Leafydex

Plant Leaf Classification & Disease Identification

We beleaf in your disease

Eshwaran Venkat & Tigran Poladian
Berkeley MIDS 207 Final Project
Section 11 - Uri Schonfeld

INTRODUCTION

Introduction to Dataset

- 4494 images … 2273 healthy leaves and 2221 diseased leaves
- 12 plant species
  - Mango, Arjun, Alstonia Scholaris, Guava, Bael, Jamun, Jatropha, Pongamia Pinnata, Basil, Pomegranate, Lemon, and Chinar
- Raw sample resolution 6k x 4k RGB pixel resolution
- The image paths have the following structure
  - Plants_2_compressed/{data_type}/{species_name} {disease_status} ({code})/{filename}.JPG'
- The dataset comes decomposed into training, testing, validation, and prediction images, therefore our EDA and model building will not further segregate data into training and other sets

https://www.kaggle.com/datasets/csafrit2/plant-leaves-for-image-classification

Project Modules Completed

- EDA & Data Engineering
- Image compression notebook
- Baseline model … sequential single layer neural net classifier
- Convolution model
  - Trained on single species data
  - Combined all species and trained on large dataset
- Generalization evaluation against real world lemon leaves
- Basic API server

EDA & Data Engineering

Make dataset more manageable

EDA - Training Images Labeled

Species name: Alstonia Scholaris, Diseased: True

Species name: Alstonia Scholaris, Diseased: False

Species name: Jamun, Diseased: True

Species name: Jamun, Diseased: False

Species name: Arjun, Diseased: True

Species name: Arjun, Diseased: False

Species name: Jatropha, Diseased: True

Species name: Jatropha, Diseased: False

Species name: Bael, Diseased: True

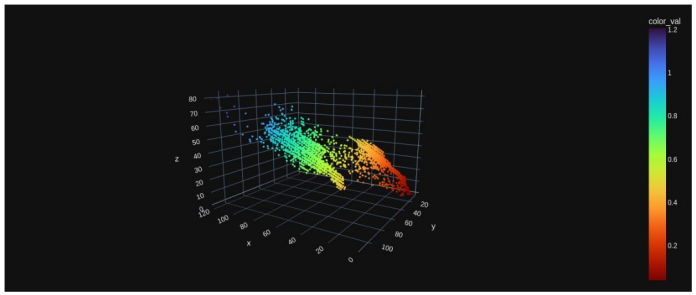
Species name: Basil, Diseased: False

- Consistent background, leaves prominent appearance
- No visible difference between raw resolution and smaller

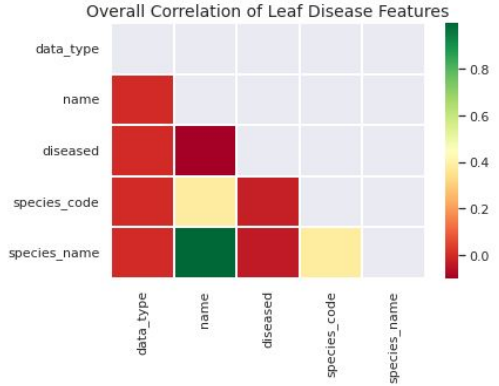
In much of the dataset the difference is very subtle

EDA - Distribution

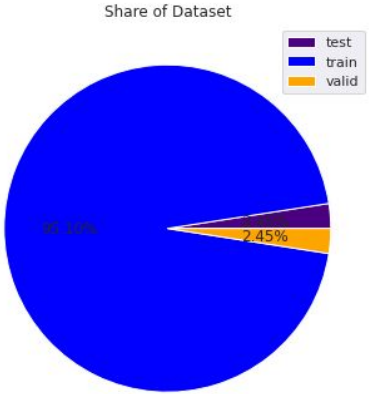
	count	unique	freq
data_type	4494	3	4274
name	4494	22	345
file	4494	4494	1
path	4494	4494	1
diseased	4494	2	2273
species_code	4494	22	345
species_name	4494	12	623

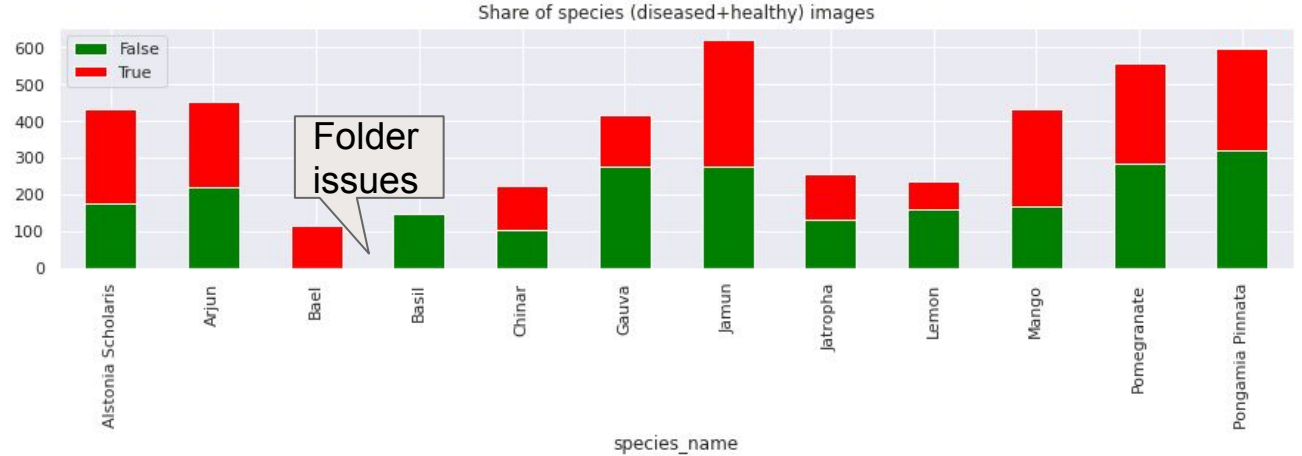


RGB spread



Expected correlation





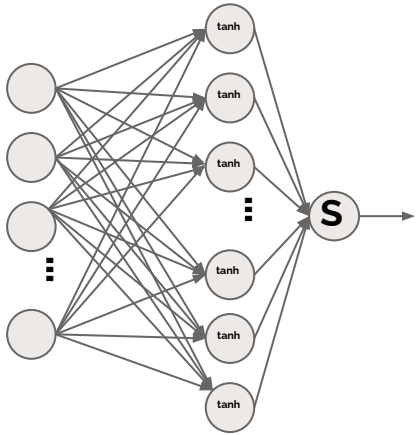
EDA - Conclusions

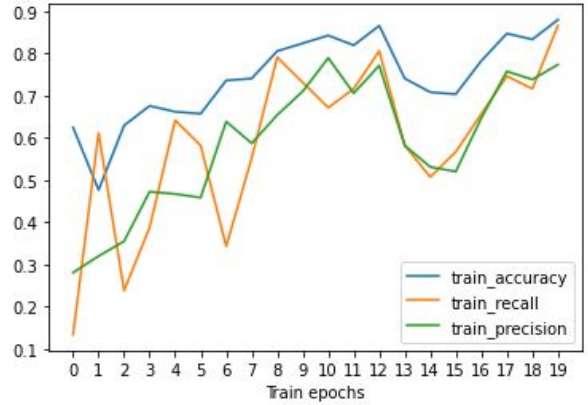
- Compression to 256x256 did not cause visible differences
- Training data not equally distributed across species
- Test and validation data pre-selected, likely not enough numbers
- Fixed kaggle dataset folder mis-spellings
- Training images are distinct from background – likely not great generalization for real world application unless pre-filtering is done

MODELS

Baseline Model - NN

- Single hidden layer binary classifier net
- Accuracy, recall, precision metrics
- Consistent 0.6 accuracy across 20 epochs
- Adam and binary cross-entropy

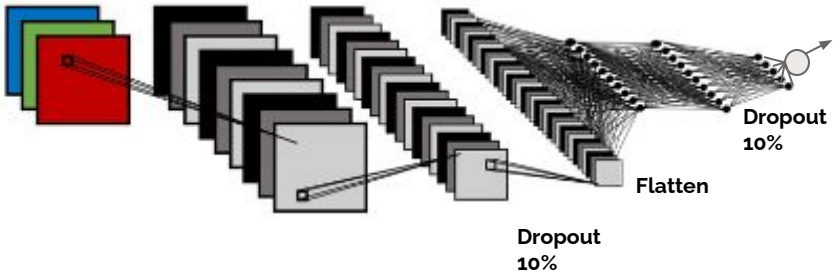





```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 786432)            0

 Output (Dense)              (None, 1)                 786433

=================================================================
Total params: 786,433
Trainable params: 786,433
Non-trainable params: 0
_________________________________________________________________
Test Accuracy: 0.6000
```

Better Model - CNN

Inputs :	Stage 1 :	Stage 2 :	Stage 3 :	Stage 4 :
RGB pictures	ConvLayer	ConvLayer	ConvLayer	fully-connected
3x256x256	128x128x32	32x32x64	8x8x126	

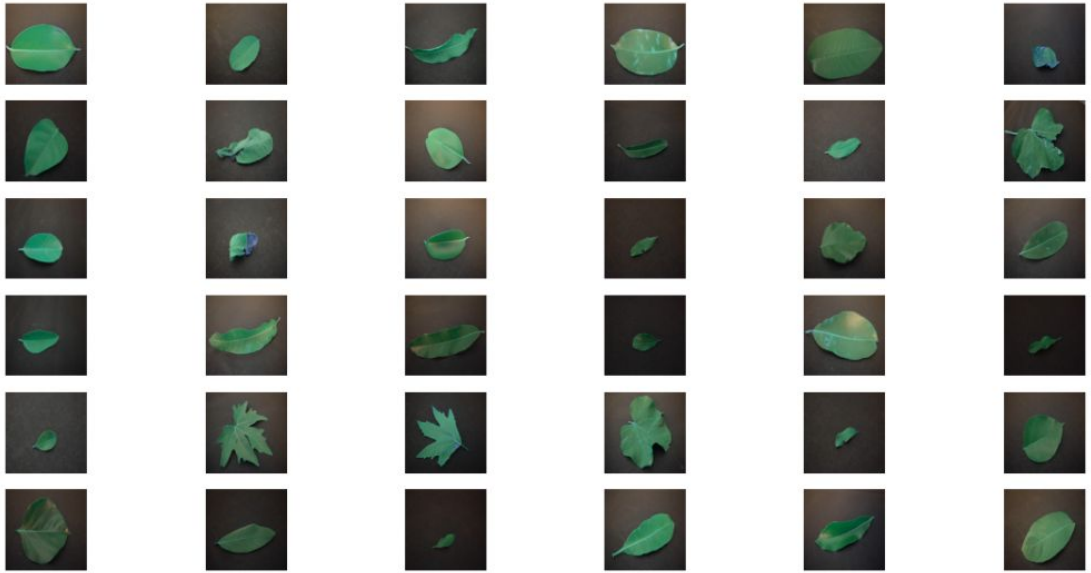


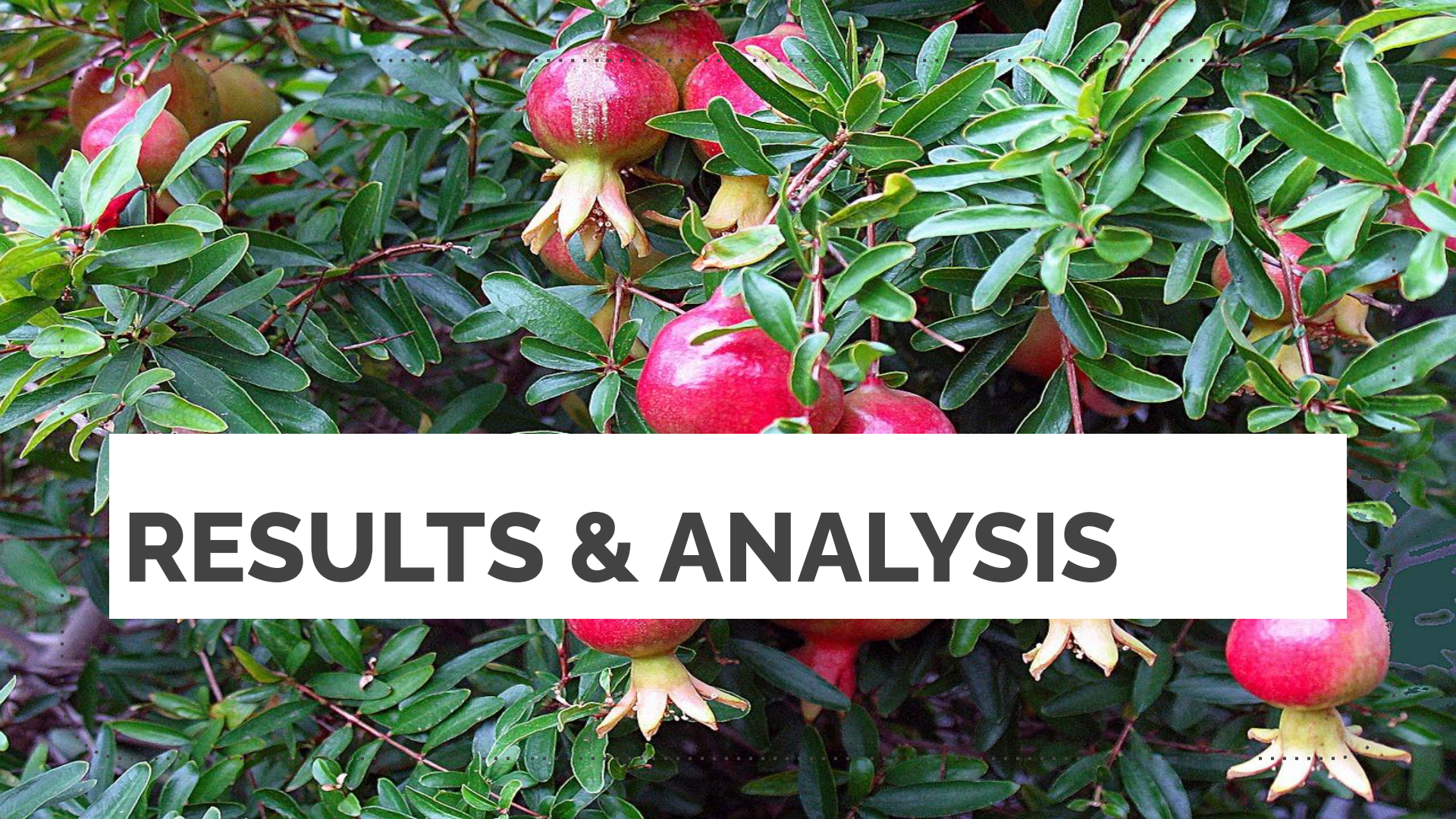
- Two dropout layers help minimize overfitting the training data
- Pixel values / 256
- Adam optimizer
- Learning rate = 0.001
- Loss function: binary cross-entropy

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv_1 (Conv2D)              (None, 128, 128, 32)      2432

pool_1 (MaxPooling2D)        (None, 64, 64, 32)        0

conv_2 (Conv2D)              (None, 32, 32, 64)        51264

pool_2 (MaxPooling2D)        (None, 16, 16, 64)        0

dropout (Dropout)            (None, 16, 16, 64)        0

conv_3 (Conv2D)              (None, 8, 8, 128)         204928

pool_3 (MaxPooling2D)        (None, 4, 4, 128)         0

flatten (Flatten)            (None, 2048)              0

fc_1 (Dense)                 (None, 1024)              2098176

dropout_1 (Dropout)          (None, 1024)              0

fc_2 (Dense)                 (None, 1)                 1025

=================================================================
Total params: 2,357,825
Trainable params: 2,357,825
Non-trainable params: 0
_________________________________________________________________
```

Better Model - CNN

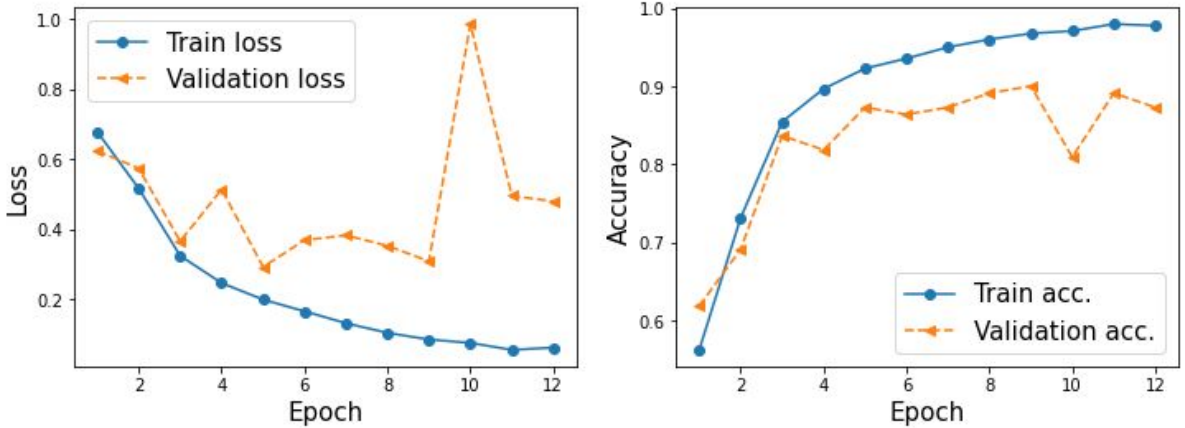
- Data Augmentation to adjust brightness, contrast, flip L-R and U-D
- Initially noticed some images with light saturation were not performing well. Adjusted contrast and added augmented training data which improved performance
- Attempted multiple
- combination of
- hyperparameters
- and the following
- summary set seems
- to work consistent
- with better results





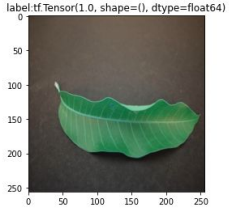
RESULTS & ANALYSIS

Results

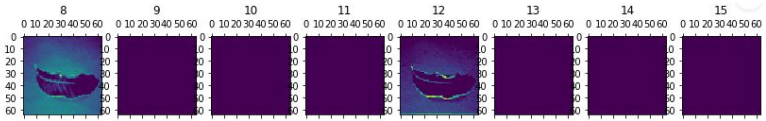


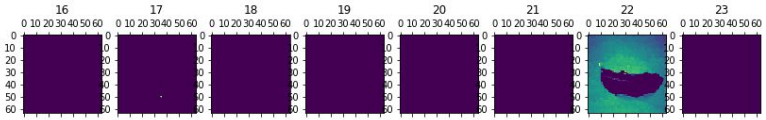
- Achieved 88.12% accuracy top result
- Varying hyperparameters resulted in a range 80 - 88%

Layers



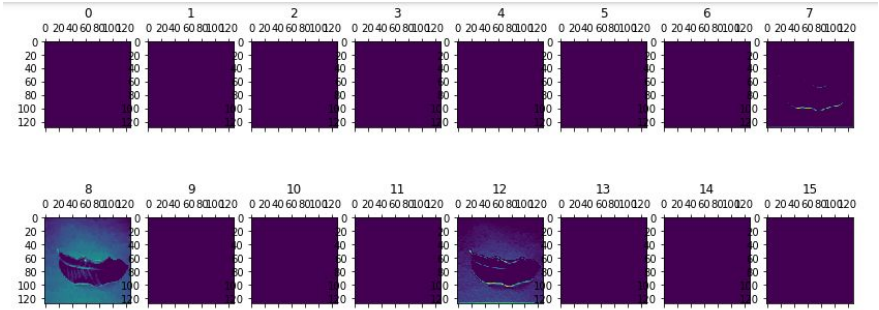
Activations of layer 1 : conv_1
Images size is 128 x 128
Number of channels is 32
Printing channels:



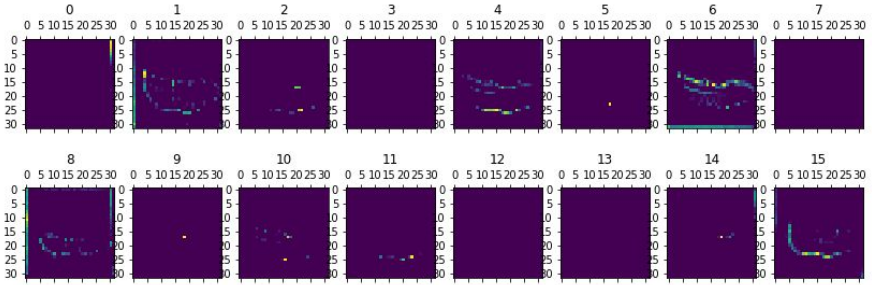


Activations of layer 2 : pool_1
Images size is 64 x 64
Number of channels is 32
Printing channels:

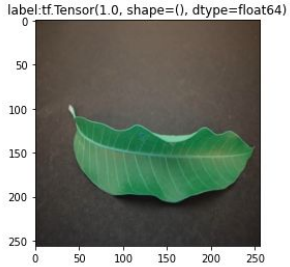
Activations of layer 3 : conv_2
Images size is 32 x 32
Number of channels is 64
Printing channels:



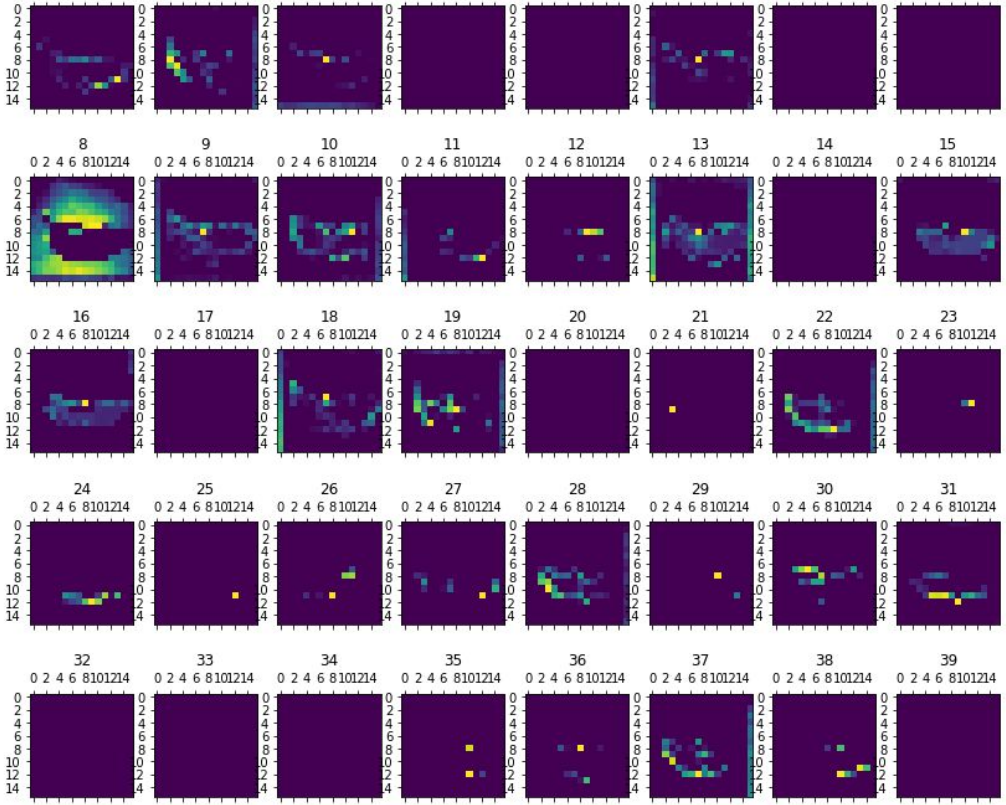
Initial convolutional filtering layers dominated by solid background



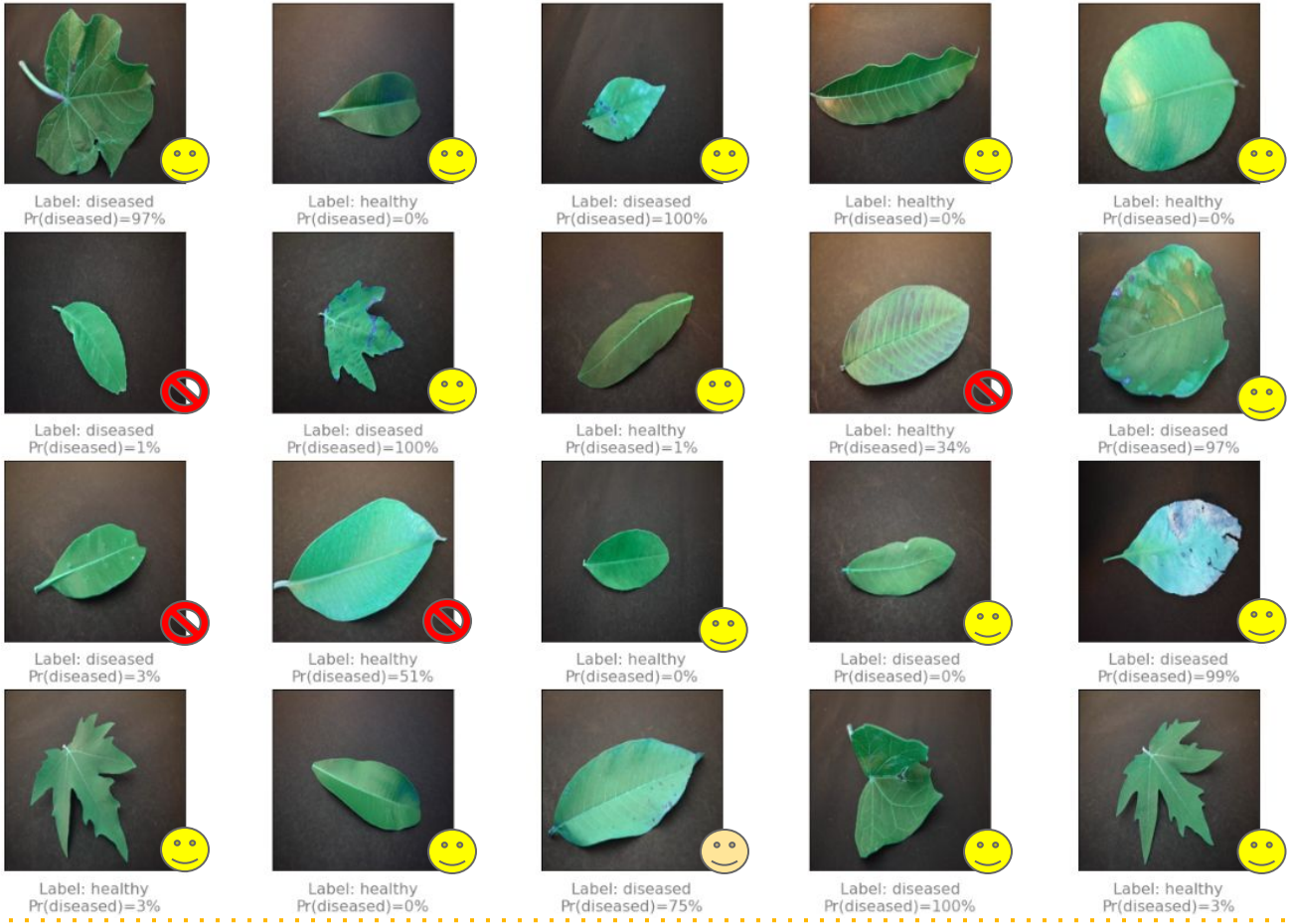
Layers



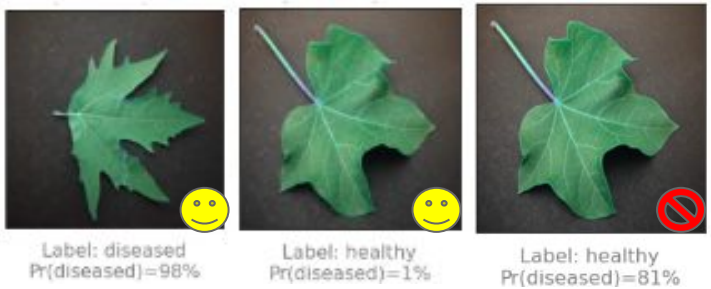
Activations of layer 4 : pool_2
Images size is 16 x 16
Number of channels is 64
Printing channels:



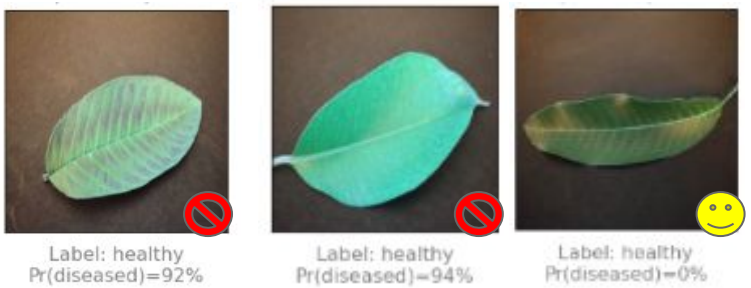
Test Set Results



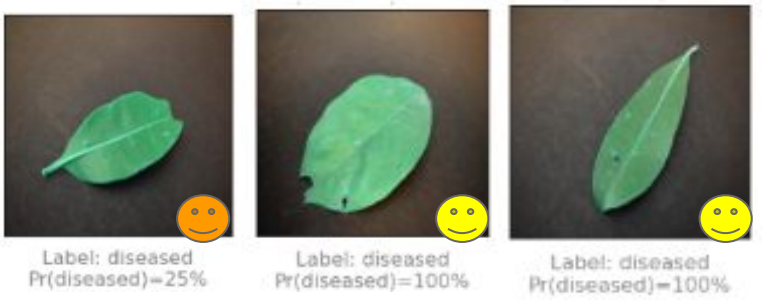
Test Results - Prominent Hits & Misses



First 2 images are correct. Multiple runs with different hyperparameters and suspected random function issues (seed not fixed) with starting weights produces different results for second image



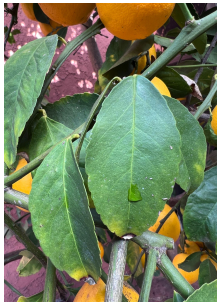
Likely color stratification and light saturation confuses first 2 images. Third image correct though given discoloration patches

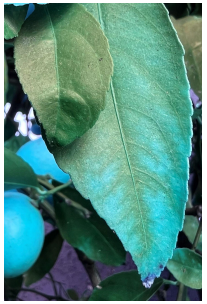


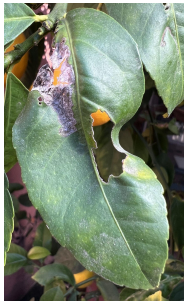
Correctly identifies class even though disease characteristics are subtle

Real world data - lemon tree

- Noisy backgrounds
- varying environmental conditions - rain drops, day, evening




Real world data - lemon tree

50% Prediction rate. Results not unexpected. Lemon training data was less diverse. Individual model performance isolated just on each species performed less well on lemon (70 - 80% range) than other species

NEXT STEPS

For Better Results

- Additional data engineering
- Ensemble approaches
- Run Monte Carlo type simulation to train for optimal hyperparameters
- Combine binary classification with species classification and feed to treatment and care recommendation engine

Operationalize - Basic Server Completed

- Create web API
- Basic API endpoints use trained model to evaluate results
  - \summary
    - returns model summary
  - \evaluate
    - takes inputs name {file_name} and label {0,1}
    - Returns accuracy against label

```
192.168.0.126:8081/evaluate?name=IMG_8026.jpg&label=0
```

Challenges

- Working with and storing high resolution images
- Problems with kaggle data
- Random seed in python packages may not be so random
- Analyzing and understanding various ML function signatures, protocols, and data structures
  - Constant recasting from DataFrame to numpy arrays, resizing matrices, and unit testing to discover expected data types

Thanks!

Any questions?

Leaf them here