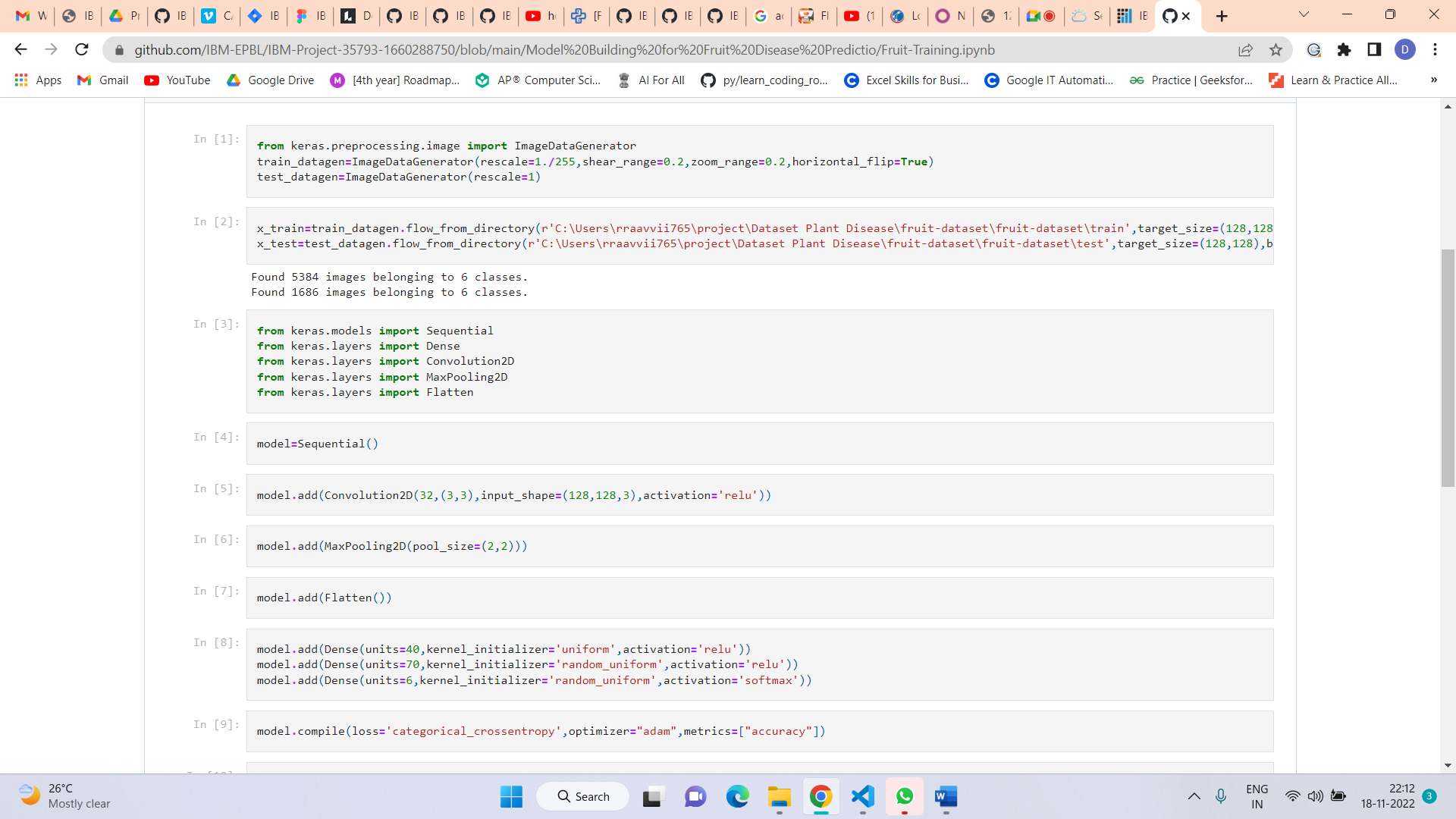
Fruit trainingibm.ipynb

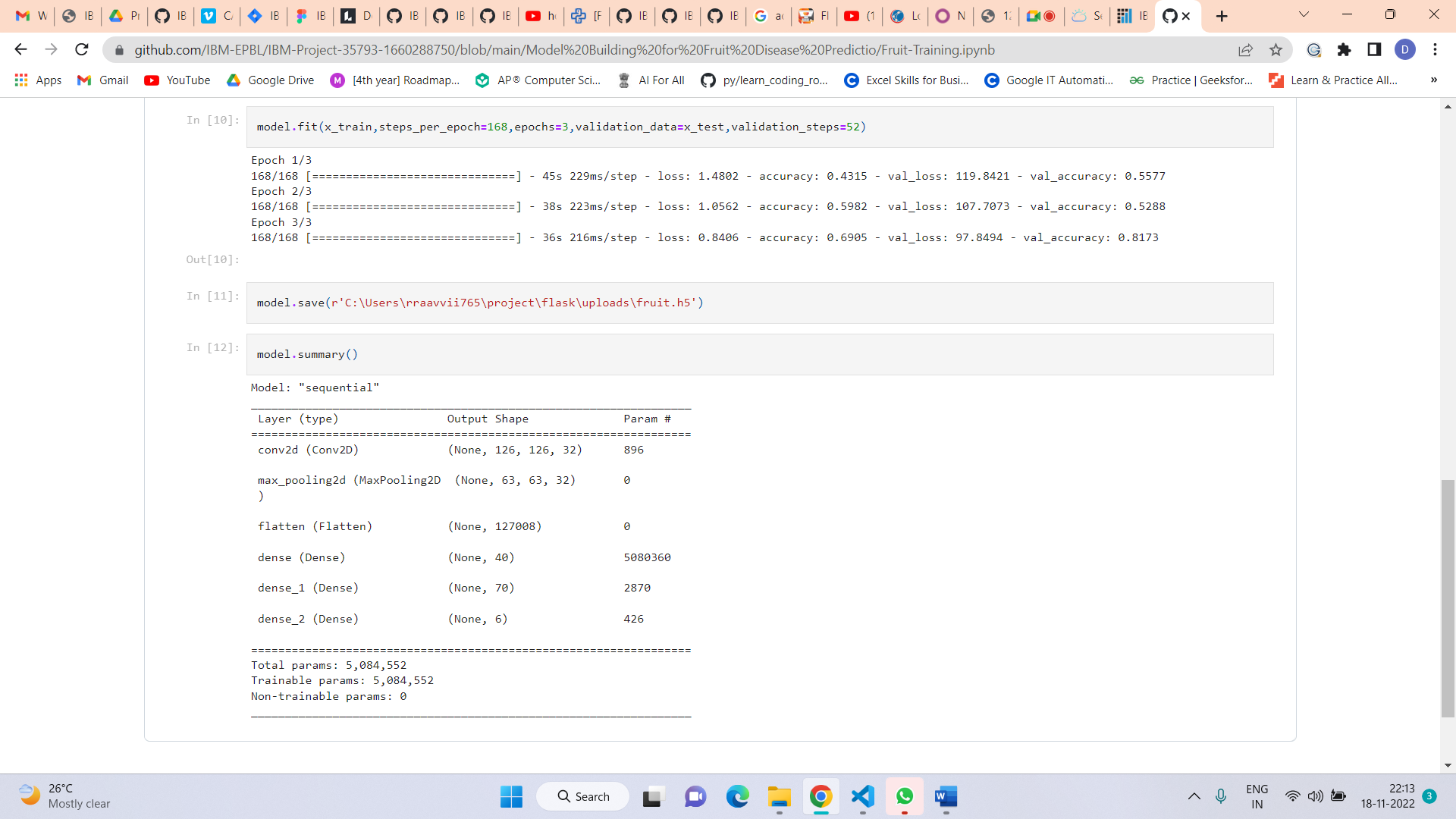


vegetable trainingibm.ipynb

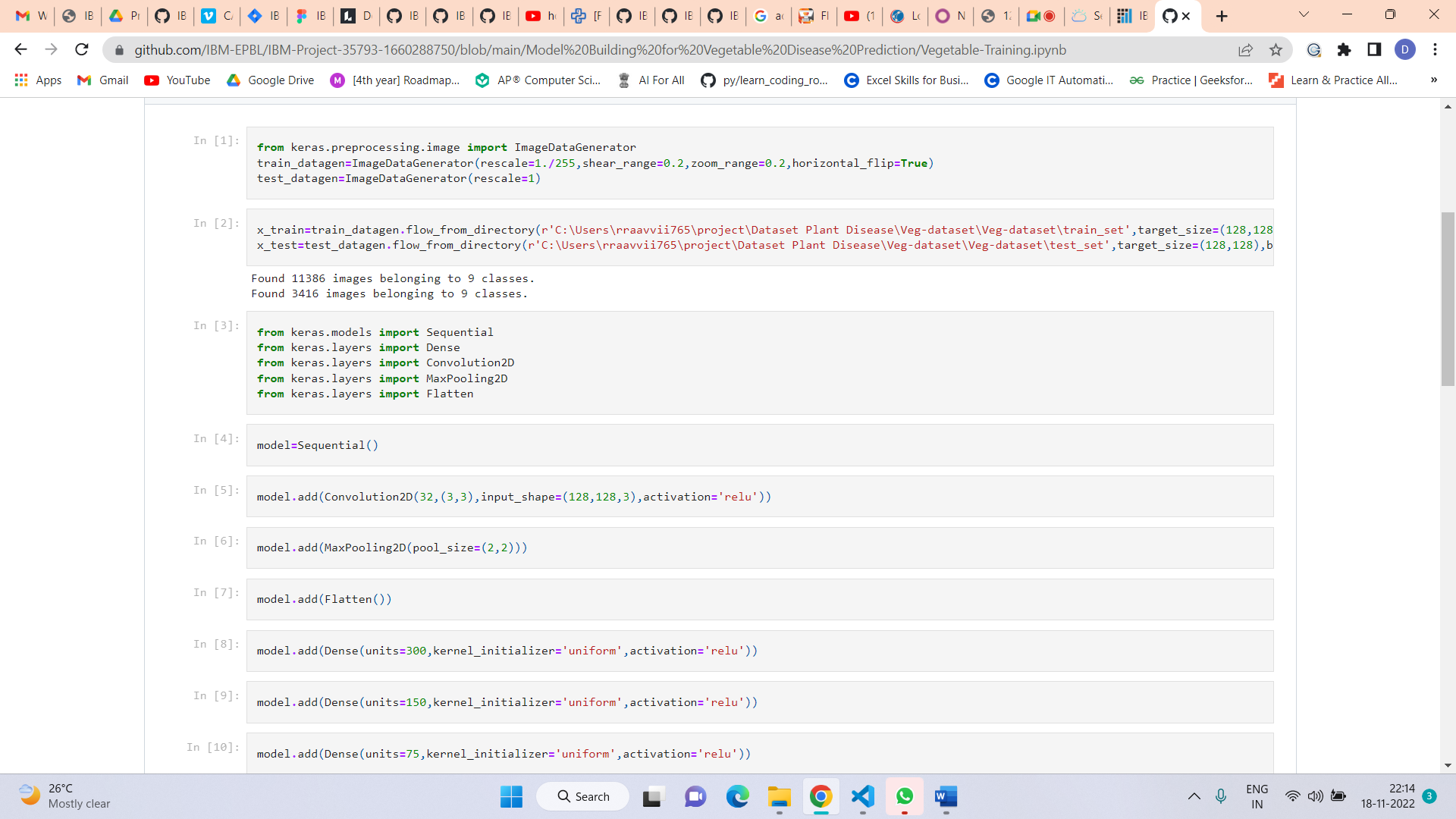


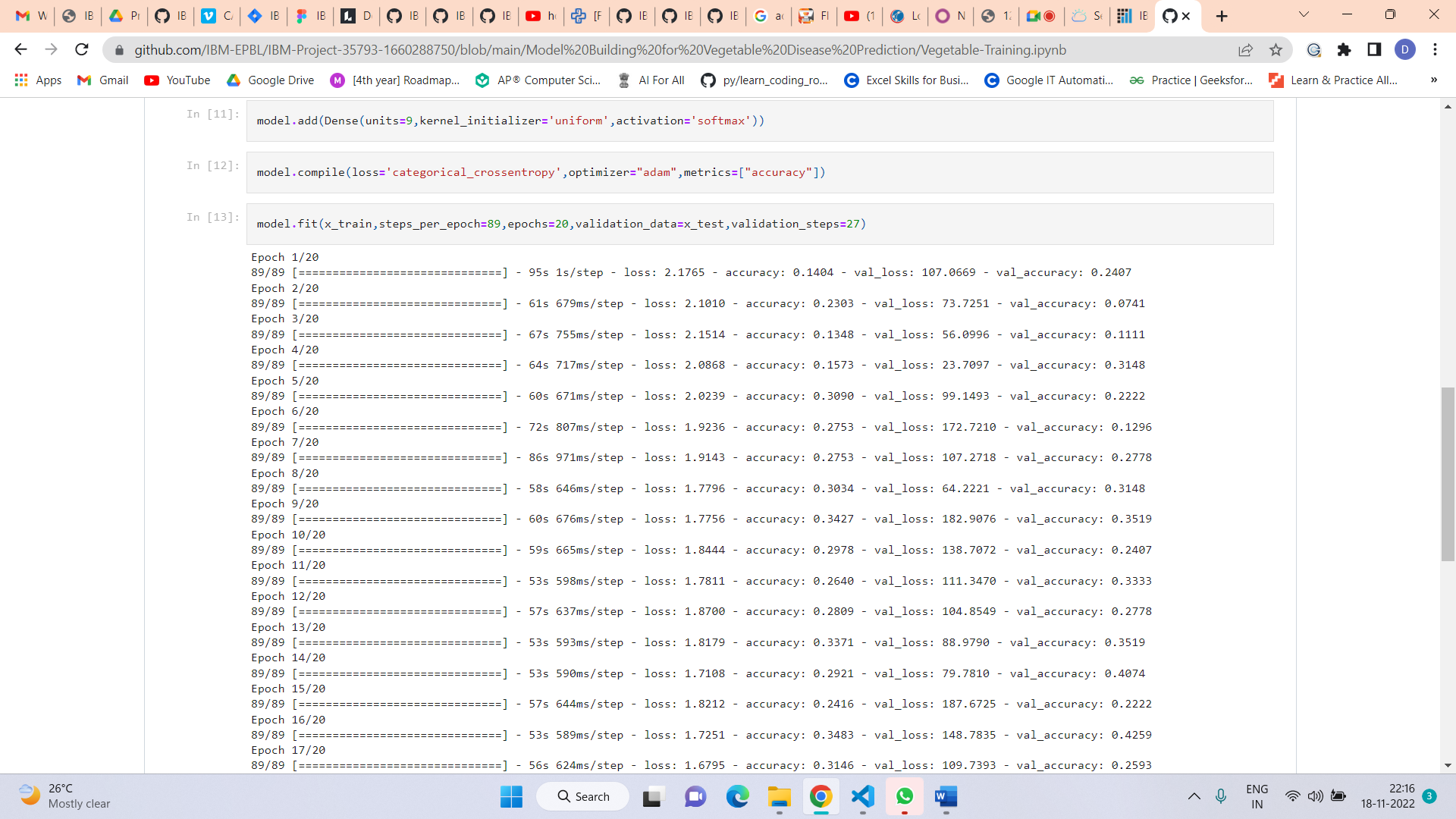
Fruit training.ipynb

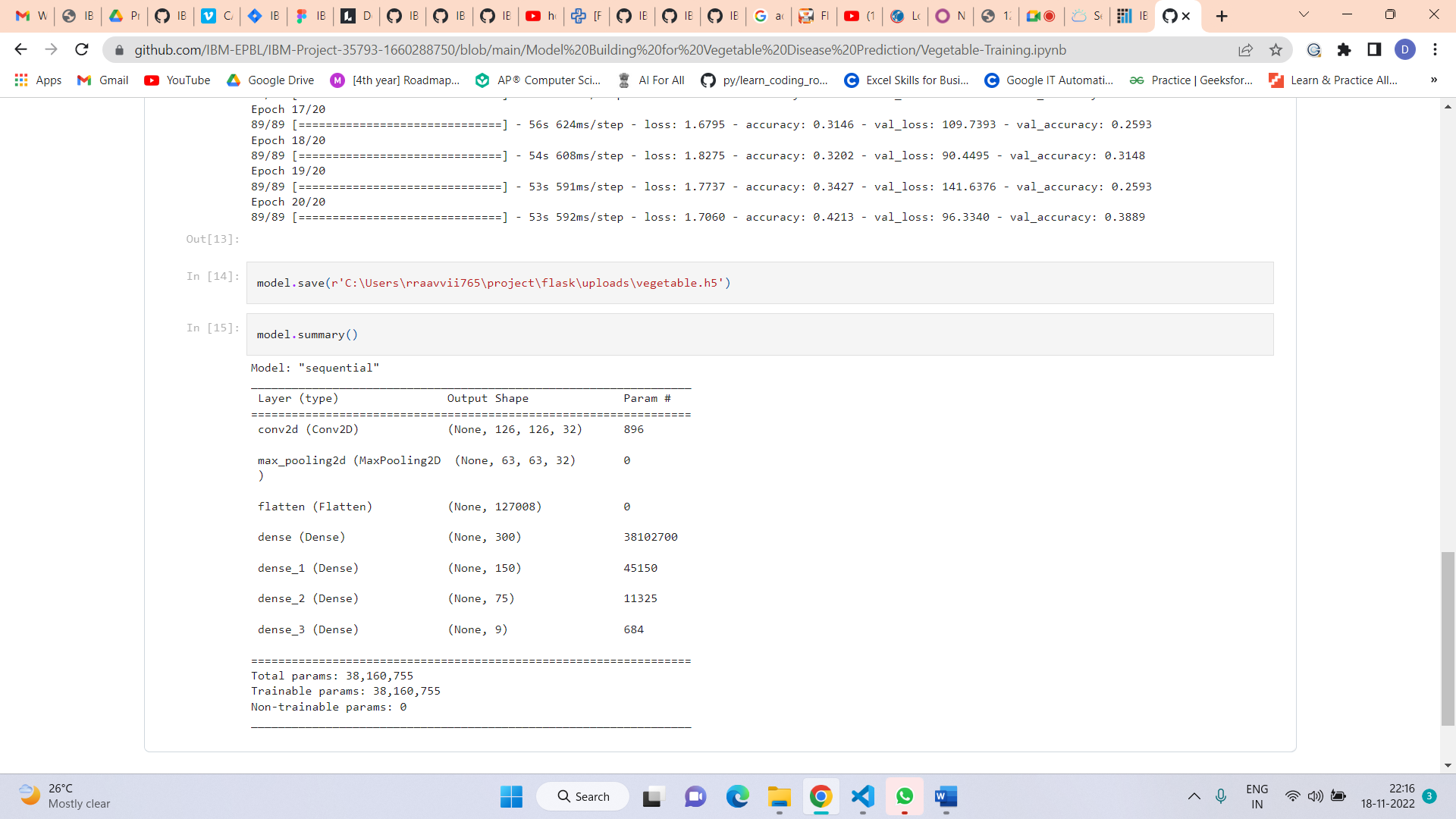


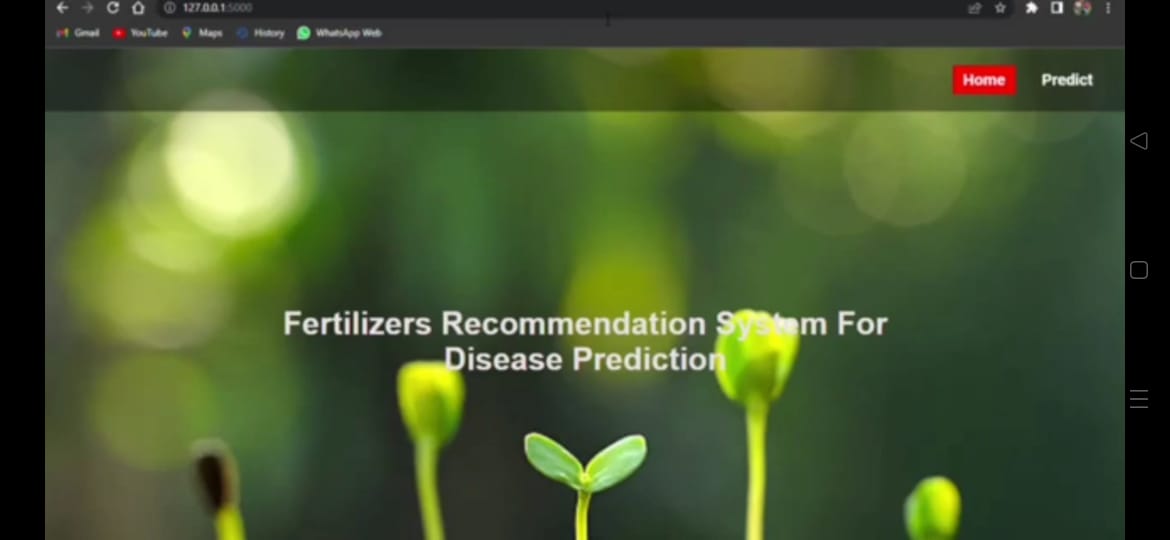


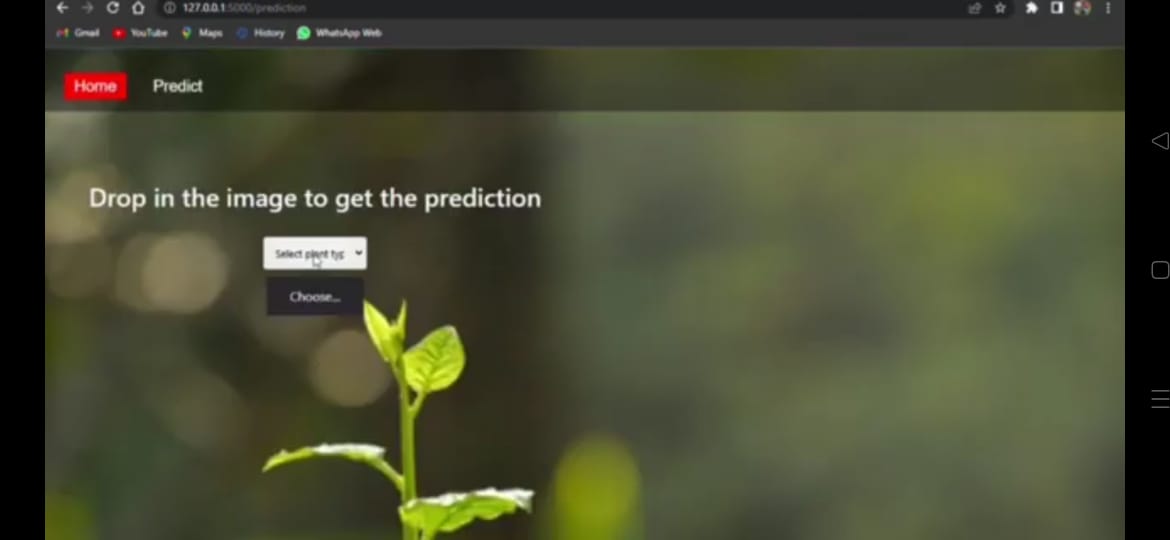
Vegetable training.ipynb

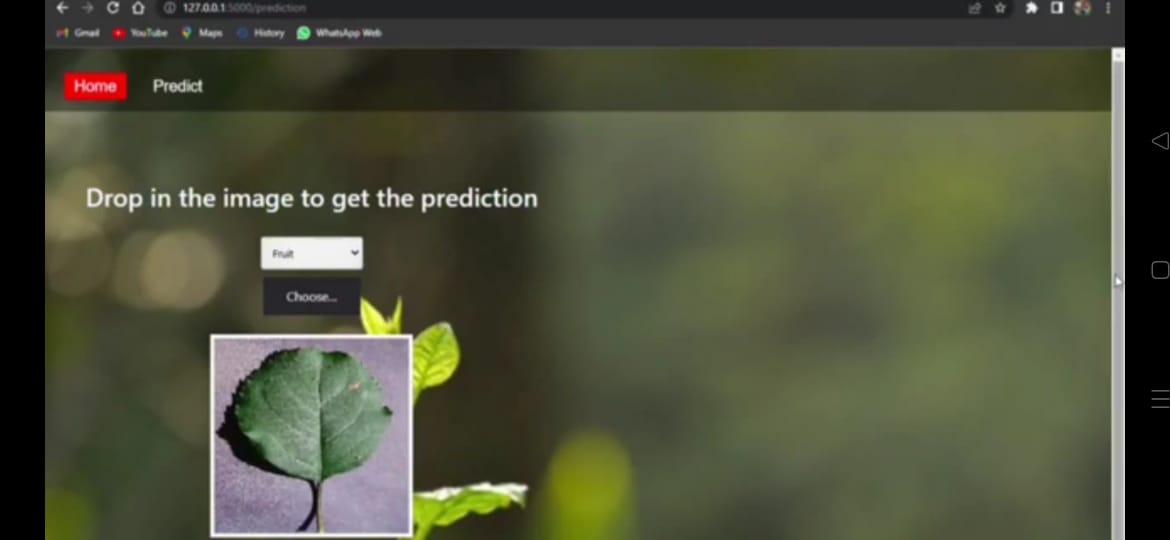


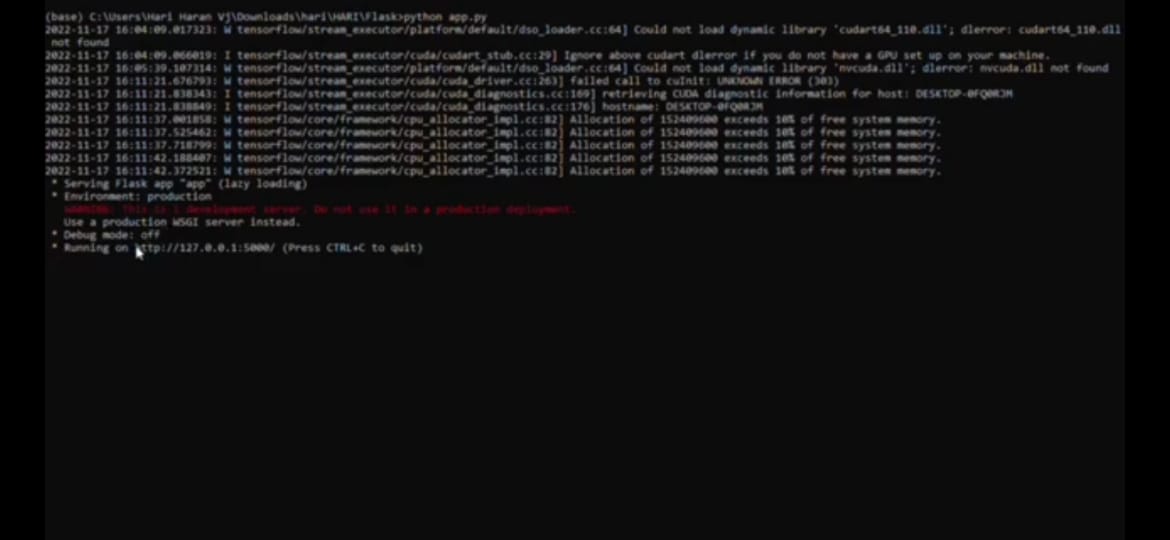


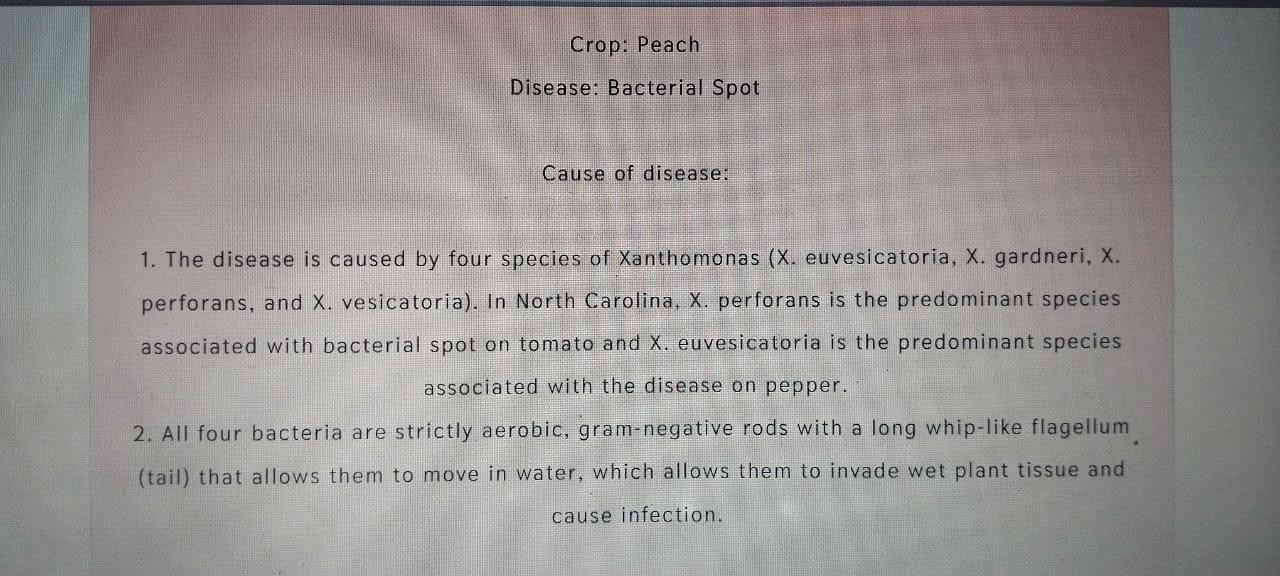


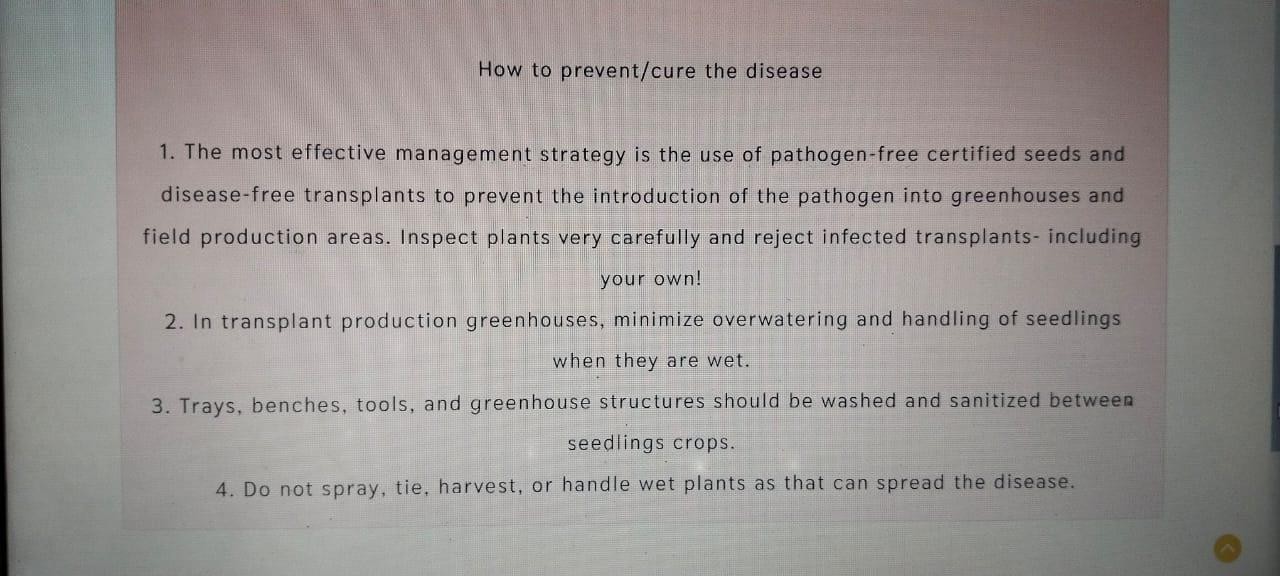












# ADVANTAGES & DISADVANTAGES

## List of advantages

* The proposed model here produces very high accuracy of classification.
* Very large datasets can also be trained and tested.
* Images of very high can be resized within the proposed itself.

## List of disadvantages

* For training and testing, the proposed model requires very high computational time.
* The neural network architecture used in this project work has high complexity.

# 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.

# CONCLUSIONS

The model proposed here involves image classification of fruit datasets and vegetable datasets. The following points are observed during model testing and training:

* 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.

### FUTURE SCOPE:

The model proposed in this project work can be extended to image recognition. The entire model can be converted into application software using Python to exe software. Real-time image classification, image recognition, and video processing are possible using the OpenCV Python library. The work in this project can be extended to security applications such as digit print recognition, iris recognition, and facial recognition.

# BIBILOGRAPHY

[1]. 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.

[2]. 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.

[3]. 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.

APPENDIX

A. Source Code (Jupyter notebook python code)

fruit.ipynb (due to limited page size the code vegetable.ipynb uploaded in github) #!/usr/bin/env python

# coding: utf-8 # In[1]: pwd

# In[2]: cd E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruit- dataset/fruit-dataset

# # Apply ImageDataGenerator functionality to Train and Test set

# # Preprocessing # In[3]: from keras.preprocessing.image import ImageDataGenerator train\_datagen =

ImageDataGenerator(rescale=1./255,shear\_range=0.2,zoom\_range=0.2,horizonta l\_fli p=True) test\_datagen = ImageDataGenerator(rescale=1) # In[4]: pwd

# In[5]: x\_train = train\_datagen.flow\_from\_directory('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruit-

dataset/fruitdataset/train',target\_size=(128,128),batch\_size=32,class\_mode='cate gorical')

# In[6]:

x\_test=test\_datagen.flow\_from\_directory('E:/IBM\_MY\_COURSE/Project/Datas et Plant Disease/fruit-dataset/fruit-dataset/test',target\_size=(128,128), batch\_size=32,class\_mode='categorical') # # Import the models

# In[7]: from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Dense,Convolution2D,MaxPool2D,Flatten

# # Initializing the models 10 # In[8]: model=Sequential() # # Add CNN Layers

# In[9]:

model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu')) # In[10]: x\_train.class\_indices

# # Add Pooling layer

# In[11]: model.add(MaxPool2D(pool\_size=(2,2))) # # Add Flatten layer # In[12]: model.add(Flatten()) # # Add Dense Layer

# In[21]: model.add(Dense(40, kernel\_initializer='uniform',activation='relu')) model.add(Dense(20, kernel\_initializer='random\_uniform',activation='relu'))

# # Add Output Layer # In[24]: model.add(Dense(6,activation='softmax', kernel\_initializer='random\_uniform'))

# # Compile the model # In[25]: model.compile(loss='categorical\_crossentropy',optimizer='adam',metrics=['accur acy' ]) # In[26]: len(x\_train)

# In[27]: 5384/32

# # Fit the Model

# In[28]:

model.fit\_generator(x\_train,steps\_per\_epoch=168,validation\_data=x\_test,validat ion\_st eps=52,epochs=3)

# # Save the Model

# In[29]: model.save("fruit.h5") # In[30]: ls

# # Test the Model

# In[32]: from keras.preprocessing import image from tensorflow.keras.preprocessing.image import img\_to\_array from tensorflow.keras.models import load\_model import numpy as np

# In[33]: model = load\_model("fruit.h5") # # Test Apple\_Healthy Class images

# In[37]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Apple healthy/00fca0da-2db3-481b- b98a9b67bb7b105c RS\_HL 7708.JPG',target\_size=(128,128)) 11

# In[39]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) # In[40]: pred = model.predict\_classes(x)

# In[41]: pred

# In[45]: index

=['Apple Black\_rot','Apple healthy','Corn\_(maize) Northern\_Leaf\_Blig ht','Corn\_( maize) healthy','Peach Bacterial\_spot','Peach healthy']

# In[46]: print('the given image belogs to=',index[pred[0]])

# # Test Apple Black Rot class images # In[54]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Apple Black\_rot/0f3d45f4-e121-42cd- a5b6- be2f866a0574 JR\_FrgE.S 2870.JPG',target\_size=(128,128))

# In[55]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) pred = model.predict\_classes(x) print('the given image belogs to=',index[pred[0]])

# # Test Corn Northern leaf Blight class images

# In[56]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-

dataset/test/Corn\_(maize) Northern\_Leaf\_Blight/00a14441-7a62- 4034-bc40- b196aeab2785 RS\_NLB 3932.JPG',target\_size=(128,128))

# In[57]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) pred = model.predict\_classes(x) print('the given image belogs to=',index[pred[0]])

# # Test Corn Healthy class images # In[58]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Corn\_(maize) healthy/0a68ef5a-027c- 41ae-b227- 159dae77d3dd R.S\_HL 7969 copy.jpg',target\_size=(128,128))

# In[59]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) pred = model.predict\_classes(x) print('the given image belogs to=',index[pred[0]]) # # Test Peach Bacterial spot class images # In[60]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Peach Bacterial\_spot/00ddc106-692e- 4c67-b2e8- 569c924caf49 Rutg.\_Bact.S 1228.JPG',target\_size=(128,128)) 12 # In[61]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) pred = model.predict\_classes(x) print('the given image belogs to=',index[pred[0]])

# # Test Peach Healthy class images

# In[62]: img = image.load\_img('E:/IBM\_MY\_COURSE/Project/Dataset Plant Disease/fruitdataset/fruit-dataset/test/Peach healthy/1a07ce54-f4fd-41cf- b088- 144f6bf71859 Rutg.\_HL 3543.JPG',target\_size=(128,128))

# In[63]: x=image.img\_to\_array(img) x=np.expand\_dims(x,axis=0) pred = model.predict\_classes(x) print('the given image belogs to=',index[pred[0]])