Plant Seedling Classification:

CNN, PreTrained ResNet34, VGG and Ensemble Approach

NAME: ARPIT GARG

ID: A1784072

MASTER OF DATA SCIENCE

THE UNIVERSITY OF ADELAIDE

INTRODUCTION

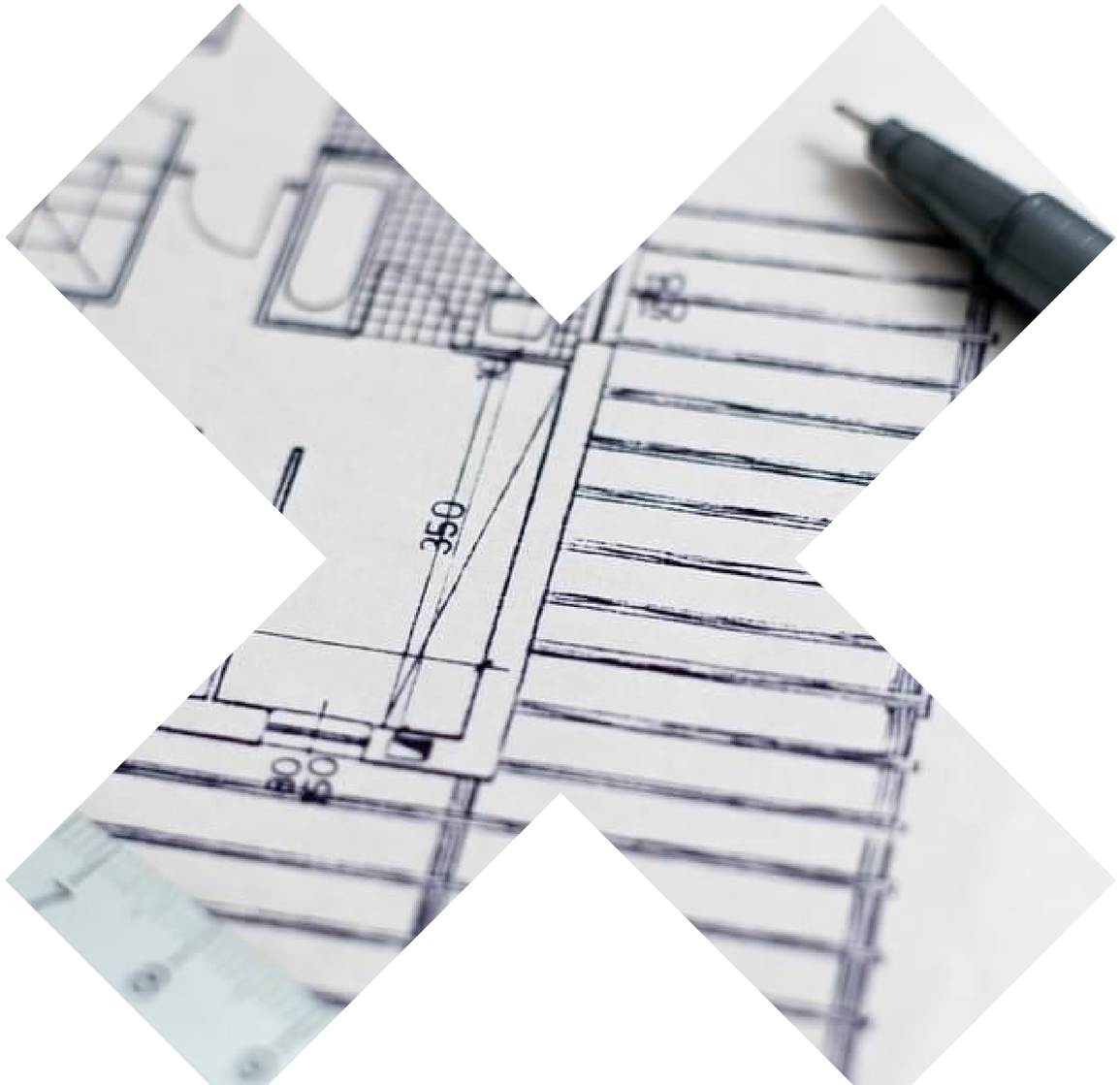
DATA LOADER

Exploratory Data Analysis (EDA)

DATA PRE-PROCESSING AND MODELLING

EVALUATION

INTRODUCTION



Aims to solve Kaggle Problem: Classify between seeds and weeds

Different categories of seedling and their images

Aarhus University Department of Engineering Signal Processing Group for hosting the original data.

Why this is important?

Global Biofertilizer Market Trends, Forecast and Competitive Analysis to 2024: Promising Opportunities in Seed Treatment and Soil Treatment Applications

Research and Markets
21 September 2020 · 4-min read

Dublin, Sept. 21, 2020 (GLOBE NEWSWIRE) -- The "Biofertilizer Market Report: Trends, Forecast and Competitive Analysis" report has been added to **ResearchAndMarkets.com's** offering.

The global biofertilizer market is expected to grow with a CAGR of 14% from 2019 to 2024

TRENDING

1. Sen. Lindsey Graham says Black people are free to 'go anywhere' in South Carolina so long as they're conservative
2. William Barr discovers that he is not immune to Trump's wrath
3. Former NJ Gov. Chris Christie released from hospital a week after testing positive for COVID-19

- Reduce labour cost
- Increase Efficiency

INTRODUCTION

DATA LOADER

Exploratory Data Analysis (EDA)

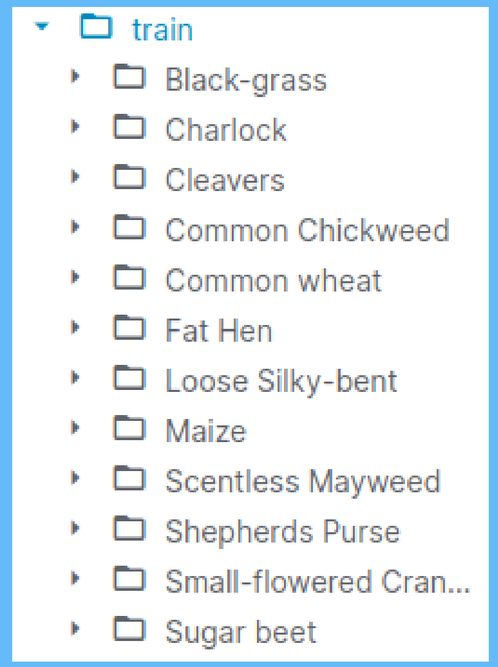
DATA PRE-PROCESSING AND MODELLING

EVALUATION

Data Loaders:

Train dataset:

```
In [5]: #transforming and loading the dataset
        transform = transforms.Compose(
            [transforms.Resize([224,224])
            ])
        train_data = datasets.ImageFolder(train_dir, transform=transform)
```

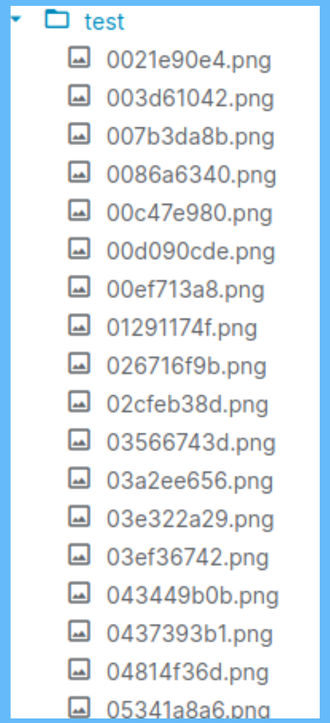



```
{0: 'Black-grass',
 1: 'Charlock',
 2: 'Cleavers',
 3: 'Common Chickweed',
 4: 'Common wheat',
 5: 'Fat Hen',
 6: 'Loose Silky-bent',
 7: 'Maize',
 8: 'Scentless Mayweed',
 9: 'Shepherds Purse',
 10: 'Small-flowered Cranesbill',
 11: 'Sugar beet'}
```

Test dataset:

```
#for Test Dataset
class SeedlingsTestDataset(Dataset):
    def __init__(self, data_path, transform=None):
        self.data_path = data_path
        self.transform = transform

        self.images = [os.path.join(self.data_path, dI) for dI in os.listdir(self.data_path)]

    def __len__(self):
        return len(self.images)

    def __getitem__(self, index):
        img_path = self.images[index]
        img = Image.open(img_path)

        if self.transform is not None:
            img = self.transform(img)

        return img
```

```
#test transform and load
test_transforms = transforms.Compose([
        transforms.Resize([224, 224]),
        transforms.ToTensor(),
        transforms.ToPILImage()
])

test_dataset = SeedlingsTestDataset(
    test_dir,
    transform=test_transforms)
```



INTRODUCTION

DATA LOADER

Exploratory Data Analysis (EDA)

DATA PRE-PROCESSING AND MODELLING

EVALUATION

Exploratory Data Analysis (EDA)

01 Print some dataset details

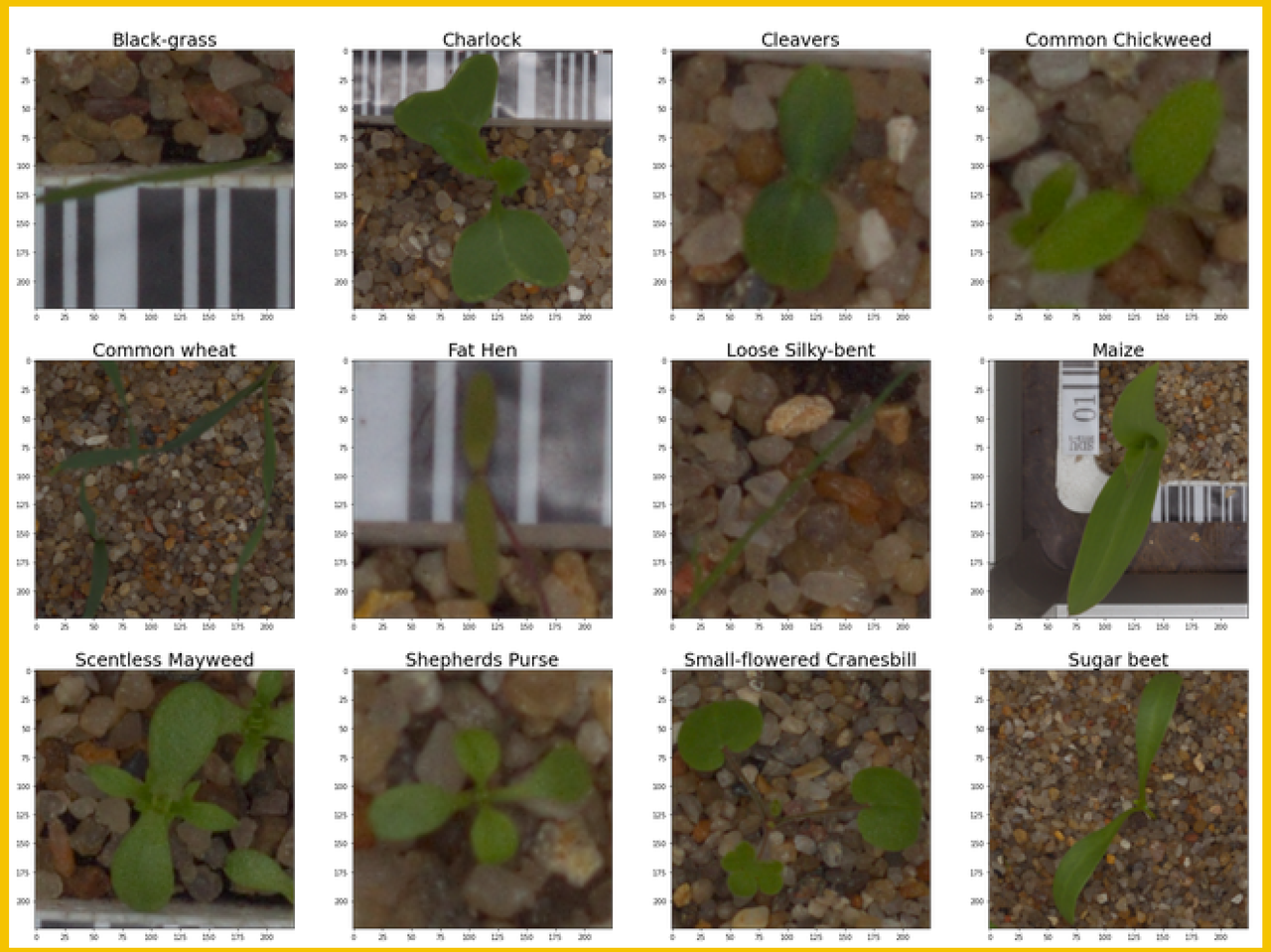
02 **Different Plots**

03 **Categories with different IDs**

Exploratory Data Analysis (EDA)

01

Print some dataset details




```
#length of the training data
data_length
```

```
4750
```

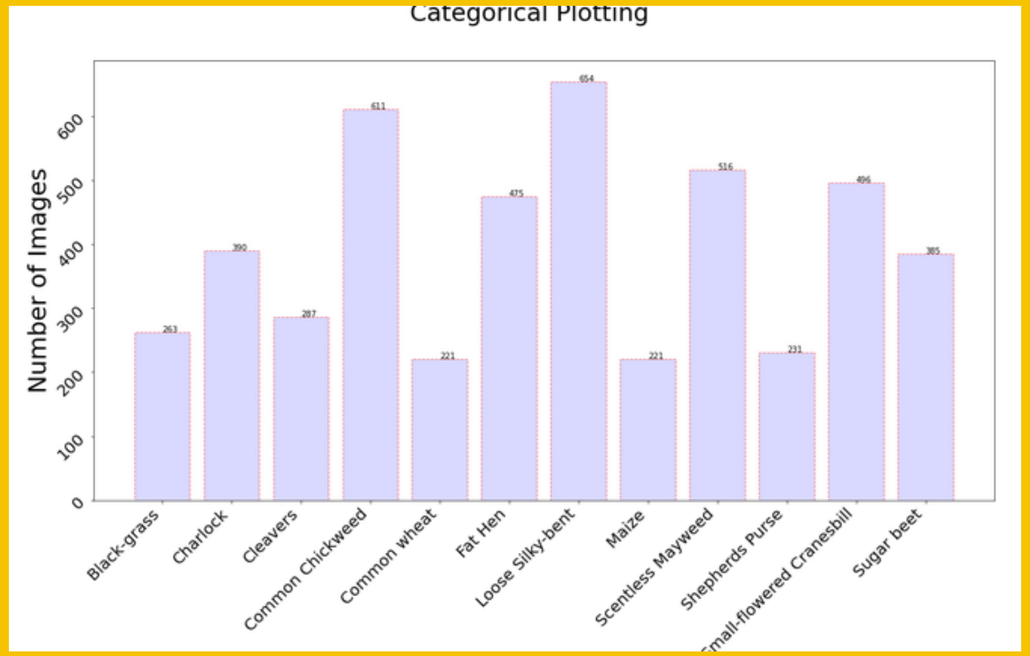
```
#shape of the image
image_shape
```

```
(224, 224, 3)
```

Exploratory Data Analysis (EDA)

02

Different Plots



02

Categories with different IDs

```
{0: 'Black-grass',
 1: 'Charlock',
 2: 'Cleavers',
 3: 'Common Chickweed',
 4: 'Common wheat',
 5: 'Fat Hen',
 6: 'Loose Silky-bent',
 7: 'Maize',
 8: 'Scentless Mayweed',
 9: 'Shepherds Purse',
 10: 'Small-flowered Cranesbill',
 11: 'Sugar beet'}
```

INTRODUCTION

DATA LOADER

Exploratory Data Analysis (EDA)

DATA PRE-PROCESSING AND MODELLING

Type something

EVALUATION

DATA PRE-PROCESSING

DATA AND LABELS

```
#Making enpty labels and data
data_shape = list(image_shape)
data_shape.insert(0, data_length)
data = np.zeros(data_shape, dtype=np.uint8)
labels = np.zeros([data_length], dtype=np.int64)
```

```
#dividing labels and data
i = 0
for image, label in tqdm(train_data, desc="Reading Images"):
    data[i] = np.array(image)
    labels[i] = label
    i += 1
```

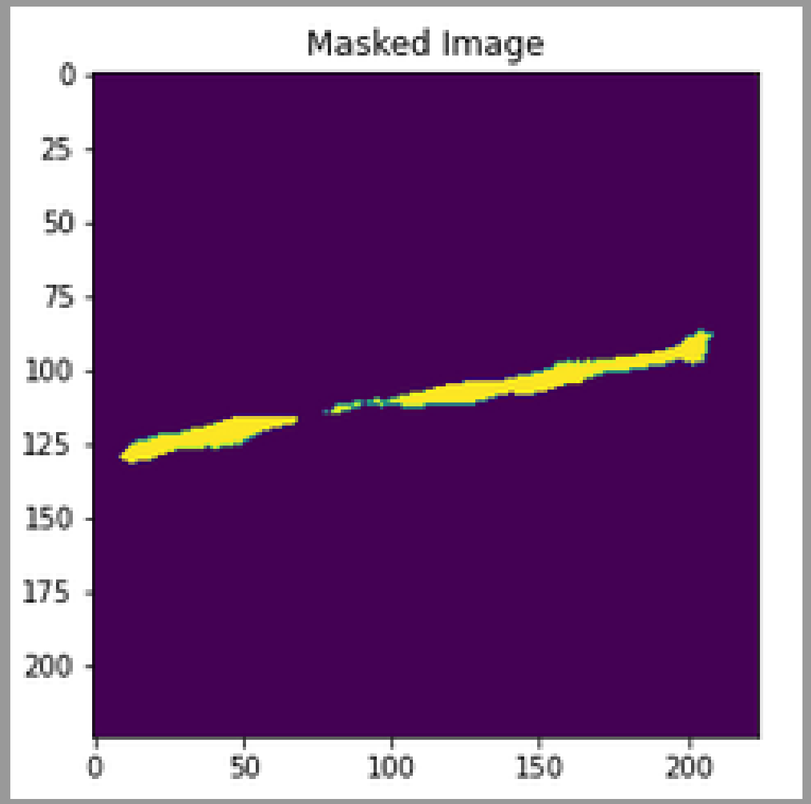
```
#Creatig dictionary where train_dict contains data ,labels and shape, train_infor contains ids and image paths
train_dict = {"data": data, "labels": labels, 'data_shape': image_shape}
train_info = {"label_names": idx_to_class, "file_paths": image_paths}
```

DATA PRE-PROCESSING

DIGITAL IMAGE PROCESSING

1. MASKING

```python
def create_mask_for_plant(image):
    image_hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    sensitivity = 35
    lower_hsv = np.array([60 - sensitivity, 100, 50])
    upper_hsv = np.array([60 + sensitivity, 255, 255])
    mask = cv2.inRange(image_hsv, lower_hsv, upper_hsv)
    kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (11,11))
    mask = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
    return mask
```

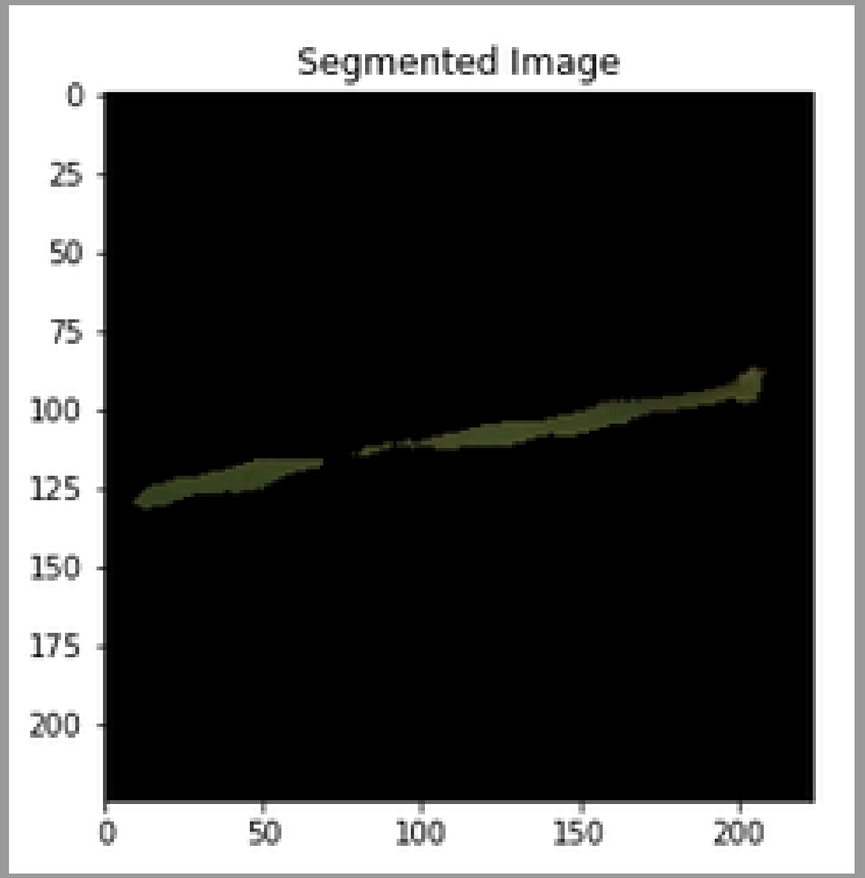


DATA PRE-PROCESSING

DIGITAL IMAGE PROCESSING

2. SEGMENTATION

```
#segmentation
def segment_plant(image):
    mask = create_mask_for_plant(image)
    output = cv2.bitwise_and(image, image, mask = mask)
    return output
```

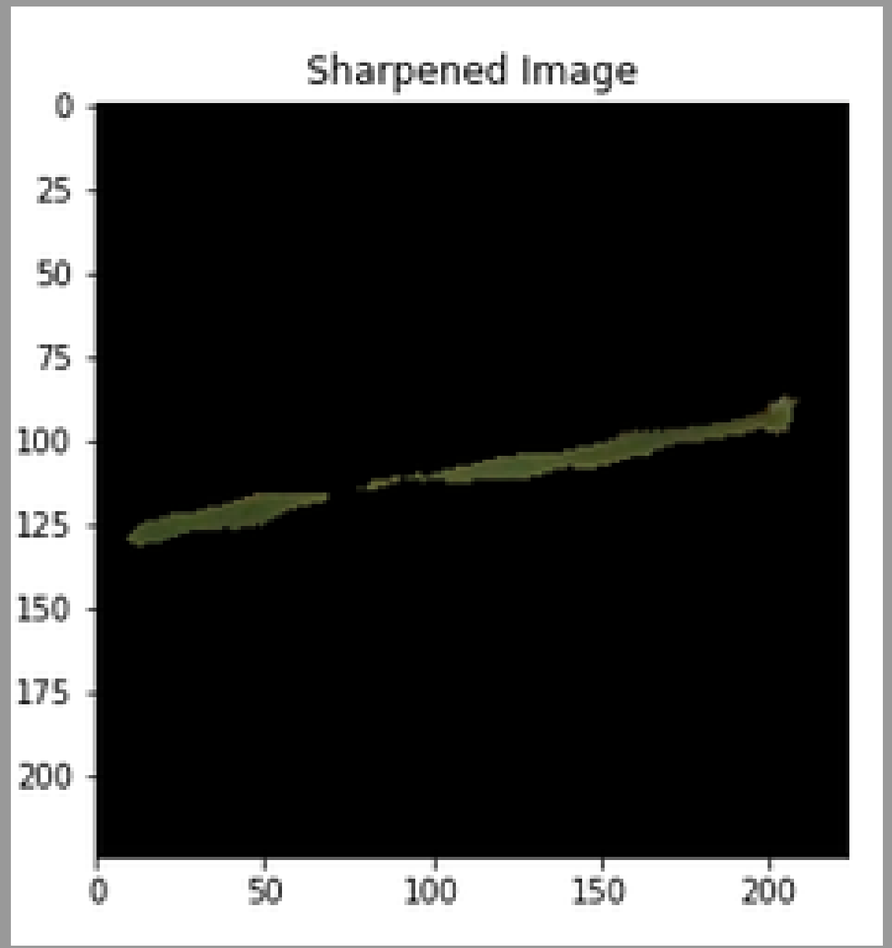


DATA PRE-PROCESSING

DIGITAL IMAGE PROCESSING

3. SHARPENING

```python
#sharpening
def sharpen_image(image):
    image_blurred = cv2.GaussianBlur(image, (0, 0), 3)
    image_sharp = cv2.addWeighted(image, 1.5, image_blurred, -0.5, 0)
    return image_sharp
```



DATA PRE-PROCESSING

CONVERTING IMAGES TO TENSOR

```
image_dataset = ImageDataset(train_dict["data"], train_dict["labels"])
#Converting images to Tensor
print(image_dataset[0][0].shape)
print(image_dataset[4610])
```

```
#Image Dataset class returns item and length
class ImageDataset(torch.utils.data.Dataset):
    def __init__(self, x_data, y_data):
        self.x_data = x_data
        self.y_data = y_data
        self.to_tensor = torchvision.transforms.ToTensor()
    def __getitem__(self, index):
        return self.to_tensor(self.x_data[index]), self.y_data[index]
    def __len__(self):
        return len(self.x_data)
```

```
torch.Size([3, 224, 224])
(tensor([[[0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         ...,
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.]],

        [[0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         ...,
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.]],

        [[0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         ...,
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.],
         [0., 0., 0.,  ..., 0., 0., 0.]]]), 11)
```

DATA PRE-PROCESSING

DIVIDING VALIDATION AND TRAINING

```
#Dividing the dataset to training set and validation set
def array_random_pick(array, pick_num):
    index = np.arange(len(array))
    pick = np.random.choice(len(array), pick_num, replace=False)
    unpick = np.equal(np.in1d(index, pick), False)
    return array[unpick], array[pick]
#10% to validation and remaining to training data
train_mask, valid_mask = array_random_pick(np.arange(len(image_dataset)), 475)
train_set = torch.utils.data.Subset(image_dataset, train_mask)
valid_set = torch.utils.data.Subset(image_dataset, valid_mask)
```

```
#length of training data
len(train_loader)
```

```
4275
```

```
#length of validation data
len(valid_loader)
```

```
475
```

DATA MODELLING

- CONVOLUTIONAL NEURAL NETWORK
- RESNET 34 (PYTORCH PRE-TRAINED WEIGHTS)
- VGG-16
- ENSEMBLE APPROACH

DATA MODELLING

CONVOLUTIONAL NEURAL NETWORK

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 6, 220, 220]             456
       BatchNorm2d-2         [-1, 6, 220, 220]              12
         MaxPool2d-3         [-1, 6, 110, 110]               0
           Dropout-4         [-1, 6, 110, 110]               0
            Conv2d-5        [-1, 16, 106, 106]           2,416
       BatchNorm2d-6        [-1, 16, 106, 106]              32
         MaxPool2d-7          [-1, 16, 53, 53]               0
           Dropout-8          [-1, 16, 53, 53]               0
            Conv2d-9          [-1, 32, 49, 49]          12,832
           Linear-10                [-1, 512]      39,338,496
             ReLU-11                [-1, 512]               0
           Linear-12                [-1, 100]          51,300
             ReLU-13                [-1, 100]               0
           Linear-14                 [-1, 12]           1,212
================================================================
Total params: 39,406,756
Trainable params: 39,406,756
Non-trainable params: 0
```

submission_cnn.csv a few seconds ago by Arpit Garg add submission details	0.96347	0.96347

DATA MODELLING

RESNET34

```
      BatchNorm2d-110           [-1, 512, 7, 7]           1,024
             ReLU-111           [-1, 512, 7, 7]               0
          Conv2d-112            [-1, 512, 7, 7]       2,359,296
     BatchNorm2d-113            [-1, 512, 7, 7]           1,024
             ReLU-114           [-1, 512, 7, 7]               0
       BasicBlock-115           [-1, 512, 7, 7]               0
          Conv2d-116            [-1, 512, 7, 7]       2,359,296
     BatchNorm2d-117            [-1, 512, 7, 7]           1,024
             ReLU-118           [-1, 512, 7, 7]               0
          Conv2d-119            [-1, 512, 7, 7]       2,359,296
     BatchNorm2d-120            [-1, 512, 7, 7]           1,024
             ReLU-121           [-1, 512, 7, 7]               0
       BasicBlock-122           [-1, 512, 7, 7]               0
AdaptiveAvgPool2d-123           [-1, 512, 1, 1]               0
          Linear-124                 [-1, 12]           6,156
          ResNet-125                 [-1, 12]               0
================================================================
Total params: 21,290,828
Trainable params: 21,059,340
Non-trainable params: 231,488
```

submission_resnet.csv 3 minutes ago by Arpit Garg	0.97732	0.97732

DATA MODELLING

VGG-16

submission.csv 3 minutes ago by Arpit Garg	0.96725	0.96725

ENSEMBLE

submission_ensemble.csv a minute ago by Arpit Garg	0.69269	0.69269

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 224, 224]           1,792
              ReLU-2         [-1, 64, 224, 224]               0
            Conv2d-3         [-1, 64, 224, 224]          36,928
              ReLU-4         [-1, 64, 224, 224]               0
         MaxPool2d-5         [-1, 64, 112, 112]               0
            Conv2d-6        [-1, 128, 112, 112]          73,856
              ReLU-7        [-1, 128, 112, 112]               0
            Conv2d-8        [-1, 128, 112, 112]         147,584
              ReLU-9        [-1, 128, 112, 112]               0
        MaxPool2d-10          [-1, 128, 56, 56]               0
           Conv2d-11          [-1, 256, 56, 56]         295,168
             ReLU-12          [-1, 256, 56, 56]               0
           Conv2d-13          [-1, 256, 56, 56]         590,080
             ReLU-14          [-1, 256, 56, 56]               0
           Conv2d-15          [-1, 256, 56, 56]         590,080
             ReLU-16          [-1, 256, 56, 56]               0
        MaxPool2d-17          [-1, 256, 28, 28]               0
           Conv2d-18          [-1, 512, 28, 28]       1,180,160
             ReLU-19          [-1, 512, 28, 28]               0
           Conv2d-20          [-1, 512, 28, 28]       2,359,808
             ReLU-21          [-1, 512, 28, 28]               0
           Conv2d-22          [-1, 512, 28, 28]       2,359,808
             ReLU-23          [-1, 512, 28, 28]               0
        MaxPool2d-24          [-1, 512, 14, 14]               0
           Conv2d-25          [-1, 512, 14, 14]       2,359,808
             ReLU-26          [-1, 512, 14, 14]               0
           Conv2d-27          [-1, 512, 14, 14]       2,359,808
             ReLU-28          [-1, 512, 14, 14]               0
           Conv2d-29          [-1, 512, 14, 14]       2,359,808
             ReLU-30          [-1, 512, 14, 14]               0
        MaxPool2d-31            [-1, 512, 7, 7]               0
AdaptiveAvgPool2d-32            [-1, 512, 7, 7]               0
           Linear-33                 [-1, 4096]     102,764,544
             ReLU-34                 [-1, 4096]               0
          Dropout-35                 [-1, 4096]               0
           Linear-36                 [-1, 4096]      16,781,312
             ReLU-37                 [-1, 4096]               0
          Dropout-38                 [-1, 4096]               0
           Linear-39                 [-1, 1000]       4,097,000
================================================================
Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0
```

INTRODUCTION

DATA LOADER

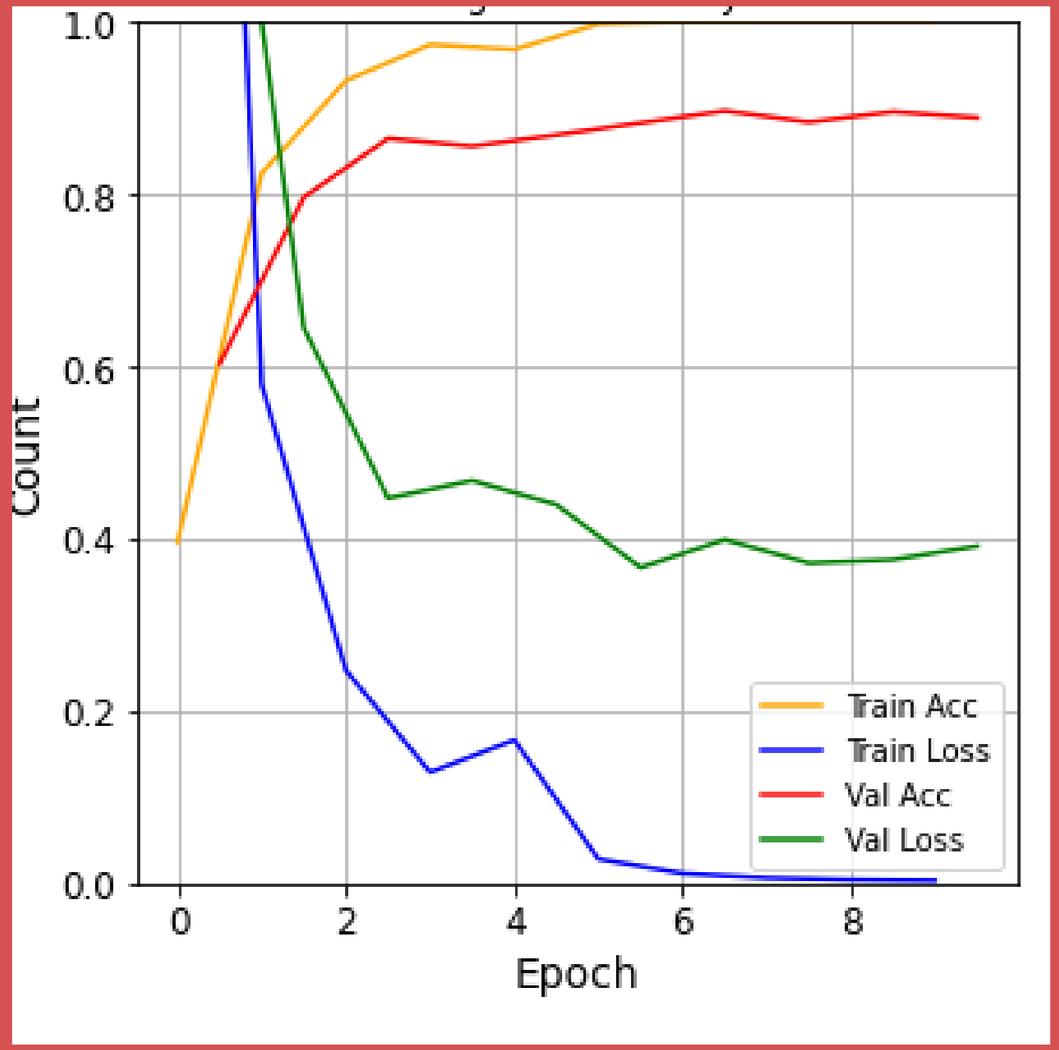
Exploratory Data Analysis (EDA)

DATA PRE-PROCESSING AND MODELLING

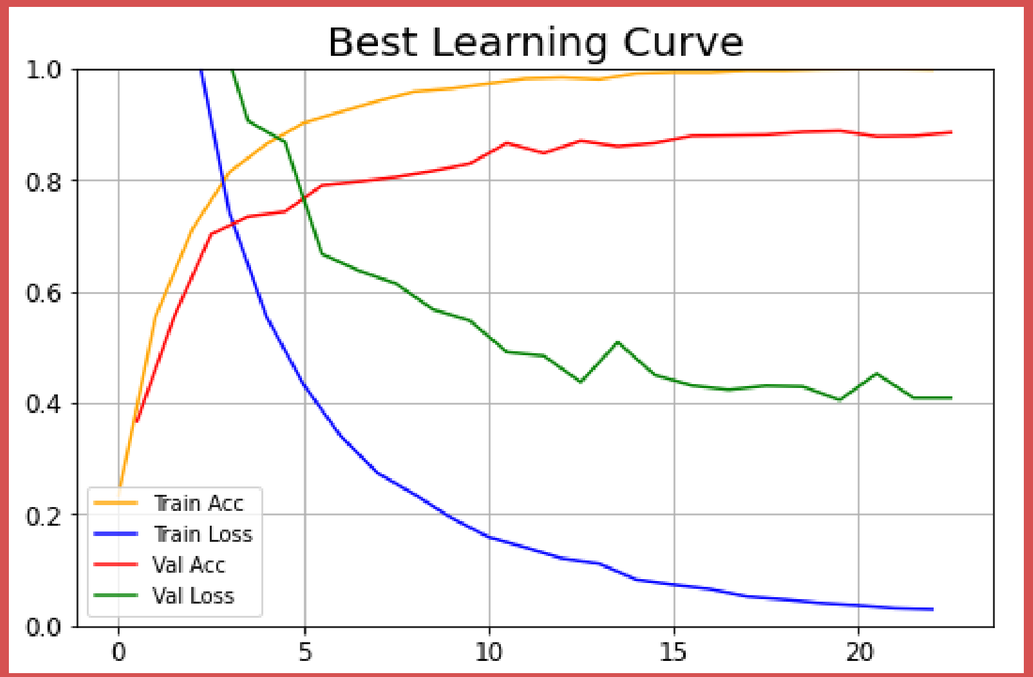
EVALUATION

EVALUATION

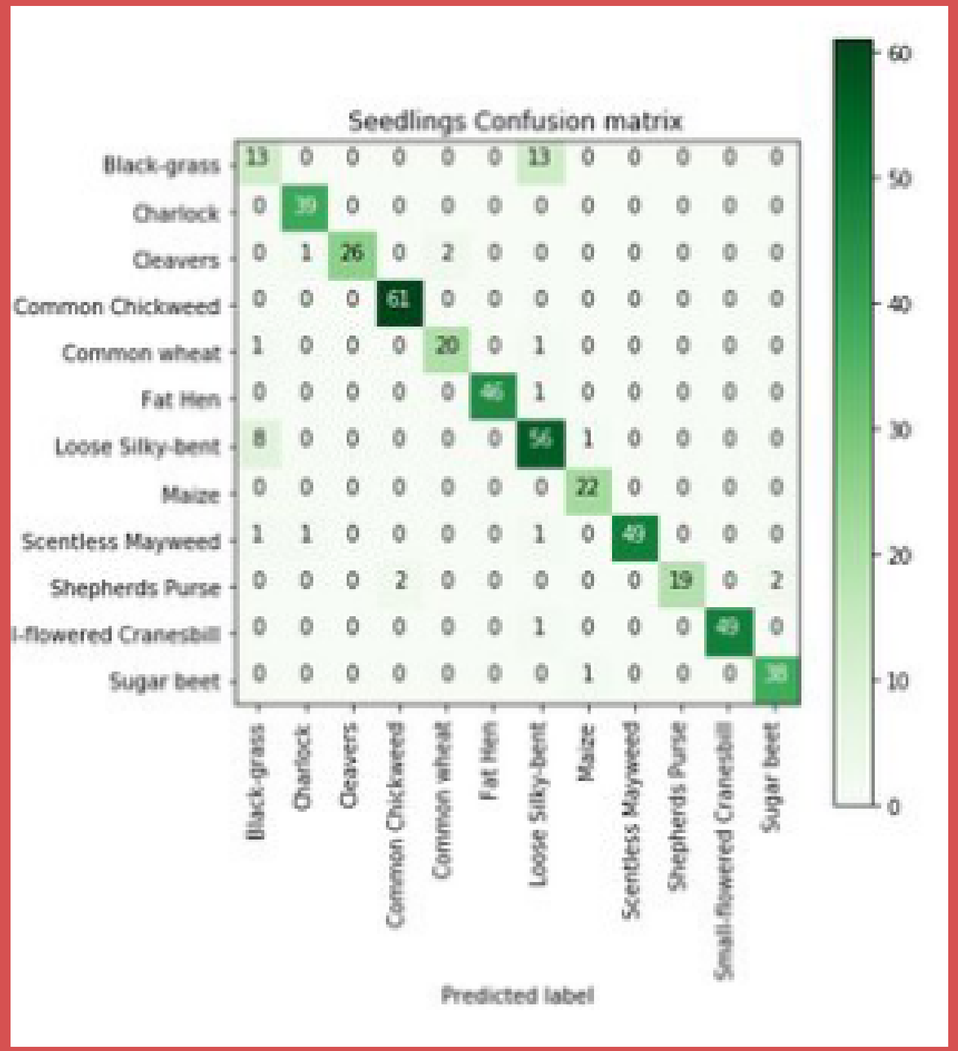
CNN LEARNING CURVE

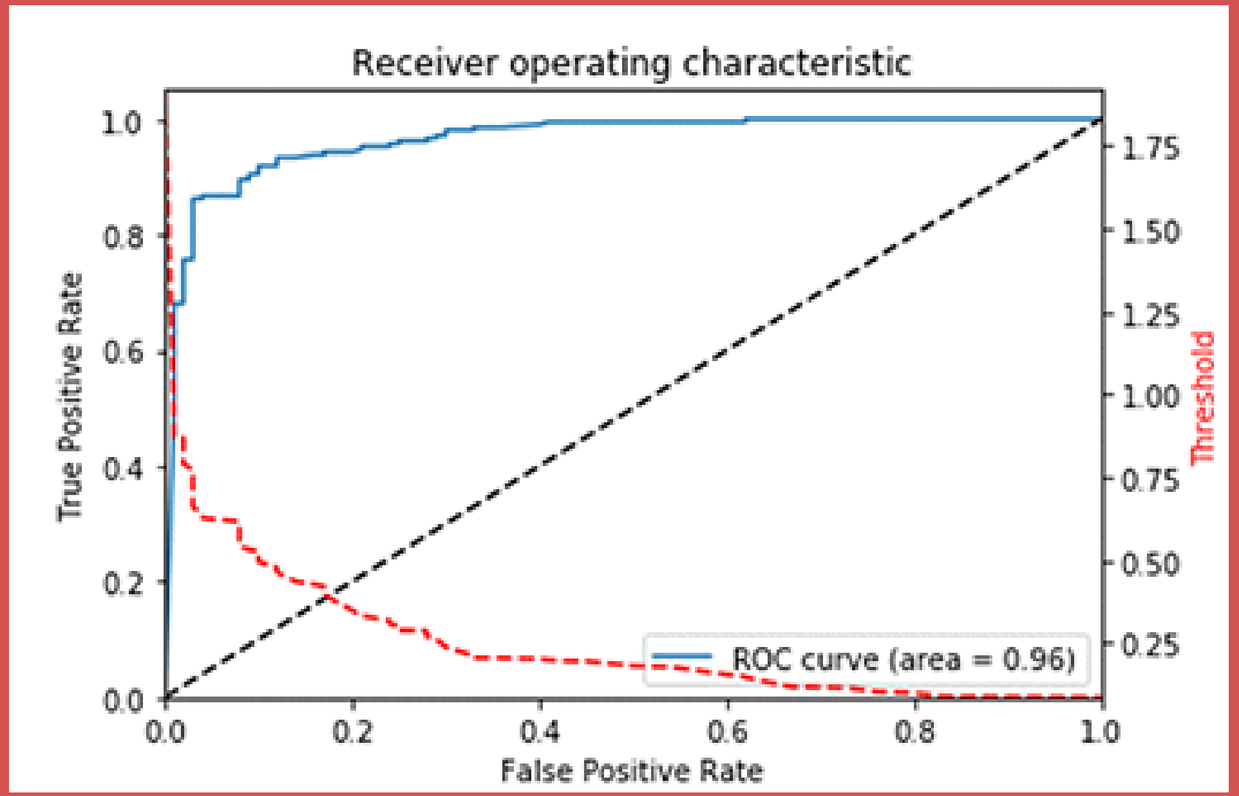


RESNET34 LEARNING CURVE



EVALUATION


Seedlings Confusion matrix
Black-grass
Charlock
Cleavers
Common Chickweed
Common wheat
Fat Hen
Loose Silky-bent
Maize
Scentless Mayweed
Shepherds Purse
I-flowered Cranesbill
Sugar beet
Predicted label


Receiver operating characteristic
True Positive Rate
False Positive Rate
Threshold
ROC curve (area = 0.96)

EVALUATION

Table 1: Evaluation Metrics

S.No.	Model	F Score
1	Convolutional Neural Network	0.96347
2	VGG-16	0.96725
3	ResNet34	0.97732
4	Ensemble	0.69269

submission_cnn.csv a few seconds ago by Arpit Garg add submission details	0.96347	0.96347
submission.csv 3 minutes ago by Arpit Garg	0.96725	0.96725
submission_resnet.csv 3 minutes ago by Arpit Garg	0.97732	0.97732
submission_ensemble.csv a minute ago by Arpit Garg	0.69269	0.69269

DANKE!
THANK YOU!
MERCI!
GRAZIE!
GRACIAS!
DANK JE WEL!

. . . . . .