Identification & Classification of Flooded Building from Satellite Images

Springboard – Data Science Career Track
Capstone Project 3

Biniam Asmerom
April 2022

Courtesy of www.houstonpublicmedia.org

Business Problem

- After a hurricane, damage assessment is critical to emergency managers for efficient response and resource allocation
- One way to gauge the damage extent is to quantify the number of flooded/damaged buildings, which is traditionally done by ground survey.
  - This process is labor-intensive, time-consuming as well as some areas could be inaccessible due to flooding.

- Emergency managers and responders (such as FEMA and Insurance companies) would greatly benefit from an automated model that would enable them to better plan for and allocate necessary resources.

Project Objective

- Build a robust, accurate and quick image identification & classification model of satellite images

Data

- Input data : 2 classes
  - Damage and no-damage
- Input data grouped into:
  - ***Training data:*** 5000 images of each class
  - ***Validation:*** 1000 images of each class
  - ***Test:*** 1000 images of each class
  - ***Test-another:*** unbalanced data with 8000/1000 images of damaged/undamaged classes

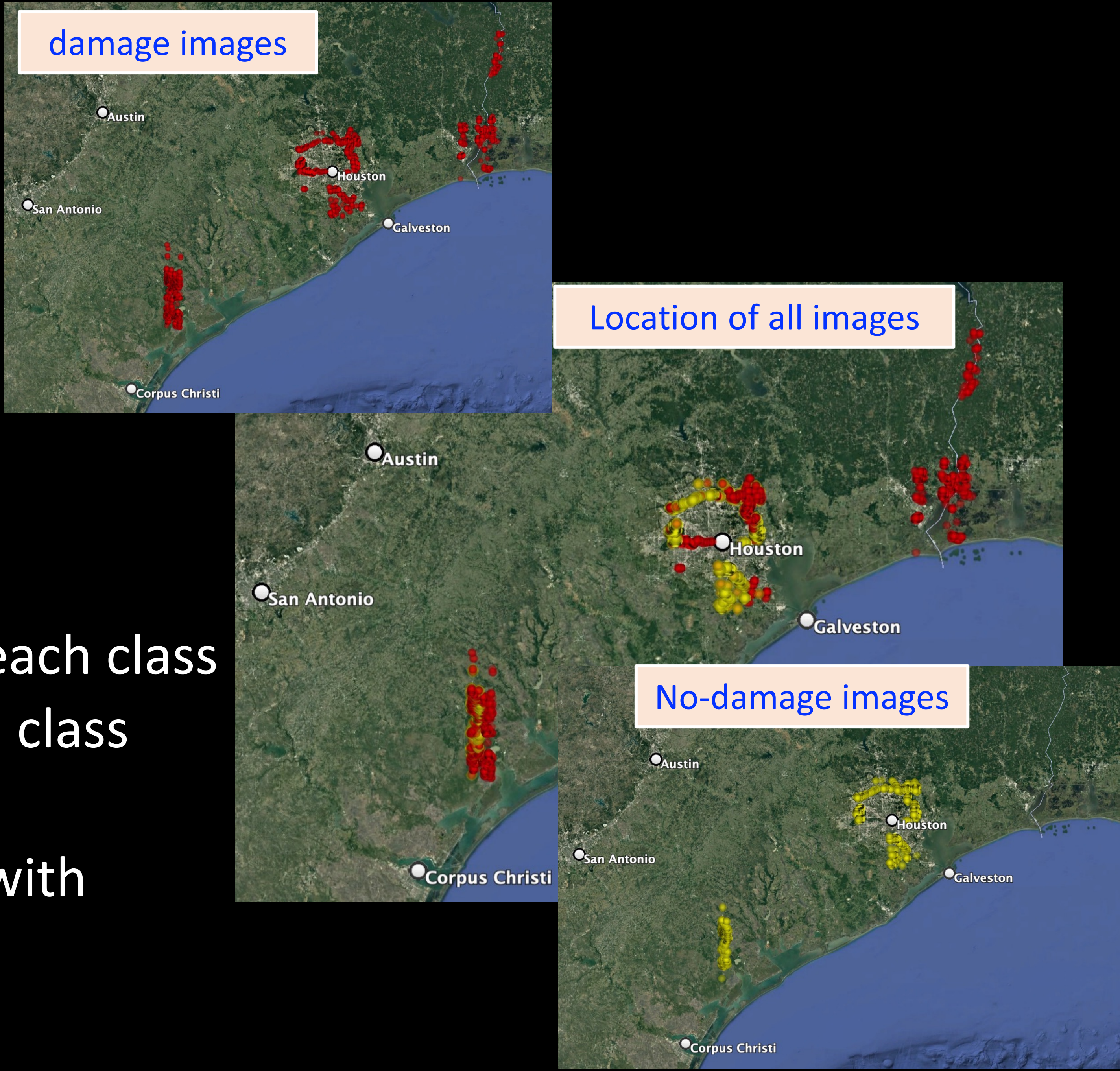
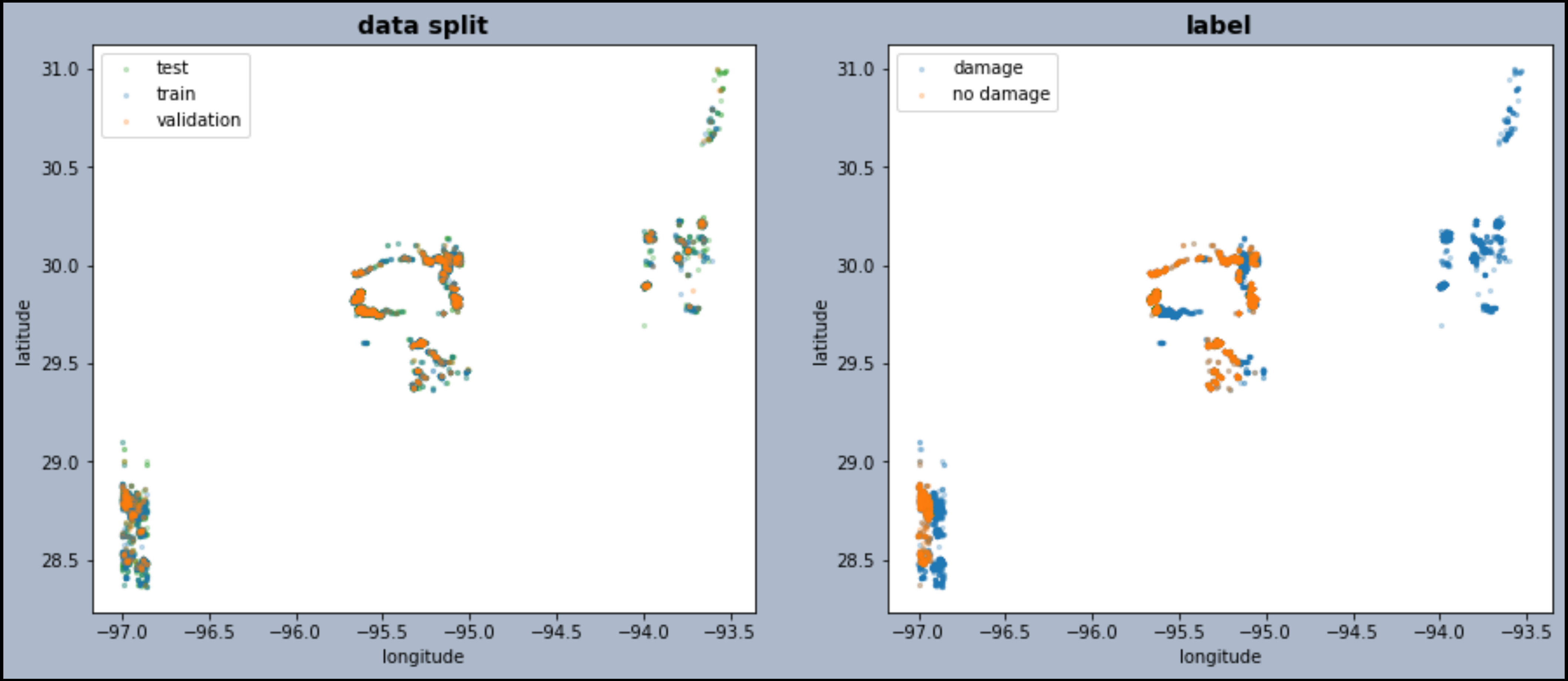
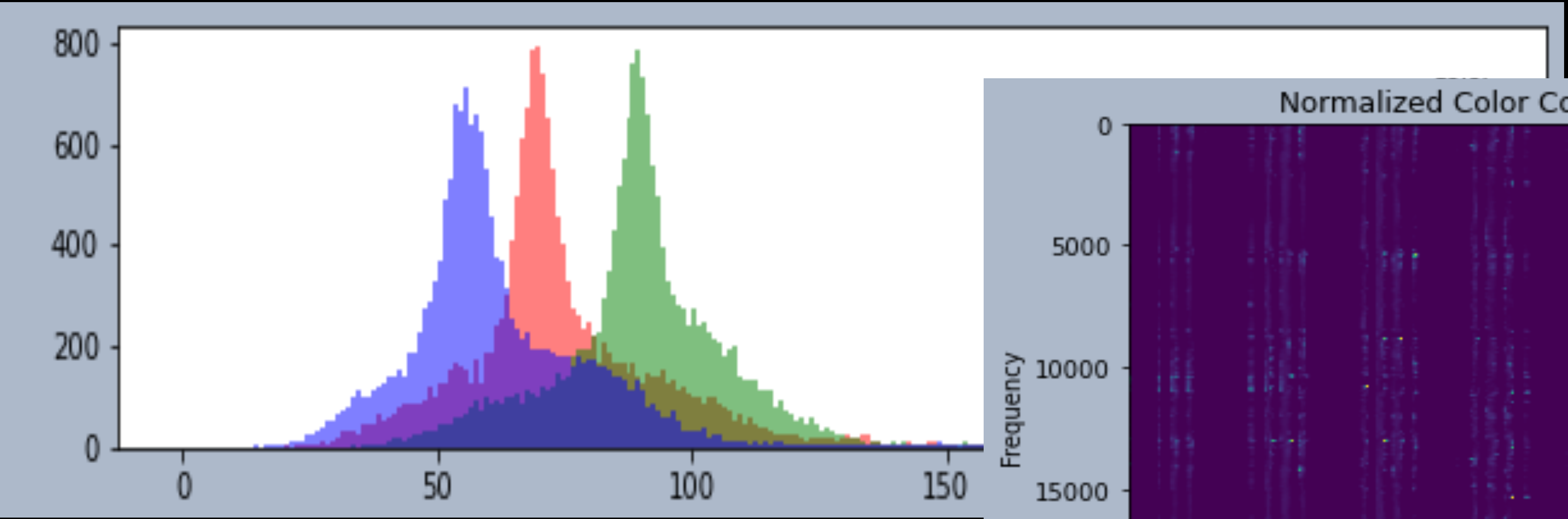


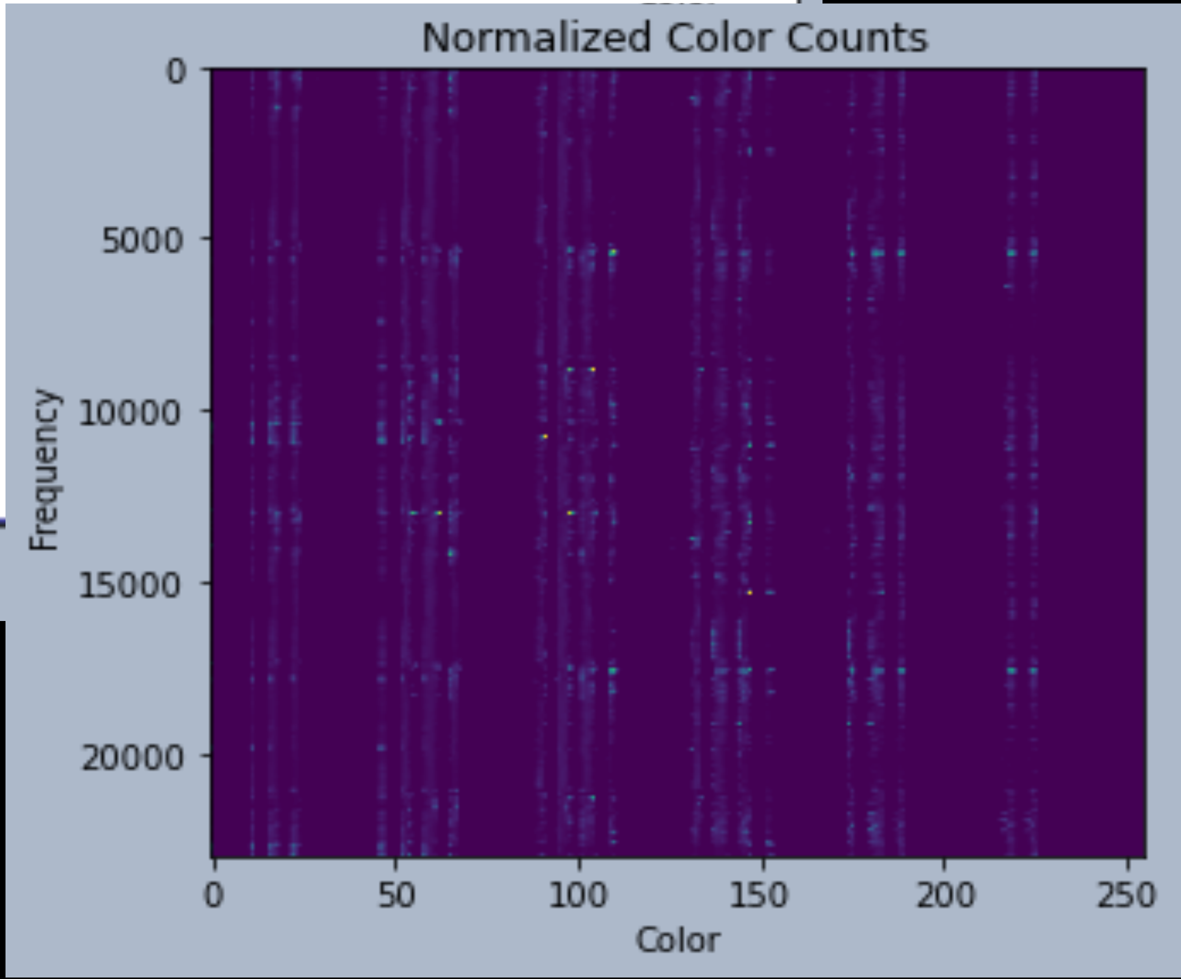
Image distribution by data type



Data Analysis

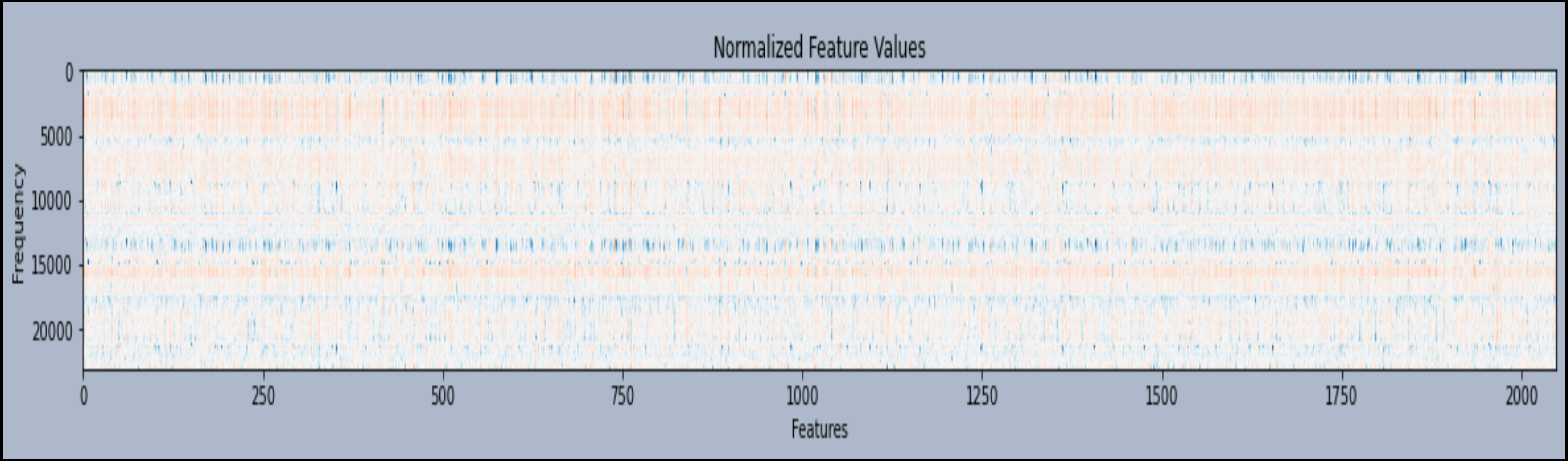
Extracting Features : color only features



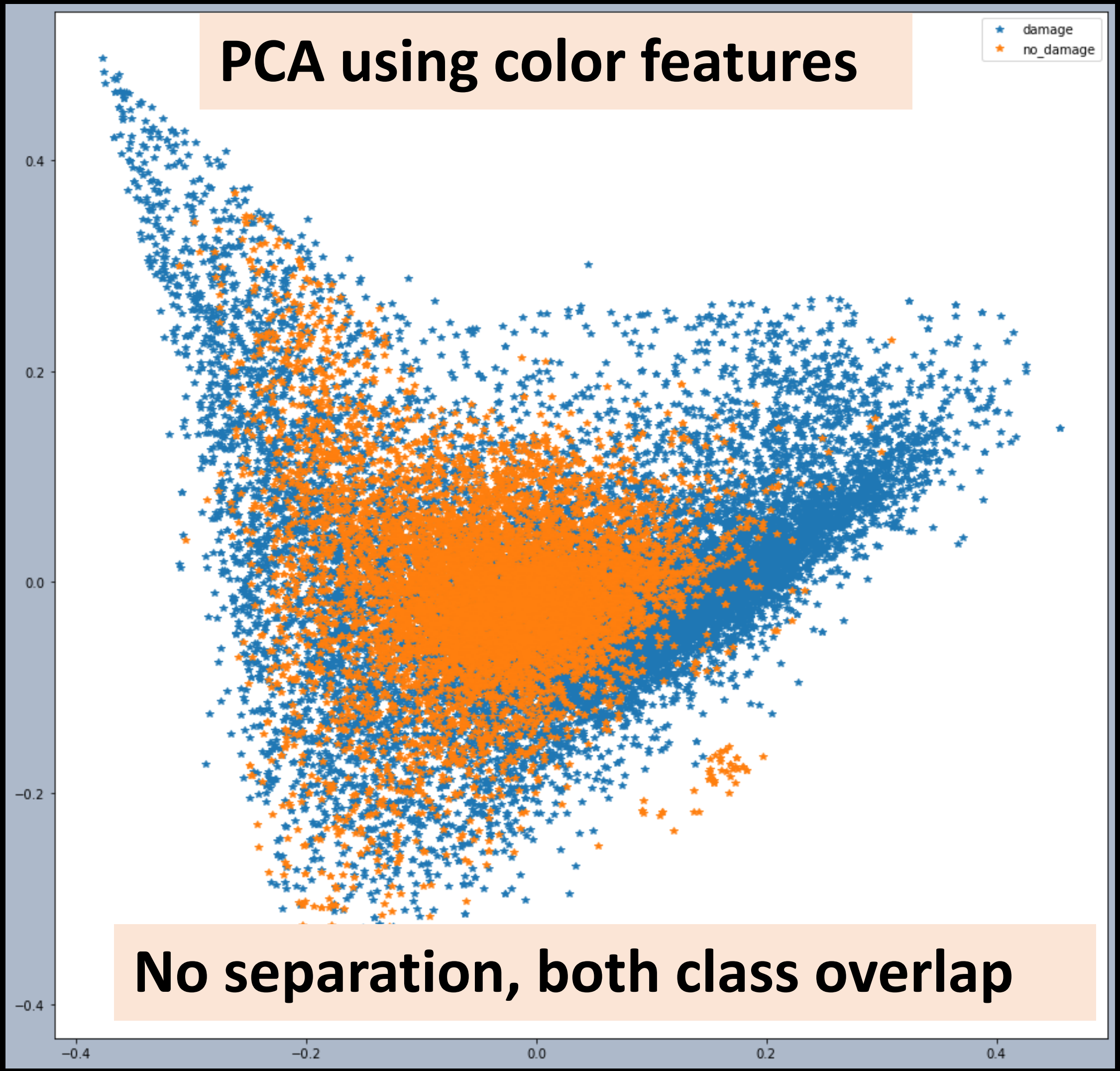


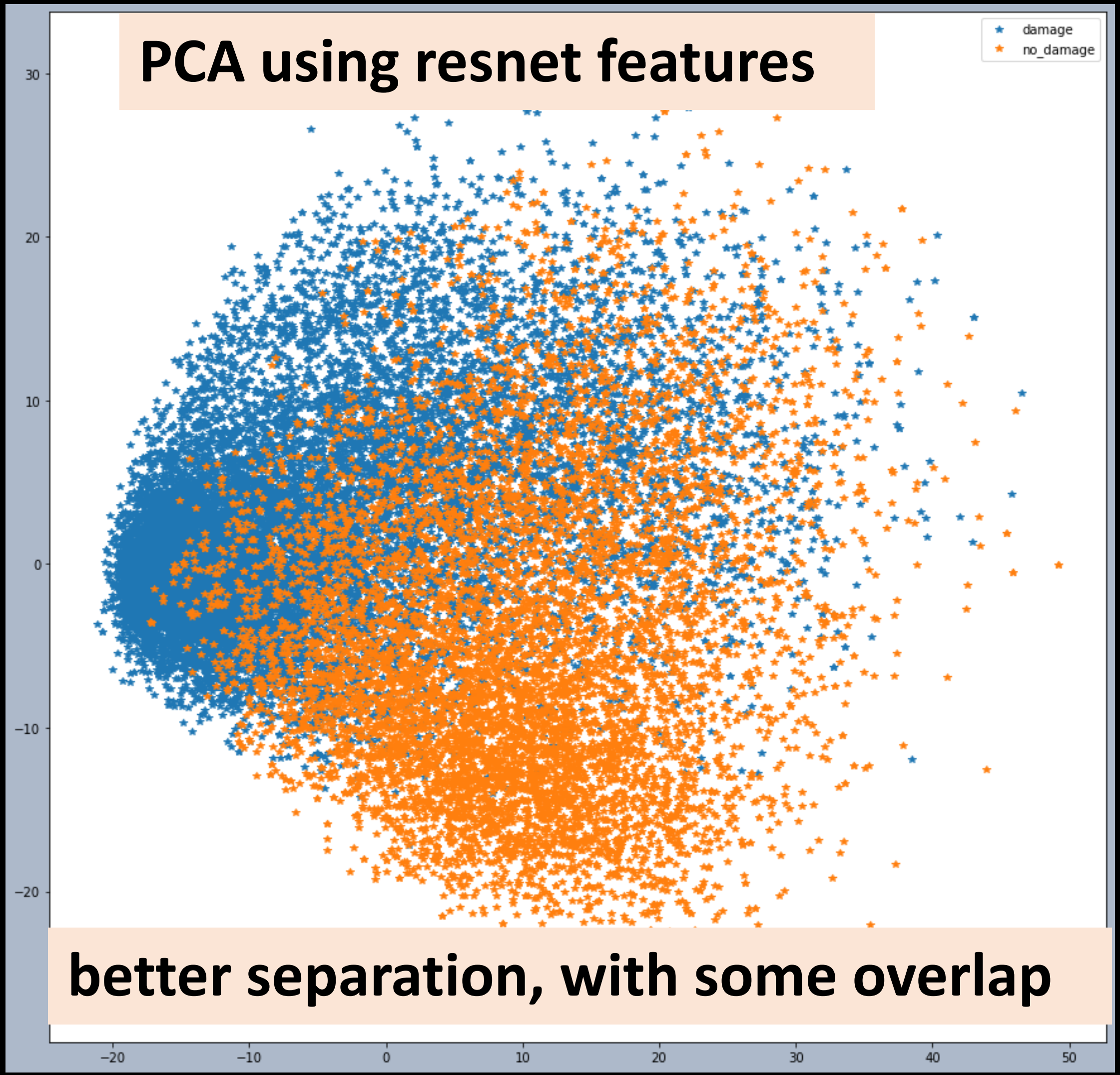
Extracting Features : Resnet50 features

- Acquired features from a pre-trained model. Besides color features, these features provide more information about the shape, corners and much more complicated features



Principal Component Analysis (PCA)





Modeling

Computer Vision

- Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs

- Core building blocks of computer vision include:
  - **Object classification**
  - **Object identification**
  - **Video motion analysis**
  - **Image segmentation**
  - **Scene reconstruction and**
  - **Image restoration**

Transfer Learning

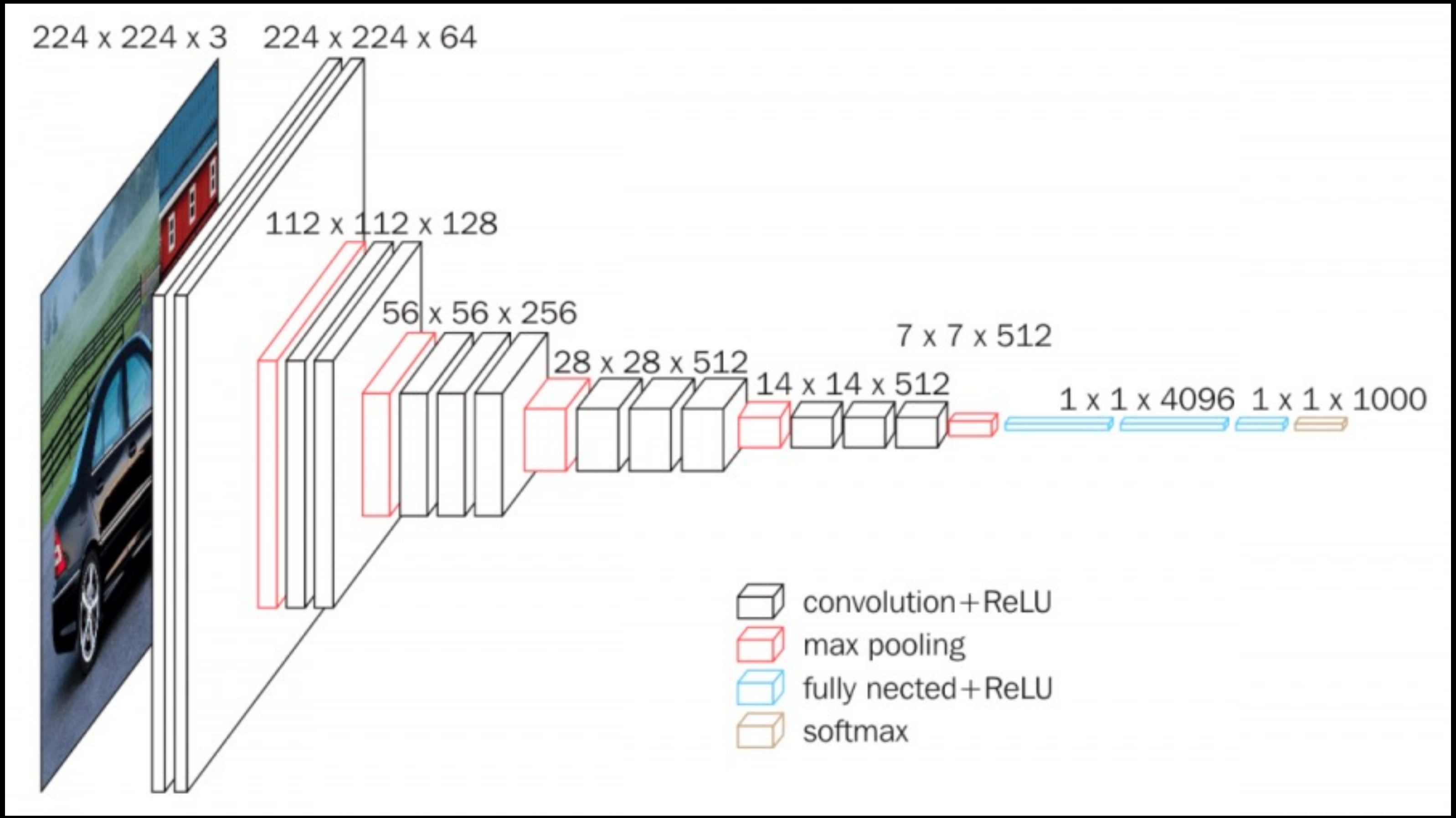
- Transfer learning is a machine learning method where a model developed for a task is reused as the starting point for a model on a second task.

- VGG16 is the model leveraged for transfer learning

- Transfer learning approach :
  - **Select Source Model --- a** pre-trained source model (in this case VGG16) is chosen from available models
  - **Reuse Model ---** re-use all or parts of the model for the second task of interest.
  - **Tune Model ---** adapted or refined on the input-output pair data available for the task of interest.

VGG16 Architecture

- VGG16 is a simple and widely used CNN Architecture used for ImageNet visual object recognition software research.



VGG16 layers

- VGG16 is composed of :
  - 13 convolutional layers --- tunable
  - 5 max-pooling layers --- non-tunable
  - 3 fully connected layers --- tunable
- The number of filters in the first block is 64 and the number is doubled in the subsequent blocks until it reaches 512.
- This model is finished by two fully connected hidden layers (each with 4096 neurons) and one output layer. with 1000 neurons.

Layers adopted for this project

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 224, 224, 64)	1792
conv2d_2 (Conv2D)	(None, 224, 224, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 112, 112, 64)	0
conv2d_3 (Conv2D)	(None, 112, 112, 128)	73856
conv2d_4 (Conv2D)	(None, 112, 112, 128)	147584
max_pooling2d_2 (MaxPooling2	(None, 56, 56, 128)	0
conv2d_5 (Conv2D)	(None, 56, 56, 256)	295168
conv2d_6 (Conv2D)	(None, 56, 56, 256)	590080
conv2d_7 (Conv2D)	(None, 56, 56, 256)	590080
max_pooling2d_3 (MaxPooling2	(None, 28, 28, 256)	0
conv2d_8 (Conv2D)	(None, 28, 28, 512)	1180160
conv2d_9 (Conv2D)	(None, 28, 28, 512)	2359808
conv2d_10 (Conv2D)	(None, 28, 28, 512)	2359808
max_pooling2d_4 (MaxPooling2	(None, 14, 14, 512)	0
conv2d_11 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_12 (Conv2D)	(None, 14, 14, 512)	2359808
conv2d_13 (Conv2D)	(None, 14, 14, 512)	2359808
max_pooling2d_5 (MaxPooling2	(None, 7, 7, 512)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_1 (Dense)	(None, 4096)	102764544
dropout_1 (Dropout)	(None, 4096)	0
dense_2 (Dense)	(None, 4096)	16781312
dropout_2 (Dropout)	(None, 4096)	0
dense_3 (Dense)	(None, 2)	8194

Total params: 134,268,738
Trainable params: 134,268,738
Non-trainable params: 0

Error Metrics

- Accuracy, precision and recall
- Confusion matrix and classification report.
- ROC/AUC values will be examined to access the quality of the classification from the models.

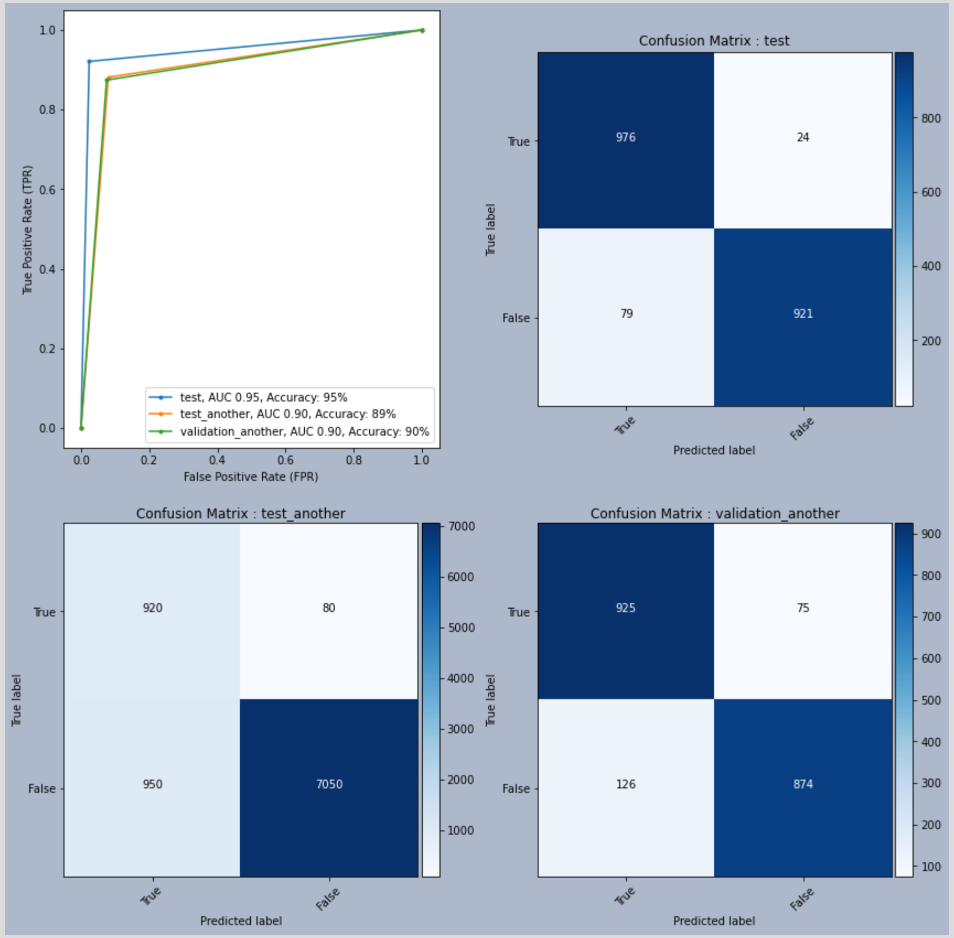
Base line model

- Generated two simple baseline models using K-Nearest Neighbors (KNN) algorithm. Two kinds of features were utilized to build the two baseline models:

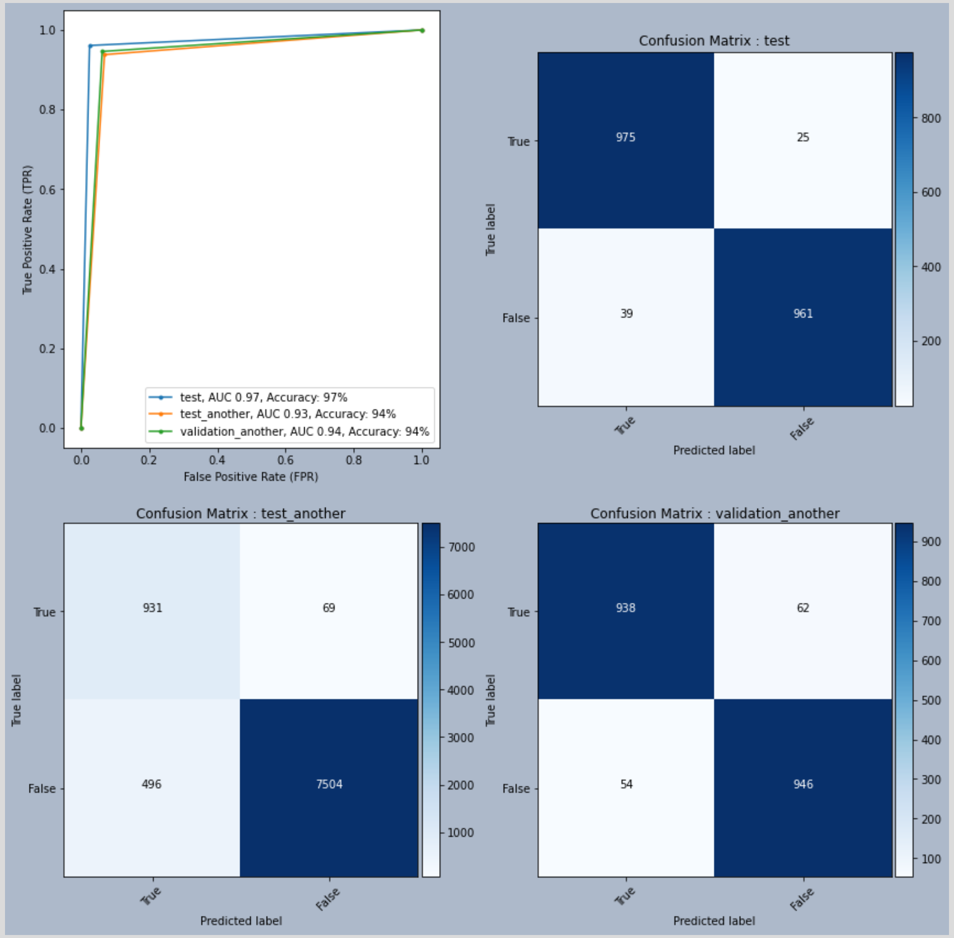
  - Baseline model 1: KNN model using color features only
  - Baseline model 2: KNN model using resnet50 features

Base line model

Baseline model 1



Baseline model 2



Base line model : Classification Report

		Base Line Model 1 - Color Features				Base Line Model 2 - Resnet Features			
Dataset		precision	recall	f1-score	support	precision	recall	f1-score	support
Test	FALSE	0.925	0.976	0.950	1000	0.962	0.975	0.968	1000
	TRUE	0.975	0.921	0.947	1000	0.975	0.961	0.968	1000
	accuracy	0.949	0.949	0.949	0.949	0.968	0.968	0.968	0.968
	macro avg	0.950	0.949	0.948	2000	0.968	0.968	0.968	2000
	weighted avg	0.950	0.949	0.948	2000	0.968	0.968	0.968	2000
Test Another	FALSE	0.492	0.920	0.641	1000	0.652	0.931	0.767	1000
	TRUE	0.989	0.881	0.932	8000	0.991	0.938	0.964	8000
	accuracy	0.886	0.886	0.886	0.886	0.937	0.937	0.937	0.937
	macro avg	0.740	0.901	0.787	9000	0.822	0.935	0.865	9000
	weighted avg	0.934	0.886	0.900	9000	0.953	0.937	0.942	9000
Validation Another	FALSE	0.880	0.925	0.902	1000	0.946	0.938	0.942	1000
	TRUE	0.921	0.874	0.897	1000	0.938	0.946	0.942	1000
	accuracy	0.900	0.900	0.900	0.900	0.942	0.942	0.942	0.942
	macro avg	0.901	0.900	0.899	2000	0.942	0.942	0.942	2000
	weighted avg	0.901	0.900	0.899	2000	0.942	0.942	0.942	2000

Extended modeling

Two models were built using transfer learning from VGG16:

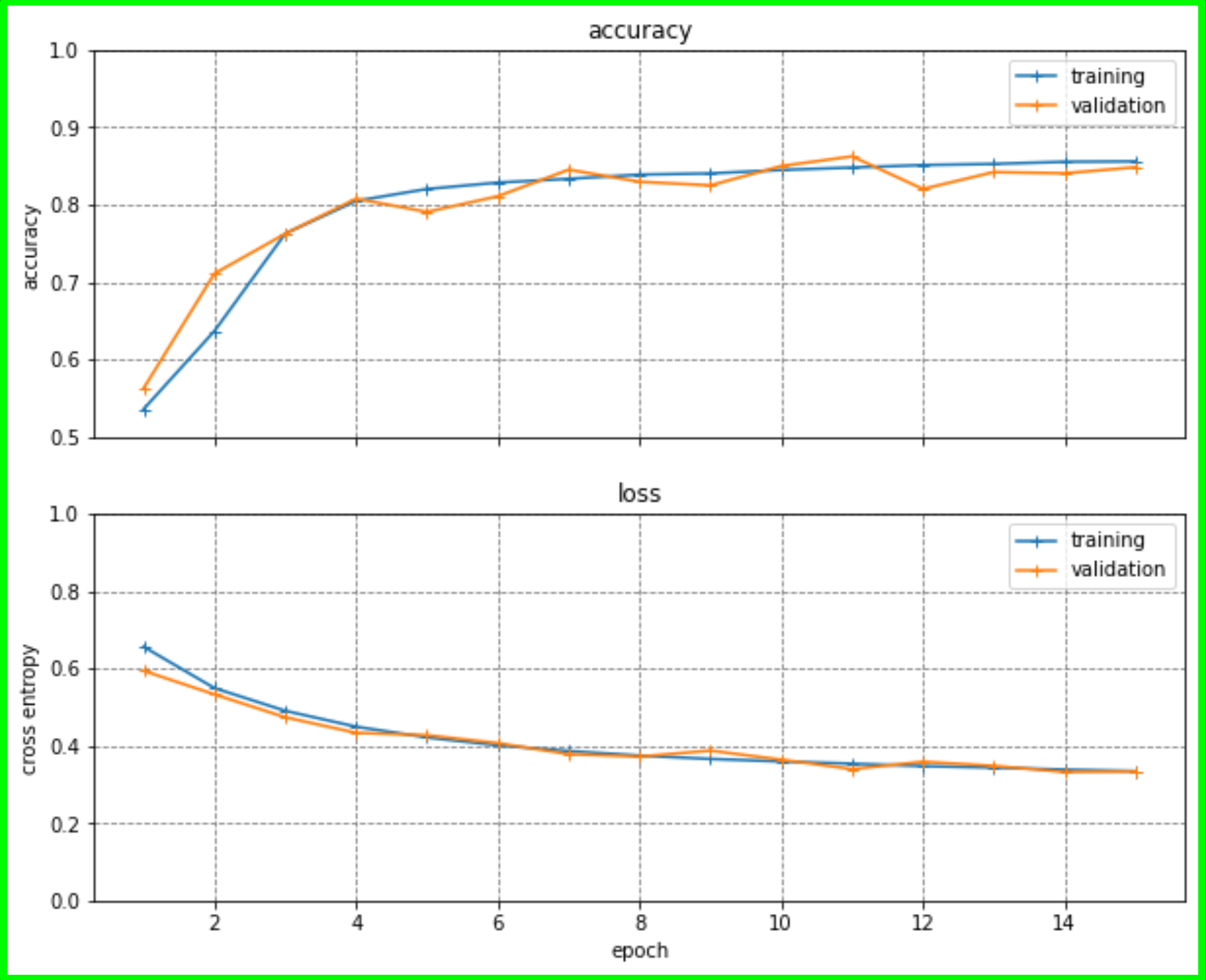
- Model 1
  - Take all the convolutional + the max pooling layers of VGG16 and freeze them
  - Add a global-average layer and a prediction layer
- Model 2
  - Take all the convolutional + the max pooling layers of VGG16 and freeze the top 15 layers only.
  - Re-train the deeper three convolutional layers using our input datasets
  - Add a global-average layer and a prediction layer

Extended modeling : Model 1

Model 1 architecture

```
Model - 1 summary

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Functional)           (None, 4, 4, 512)         14714688
_________________________________________________________________
global_average_pooling2d (Gl (None, 512)               0
_________________________________________________________________
dense_21 (Dense)             (None, 1)                 513
=================================================================
Total params: 14,715,201
Trainable params: 513
Non-trainable params: 14,714,688
```



- Model 1 accuracy is only 86%
- Lower than the base line model

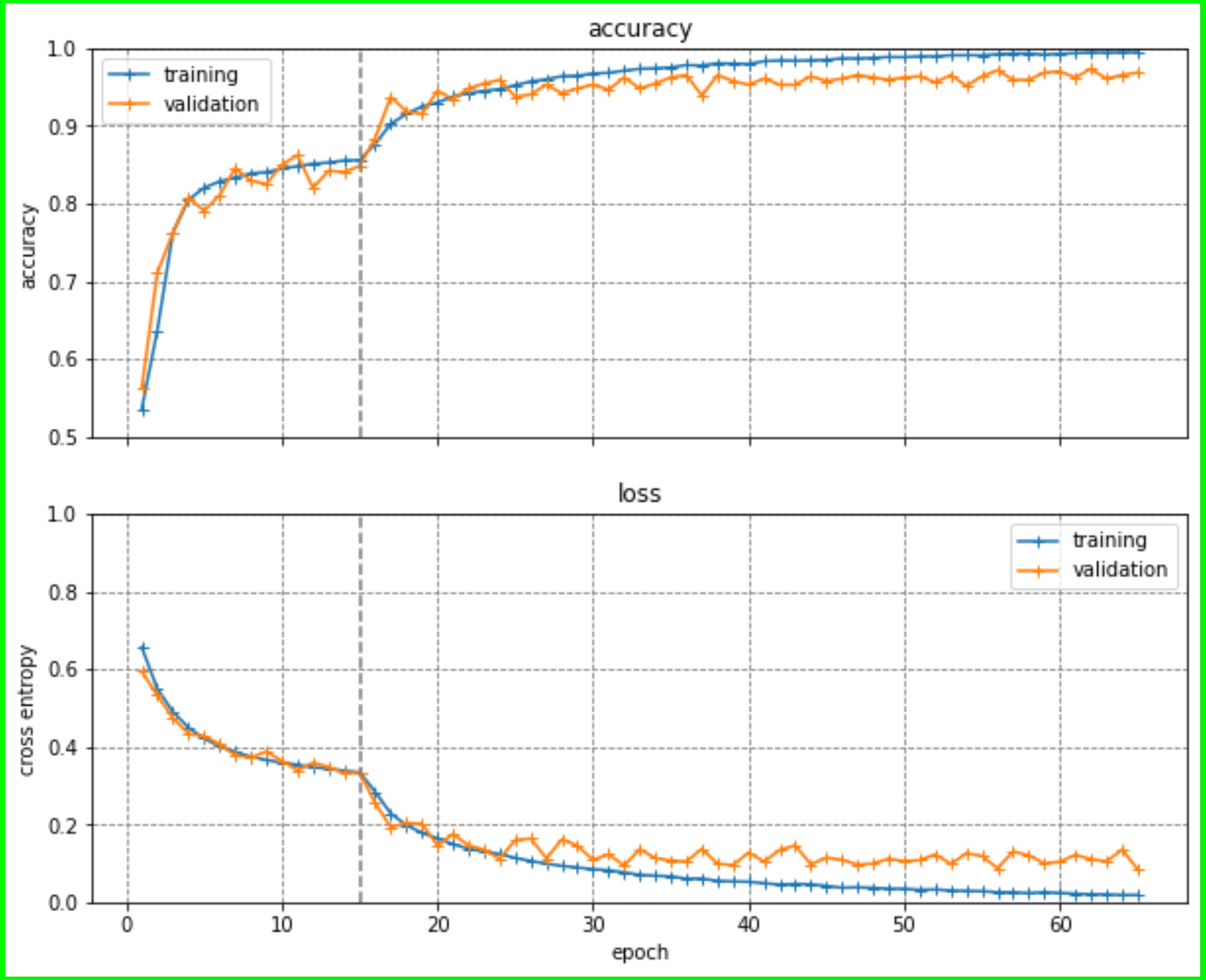
Extended modeling : Model 2

Model 2 architecture

Model - 2 summary

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 4, 4, 512)	14714688
global_average_pooling2d	(Gl (None, 512)	0
dense_21 (Dense)	(None, 1)	513

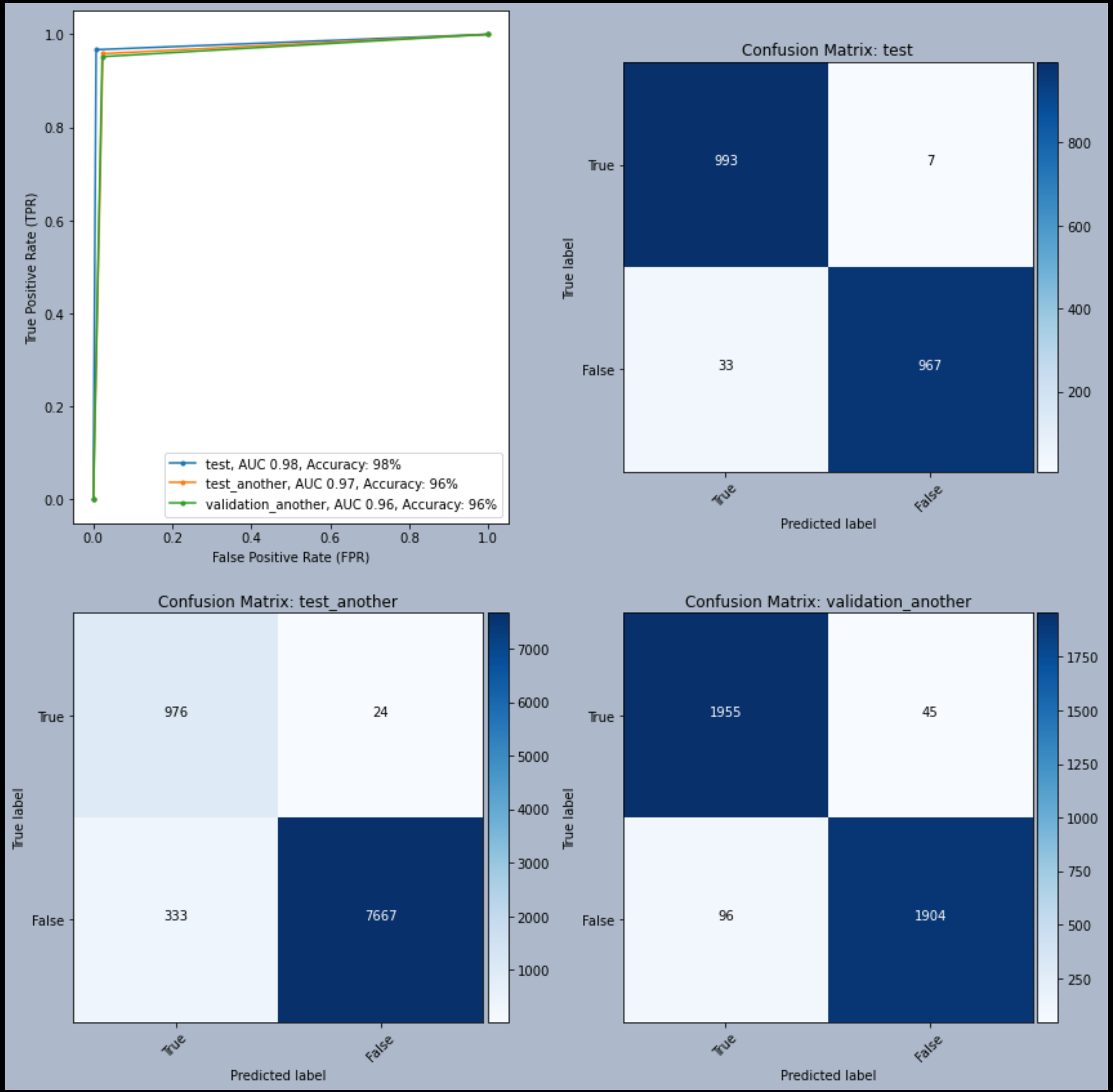
Total params: 14,715,201
Trainable params: 7,079,937
Non-trainable params: 7,635,264



Extended modeling : Model 2

Model 2 results :

- Accuracy of 96%, an improvement of 10% from model 1
- This suggests that the deeper layers, which create features that are specific to the dataset, need to get re-trained on the dataset at hand.



Model Comparison

		Base Line Model - Color Features				Base Line Model - Resnet Features				Final Model			
Dataset		precision	recall	f1-score	support	precision	recall	f1-score	support	precision	recall	f1-score	support
Test	FALSE	0.925	0.976	0.950	1000	0.962	0.975	0.968	1000	0.968	0.993	0.980	1000
	TRUE	0.975	0.921	0.947	1000	0.975	0.961	0.968	1000	0.993	0.967	0.980	1000
	accuracy	0.949	0.949	0.949	0.949	0.968	0.968	0.968	0.968	0.980	0.980	0.980	0.980
	macro avg	0.950	0.949	0.948	2000	0.968	0.968	0.968	2000	0.980	0.980	0.980	2000
	weighted avg	0.950	0.949	0.948	2000	0.968	0.968	0.968	2000	0.980	0.980	0.980	2000
Test Another	FALSE	0.492	0.920	0.641	1000	0.652	0.931	0.767	1000	0.746	0.976	0.845	1000
	TRUE	0.989	0.881	0.932	8000	0.991	0.938	0.964	8000	0.997	0.958	0.977	8000
	accuracy	0.886	0.886	0.886	0.886	0.937	0.937	0.937	0.937	0.960	0.960	0.960	0.960
	macro avg	0.740	0.901	0.787	9000	0.822	0.935	0.865	9000	0.871	0.967	0.911	9000
	weighted avg	0.934	0.886	0.900	9000	0.953	0.937	0.942	9000	0.969	0.960	0.963	9000
Validation Another	FALSE	0.880	0.925	0.902	1000	0.946	0.938	0.942	1000	0.953	0.978	0.965	2000
	TRUE	0.921	0.874	0.897	1000	0.938	0.946	0.942	1000	0.977	0.952	0.964	2000
	accuracy	0.900	0.900	0.900	0.900	0.942	0.942	0.942	0.942	0.965	0.965	0.965	0.965
	macro avg	0.901	0.900	0.899	2000	0.942	0.942	0.942	2000	0.965	0.965	0.965	4000
	weighted avg	0.901	0.900	0.899	2000	0.942	0.942	0.942	2000	0.965	0.965	0.965	4000

Model Comparison

- Comparison to the work done by Quoc Dung Cao and Youngjun Choe (original study associated with the data)
  - model architecture --- CNN + data augmentation + 50% dropout using Adam optimizer.
  - accuracy of 98%, 97.29% and 97.03% for the validation, test data(balanced) and another test (unbalanced data)

- model 2 has --- an accuracy of 96%, 98% and 96% for the validation, test data(balanced) and another test (unbalanced data).
  - by doing transfer learning we can achieve similar performance to the ones built from scratch.

Conclusion

- We demonstrated that transfer learning + fine tuning of the deeper layers can achieve similar results to the CNN models built from scratch
  - This will save a lot of compute power and more importantly time.

- For emergency managers and responders, a quick and accurate identification of damage assessment is beneficial. This can be done into two stages:
  - Stage1 --- Extract features from a pre-trained model and use the features as an exogenous parameter and run a quick classification model such as KNN or logistic linear regression. Though the accuracy is not at the level of the CNN models, it is very quick with decent accuracy.
  - Stage2 --- While the emergency managers are utilizing the results from stage1, a new model can be trained using transfer learning + fine-tuning which would greatly improve the accuracy with minimum runtime.

Future Works

- The baseline model generated based on features extracted from resnet50 produced really good results. This could be either:
  - The dataset came from a small geographic location and may have similar building features and the classification problem is binary
  - The features extract from restnet50 are really good and were able to generalize over many images.
- Extract features from VGG16 and use KNN to build a model and compare results to that of the resnet50.
  - The comparison will confirm whether the good results of the baseline model were due to the features of the resnet50 or similar accuracy can be achieved by utilizing another pre-trained model.
- Do transfer learning using the resnet50 architecture and compare it to our best model, which was generated by doing transfer learning from VGG16.

Recommendations

- Instead of building CNN from scratch, it is good to consider and explore transfer learning first. There are several pre-trained models easily available

- We can use these pre-trained models in several ways:

  - Extract features from a pre-trained model and use the features as an exogenous parameter and run a quick classification model such as KNN or logistic linear regression. Though the accuracy is not at the level of the CNN models, it is very quick with decent accuracy.

  - Use transfer learning + fine-tuning which would greatly improve the accuracy with minimum runtime.

References

[1] https://www.kaggle.com/kmader/satellite-images-of-hurricane-damage
[2] https://www.datarobot.com/blog/introduction-to-computer-vision-what-it-is-and-how-it-works/
[3] https://machinelearningmastery.com/transfer-learning-for-deep-learning/
[4] https://medium.com/mlearning-ai/an-overview-of-vgg16-and-nin-models-96e4bf398484
[5] https://arxiv.org/abs/1807.01688