pwd

In [4]:

```
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm
```

In [2]:

```
path="/home/rogueware/Documents/Dataset/PlantVillage-Dataset/raw/color"
a=[]
training_data=[]
classlist=[]
import os
from tqdm import tqdm
for i in tqdm(os.listdir(path)):
    classlist.append(i)
    for j in (os.listdir(path+"/"+i)):
        try:
            img_array = cv2.imread(path+"/"+i+"/"+j ,cv2.IMREAD_COLOR)
            #print(img_array.shape)
            # convert to array
            new_array = cv2.resize(img_array, (120,120))  # resize to normalize dat
            #print(new_array)
            training_data.append([new_array,i])  # add this to our training_data
        except Exception as e:  # in the interest in keeping the output clean...
            pass
```

```
100%|██████████| 38/38 [01:23<00:00,  1.83s/it]
```

In [3]:

```
classdict={}
count=0
for i in classlist:
    classdict[i]=count
    count+=1
```

In [4]:

```
for i in tqdm(range(len(training_data))):
    training_data[i][1]=classdict[training_data[i][1]]
```

```
100%|██████████| 54305/54305 [00:00<00:00, 1245354.92it/s]
```

In [11]:

```
classdict
```

Out[11]:

```
{'Tomato___healthy': 0,
 'Tomato___Septoria_leaf_spot': 1,
 'Corn_(maize)___healthy': 2,
 'Peach___healthy': 3,
 'Apple___Cedar_apple_rust': 4,
 'Squash___Powdery_mildew': 5,
 'Grape___healthy': 6,
 'Tomato___Tomato_mosaic_virus': 7,
 'Tomato___Bacterial_spot': 8,
 'Corn_(maize)___Common_rust_': 9,
 'Cherry_(including_sour)___Powdery_mildew': 10,
 'Apple___Apple_scab': 11,
 'Potato___Late_blight': 12,
 'Strawberry___Leaf_scorch': 13,
 'Orange___Haunglongbing_(Citrus_greening)': 14,
 'Corn_(maize)___Northern_Leaf_Blight': 15,
 'Pepper,_bell___healthy': 16,
 'Grape___Black_rot': 17,
 'Pepper,_bell___Bacterial_spot': 18,
 'Tomato___Early_blight': 19,
 'Blueberry___healthy': 20,
 'Cherry_(including_sour)___healthy': 21,
 'Potato___healthy': 22,
 'Apple___Black_rot': 23,
 'Grape___Leaf_blight_(Isariopsis_Leaf_Spot)': 24,
 'Tomato___Target_Spot': 25,
 'Tomato___Spider_mites Two-spotted_spider_mite': 26,
 'Tomato___Tomato_Yellow_Leaf_Curl_Virus': 27,
 'Apple___healthy': 28,
 'Soybean___healthy': 29,
 'Grape___Esca_(Black_Measles)': 30,
 'Raspberry___healthy': 31,
 'Strawberry___healthy': 32,
 'Peach___Bacterial_spot': 33,
 'Potato___Early_blight': 34,
 'Corn_(maize)___Cercospora_leaf_spot Gray_leaf_spot': 35,
 'Tomato___Leaf_Mold': 36,
 'Tomato___Late_blight': 37}
```

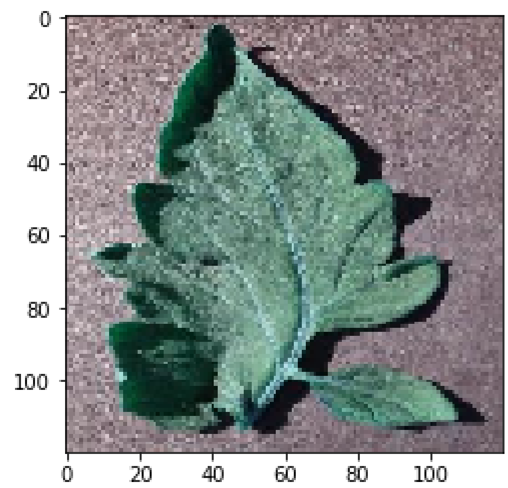
In [6]:

```
for i in training_data:
    print(i[1],end=" ")
```

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

In [5]:

```
for i in range(len(training_data)):
    if training_data[i][1]!=training_data[i+1][1]:
        plt.imshow(training_data[i][0])
        plt.show()
        print(training_data[i][1])
```




```
0
```

In [22]:

```
import random
random.shuffle(training_data)
for sample in training_data[:10]:
    print(sample[1])
```

```
---------------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
 last)
<ipython-input-22-35b6d771cfd1> in <module>
      1 import random
----> 2 random.shuffle(training_data)
      3 for sample in training_data[:10]:
      4     print(sample[1])

NameError: name 'training_data' is not defined
```

In [21]:

```
import keras
X = []
y = []

for features,label in training_data:
    X.append(features)
    y.append(label)

#print(X[0].reshape(-1, IMG_SIZE, IMG_SIZE, 1))

X = np.array(X).reshape(-1, 120, 120,3)
print(X.shape)
y = keras.utils.to_categorical(y, 38)
#print(y[0:5])
print(X[0])
print(y[0])
```

```
---------------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
 last)
<ipython-input-21-aed630575ca3> in <module>
      3 y = []
      4
----> 5 for features,label in training_data:
      6     X.append(features)
      7     y.append(label)

NameError: name 'training_data' is not defined
```

In [10]:

```
import pickle

pickle_out = open("X.pickle","wb")
pickle.dump(X, pickle_out)
pickle_out.close()

pickle_out = open("Y.pickle","wb")
pickle.dump(y, pickle_out)
pickle_out.close()
```

In [12]:

```
pickle_out1 = open("classdict.pickle","wb")
pickle.dump(classdict, pickle_out1)
pickle_out1.close()
```

In [1]:

```
import pickle
infile = open("X.pickle","rb")
X = pickle.load(infile)
infile.close()
infile1 = open("Y.pickle","rb")
Y = pickle.load(infile1)
infile1.close()
infile2 = open("classdict.pickle","rb")
classdict = pickle.load(infile2)
infile2.close()
```

In [2]:

```
import pickle
import matplotlib.pyplot as plt
import numpy as np
from keras import backend
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPooling2D, Activation
from sklearn.model_selection import train_test_split
from tensorflow.keras.callbacks import TensorBoard
from keras.applications.mobilenet import MobileNet
from keras.applications.vgg16 import preprocess_input, decode_predictions
from keras.applications.xception import xception
```

```
Using TensorFlow backend.
```

In [3]:

```
input_shape=X.shape
print(input_shape[1:])
```

```
(120, 120, 3)
```

In [25]:

```
del(model)
model = Sequential()
model.add(Conv2D(128, kernel_size=(7, 7), activation='relu', input_shape=input_shap
#model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(5, 5)))
model.add(Conv2D(256, kernel_size=(5, 5), activation='relu'))
#model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(3, 3)))
model.add(Conv2D(256, (3, 3), activation='relu'))
#model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(200, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(100, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(100, activation='relu'))
model.add(Dropout(0.1))
model.add(Dense(38, activation='softmax'))
model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_16 (Conv2D)           (None, 114, 114, 128)     18944
_________________________________________________________________
max_pooling2d_15 (MaxPooling (None, 22, 22, 128)       0
_________________________________________________________________
conv2d_17 (Conv2D)           (None, 18, 18, 256)       819456
_________________________________________________________________
max_pooling2d_16 (MaxPooling (None, 6, 6, 256)         0
_________________________________________________________________
conv2d_18 (Conv2D)           (None, 4, 4, 256)         590080
_________________________________________________________________
max_pooling2d_17 (MaxPooling (None, 2, 2, 256)         0
_________________________________________________________________
flatten_5 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_17 (Dense)             (None, 200)               205000
_________________________________________________________________
dropout_10 (Dropout)         (None, 200)               0
_________________________________________________________________
dense_18 (Dense)             (None, 100)               20100
_________________________________________________________________
dropout_11 (Dropout)         (None, 100)               0
_________________________________________________________________
dense_19 (Dense)             (None, 100)               10100
_________________________________________________________________
dropout_12 (Dropout)         (None, 100)               0
_________________________________________________________________
dense_20 (Dense)             (None, 38)                3838
=================================================================
Total params: 1,667,518
Trainable params: 1,667,518
Non-trainable params: 0
_________________________________________________________________
```

In [26]:

```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.30, random_st
```

In [27]:

```
x_train, x_val, y_train, y_val = train_test_split(x_train, y_train, test_size=0.1,
```

In [7]:

```
del(X)
del(Y)
```

In [28]:

```
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
```

In [32]:

```
#NAME = "PlantDisease-CNN"
#tensorboard = TensorBoard(log_dir="C:\Users\admin\Desktop\pickle\current_logs")


model.compile(loss=keras.losses.categorical_crossentropy,
              optimizer=keras.optimizers.Adam(),
              metrics=['accuracy'])


model.fit(x_train, y_train,
          batch_size=500,
          epochs=20,
          verbose=1,
          validation_data=(x_test, y_test),
         )
```

```
Train on 34211 samples, validate on 16292 samples
Epoch 1/20
34211/34211 [==============================] - 30s 864us/step - loss:
3.1328 - acc: 0.1766 - val_loss: 2.6689 - val_acc: 0.2746
Epoch 2/20
34211/34211 [==============================] - 28s 822us/step - loss:
2.1652 - acc: 0.3909 - val_loss: 1.7442 - val_acc: 0.4963
Epoch 3/20
34211/34211 [==============================] - 29s 835us/step - loss:
1.5185 - acc: 0.5594 - val_loss: 1.1845 - val_acc: 0.6540
Epoch 4/20
34211/34211 [==============================] - 28s 827us/step - loss:
1.1838 - acc: 0.6472 - val_loss: 0.8951 - val_acc: 0.7314
Epoch 5/20
34211/34211 [==============================] - 28s 821us/step - loss:
0.9517 - acc: 0.7086 - val_loss: 0.7329 - val_acc: 0.7712
Epoch 6/20
34211/34211 [==============================] - 29s 833us/step - loss:
0.8103 - acc: 0.7479 - val_loss: 0.7343 - val_acc: 0.7691
Epoch 7/20
34211/34211 [==============================] - 29s 843us/step - loss:
0.7004 - acc: 0.7803 - val_loss: 0.5811 - val_acc: 0.8156
Epoch 8/20
34211/34211 [==============================] - 28s 830us/step - loss:
0.6110 - acc: 0.8061 - val_loss: 0.6313 - val_acc: 0.8019
Epoch 9/20
34211/34211 [==============================] - 28s 832us/step - loss:
0.5396 - acc: 0.8281 - val_loss: 0.5288 - val_acc: 0.8303
Epoch 10/20
34211/34211 [==============================] - 29s 837us/step - loss:
0.4693 - acc: 0.8474 - val_loss: 0.4444 - val_acc: 0.8581
Epoch 11/20
34211/34211 [==============================] - 28s 828us/step - loss:
0.4335 - acc: 0.8590 - val_loss: 0.3993 - val_acc: 0.8693
Epoch 12/20
34211/34211 [==============================] - 28s 829us/step - loss:
0.3836 - acc: 0.8763 - val_loss: 0.3651 - val_acc: 0.8829
Epoch 13/20
34211/34211 [==============================] - 29s 835us/step - loss:
0.3395 - acc: 0.8893 - val_loss: 0.3523 - val_acc: 0.8884
Epoch 14/20
34211/34211 [==============================] - 29s 839us/step - loss:
0.3170 - acc: 0.8948 - val_loss: 0.3837 - val_acc: 0.8814
```

```
Epoch 15/20
34211/34211 [==============================] - 28s 828us/step - loss:
0.2786 - acc: 0.9082 - val_loss: 0.3307 - val_acc: 0.8961
Epoch 16/20
34211/34211 [==============================] - 28s 828us/step - loss:
0.2519 - acc: 0.9170 - val_loss: 0.2935 - val_acc: 0.9063
Epoch 17/20
34211/34211 [==============================] - 29s 841us/step - loss:
0.2228 - acc: 0.9266 - val_loss: 0.2813 - val_acc: 0.9116
Epoch 18/20
34211/34211 [==============================] - 28s 828us/step - loss:
0.2047 - acc: 0.9318 - val_loss: 0.2647 - val_acc: 0.9174
Epoch 19/20
34211/34211 [==============================] - 29s 837us/step - loss:
0.2067 - acc: 0.9314 - val_loss: 0.2911 - val_acc: 0.9080
Epoch 20/20
34211/34211 [==============================] - 28s 833us/step - loss:
0.1827 - acc: 0.9394 - val_loss: 0.3226 - val_acc: 0.9000
```

Out[32]:

```
<keras.callbacks.History at 0x195e3ddb550>
```

In [21]:

```
model.save("Tanmay_Project_model")
```

In [37]:

```
arr=model.predict_classes(x_val[:])
#print(arr)
#plt.imshow(x_val[15])
#plt.show()
#print(y_val[15])
```

In [38]:

```
count=0

ar=[]
for i in range(len(y_val)):
    for j in range(len(y_val[i])):
        if y_val[i][j]==1:
            ar.append(j)
for i in range(len(ar)):
    #print(arr[i],ar[i])
    if arr[i]==ar[i]:
        count+=1

print((count/len(ar))*100)
```

```
76.80852183061546
```

In [ ]:

```

```