

Project Objectives

Team ID	PNT2022TMID30319
Project Name	Fertilizers Recommendation System For Disease Prediction

Preprocess the image

Image preprocessing is improving image statistics so that undesired distortions are suppressed and image capabilities. We will use CNN machine learning algorithm for image classification.

▼ Import Library

```
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow
import keras
import os
import cv2
import glob
from skimage import io
import random
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
from keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator load_img img to array
https://colab.research.google.com/drive/1NIS8JU55G-edkltz8nSNJIWeRst_Mld#scrollTo=PRGILyTeC2y&printMode=true 1/10
```

10/30/22, 2:43 PM

Image_Preprocessing.ipynb - Colaboratory

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array
from numpy import expand_dims
```

```
path = 'Dataset Plant Disease'
```

```
path1 = 'Dataset Plant Disease/fruit-dataset/fruit-dataset/train'
```

```
path2 = 'Dataset Plant Disease/fruit-dataset/fruit-dataset/test'
```

```
train_data_gen = ImageDataGenerator(rescale = 1./255,
                                    shear_range = 0.2,
                                    zoom_range = 0.2,
                                    horizontal_flip = True,
                                    validation_split = 0.30)
test_data_gen = ImageDataGenerator(rescale = 1./255, validation_split = 0.30)
```

▼ Train and Test

```
training_set = train_data_gen.flow_from_directory(path,
                                                  target_size=(64,64),
                                                  batch_size=100,
                                                  class_mode='categorical',
                                                  shuffle=True,
                                                  color_mode='rgb',
                                                  subset = 'training')
```

```
testing_set = test_data_gen.flow_from_directory(path,
                                                  target_size=(64,64),
                                                  batch_size=100,
                                                  class_mode='categorical',
                                                  shuffle=True,
                                                  color_mode='rgb',
                                                  subset = 'validation')
```

Found 15311 images belonging to 2 classes.

Found 6561 images belonging to 2 classes.

Applying CNN Algorithm

For different batch size, the CNN gives different accuracies. The batch size determines the number of iterations per epoch. Another important hyper parameter is the number of epochs.

```
In [7]: model=Sequential()

In [8]: model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
        model.add(MaxPooling2D(pool_size=(2,2)))
        model.add(Flatten())

        32*(3*3*3+1)
        model.add(Dense(300,activation='relu'))
        model.add(Dense(150,activation='relu'))
        model.add(Dense(6,activation='softmax'))
        model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
=====		
conv2d (Conv2D)	(None, 126, 126, 32)	896
max_pooling2d (MaxPooling2D)	(None, 63, 63, 32)	0
flatten (Flatten)	(None, 127008)	0
dense (Dense)	(None, 300)	38102700
dense_1 (Dense)	(None, 150)	45150
dense_2 (Dense)	(None, 6)	906
=====		
Total params: 38,149,652		
Trainable params: 38,149,652		
Non-trainable params: 0		

Deep neural networks detect the disease

It is more efficient to detect plant leaf disease and easier to obtain higher accuracy when using a novel deep learning approach based on CNNs.

```
Upload=files.upload()

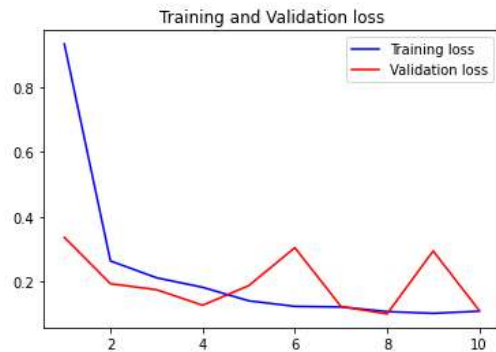
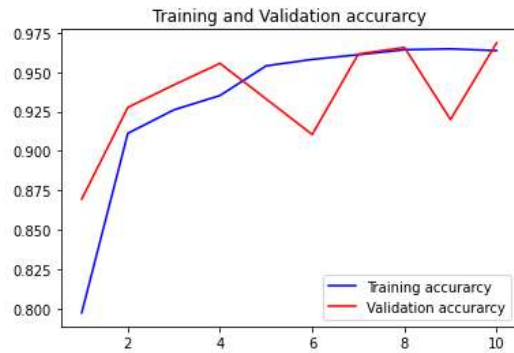
for path in Upload.keys():
    input=image.load_img(path,target_size=(128,128))
    x=image.img_to_array(input)
    x=np.expand_dims(x,axis=0)
    y=np.argmax(model.predict(x),axis=1)
    index=['Pepper,_bell___Bacterial_spot','Pepper,_bell___healthy','Potato___Early_blight','Potato___Late_blight','Potato___healthy','Tomato___Bacteria']
    index[y[0]]
    print(index[y[0]])
```

Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving 00d8f10f-5038-4e0f-bb58-0b885ddc0cc5___RS_Early.B 8722.JPG to 00d8f10f-5038-4e0f-bb58-0b885ddc0cc5___RS_Early.B 8722.JPG
Saving 0a8a68ee-f587-4dea-beec-79d02e7d3fa4___RS_Early.B 8461.JPG to 0a8a68ee-f587-4dea-beec-79d02e7d3fa4___RS_Early.B 8461.JPG
Saving 0a47f32c-1724-4c8d-bfe4-986cedd3587b___RS_Early.B 8001.JPG to 0a47f32c-1724-4c8d-bfe4-986cedd3587b___RS_Early.B 8001.JPG
Saving 0a0744dc-8486-4fbb-a44b-4d63e6db6197___RS_Early.B 7575.JPG to 0a0744dc-8486-4fbb-a44b-4d63e6db6197___RS_Early.B 7575.JPG
Saving 0a6983a5-895e-4e68-9edb-88adf79211e9___RS_Early.B 9072.JPG to 0a6983a5-895e-4e68-9edb-88adf79211e9___RS_Early.B 9072.JPG
Saving 0a79700b-f834-41f5-ae51-6ceda6f67a48___RS_Early.B 8951.JPG to 0a79700b-f834-41f5-ae51-6ceda6f67a48___RS_Early.B 8951.JPG
Saving 0ad3ba53-f01b-403b-a99d-5991eed85045___RS_Early.B 7600.JPG to 0ad3ba53-f01b-403b-a99d-5991eed85045___RS_Early.B 7600.JPG
Saving 0bbb8bce-2020-416b-8bd6-c160c2db9921___RS_Early.B 8386.JPG to 0bbb8bce-2020-416b-8bd6-c160c2db9921___RS_Early.B 8386.JPG
1/1 [=====] - 0s 423ms/step
Tomato___Septoria_leaf_spot
1/1 [=====] - 0s 31ms/step
Potato___Early_blight
1/1 [=====] - 0s 31ms/step
Pepper,_bell___healthy
1/1 [=====] - 0s 31ms/step
Tomato___Septoria_leaf_spot
1/1 [=====] - 0s 31ms/step
Potato___Early_blight
1/1 [=====] - 0s 29ms/step
Potato___Early_blight
1/1 [=====] - 0s 30ms/step
Tomato___Septoria_leaf_spot
1/1 [=====] - 0s 35ms/step
Potato___Early_blight
```

Accuracy of the model

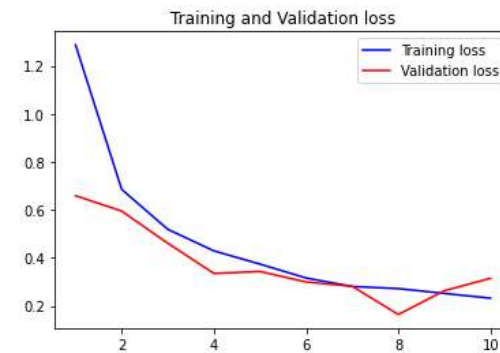
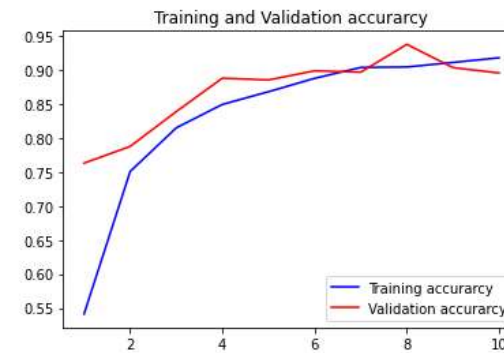
Fruit



Test accuracy : 97.1

Train accuracy : 97.9

Vegetable

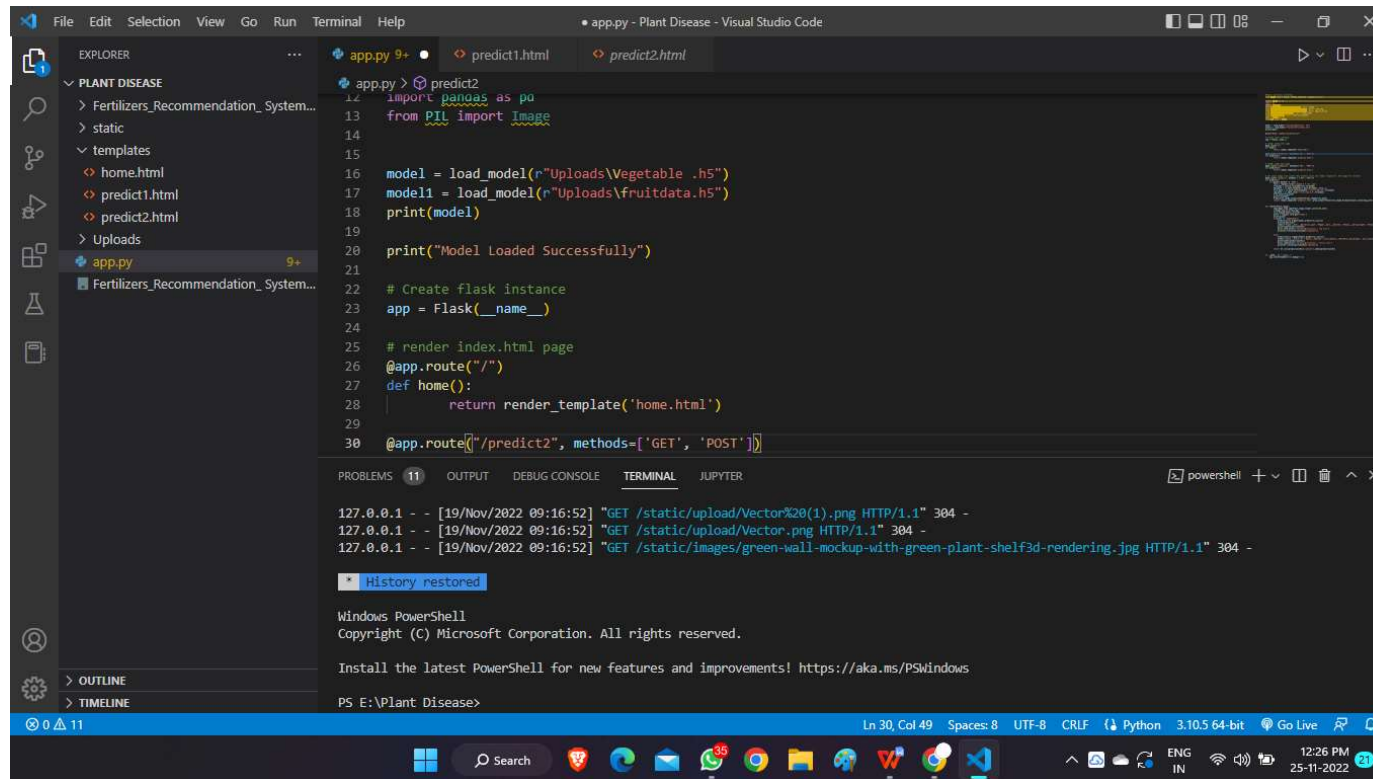


Test accuracy : 89.5

Train accuracy : 89.5

Web application of flask framework

Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python.



```
File Edit Selection View Go Run Terminal Help
• app.py - Plant Disease - Visual Studio Code

EXPLORER
  PLANT DISEASE
    Fertilizers_Recommendation_System...
    static
    templates
      home.html
      predict1.html
      predict2.html
    Uploads
      app.py 9+
      Fertilizers_Recommendation_System...

app.py 9+
12 import pandas as pd
13 from PIL import Image
14
15
16 model = load_model(r"Uploads\Vegetable .h5")
17 model1 = load_model(r"Uploads\fruitdata.h5")
18 print(model)
19
20 print("Model Loaded Successfully")
21
22 # Create flask instance
23 app = Flask(__name__)
24
25 # render index.html page
26 @app.route("/")
27 def home():
28     return render_template('home.html')
29
30 @app.route("/predict2", methods=['GET', 'POST'])

TERMINAL
127.0.0.1 - - [19/Nov/2022 09:16:52] "GET /static/upload/Vector%20(1).png HTTP/1.1" 304 -
127.0.0.1 - - [19/Nov/2022 09:16:52] "GET /static/upload/Vector.png HTTP/1.1" 304 -
127.0.0.1 - - [19/Nov/2022 09:16:52] "GET /static/images/green-wall-mockup-with-green-plant-shelf3d-rendering.jpg HTTP/1.1" 304 -
* History restored
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS E:\Plant Disease>
```