Team-14

Project Name:
Seed classification

TEAM MEMBERS

Pothuri Akish	Arvind Karthik
akishpothuri@gmail.com	arvindkarthik2000@gamil.com
Dhanush Nunna	Kiran Lakhani
dhanush.nunna@gamil.com	kiranlakhani20@gamil.com

TASK PROGRESS

IMPORTING ImageDataGenerator CLASS:

```
from keras.preprocessing.image import ImageDataGenerator
```

CONFIGURING ImageDataGenerator CLASS:

```
train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.5,
horizontal_flip=True,vertical_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
```

APPLYING ImageDataGenerator FUNCTIONALITY TO TRAINSET & TESTSET:

```
x_train=train_datagen.flow_from_directory(r'C:\Users\KIRAN
LAKHANI\Desktop\project1\train',target_size=(64,64),
batch_size=4,class_mode='categorical')

x_test=test_datagen.flow_from_directory(r'C:\Users\KIRAN
LAKHANI\Desktop\project1\test',
target_size=(64,64),batch_size=4,class_mode='categorical')

print(x_train.class_indices)

model.compile(loss='categorical_crossentropy', optimizer='adam',metrics=['accuracy'])
```

IMPORTING THE MODEL LIBRARIES

```
_from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

INITIALIZE THE MODEL

```
model = Sequential()
```

ADDING CNN LAYERS

```
model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = 'relu'))

model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Convolution2D(64,(3,3),activation = 'relu'))

model.add(MaxPooling2D(pool_size = (2,2)))

model.add(Flatten())
```

ADDING DENSE LAYERS

```
model.add(Dense(output_dim = 600 ,init = 'uniform',activation = 'relu'))

model.add(Dense(output_dim = 12,activation = 'softmax'))
```

IMAGE DATA GENERATOR FUNCTIOINALITY

```
from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range =
0.2,horizontal_flip = True)

test_datagen =ImageDataGenerator(rescale = 1)
```

TRAIN AND TEST THE MODEL

```
x_train = train_datagen.flow_from_directory(r'E:\python_basics\project\train',target_size =
(64,64),batch_size = 32, class_mode = 'categorical')

x_test =  test_datagen.flow_from_directory(r'E:\python_basics\project\test',target_size =
(64,64),batch_size = 32, class_mode = 'categorical')

print(x_train.class_indices)

model.compile(optimizer = 'adam',loss = 'categorical_crossentropy',metrics = ['accuracy'])

model.fit_generator(x_train, steps_per_epoch = 150,epochs=30,validation_data =
x_test,validation_steps = 30)
```

SAVE THE MODEL

```
model.save("seedling.h5")
```

REPORT

1 INTRODUCTION

a. Overview:

Seed classification is a process in which different varieties of seeds are categorized into different classes on the basis of their morphological features. In the present work We performed seed classification using CNN-model.The data was collected from Agriculture website's database.The classifiers used from these methods are Multilayer Perceptron, Classifiers, Multi Class Classifier.Multi fold Cross Validation as well as Training Set method. Multilayer Perceptron and Logistics from function method gives higher accuracy than all other classifiers, which is 95% for both the classifiers using multi fold Cross Validation. We also analyzed the results using 5 fold and 2 fold Cross Validation,we decrease the fold value except the Multi fold Perceptron classifier that gives the highest accuracy value 95% using multi fold Cross Validation. Finally we also observed that Training Set method gives higher accuracy than Cross Validation during the classification process.

1.2 Purpose:

Seed is the basic and most critical input for sustainable agriculture. The response of all other inputs depends on quality of seeds to a large extent. It is estimated that the direct contribution of quality seed alone to the total production is about 15 – 20% depending upon the crop and it can be further raised up to 45% with efficient management of other inputs.By the seed classification farmer's will know what is the preferable seed and in which seed they get yield according to their soil type.

In the India 70% of the farmers depends on the daily labour to remove unwanted weeds in their fields.By the seed classification they may get help to recognize the weeds in their fields by their own.

2 LITERATURE SURVEY

2.1 Existing problem:

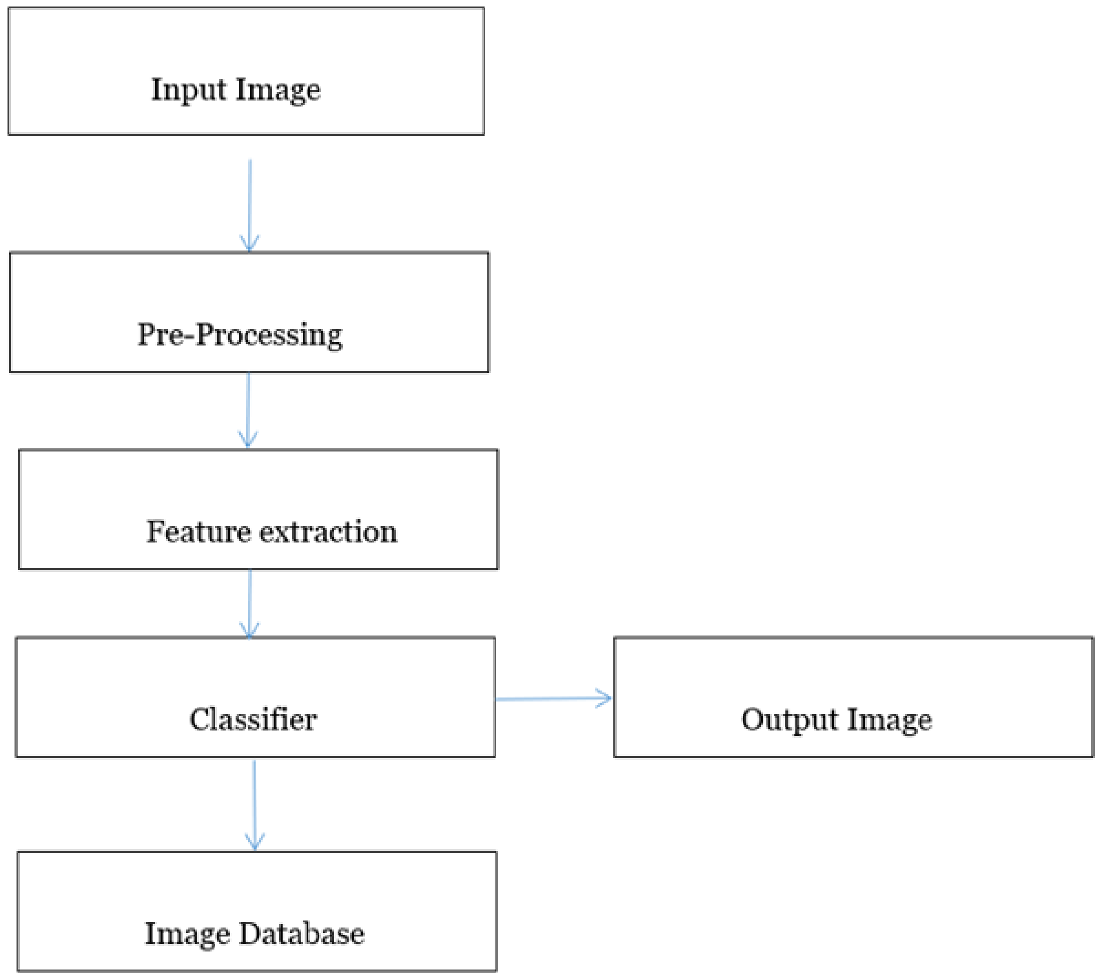
In recent times, the state of agriculture and the amount of work people need to put in to check if plants/food is growing correctly is phenomenal, because it is 2019 and workers still need to organize and recognize the difference between different plants and weeds. People who are working in the agriculture field still have to have the ability to sort and recognize different plants and weeds, which does take a lot of time and a lot of effort in the long term.This is where Artificial Intelligence can actually help benefit those workers, as the time and energy to identify plant seedlings will be much shortened. The ability to do so effectively can mean better crop yields and in the long term will result in better care for the environment. As by identifying the difference among the plants and weeds in a timely manner where it is highly accurate can positively impact agriculture.

2.2 Proposed solution:

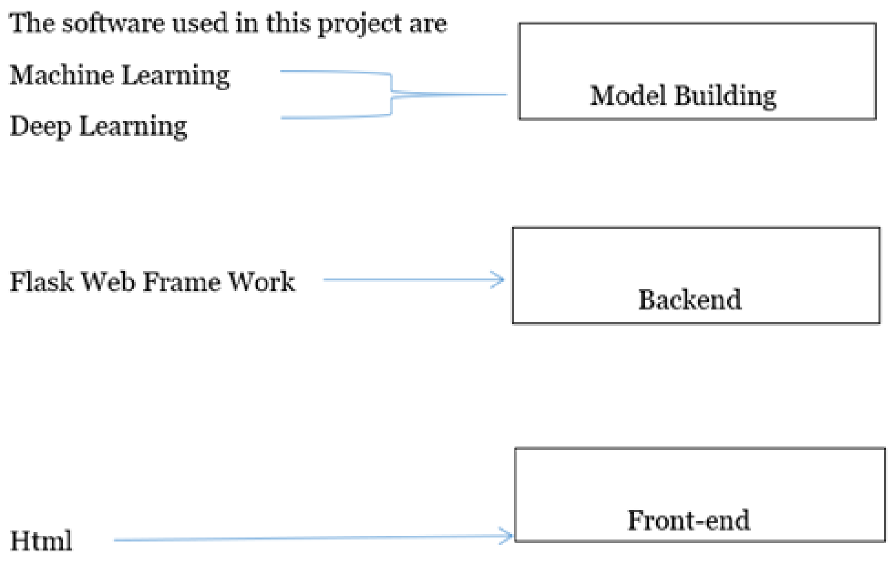
In this work, we explore the performance of traditional computer vision methods on this task and show that a Deep Convolutional Neural Network (CNN) does the best job at classifying plant seedlings. In this work, CNN is adopted for plant seedling classification to automatically discriminate between weed species and crops at early growth stages. . In this project by using AI system will make a machine to identify the plant seeds from the weeds, so this was helpful to the farmers and they can identify the crop growth or plant seeds from weed.

3 THEORICAL ANALYSIS

3.1 Block Diagram:



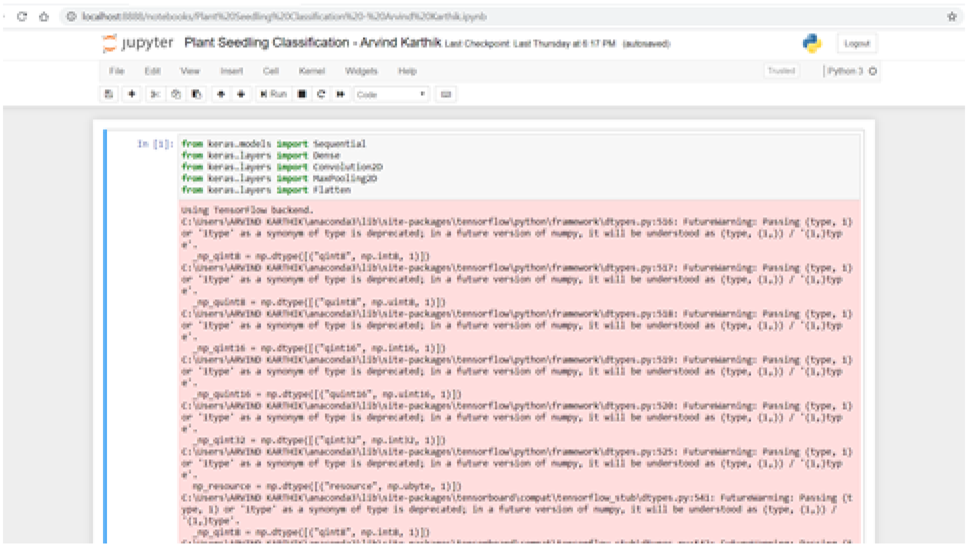
3.2 Software designing:



4 . EXPERIMENTAL INVESTIGATION

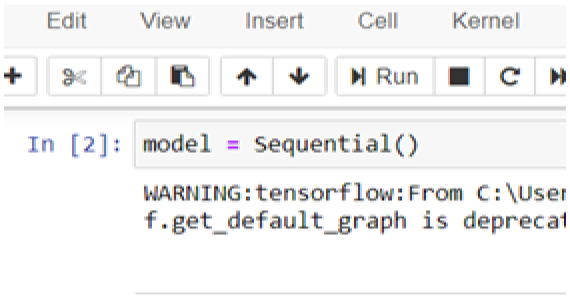
In making our Seedling classification model, we followed a few steps. They are-

1. importing the libraries


localhost:8888/notebooks/Plant%20Seedling%20Classification%20-%20Arvind%20Karthik.ipynb
jupyter Plant Seedling Classification - Arvind Karthik Last Checkpoint: Last Thursday at 6:17 PM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [1]: from keras.models import Sequential
        from keras.layers import Dense
        from keras.layers import Convolution2D
        from keras.layers import MaxPooling2D
        from keras.layers import Flatten

        Using TensorFlow backend.
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:516: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          _np_qint8 = np.dtype([("qint8", np.int8, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:517: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:518: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          _np_qint16 = np.dtype([("qint16", np.int16, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:519: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:520: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          _np_qint32 = np.dtype([("qint32", np.int32, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py:525: FutureWarning: Passing (type, 1)
        or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,)typ
        e'.
          np_resource = np.dtype([("resource", np.ubyte, 1)])
        C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorboard\compat\tensorflow_stub\dtypes.py:541: FutureWarning: Passing (t
        ype, 1) or '1type' as a synonym of type is deprecated; in a future version of numpy, it will be understood as (type, (1,)) /
        '(1,)type'.
          _np_qint8 = np.dtype([("qint8", np.int8, 1)])
```

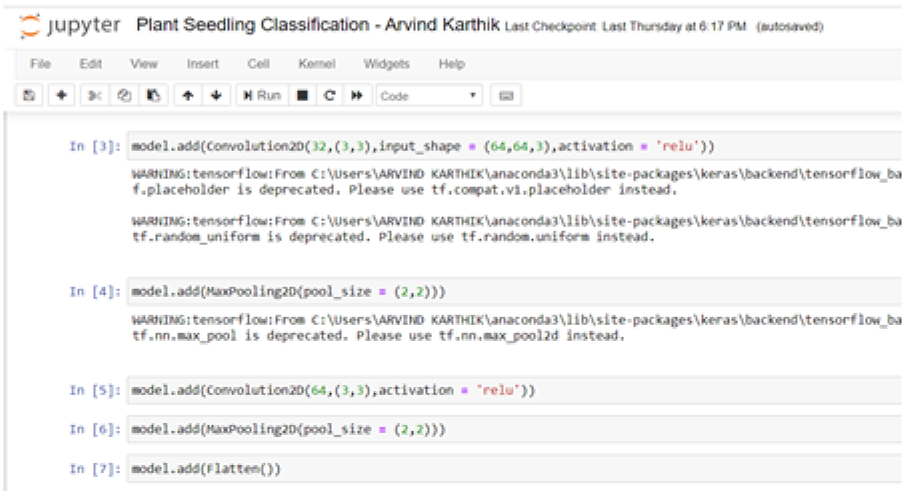
2. initialize the model



```
In [2]: model = Sequential()

        WARNING:tensorflow:From C:\User
        f.get_default_graph is deprecat
```

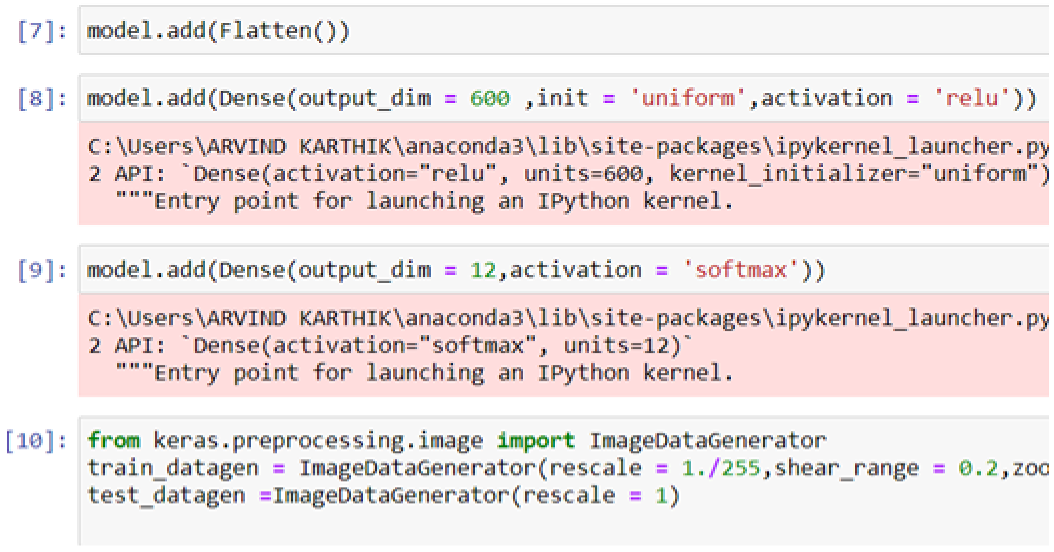
3. loading the preprocessed data

4. adding cnn layers


jupyter Plant Seedling Classification - Arvind Karthik Last Checkpoint: Last Thursday at 6:17 PM (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run Code

```
In [3]: model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = 'relu'))

        WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\backend\tensorflow_ba
        f.placeholder is deprecated. Please use tf.compat.v1.placeholder instead.

        WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\backend\tensorflow_ba
        tf.random_uniform is deprecated. Please use tf.random.uniform instead.

In [4]: model.add(MaxPooling2D(pool_size = (2,2)))

        WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\backend\tensorflow_ba
        tf.nn.max_pool is deprecated. Please use tf.nn.max_pool2d instead.

In [5]: model.add(Convolution2D(64,(3,3),activation = 'relu'))

In [6]: model.add(MaxPooling2D(pool_size = (2,2)))

In [7]: model.add(Flatten())
```

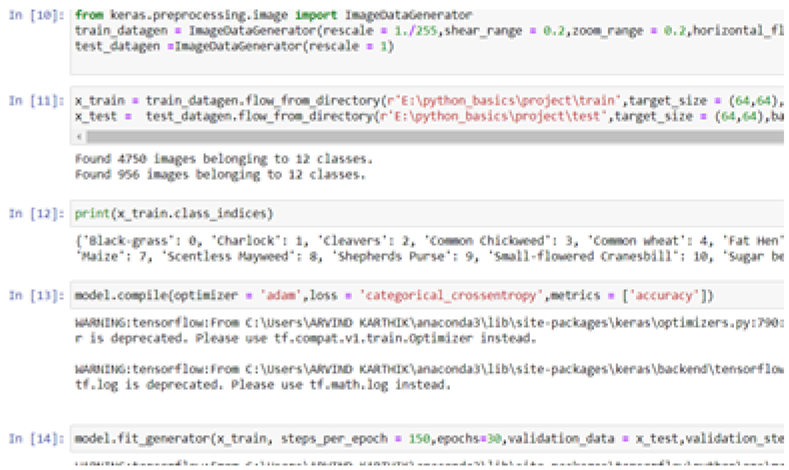
5) adding dense layers



```
[7]: model.add(Flatten())

[8]: model.add(Dense(output_dim = 600 ,init = 'uniform',activation = 'relu'))

     C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\ipykernel_launcher.py
     2 API: `Dense(activation="relu", units=600, kernel_initializer="uniform")
       """Entry point for launching an IPython kernel.

[9]: model.add(Dense(output_dim = 12,activation = 'softmax'))

     C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\ipykernel_launcher.py
     2 API: `Dense(activation="softmax", units=12)`
       """Entry point for launching an IPython kernel.

[10]: from keras.preprocessing.image import ImageDataGenerator
      train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoo
      test_datagen =ImageDataGenerator(rescale = 1)
```

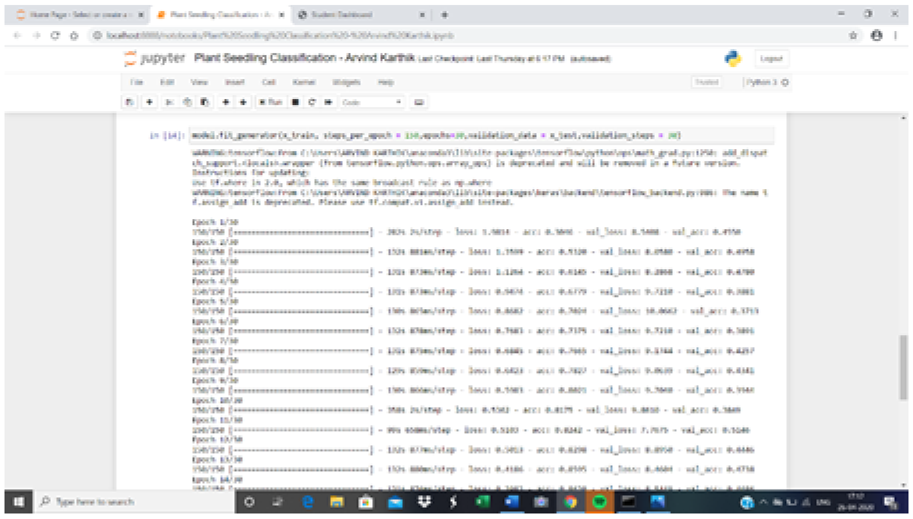
6) configure the learning process

7) train and test the model



```
In [10]: from keras.preprocessing.image import ImageDataGenerator
         train_datagen = ImageDataGenerator(rescale = 1./255,shear_range = 0.2,zoom_range = 0.2,horizontal_fl
         test_datagen =ImageDataGenerator(rescale = 1)

In [11]: x_train = train_datagen.flow_from_directory(r'E:\python_basics\project\train',target_size = (64,64),
         x_test =  test_datagen.flow_from_directory(r'E:\python_basics\project\test',target_size = (64,64),ba

         Found 4750 images belonging to 12 classes.
         Found 956 images belonging to 12 classes.

In [12]: print(x_train.class_indices)

         {'Black-grass': 0, 'Charlock': 1, 'Cleavers': 2, 'Common Chickweed': 3, 'Common wheat': 4, 'Fat Hen'
         'Maize': 7, 'Scentless Mayweed': 8, 'Shepherds Purse': 9, 'Small-flowered Cranesbill': 10, 'Sugar be

In [13]: model.compile(optimizer = 'adam',loss = 'categorical_crossentropy',metrics = ['accuracy'])

         WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\optimizers.py:790:
         r is deprecated. Please use tf.compat.v1.train.Optimizer instead.

         WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\backend\tensorflow
         tf.log is deprecated. Please use tf.math.log instead.

In [14]: model.fit_generator(x_train, steps_per_epoch = 150,epochs=30,validation_data = x_test,validation_ste
```

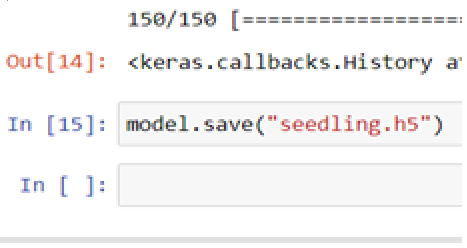
8) optimise the model



jupyter Plant Seedling Classification - Arvind Karthik Last Checkpoint: Last Thursday at 6:17 PM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Type here to search

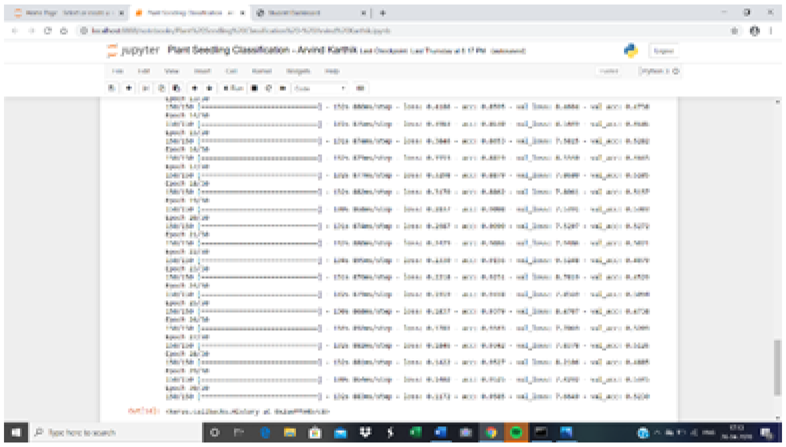
```
In [14]: model.fit_generator(x_train, steps_per_epoch = 150,epochs=30,validation_data = x_test,validation_steps = 30)

         WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\tensorflow\python\ops\math_grad.py:1250: add_dispat
         ch_support.<locals>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.
         Instructions for updating:
         Use tf.where in 2.0, which has the same broadcast rule as np.where
         WARNING:tensorflow:From C:\Users\ARVIND KARTHIK\anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:986: The name t
         f.assign_add is deprecated. Please use tf.compat.v1.assign_add instead.

         Epoch 1/30
         150/150 [==============================] - 302s 2s/step - loss: 1.9814 - acc: 0.3090 - val_loss: 8.5408 - val_acc: 0.4550
         Epoch 2/30
         150/150 [==============================] - 132s 881ms/step - loss: 1.3509 - acc: 0.5120 - val_loss: 8.0580 - val_acc: 0.4958
         Epoch 3/30
         150/150 [==============================] - 131s 873ms/step - loss: 1.1264 - acc: 0.6145 - val_loss: 8.2868 - val_acc: 0.4790
         Epoch 4/30
         150/150 [==============================] - 131s 873ms/step - loss: 0.9474 - acc: 0.6779 - val_loss: 9.7210 - val_acc: 0.3881
         Epoch 5/30
         150/150 [==============================] - 130s 865ms/step - loss: 0.8682 - acc: 0.7024 - val_loss: 10.0662 - val_acc: 0.3713
         Epoch 6/30
         150/150 [==============================] - 132s 878ms/step - loss: 0.7683 - acc: 0.7375 - val_loss: 9.7210 - val_acc: 0.3891
         Epoch 7/30
         150/150 [==============================] - 131s 873ms/step - loss: 0.6885 - acc: 0.7665 - val_loss: 9.3744 - val_acc: 0.4257
         Epoch 8/30
         150/150 [==============================] - 129s 859ms/step - loss: 0.6421 - acc: 0.7827 - val_loss: 9.0639 - val_acc: 0.4381
         Epoch 9/30
         150/150 [==============================] - 130s 866ms/step - loss: 0.5983 - acc: 0.8025 - val_loss: 9.7660 - val_acc: 0.3944
         Epoch 10/30
         150/150 [==============================] - 358s 2s/step - loss: 0.5582 - acc: 0.8175 - val_loss: 9.0858 - val_acc: 0.3660
         Epoch 11/30
         150/150 [==============================] - 99s 658ms/step - loss: 0.5183 - acc: 0.8242 - val_loss: 7.7675 - val_acc: 0.5240
         Epoch 12/30
         150/150 [==============================] - 132s 879ms/step - loss: 0.5013 - acc: 0.8298 - val_loss: 8.8958 - val_acc: 0.4446
         Epoch 13/30
         150/150 [==============================] - 132s 880ms/step - loss: 0.4186 - acc: 0.8595 - val_loss: 8.4684 - val_acc: 0.4758
         Epoch 14/30
```

9) save the model



```
         150/150 [=================

Out[14]: <keras.callbacks.History a

In [15]: model.save("seedling.h5")

In [ ]:
```

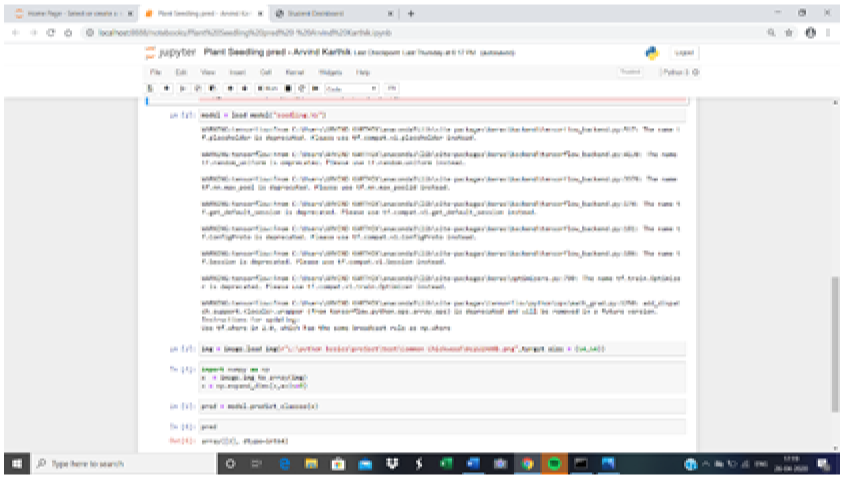
Our model reached an accuracy of 95% at the 30th epoch.



jupyter Plant Seedling Classification - Arvind Karthik

```
[illegible]
```

This is a sample prediction before we went for Application building

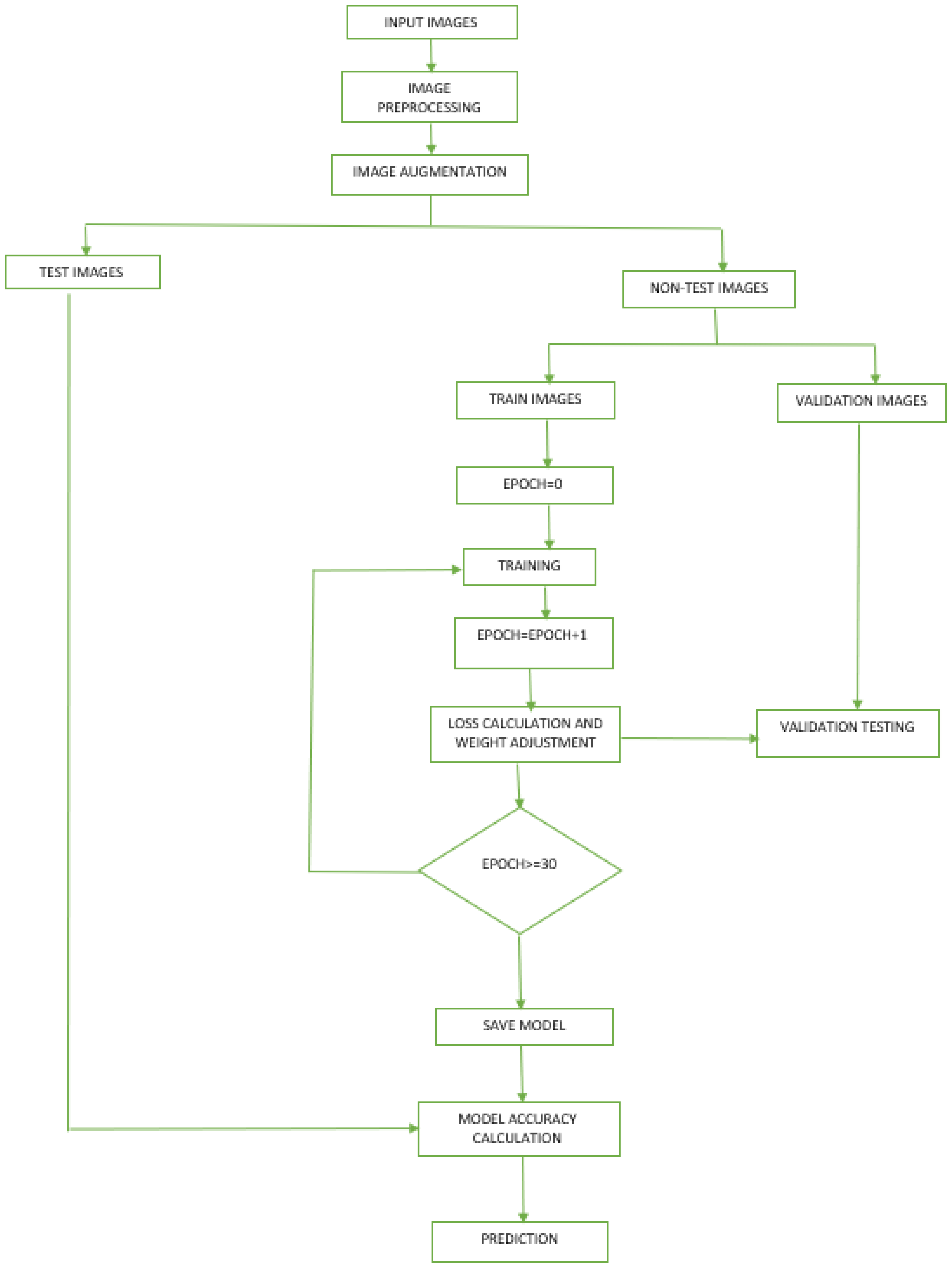


jupyter Plant Seedling pred - Arvind Karthik

```
[illegible]

import numpy as np
x = image.img_to_array(img)
x = np.expand_dims(x,axis=0)

pred = model.predict_classes(x)

pred

array([3], dtype=int64)
```

As seen in the image, our model predicted the Common Chickweed image correctly as 3. The number given to Common Chickweed class was 3. Hence our model predicted 3.

5. FLOWCHART


INPUT IMAGES
IMAGE PREPROCESSING
IMAGE AUGMENTATION
TEST IMAGES
NON-TEST IMAGES
TRAIN IMAGES
VALIDATION IMAGES
EPOCH=0
TRAINING
EPOCH=EPOCH+1
LOSS CALCULATION AND WEIGHT ADJUSTMENT
VALIDATION TESTING
EPOCH>=30
SAVE MODEL
MODEL ACCURACY CALCULATION
PREDICTION

6 . RESULTS

We attempt to use a CNN for this problem. CNNs have been widely used for diverse image classification tasks.
This Seedling classification model has 2 convolution layers. Each is followed with a rectified linear unit (ReLU). The first convolutional layer has 32 filters, the next has 64 . After each convolutional layer, we have a max pooling layer for dimensionality reduction , with a pool size of (2,2). Then we add a flatten later. At the end of the two convolutional layers are 2 fully connected layers called the Dense layers. The first Dense layer has Relu activation function . The last fully connected layer has a softmax activation function , as there are 12 classes, which outputs probability distribution for each of the 12 classes. We use Adam optimizer with a batch size of 32 for each step and a weighted cross-entropy loss, to handle the imbalanced number of pixels for each class.
To fit the model, we use steps_per_epoch as 150, epochs as 30,validation_steps as 30.
Our model reached an accuracy of 95% at the 30th epoch.
Then we saved our model by using- model.save("seedling.h5").

7. ADVANTAGES AND DISADVANTAGES

Advantages

Following are the advantages:

➡Features are automatically deduced and optimally tuned for desired outcome.

Features are not required to be extracted ahead of time. ➡Robustness to natural

variations in the data is automatically learned.

➡The same neural network based approach can be applied to many different applications and data types.

➡Massive parallel computations can be performed using GPUs and are scalable for large volumes of data. Moreover it delivers better performance results when amount of data are huge.

➡The CNN architecture is flexible to be adapted to new problems in the future.

<u>Disadvantages</u>

Following are the disadvantages:

➡It requires very large amount of data in order to perform better than other techniques.

➡It is extremely expensive to train due to complex data models. Moreover deep learning requires expensive GPUs and hundreds of machines. This increases cost to the users.

➡It is difficult to be adopted by less skilled people.

➡It is not easy to comprehend output based on mere learning and requires classifiers to do so. Convolutional neural network based algorithms perform such tasks.

8. APPLICATIONS

Image recognition and classification is the primary field of convolutional neural networks use. It is also the one use case that involves the most progressive frameworks. Our Seedling classification model uses CNN algorithm to analyze the input image and predict the class of the image.

The purpose of the CNN image classification is the following:

- Deconstruct an image and identify its distinct feature. For that, the system uses supervised machine learning classification algorithm.
- Reduce the description of its essential credentials
- The following fields are using this process:
- Image tagging algorithms are the most basic type of image classification. The image tag is a word or a word combination that describes the images and makes it easier to find.
- Visual Search – this technique involves matching an input image with the available database. Besides, the visual search analyzes the image and looks for images with similar credentials. For example, our seedling classification model analyzes the input image and predicts the class of the image.
- Recommender engines is another field to apply image classification .

9. CONCLUSION:

People working in Agricultural sectors have a hard time to sort and recognize different plants and weeds as it is a very time consuming process and includes lots of effort in the long term. This is where Artificial Intelligence can aid in benefitting those workers by reducing the amount of time and effort to do so. Our Convolution Neural network model can help them in plant seedlings classification. Our dataset consists plants belonging to 12 different species at several growth stages. Our model can differentiate between a weed and a plants. Our Model achieves 95% accuracy in doing so.

10. FUTURE SCOPE:

Of course, future enhancements can be made to make it more convenient and effective to use. One thing we can do is by increasing the models accuracy by increasing the no. of epochs and decreasing the batch size, also number of hidden layers can be increased.

Apart from the changes in model, we can extend our work by using robotic arms for separation in actual farmland. This robotic arm will be assisted by a camera attached to it on the top to capture crop images and feed them directly to our Algorithm. Secondly, a better and more informative dataset can be chosen, the size of the dataset can be increased and testing the model using images with multiple plants in a scene can help us perform much better.

We believe that with promising results and mere perfect accuracy, we will be able to automate the entire process of weed control in farm and agricultural industry and thereby, reducing the cost and labor inculcated in it while improving crop yield and productivity.

11. BIBILOGRAPHY:

[1]. Plant Seedling Classification Dataset -
https://www.kaggle.com/c/plant-seedlings-classification/data

[2]. Research Paper –

https://www.researchgate.net/publication/329087077_Deep_Convolutional_Neural_Network_for_Plant_Seedlings_Classification

[3]. Model Preparation Guide – Smart bridge documents

https://thesmartbridge.com/documents/spsaimldocs/CNNprep.pdf

APPENDIX:

A. **SOURCE CODE:**

CNN MODEL BUILDING:

```
from keras.models import Sequential

from keras.layers import Dense

from keras.layers import Convolution2D

from keras.layers import MaxPooling2D

from keras.layers import Flatten


model=Sequential()


model.add(Convolution2D(32,(3,3),
input_shape=(64,64,3),activation='relu'))


model.add(MaxPooling2D(2,2))


model.add(Flatten())


model.add(Dense(units=600,
init='uniform',activation='relu'))


model.add(Dense(units=12, init='uniform',
activation='softmax'))


from keras.preprocessing.image import
ImageDataGenerator

train_datagen=ImageDataGenerator(rescale=1./255,
shear_range=0.2, zoom_range=0.2,
horizontal_flip=True,vertical_flip=True)

test_datagen=ImageDataGenerator(rescale=1./255)


x_train=train_datagen.flow_from_directory(r'C:\Users\
KIRAN
LAKHANI\Desktop\project1\train',target_size=(64,64),
batch_size=32,class_mode='categorical')


x_test=test_datagen.flow_from_directory(r'C:\Users\KI
RAN LAKHANI\Desktop\project1\test',


target_size=(64,64),batch_size=32,class_mode='categor
ical')


print(x_train.class_indices)


model.compile(loss='categorical_crossentropy',
optimizer='adam',metrics=['accuracy'])


model.fit_generator(x_train,steps_per_epoch=150,
epochs=30, validation_data=x_test,validation_steps=63)


model.save("seedling.h5")
```

CNN MODEL PREDICTION:

```
from keras.models import load_model

from keras.preprocessing import image

import numpy as np

import cv2


model=load_model('seedling.h5')


img = image.load_img(r'C:\Users\KIRAN
LAKHANI\Desktop\black-grass.png',
target_size=(64,64))

x = image.img_to_array(img)

x = np.expand_dims(x,axis=0)


x.shape


pred = model.predict_classes(x)
```

```
    pred
```

FLASK MODEL:

```
from __future__ import division, print_function

# coding=utf-8

import sys

import os

import glob

import numpy as np

from keras.preprocessing import image



from keras.applications.imagenet_utils import
preprocess_input, decode_predictions


from keras.models import load_model

from keras import backend

from tensorflow.keras import backend


import tensorflow as tf
```

```
global graph

graph=tf.get_default_graph()


#global graph

#graph = tf.get_default_graph()




from skimage.transform import resize


# Flask utils

from flask import Flask, redirect, url_for, request,
render_template

from werkzeug.utils import secure_filename

from gevent.pywsgi import WSGIServer


# Define a flask app

app = Flask(__name__)


# Model saved with Keras model.save()

MODEL_PATH = 'models/seedling.h5'


# Load your trained model
```

```
model = load_model(MODEL_PATH)

        # Necessary

# print('Model loaded. Start serving...')


# You can also use pretrained model from Keras

# Check https://keras.io/applications/

#from keras.applications.resnet50 import ResNet50

#model = ResNet50(weights='imagenet')

#model.save('')

print('Model loaded. Check http://127.0.0.1:5000/')






@app.route('/', methods=['GET'])

def index():

    # Main page

    return render_template('index.html')




@app.route('/predict', methods=['GET', 'POST'])

def upload():
    if request.method == 'POST':

        # Get the file from post request

        f = request.files['file']



        # Save the file to ./uploads

        basepath = os.path.dirname(__file__)

        file_path = os.path.join(

            basepath, 'uploads', secure_filename(f.filename))

        f.save(file_path)

        img = image.load_img(file_path, target_size=(64, 64))

        x = image.img_to_array(img)

        x = np.expand_dims(x, axis=0)



        with graph.as_default():

            preds = model.predict_classes(x)

        index = ['Black-grass','Charlock','Cleavers','Common
Chickweed','Common wheat','Fat Hen','Loose
Silky-bent','Maize','Scentless Mayweed','Shepherds
Purse','Small-flowered Cranesbill','Sugar beet']

        text = "prediction : "+index[preds[0]]



               # ImageNet Decode
        return text



if __name__ == '__main__':
    app.run(debug=False,threaded = False)
```

base.html:

```
<html lang="en">


<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible"
content="ie=edge">
    <title>Artificial Intelligence </title>
    <link
href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.
min.css" rel="stylesheet">
    <script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.
```

```
min.js"></script>

    <script
src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></
script>

    <script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.mi
n.js"></script>

    <link href="{{ url_for('static', filename='css/main.css')
}}" rel="stylesheet">

            <style>

            .bg-dark {

                        background-color: #42678c!important;

            }

            #result {

                        color: #0a1c4ed1;

            }

            </style>

</head>


<body>

    <nav class="navbar navbar-dark bg-dark">

        <div class="container">

            <a class="navbar-brand" href="#">Team 14</a>

        </div>
```

```
  </nav>

  <div class="container">

    <div id="content" style="margin-top:2em">

                <div class="container">

                  <div class="row">

                            <div class="col-sm-6 bd" >

                              <h3>Seedling Classification : </h3>

                              <br>

         <div class="col-sm-60 bd" >

                              <p>

                              In recent times, the state of
agriculture and the amount of work people need to put in to
check if plants/food is growing correctly is phenomenal,
because it is 2019 and workers still need to organize and
recognize the difference between different plants and
weeds.

                              People who are working in the
agriculture field still have to have the ability to sort and
recognize different plants and weeds, which does take a lot
of time and a lot of effort in the long term.This is where
Artificial Intelligence can actually help benefit those
workers, as the time and energy to identify plant seedlings
will be much shortened. The ability to do so effectively can
mean better crop yields and in the long term will result in
better care for the environment. As by identifying the
difference among the plants and weeds in a timely manner
where it is highly accurate can positively impact
agriculture.
```

```
                                   </p>

                                   <br>

                                        <img
src="https://encrypted-tbn0.gstatic.com/images?q=tbn%3A
ANd9GcTSp-XwjjfvbjHfcECQvkdhtVOZJfBn7ZaFTNXP1
jy0PBnvzu2n&usqp=CAU.PNG"
style="height:350px"class="img-rounded" alt="Gesture">

                                  </div>




                                  <div class="container">

    <div id="content" style="margin-top:2em">{% block
content %}{% endblock %}</div>

  </div>


                                  <div class="loader"
style="display:none;"></div>



                            </div>

                                  </div>



                             </div>

                            </div>

                            </div>
    </div>

    </div>

</body>


<footer>

    <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>

</footer>


</html>
```

Index.html:

```
{% extends "base.html" %} {% block content %}


<h2>Seedling classification</h2>


<div>

    <form id="upload-file" method="post"
enctype="multipart/form-data">

        <label for="imageUpload" class="upload-label">

            Choose...

        </label>

        <input type="file" name="file" id="imageUpload"
```

```
accept=".png, .jpg, .jpeg">

    </form>



    <div class="image-section" style="display:none;">

        <div class="img-preview">

            <div id="imagePreview">

            </div>

        </div>

        <div>

            <button type="button" class="btn btn-primary btn-lg
" id="btn-predict">Predict!</button>

        </div>

    </div>

    <div class="loader" style="display:none;"></div>



    <h3 id="result">

        <span> </span>

    </h3>

</div>



{% endblock %}
```

CSS FILE:

```
.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

.img-preview>div {
    width: 100%;
    height: 100%;
    background-size: 256px 256px;
    background-repeat: no-repeat;
    background-position: center;
}

input[type="file"] {
    display: none;
}
```

```
.upload-label{
    display: inline-block;
    padding: 12px 30px;
    background: #39D2B4;
    color: #fff;
    font-size: 1em;
    transition: all .4s;
    cursor: pointer;
}

.upload-label:hover{
    background: #34495E;
    color: #39D2B4;
}

.loader {
    border: 8px solid #f3f3f3; /* Light grey */
    border-top: 8px solid #3498db; /* Blue */
    border-radius: 50%;
    width: 50px;
    height: 50px;
    animation: spin 1s linear infinite;
}
body {
  background-color: lightblue;
}
@keyframes spin {
    0% { transform: rotate(0deg); }
    100% { transform: rotate(360deg); }
}
```

JS FILE:

```
$(document).ready(function () {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();

    // Upload Preview
    function readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function (e) {
                $('#imagePreview').css('background-image', 'url('
+ e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            }
            reader.readAsDataURL(input.files[0]);
        }
    }
    $("#imageUpload").change(function () {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function () {
        var form_data = new FormData($('#upload-file')[0]);

        // Show loading animation
        $(this).hide();
        $('.loader').show();
        // Make prediction by calling api /predict

        $.ajax({

            type: 'POST',

            url: '/predict',

            data: form_data,

            contentType: false,

            cache: false,

            processData: false,

            async: true,

            success: function (data) {

                // Get and display the result

                $('.loader').hide();

                $('#result').fadeIn(600);

                $('#result').text(' Result:  ' + data);

                console.log('Success!');

            },

        });

    });


});
```