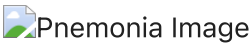
Final Project Submission

- Student name: Deztany Jackson
- Student pace: DS Flex
- Scheduled project review date/time: March 30, 2023
- Instructor name: Morgan Jones
- Blog post URL: http://dmvinedata.com/systembias/

X-ray Image Predictions: Pneumonia or Normal


Pnemonia Image

Summary

A local hospital wants to explore their image recognition options for pneumonia cases. They want eventually replace a few doctors due to staffing issues. They want to test it out for a trial version. Worldwide Pneumonia s is a deadly and expensive condition, especially for children and the elderly. Using Convultion Neural Networks, X-Ray images of Normal and Pneumonia patients, data scienties will attempt to build models that detect pneumonia with the goal of minimizing False Negatives (predicting no pneumonia, while there is). Because of the severity this is a main focus.

The dataset is from Chest X-Ray of "Normal" and "Pneumonia" images from Kaggle with a total of 5856 .jpeg files.

Based on the business goal and data, the main metrics will be Recall, F1-Score, and Loss, Accuracy will be secondary.

There is a data imbalance between categories and splits. After the images are loaded, the Train and Validation images are combined and then split 80/20 for each respective category (Normal, Pneumonia). This does not solve the data and split imbalance but improves the offset to support more accurate modeling evaluation metrics.

Modeling was down over 10 iterations (including a Transfor Model) with Hyperparameter tweeks: (Batch Size, Epochs, Dropout, Activation, Class Weight). The best models had no False Negatives or False Positives with scores of 1. The chosen model had the smallest loss.

Hospital Recommendations:

- Usage: The model is best as a learning tool and not an official diagnosis.

- Strategy: Use the model as an initial reviewer of the images.
- Staffing: The model is best used with a doctor, not standalone.

Technical Recommendations:

- Scope, review and process images more beforehand
- Visualize the activation functions to see better what areas the model layers are diagnosing
- Iterate model improvement with with augmented data
- Visually inspect the images that were FN and FP.

Import Libraries

In [1]:
```
import os
import glob

#from google.colab import drive #Google Collab

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pickle

import tensorflow as tf

from tensorflow import keras
from tensorflow.keras.metrics import Accuracy,Precision,Recall
from tensorflow.keras.preprocessing.image import ImageDataGenerator, array_to_i
from tensorflow.keras import models, layers, optimizers, activations
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropo
from tensorflow.keras.utils import plot_model

from sklearn.model_selection import train_test_split


from sklearn.model_selection import GridSearchCV
from tensorflow.keras.wrappers.scikit_learn import KerasClassifier
from keras.optimizers import Adam

from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
from tensorflow.keras.applications import  VGG19 #Pretrained Transfer Modeling
from sklearn.utils.class_weight import compute_class_weight

from keras.preprocessing import image
from PIL import Image

from sklearn.metrics import make_scorer
from sklearn.metrics import precision_score, f1_score, recall_score,accuracy_sc
from sklearn.metrics import precision_recall_curve, PrecisionRecallDisplay, ave
from sklearn.metrics import confusion_matrix as cm
from sklearn.metrics import classification_report, ConfusionMatrixDisplay
from sklearn.utils.class_weight import compute_class_weight

import datetime
```

```
#Used to move files
import shutil

#Ignore warnings
import warnings
warnings.filterwarnings('ignore')
tf.__version__

# Stop Tensorflow errors from showing especially
import logging
#tf.get_logger().setLevel(logging.ERROR)

# Showing the areas of the image boundaries
import lime
from lime import lime_base
from lime import lime_image
from skimage.segmentation import mark_boundaries
```

In [2]:
```
#Getting start time
original_start = datetime.datetime.now()
```

The following modeling effort was developed on a Apple M2 Max 64GB computer. There may be tasks that may longer execute on other hardware

Data Understanding

The dataset is from Chest X-Ray of "Normal" and "Pneumonia" images from Kaggle with a total of 5856 .jpeg files. This data set is already broken up into three folders (train, validation and test) and a folder for each category ("NORMAL" or "PNEUMONIA").

Original Dataset Splits

- **Train [1341 NORMAL, 3875 PNEUMONIA]**
- **Validation [8 NORMAL, 8 PNEUMONIA]**
- **Test [234 NORMAL, 390 PNEUMONIA]**

There is a data imbalance between categories and splits. After the images are loaded, the Train and Validation images are combined and then split 80/20 for each respective category (Normal, Pneumonia). This does not solve the data and split imbalance but improves the offset to support more accurate modeling evaluation metrics.

The original dataset is already split by the time this notebook is turned it. The following code will still work with the original dataset and previously shifted images

Loading Images

Images are loaded from the current directory joined with the location of the datset. The python "glob" module Glob, Tara Boyle, 2021 is used to retrieve files recursively.

Without knowing all the file names "*" astericks are used in place to get all fo the files.

In [3]:
```
#Get Current directory
cur_dir = os.getcwd()
cur_dir
```

Out[3]:
```
'/Users/deztanyjackson/Documents/0_Flatiron/Phase_4/Project_4/Image_Classification_P4'
```

Understanding the total images split among the respective directories

In [4]:
```
def print_image_lengths():
    """ Retrieving data and the lengths of the training, validation and test fo
    #Retrieving dataset in the respective folders
    train_data = glob.glob(os.path.join(cur_dir,'data/chest_xray/train/**/*.jpe
    val_data = glob.glob(os.path.join(cur_dir,'data/chest_xray/val/**/*.jpeg'))
    test_data = glob.glob(os.path.join(cur_dir,'data/chest_xray/test/**/*.jpeg'

    #Printing total number of images in different directories
    print(f"Training Set has: {len(train_data)} images")
    print(f"Validation Set has: {len(val_data)} images")
    print(f"Testing Set has: {len(test_data)} images")

    return train_data, val_data, test_data
```

Understanding the total categories (Normal and Pneumonia) split among the respective directories

---

Initial Data Balancing

The data splitting and moving will not solve the data imbalance for training, but supports better modeling metric evaluations.

- Orinally started with 16 Validation Images
- Moving validation images to train set.
- Perform Random Train/Valid split 80/20 on Normal and Pneumonia images
- Move 20% of each back to validation folder

Retrieves Normal and Pneumonia specific images for each dataset

In [5]:
```
#Pulling in Normal and Pneumonia specific data

#Training Data
t_norm = glob.glob(os.path.join(cur_dir,'data/chest_xray/train/NORMAL/*.jpeg'))
t_pneu = glob.glob(os.path.join(cur_dir,'data/chest_xray/train/PNEUMONIA/*.jpeg
#Validation Data
v_pneu = glob.glob(os.path.join(cur_dir,'data/chest_xray/val/PNEUMONIA/*.jpeg')
v_norm = glob.glob(os.path.join(cur_dir,'data/chest_xray/val/NORMAL/*.jpeg'))
#Test Data
test_norm = glob.glob(os.path.join(cur_dir,'data/chest_xray/test/NORMAL/*.jpeg'
test_pneu = glob.glob(os.path.join(cur_dir,'data/chest_xray/test/PNEUMONIA/*.jp
```

Moving and Splitting Files (If Necessary)

Moving the validation data to the respective Normal and Pneumonia Training folders if the amount of files in the folders are less than 20 (This number is arbitrary). . If so, there will be a 80/20 split of the data and then the 20% of files are moved back to the respective Validation folder. The original dataset had 8 images per folder. The goal is to only move files if there is an huge value offset. If not, then the files don't need to move.

In [6]:
```
if (len(v_pneu) <= 20) & (len(v_norm) <= 20): # Only perform action if validat
    #Moving Validation to Test
    cat = [v_norm,v_pneu]
    for c in cat:
        if c == v_norm:
            images = [f for f in v_norm if '.jpeg' in f.lower()]
            for img in images:
                new_path = os.path.join(cur_dir,"data/chest_xray/train/NORMAL/"
                shutil.move(img,new_path)
        elif c == v_pneu:
            images = [f for f in v_pneu if '.jpeg' in f.lower()]
            for img in images:
                new_path = os.path.join(cur_dir,"data/chest_xray/train/PNEUMONI
                shutil.move(img,new_path)
        else:
            pass

        #Splitting the data 80/20 for the Normal and Pnuemonia categories.
        #Moving the 20% split to the respective Validation directories.
    train_cat = [t_norm,t_pneu]
    for cat in train_cat:
        train,val = train_test_split(cat, train_size=.85, random_state=42)
        if cat == t_norm:
            for img in val:
                new_path = os.path.join(cur_dir,"data/chest_xray/val/NORMAL/")
                shutil.move(img,new_path)
        elif cat == t_pneu:
            for img in val:
                new_path = os.path.join(cur_dir,"data/chest_xray/val/PNEUMONIA/
                shutil.move(img,new_path)
        else:
            pass
else: #If validation folders doesnt have less than 20 images then don't do anyt
    print("The folders are split well")
```

```
The folders are split well
```

In [7]:
```
#Printing Dataset values
train_data, val_data, test_data = print_image_lengths()
print("\n")
#Training Data
print(f"Training Pneumonia Amount: {len(t_pneu)} images")
print(f"Training Normal Amount: {len(t_norm)} images")
print(f"Training Normal to Pneumonia Ratio: {round(len(t_norm)/len(t_pneu),2)}"
print("\n")
#Validation Data
print(f"Validation Pneumonia Amount: {len(v_pneu)} images")
```

```
print(f"Validation Normal Amount: {len(v_norm)} images")
print(f"Validation Normal to Pneumonia Ratio: {round(len(v_norm)/len(v_pneu),2)
```

```
Training Set has: 4509 images
Validation Set has: 723 images
Testing Set has: 624 images


Training Pneumonia Amount: 3347 images
Training Normal Amount: 1162 images
Training Normal to Pneumonia Ratio: 0.35


Validation Pneumonia Amount: 536 images
Validation Normal Amount: 187 images
Validation Normal to Pneumonia Ratio: 0.35
```

There is still imbalance but better offset.

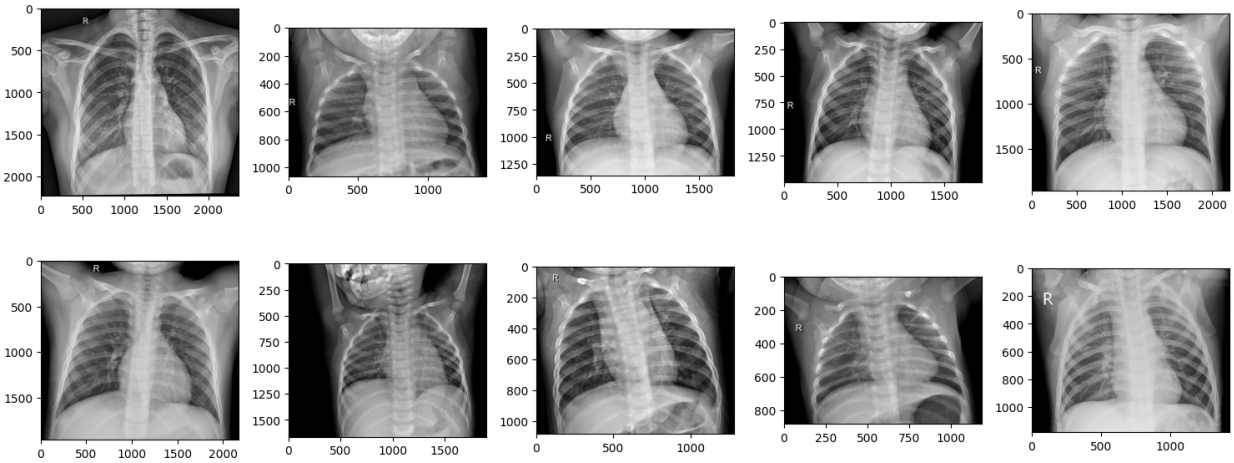
Normal and Pneumonia Images

Displaying 10 of the first images in the Normal and Pneumonia Training folders. ".imread()" method reads images from file into an array. All the images will be displayed a grid to show size and in the color gray. For the images below, the higher pixel intensity is associated with lighter color (possible, light gray or white). Because of the cloudiness that Pneumonia displays, it would have a higher pixel intensity.

NORMAL IMAGES

In [8]:
```
#Plotting 10 images from the train folder
plt.figure(figsize = (15,15))
print ("Sample of Normal Pictures")
for i in range(10):
    plt.subplot(5,5,i+1)
    #Reads images from file into an array
    img = plt.imread(t_norm[i])
    plt.imshow(img, cmap = "gray")
    plt.tight_layout()
plt.savefig('images/Img_Norm.png')
```

```
Sample of Normal Pictures
```

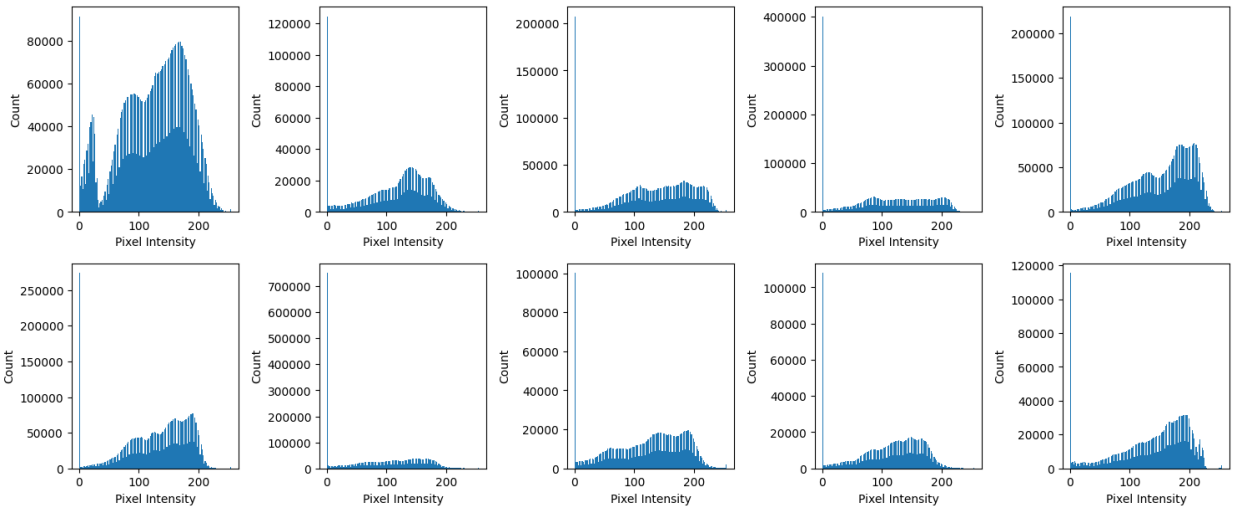


NORMAL PIXEL INTENSITY

Displaying pixel intensity associated with the plotted images. An image has 256 values (0,255). Zero is associated to the color black and 255 associated to the color white. Pixel Understanding

In [9]:
```
#Plotting 10 image pixels from the train folder
plt.figure(figsize = (15,15),facecolor="White")
print ("Associated Normal Pixel Intensities")
for i in range(10):
    plt.subplot(5,5,i+1)
    img = plt.imread(t_norm[i])
    plt.hist(img.ravel(), bins=150)
    plt.tight_layout()
    plt.ylabel("Count")
    plt.xlabel("Pixel Intensity")
plt.savefig('images/Hist_Norm.png')
    #plt.axis('off')
```

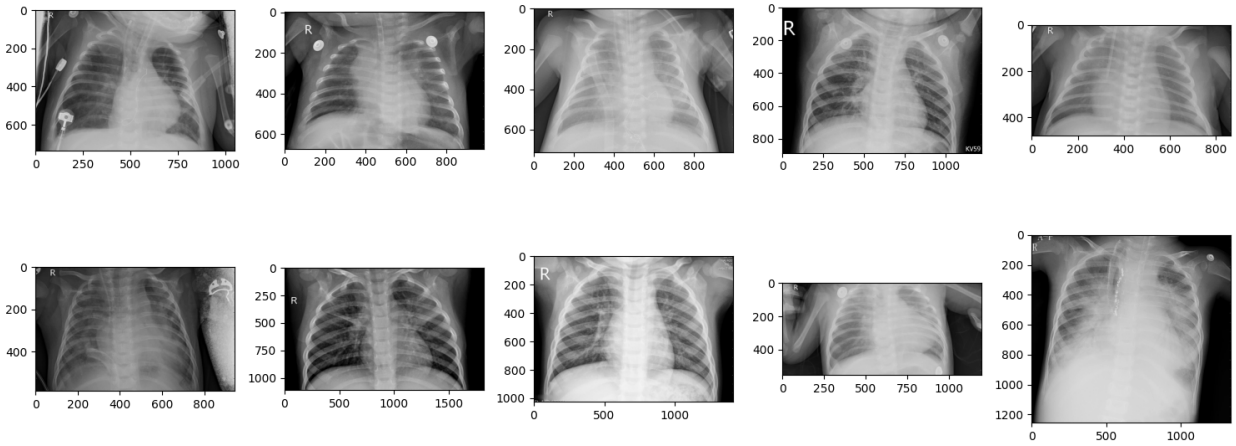
```
Associated Normal Pixel Intensities
```



PNEUMONIA IMAGES

In [10]:
```
plt.figure(figsize = (15,15))
print ("Sample of Random Pneumonia Pictures")
for i in range(10):
    plt.subplot(5,5,i+1)
    img = plt.imread(t_pneu[i])
    plt.imshow(img, cmap = "gray")
    plt.tight_layout()
plt.savefig('images/Img_Pneu.png')
    #plt.axis('off')
```

```
Sample of Random Pneumonia Pictures
```



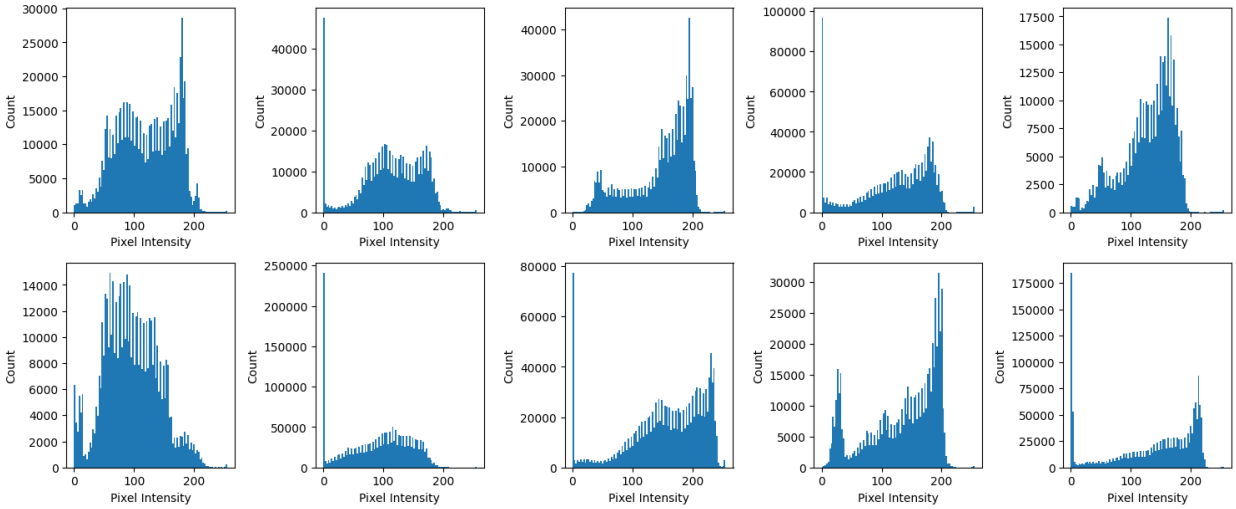
Visually, both the Normal and Pneumonia pictures vary in size. Pneumonia pictures have a cloudier visual look.

PNEUMONIA PIXEL IMAGES

Displaying pixel intensity associated with the plotted images.

In [11]:
```
plt.figure(figsize = (15,15),facecolor="White")
print ("Associated Pneumonia Pixel Intensities")
for i in range(10):
    plt.subplot(5,5,i+1)
    img = plt.imread(t_pneu[i])
    plt.hist(img.ravel(), bins=100)
    plt.tight_layout()
    plt.ylabel("Count")
    plt.xlabel("Pixel Intensity")
plt.savefig('images/Hist_Pneu.png')
    #plt.axis('off')
```

```
Associated Pneumonia Pixel Intensities
```



It appears the pixel intensity is more with Pneumonia.

Preprocessing Images

Rescaling the image Rescaling

For the intial dataset, the only preprocessing will be rescaling, creating target size and scoping the batch size. Further data augmentation will be done during the modeling.

Image Generator

Generating batches of tensor image data with real-time data augmentation.

In [12]:
```
#ImageDataGenerator without pre data augmentation
train_datagen = ImageDataGenerator(rescale=1./255)
val_datagen = ImageDataGenerator(rescale=1./255)
test_datagen = ImageDataGenerator(rescale=1./255)
```

Creating directories for augmentation. The subdirectories will be understood by the operation.

In [13]:
```
#Create Current Directories
train_directory= 'data/chest_xray/train'
validation_directory= 'data/chest_xray/val'
test_directory = 'data/chest_xray/test'
```

With the tensor image data and directory we will generate batches of augmented data.

Describe:

- **Target Size - Dimensions the images will be resized to**
- **Batch Size - Size of batches of data**
- **Classes - Class names**

- Class Mode - "Binary" 1D Binary labels
- Shuffle - Sorts data in a random matter, Default is True.
- Seed - Random seed for shuffling

In [14]:
```
#Using a generator for the images to work with.
train_gen = train_datagen.flow_from_directory(train_directory,
                                              target_size = (224, 224),
                                              batch_size=300,
                                              classes = ["NORMAL","PNEUMON
                                              class_mode = 'binary',
                                              seed = 42)

val_gen = val_datagen.flow_from_directory(validation_directory,
                                          target_size = (224, 224),
                                          batch_size = 100,
                                          classes = ["NORMAL","PNEU
                                          class_mode = 'binary',
                                          seed = 42)
#Testing data will be used on Final Model only
test_gen = test_datagen.flow_from_directory(test_directory,
                                          target_size = (224, 224),
                                          batch_size = 624,
                                          classes = ["NORMAL","PNEU
                                          class_mode = 'binary',
                                          seed = 42,
                                          shuffle= False)
```

```
Found 4509 images belonging to 2 classes.
Found 723 images belonging to 2 classes.
Found 624 images belonging to 2 classes.
```

In [15]:
```
#Checking the class label and index match
print(train_gen.class_indices)
print(val_gen.class_indices)
```

```
{'NORMAL': 0, 'PNEUMONIA': 1}
{'NORMAL': 0, 'PNEUMONIA': 1}
```

Iterating through the generated images. X and y are the images and labels respectively. It is a sample equal to the batch size above.

In [16]:
```
X_train, y_train = next(train_gen)
X_val, y_val = next(val_gen)
X_test, y_test = next(test_gen)

print("X_train:",len(X_train),"y_train:",len(y_train))
print("X_val:",len(X_val),"y_val:",len(y_val))
print("X_test:",len(X_test),"y_test:",len(y_test))
```

```
X_train: 300 y_train: 300
X_val: 100 y_val: 100
X_test: 624 y_test: 624
```

Modeling

Custom Functions for Model Evaluation

Visualization Metrics,A.Jang 2023

In [17]:
```
#Create a new directory for saved files
directory = "saved_files"
save_path = os.path.join(cur_dir,directory)
try:
    os.mkdir(save_path)
except OSError as error:
    print(error)
```

```
[Errno 17] File exists: '/Users/deztanyjackson/Documents/0_Flatiron/Phase_4/Pr
oject_4/Image_Classification_P4/saved_files'
```

In [18]:
```
#Using Pickle in a function to save files
def save_file(name,file):
    """ Saving file to a specific directory using pickle"""
    saved_name = os.path.join(save_path,name)
    with open(saved_name, 'wb') as f:
        pickle.dump(file, f)
```

In [19]:
```
#loading Pickle Files
def load_file(name):
    """ Loading file from a specific directory using pickle"""
    saved_name = os.path.join(save_path,name)
    with open(saved_name, 'rb') as f:
        file = pickle.load(f)
    return file
```

In [20]:
```
#Printing metric subplots of training and validation per epoch
def eval_metrics(history):
    """ Plots Train and Validation metrics["acc","loss","recall","precision"]
    from a model's evaluation results."""

    fig, ax = plt.subplots(1,4, figsize = (25,5))
    fig.suptitle('Model Metrics Results')

    metrics = ["loss","recall","acc","precision"]

    for i, metric in enumerate(metrics):

        ax[i].plot(history.history[metric])
        ax[i].plot(history.history["val_" + metric])
        ax[i].set_title('Model {}'.format(metric))
        ax[i].set_ylabel(metric)
        ax[i].set_xlabel("Epochs")
        ax[i].legend(['train', 'val'])
```

Predictor Algorithm with Shuffle,by Jakob, 2021

In [21]:
```
def predict_results(model,generator):
    """Generates prediction results and true labels from a specific model and s
    This formula works well for shuffled generated data."""

    # Create lists for storing the predictions and labels
    predictions = []
```

```
    labels = []

    # Get the total number of labels in generator
    # (i.e. the length of the dataset where the generator generates batches fro
    n = len(generator.labels)

    # Loop over the generator
    for data, label in generator:
        # Make predictions on data using the model. Store the results.
        predictions.extend(model.predict(data).flatten())

        # Store corresponding labels
        labels.extend(label)

        # We have to break out from the generator when we've processed
        # the entire once (otherwise we would end up with duplicates).
        if (len(label) < generator.batch_size) and (len(predictions) == n):
            break
        return labels, predictions #y_true, y_hat_train/y_pred
```

In [55]:
```
def conf_matrix(y_true,y_pred):
    """Prints confusion matrix for the Adopted and Not adopted true and predict
    y_pred = np.round(y_pred,0)
    fig, ax = plt.subplots(1,2, figsize = (15,4), facecolor = "white")

    ax[0].set_title("Confusion Matrix")
    ConfusionMatrixDisplay.from_predictions(y_true, y_pred, normalize=None, dis
    ax[1].set_title("Confusion Matrix: Normalized")
    ConfusionMatrixDisplay.from_predictions(y_true, y_pred, normalize="all", di
    ax[0].grid(False)
    ax[1].grid(False)
```

In [22]:
```
#def conf_matrix(y_true,y_pred, norm_bool):
    """Prints confusion matrix for the Normal and Pneumonia true and predicted
    y_pred = np.round(y_pred,0) #Rounding to the nearest 0 or 1, assumes thresh
    cmatrix= cm(y_true, y_pred, normalize=norm_bool)# Allows normalized matrix
    matrix_disp= ConfusionMatrixDisplay(cmatrix, display_labels= ["NORMAL", "PN
    matrix_disp.plot()

    plt.xlabel('Predictions')
    plt.ylabel('Actual')
    plt.title('Model Confusion Matrix')
    plt.show()
```

In [23]:
```
def eval_report(y_true, y_pred,gen,history):
    """ Displaying the classification report and loss from the model results"""
    y_pred = np.round(y_pred,0) # Rounding to the nearest 0 or 1
    #Display Precision, Recall, F1 and Accuracy
    print(classification_report(y_true, y_pred, digits = 3))
    print('\n')

    #Display Model Metrics
    loss, _,_,_ = history.model.evaluate(gen)

    print(f'Loss: {np.round(loss,4)}')
```

In [24]:
```
# set up the class weights using y_train
def calc_weight(labels):
    """ Setting the weights and pairing it with the classes for the class weigh
    Labels can also be thought of as y_train"""
    classes = [0,1]
    weights = compute_class_weight(class_weight='balanced', classes=classes, y=
    class_weights = dict(zip(classes, weights))
    return class_weights
```

The following is the outline of each model iteration:

- Sequence
- Compile
- Summary
- Fit
- Predict
- Conf_matrix
- Eval report
- Model Evaluation

Each iteration will have slight changes for improvement. They will be described more at the General descriptions of specific iterations.

The main metrics focused on are:

- Recall, F1 Score, Loss (Primary)
- and Accuracy (Secondary)

Because the goal is to minimized False Negatives of Pneumonia (predicting no Pneumonia, while there is). The True Positives will be looked at secondary.

There are a total of 10* model iterations with a Final Model. The model iterations are trained on the training data and tested on validation data. Both the training and validation data metrics will be displayed.

The Final Model will be chosen based on the main metrics. This final model will be used with the test data.

Main Hyperparameters & Parameters

NN Hyperparameters, Rukshan P, 2022

- Model
  - layers:
    - Conv2D:
    - MaxPooling2D:
    - Dense output:

    - filters:
    - activation:
        - Relu
        - Sigmoid:
    - dropout: Regularization technique that supports overfitting. Smaller network allows parameters to update more.
- Model Compile
    - loss:
        - binary_crossentropy
    - optimizer:
        - adam
    - learning rate:
- Fit
    - batchsize:
    - epochs:
    - early stopping: After convergence, stopping the training process after the validation and training error diverge
    - class weights:

Baseline Modeling (1)

Description of the main model hyperparameters and data parameters:

- Model
    - layers: 3 Conv2D each w/ MaxPooling2D, 2 Dense output layers
    - filters: 32,64,96 for Conv2D, 64,1 Dense layer
    - activation: Relu for all except Sigmoid for output layer
- Model Compile
    - loss: binary_crossentropy
    - optimizer:adam
    - learning rate: .001 (Default)
    - metrics: Accuracy,Recall,Precision
- Fit
    - data: train_gen, val_gen
    - train batchsize: 64
    - epochs: 10
- Additional Comments
    - Conv2D model is best for images
    - MaxPooling 2D is a downsampling strategy CNNs

In [25]:
```
# Baseline Model Stack
baseline = models.Sequential([
Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
       padding = 'same', input_shape=(224,224,3)),
```

```
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),
Flatten(),
Dense(units=64, activation='relu'),
Dense(units=1, activation='sigmoid'),
])

#Compile Model
baseline.compile(loss='binary_crossentropy',
                 optimizer='adam',
                 metrics=["acc","Recall","Precision"])
```

```
Metal device set to: Apple M2 Max
```

```
2023-04-06 21:16:01.496135: I tensorflow/core/common_runtime/pluggable_device/
pluggable_device_factory.cc:305] Could not identify NUMA node of platform GPU
ID 0, defaulting to 0. Your kernel may not have been built with NUMA support.
2023-04-06 21:16:01.496292: I tensorflow/core/common_runtime/pluggable_device/
pluggable_device_factory.cc:271] Created TensorFlow device (/job:localhost/rep
lica:0/task:0/device:GPU:0 with 0 MB memory) -> physical PluggableDevice (devi
ce: 0, name: METAL, pci bus id: <undefined>)
```

The model summary provides a description of the sequential Layers, Output Shapes and Learning Parameters.

In [26]:
```
#Display model summary ()
baseline.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896

 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 56, 56, 96)        55392

 max_pooling2d_2 (MaxPooling  (None, 28, 28, 96)       0
 2D)

 flatten (Flatten)           (None, 75264)             0

 dense (Dense)               (None, 64)                4816960

 dense_1 (Dense)             (None, 1)                 65

=================================================================
Total params: 4,891,809
Trainable params: 4,891,809
Non-trainable params: 0
_________________________________________________________________
```

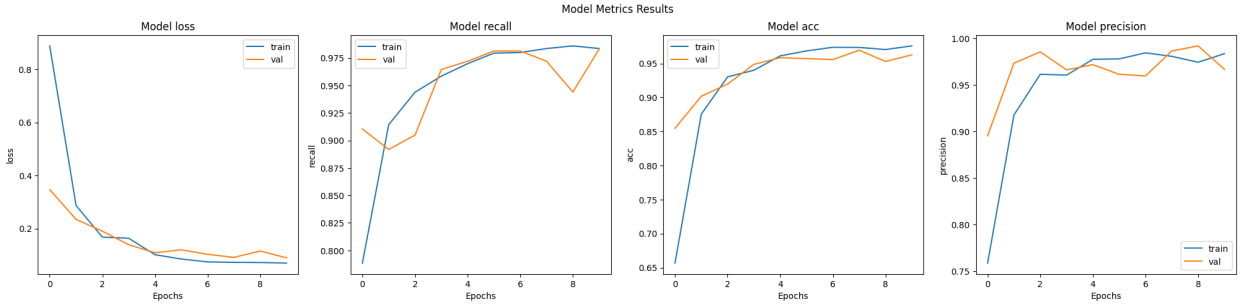
Returning a model object that contains model evaluation metrics, and hyperparameter attributes

In [27]:
```
#Fiting the data
history = baseline.fit(train_gen,
                       batch_size= 50,
                       epochs=10,
                       verbose = 0,
                       validation_data=val_gen)
```

```
2023-04-06 21:16:03.287055: W tensorflow/core/platform/profile_utils/cpu_util
s.cc:128] Failed to get CPU frequency: 0 Hz
2023-04-06 21:16:03.589461: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-06 21:16:29.991321: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

Baseline Metrics

In [28]:
```
#Plotting chosen Train and Validation Metrics
eval_metrics(history)
```

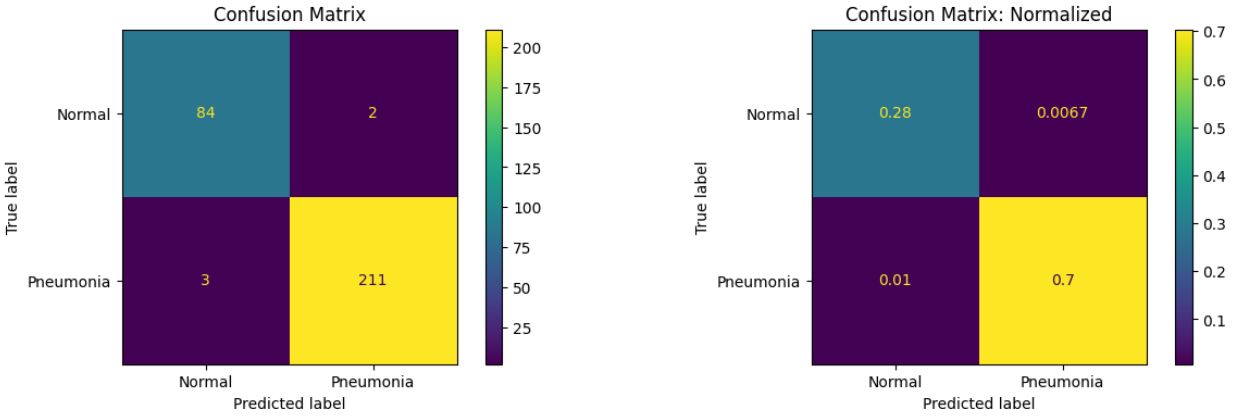


In [86]:
```
#Returns Train and Validation labels and predictions
train_labels, train_predictions = predict_results(baseline,train_gen)
val_labels, val_predictions = predict_results(baseline,val_gen)
```

```
10/10 [==============================] - 0s 42ms/step
4/4 [==============================] - 0s 47ms/step
```

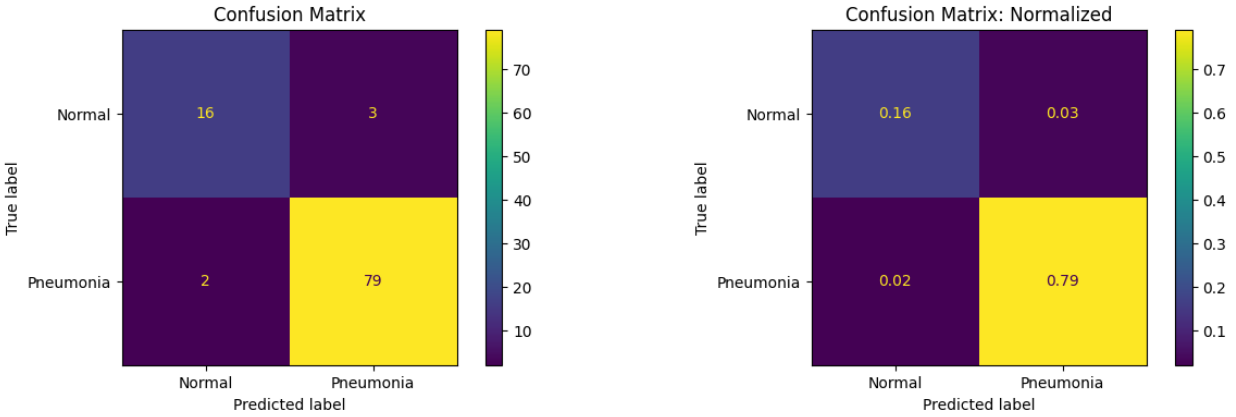
In [87]:
```
#Printing regular and normalized Confusion Matrices
print("Training Confusion Matrices:\n")
conf_matrix(train_labels,train_predictions)
```

```
Training Confusion Matrices:
```



In [88]:
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```



In [31]:
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      1.000     0.907     0.951        75
         1.0      0.970     1.000     0.985       225

    accuracy                         0.977       300
   macro avg      0.985     0.953     0.968       300
weighted avg      0.977     0.977     0.976       300


16/16 [==============================] - 21s 1s/step - loss: 0.0566 - acc: 0.9
809 - recall: 0.9919 - precision: 0.9825
Loss: 0.0566
```

In [32]:
```
#Validation Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      1.000     0.929     0.963        28
         1.0      0.973     1.000     0.986        72

    accuracy                         0.980       100
   macro avg      0.986     0.964     0.975       100
weighted avg      0.981     0.980     0.980       100


8/8 [==============================] - 3s 412ms/step - loss: 0.0897 - acc: 0.9
627 - recall: 0.9832 - precision: 0.9670
Loss: 0.0897
```

Baseline (1) Model Evaluation

- **Model Plots: The baseline model is actually really good. There is slight variation and divergence but that may be due to the limited epochs.**
- **Condusion Matrix: With the current batch size and imbalanced data, the model predicted less than 2% False Negatives. This is with the validation model having 40% Normal images.**
- **Report: The loss in the validation model is only slightly worse than the training model.**
- **The is a solid model. To help with the lingering data balance, we will work to improve that.**

Baseline Model (2) with Class Weights

Change Description:

- **Adding class weights to see it helps with class imbalance.**

Description of the model hyperparameters and data parameters:

- **Model**
  - **layers: 3 Conv2D each w/ MaxPooling2D, 2 Dense output layers**
  - **filters: 32,64,96 for Conv2D, 64,1 Dense layer**
  - **activation: Relu for all except Sigmoid for output layer**
- **Model Compile**
  - **loss: binary_crossentropy**
  - **optimizer:adam**
  - **learning rate: .001 (Default)**
  - **metrics: Accuracy,Recall,Precision**
- **Fit**
  - **data: train_gen, val_gen**
  - **train batchsize: 64**
  - **epochs: 10**
  - **Added class weights: {0: 1.6, 1: 0.7272727272727273}**
- **Additional Comments**
  - **Conv2D model is best for images**
  - **MaxPooling 2D is a downsampling strategy CNNs**

In [33]:
```
# Baseline Model Stack
baseline2 = models.Sequential([
Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
       padding = 'same', input_shape=(224,224,3)),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),
Flatten(),
Dense(units=64, activation='relu'),
Dense(units=1, activation='sigmoid'),
])

#Compile Model
baseline2.compile(loss='binary_crossentropy',
                  optimizer='adam',
                  metrics=["acc","Recall","Precision"])
```

Calculating class weights help to balance the dataset by weighting respecively.

In [34]:
```
#Calculating class weights for imbalance
class_weights = calc_weight(y_train)
class_weights
```

Out[34]:
```
{0: 1.8987341772151898, 1: 0.6787330316742082}
```

In [35]:
```
#Fitting the data on the baseline model with class weight
history2 = baseline2.fit(train_gen,
                         epochs=10,
                         verbose = 0,
                         class_weight=class_weights,
                         validation_data=val_gen)
```

```
2023-04-06 21:21:14.247934: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-06 21:21:37.535263: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [36]:
```
#Baseline2 Summary
baseline2.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 224, 224, 32)      896

 max_pooling2d_3 (MaxPooling  (None, 112, 112, 32)     0
 2D)

 conv2d_4 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_4 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 conv2d_5 (Conv2D)           (None, 56, 56, 96)        55392

 max_pooling2d_5 (MaxPooling  (None, 28, 28, 96)       0
 2D)

 flatten_1 (Flatten)         (None, 75264)             0

 dense_2 (Dense)             (None, 64)                4816960

 dense_3 (Dense)             (None, 1)                 65

=================================================================
Total params: 4,891,809
Trainable params: 4,891,809
Non-trainable params: 0
_________________________________________________________________
```

In [37]:
```
print("Basline Evaluation Metrics w/ Class Weights :\n")
eval_metrics(history2)
```

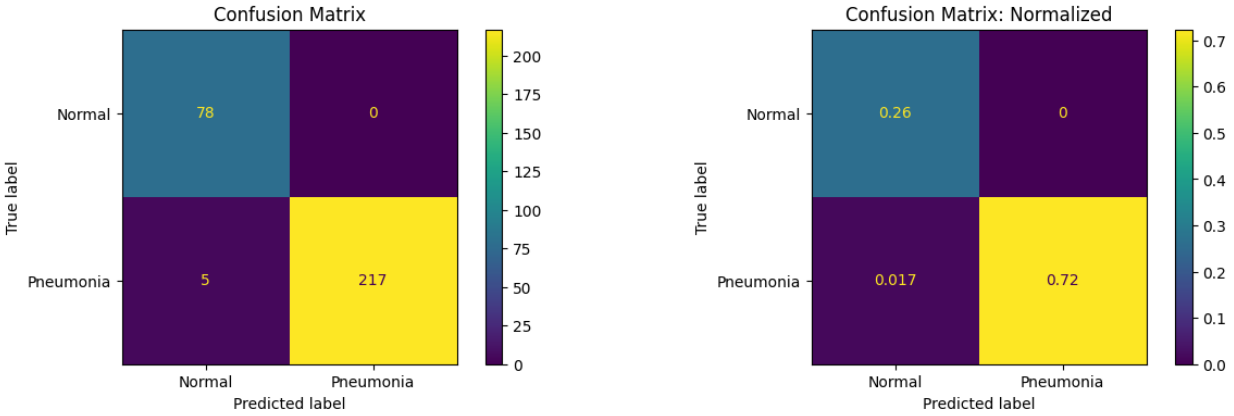
```
Basline Evaluation Metrics w/ Class Weights :
```

In [89]:
```
#Returns Train and Validation labels and predictions
train_labels, train_predictions = predict_results(baseline2,train_gen)
val_labels, val_predictions = predict_results(baseline2,val_gen)
```

```
10/10 [==============================] - 0s 52ms/step
1/1 [==============================] - 0s 178ms/step
```

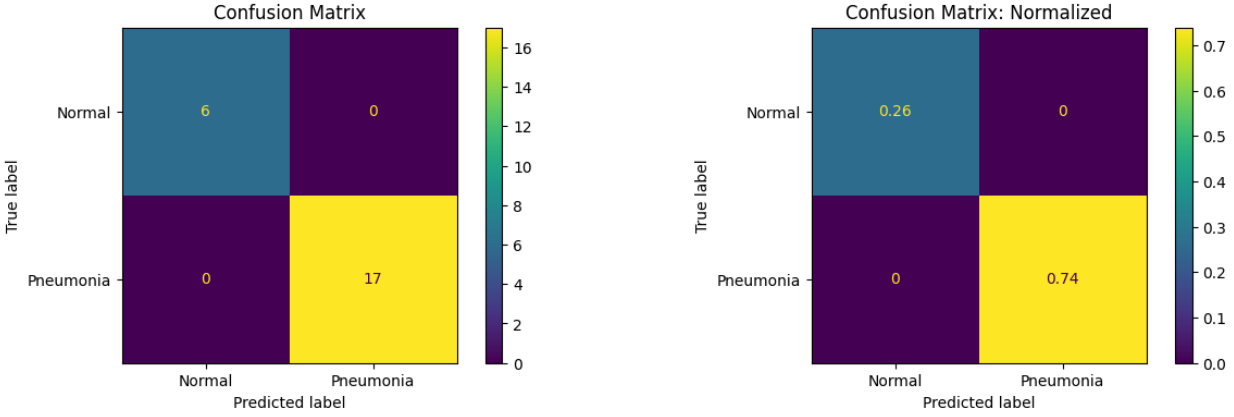
In [90]:
```
#Printing regular and normalized Confusion Matrices
print("Training Confusion Matrices:\n")
conf_matrix(train_labels,train_predictions)
```

```
Training Confusion Matrices:
```



In [91]:
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```



In [40]:
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history2)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.935     1.000     0.966        86
         1.0      1.000     0.972     0.986       214

    accuracy                          0.980       300
   macro avg      0.967     0.986     0.976       300
weighted avg      0.981     0.980     0.980       300
```

```
16/16 [==============================] - 22s 1s/step - loss: 0.0550 - acc: 0.9
794 - recall: 0.9734 - precision: 0.9988
Loss: 0.055
```

In [41]:
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history2)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.935     1.000     0.967        29
         1.0      1.000     0.972     0.986        71

    accuracy                          0.980       100
   macro avg      0.968     0.986     0.976       100
weighted avg      0.981     0.980     0.980       100
```

```
8/8 [==============================] - 3s 410ms/step - loss: 0.0912 - acc: 0.9
710 - recall: 0.9664 - precision: 0.9942
Loss: 0.0912
```

Baseline (2) Model Evaluation

- **Model Plots: Slight improvement with the class weight. The recall and accuracy do not diverge as much.**
- **Confusion Matrix: There are no FN with the validation set and no False Positives. The validation set had 25% Normal images this time.**
- **Report: The loss in the validaiton set increased from the previous model, but slightly higher than the current training set**
- **Ths model wasn't bad either. The loss increased but the amount of FN decreased.. The validation and training set are getting comparable results. The next models will add and tweek some hyperparameters to see if we can improve.**

Model Iteration 3

Change Description:

- **Increase Epochs, Adding EarlyStop and Dropout between Dense layers**

Description of the model hyperparameters and data parameters:

- Model
  - layers: 3 Conv2D each w/ MaxPooling2D, 2 Dense output layers
  - filters: 32,64,96 for Conv2D, 64,1 Dense layer
  - activation: Relu for all except Sigmoid for output layer
  - dropout: .3
- Model Compile
  - loss: binary_crossentropy
  - optimizer:adam
  - learning rate: .001 (Default)
  - metrics: Accuracy,Recall,Precision
- Fit
  - data: train_gen, val_gen
  - train batchsize: 64
  - epochs: 30
  - early stopping : True
  - class weights: {0: 1.6, 1: 0.7272727272727273}
- Additional Comments
  - Conv2D model is best for images
  - MaxPooling 2D is a downsampling strategy CNNs

In [42]:
```
#Model 3 Sequential Layers
model_3= models.Sequential([
Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
       padding = 'same', input_shape=(224,224,3)),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),
Flatten(),
Dense(units=64, activation='relu'),
Dropout(0.3),
Dense(units=1, activation='sigmoid'),
])
#Compile model
model_3.compile(loss='binary_crossentropy',
                optimizer='adam',
```

```
                  metrics=["acc","Recall","Precision"]
                   )
```

Early stopping allows the execution of the model fitting to stop when the model has stopped improving.

In [43]:
```
#Adding early stopping
es = EarlyStopping(monitor='val_loss', mode='min', verbose=0, patience=2)
#Calculating class weights for imbalance
```

In [44]:
```
history3 = model_3.fit(train_gen,
                       epochs=30,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es)
```

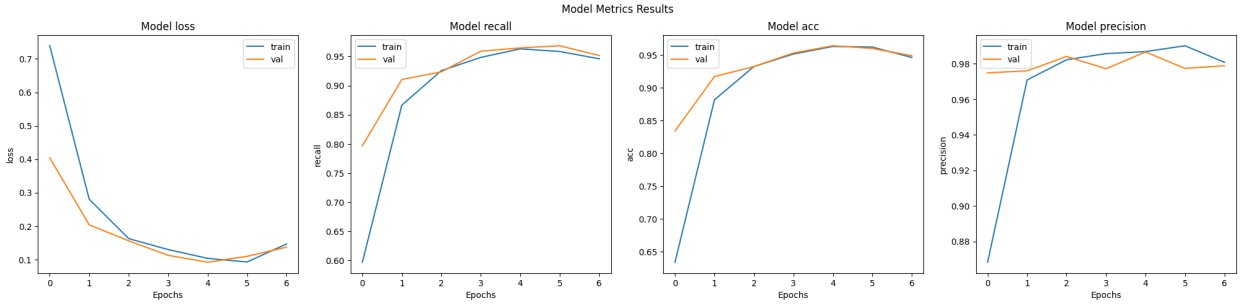
```
2023-04-06 21:26:20.211676: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-06 21:26:43.017405: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [45]:
```
model_3.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_6 (Conv2D)           (None, 224, 224, 32)      896

 max_pooling2d_6 (MaxPooling  (None, 112, 112, 32)     0
 2D)

 conv2d_7 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_7 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 conv2d_8 (Conv2D)           (None, 56, 56, 96)        55392

 max_pooling2d_8 (MaxPooling  (None, 28, 28, 96)       0
 2D)

 flatten_2 (Flatten)         (None, 75264)             0

 dense_4 (Dense)             (None, 64)                4816960

 dropout (Dropout)           (None, 64)                0

 dense_5 (Dense)             (None, 1)                 65

=================================================================
Total params: 4,891,809
Trainable params: 4,891,809
Non-trainable params: 0
_________________________________________________________________
```

In [46]:
```
print("Model 3: Dropout, EarlyStopping and 30 Epochs :\n")
eval_metrics(history3)
```

Model 3: Dropout, EarlyStopping and 30 Epochs :

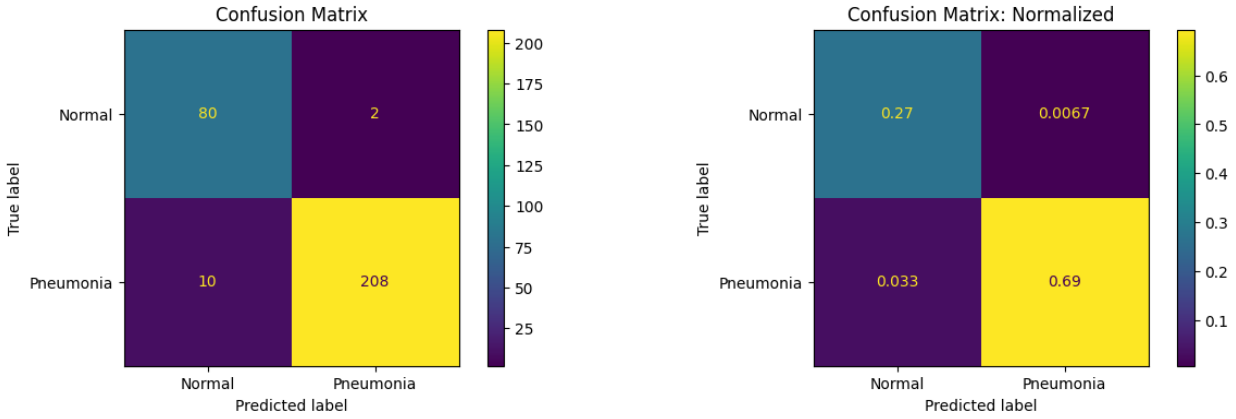


In [92]:
```
train_labels, train_predictions = predict_results(model_3,train_gen)
val_labels, val_predictions = predict_results(model_3,val_gen)
```

```
10/10 [==============================] - 1s 72ms/step
4/4 [==============================] - 0s 46ms/step
```

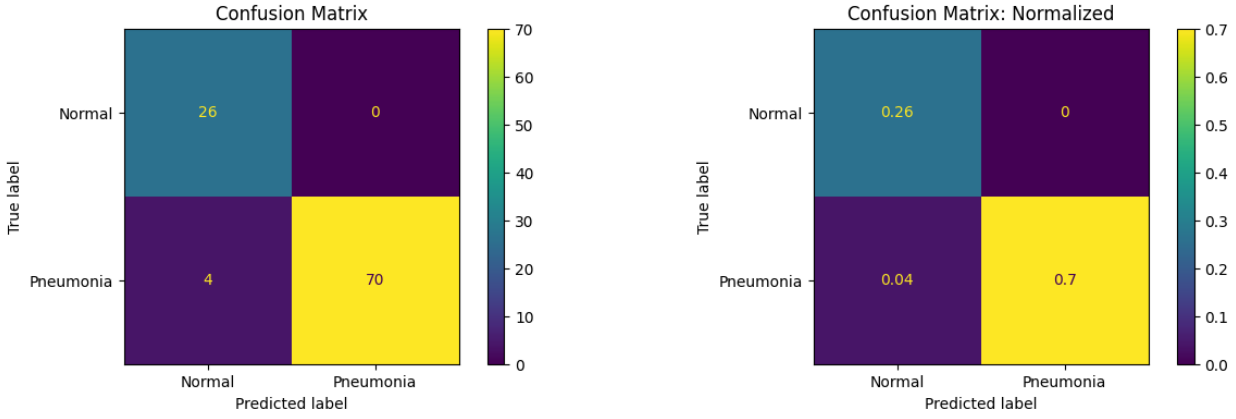
In [93]:
```
#Printing regular and normalized Confusion Matrices
print("Training Confusion Matrices:\n")
conf_matrix(train_labels,train_predictions)
```

Training Confusion Matrices:



In [94]:
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

Validation Confusion Matrices:



In [49]:
```
#Training Classification Report
print("Training Classificaiton Report:\n")
```

```
eval_report(train_labels,train_predictions,train_gen,history3)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.929     0.987     0.957        79
         1.0      0.995     0.973     0.984       221

    accuracy                         0.977       300
   macro avg      0.962     0.980     0.971       300
weighted avg      0.978     0.977     0.977       300


16/16 [==============================] - 23s 1s/step - loss: 0.1169 - acc: 0.9
610 - recall: 0.9528 - precision: 0.9944
Loss: 0.1169
```

- Model Plots: The validation data seems to be fitting well. The model isn't overfit.
- Confusion Matrix: Both matrices have less than 4% FN.
- Report: The validation loss is slightly more than the training model. Recall and F1 are better than last model, but not as good as Baseline 1.
- It isn't clear what to improve, optimizing this model with gridsearch will be the next step.

In [50]:
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history3)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.771     0.900     0.831        30
         1.0      0.954     0.886     0.919        70

    accuracy                         0.890       100
   macro avg      0.863     0.893     0.875       100
weighted avg      0.899     0.890     0.892       100


8/8 [==============================] - 4s 425ms/step - loss: 0.1369 - acc: 0.9
488 - recall: 0.9515 - precision: 0.9789
Loss: 0.1369
```

Model 3 Evaluation

Model Iterations 4-7

These models will use Keras wrappers amd GridsearchCV to optimize various hyperparameters in the models. The best parameters will be applied along to the model.

Gridsearch Keras, Jason Brownlee, 2022
Gridsearch NN,AARYN DHOR, 2020
Hyperparamters, Rukshan Pramoditha, 2022

Model 4

Using gridsearch and keras wrappers to do some hyperparameter optimization

- batch_size = [32,64,96,128]
- epochs = [30,45,60,90]

Description of the model hyperparameters and data parameters:

- Model
  - layers: 3 Conv2D each w/ MaxPooling2D, 2 Dense output layers
  - filters: 32,64,96 for Conv2D, 64,1 Dense layer
  - activation: Relu for all except Sigmoid for output layer
  - dropout: .3
- Model Compile
  - loss: binary_crossentropy
  - optimizer:adam
  - learning rate: .001 (Default)
  - metrics: Accuracy,Recall,Precision
- Fit
  - data: train_gen, val_gen
  - train batchsize: 128
  - epochs: 90
  - early stopping : True
  - class weights: {0: 1.6, 1: 0.7272727272727273}
- Additional Comments
  - Conv2D model is best for images
  - MaxPooling 2D is a downsampling strategy CNNs

In [51]:
```
# Creating a function to supply to the Keras wrapper
def buildModel():

    model= models.Sequential([
    Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
           padding = 'same', input_shape=(224,224,3)),
    MaxPooling2D(pool_size=(2, 2), strides=2),
```

```
    Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Flatten(),
    Dense(units=64, activation='relu'),
    Dropout(0.3),
    Dense(units=1, activation='sigmoid'),])

    model.compile(loss='binary_crossentropy',
                optimizer="adam",
                metrics=["acc","Recall","Precision"]
                 )
    return model
```

Choosing hyperparameters range to optimize. Being mindful of hardware and software resources.

In [52]:
```
#Choosing Paramers for Optimization
batch_size = [32,64,96,128]
epochs = [30,45,60,90]
parameters = dict(batch_size=batch_size, epochs=epochs )
```

In [53]:
```
#Early Stopping
es = EarlyStopping(monitor='val_loss', mode='min', verbose=0, patience=2)
```

GridSearchCV with CNN, Aaryn Dhore, 2020
GridSearch for Deep Learning, J. Brownlee, 2022

KerasClassifier creates a classifier from the Model Sequence build model to use as an estimator. The created parameters will be used as the parameters in the Gridsearch to help optimize.

In [62]:
```
#Creating a wrapped classifier
classifier = KerasClassifier(build_fn=buildModel, class_weight = class_weights,

#Gridsearch Creating
grid_search = GridSearchCV(estimator = classifier,
                           param_grid = parameters,
                           refit = "Acc",
                           scoring = "accuracy",
                           cv = 3,return_train_score=True)
#Fitting Gridsearch
gs = grid_search.fit(X_train, y_train, verbose = 0)
```

```
2023-04-06 21:57:41.380497: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>.....] - ETA: 0s
2023-04-06 21:57:55.201560: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 52ms/step
7/7 [==============================] - 0s 41ms/step
```

```
2023-04-06 21:57:56.104875: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>........] - ETA: 0s
2023-04-06 21:58:09.982518: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 48ms/step
7/7 [==============================] - 0s 42ms/step
2023-04-06 21:58:11.145874: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 21:58:25.090690: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 55ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 21:58:26.125023: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 21:58:45.859034: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 21:58:46.994018: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 21:59:07.052539: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 70ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 21:59:08.533484: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>........] - ETA: 0s
2023-04-06 21:59:28.373334: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 72ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 21:59:29.425392: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 21:59:54.413945: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 63ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 21:59:55.418788: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:00:19.866689: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 62ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:00:20.869924: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>........] - ETA: 0s
2023-04-06 22:00:45.830346: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

```
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-06 22:00:47.217995: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:01:22.631502: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:01:23.672929: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:01:58.751156: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 65ms/step
7/7 [==============================] - 0s 48ms/step
2023-04-06 22:01:59.794719: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:02:34.658886: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:02:35.700234: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:02:48.186948: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 73ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:02:49.409485: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>........] - ETA: 0s
2023-04-06 22:03:01.726045: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:03:03.235367: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:03:15.216962: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 58ms/step
7/7 [==============================] - 0s 54ms/step
2023-04-06 22:03:16.257556: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:03:32.574939: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 54ms/step
7/7 [==============================] - 0s 52ms/step
2023-04-06 22:03:33.585585: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
```

```
2023-04-06 22:03:49.687466: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 63ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 22:03:50.746584: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>........] - ETA: 0s
2023-04-06 22:04:06.925891: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 60ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:04:07.920472: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:04:28.087675: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 62ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:04:29.572100: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:04:49.605725: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 69ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:04:50.648270: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:05:10.975152: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 69ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:05:12.044148: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:05:39.798074: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 65ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 22:05:40.827735: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.......................] - ETA: 0s
2023-04-06 22:06:08.298435: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:06:09.355733: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [=====================>........] - ETA: 0s
2023-04-06 22:06:37.519775: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 68ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:06:38.971313: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

```
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:06:50.346422: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 48ms/step
2023-04-06 22:06:51.466702: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:07:03.004398: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 72ms/step
7/7 [==============================] - 0s 45ms/step
2023-04-06 22:07:04.035835: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:07:15.418741: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:07:16.488365: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:07:31.767807: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:07:32.817079: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:07:47.434806: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 62ms/step
7/7 [==============================] - 0s 44ms/step
2023-04-06 22:07:49.138868: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:08:03.676344: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:08:04.671206: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:08:23.754617: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 72ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-06 22:08:24.849202: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:08:43.917250: I tensorflow/core/grappler/optimizers/custom_graph_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 70ms/step
7/7 [==============================] - 0s 45ms/step
```

```
2023-04-06 22:08:44.973368: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:09:03.839222: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:09:05.358090: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:09:31.339229: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 56ms/step
7/7 [==============================] - 0s 52ms/step
2023-04-06 22:09:32.654355: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:09:59.617658: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 48ms/step
2023-04-06 22:10:00.651738: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:10:27.337983: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 69ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:10:28.438839: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:10:39.592533: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 68ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:10:40.680337: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:10:51.155730: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 69ms/step
7/7 [==============================] - 0s 48ms/step
2023-04-06 22:10:52.789377: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:11:03.153512: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 63ms/step
7/7 [==============================] - 0s 52ms/step
2023-04-06 22:11:04.249885: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:11:18.264514: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

```
4/4 [==============================] - 0s 71ms/step
7/7 [==============================] - 0s 46ms/step
2023-04-06 22:11:19.313782: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:11:32.802840: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 69ms/step
7/7 [==============================] - 0s 48ms/step
2023-04-06 22:11:33.856991: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:11:48.024190: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 65ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-06 22:11:49.105330: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:12:06.841931: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 1s 97ms/step
2023-04-06 22:12:08.758243: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:12:26.149194: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 65ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:12:27.193363: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:12:44.592625: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 73ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:12:45.705579: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-06 22:13:10.801305: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 73ms/step
7/7 [==============================] - 0s 47ms/step
2023-04-06 22:13:11.903707: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
3/4 [====================>.........] - ETA: 0s
2023-04-06 22:13:36.307241: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 62ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 22:13:37.908110: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
```

```
2023-04-06 22:14:02.875050: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-06 22:14:03.946659: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [63]:
```
#Printing best score and parameters
best_batch = gs.best_params_["batch_size"]
best_epoch = gs.best_params_["epochs"]
print("Best: %f using %s" % (gs.best_score_, gs.best_params_))
```

```
Best: 0.923333 using {'batch_size': 128, 'epochs': 60}
```

Plotting Model 4 with best parameters

In [64]:
```
# Model 4
model_4= models.Sequential([
Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
       padding = 'same', input_shape=(224,224,3)),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
       padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Flatten(),
Dense(units=64, activation='relu'),
Dropout(0.3),
Dense(units=1, activation='sigmoid'),
])

model_4.compile(loss='binary_crossentropy',
                optimizer='adam',
                metrics=["acc","Recall","Precision"]
                 )
```

In [65]:
```
#Fitting model with best parameters
history4 = model_4.fit(train_gen,batch_size= best_batch,
                       epochs=best_epoch,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es) #stopping early because of time resources
```

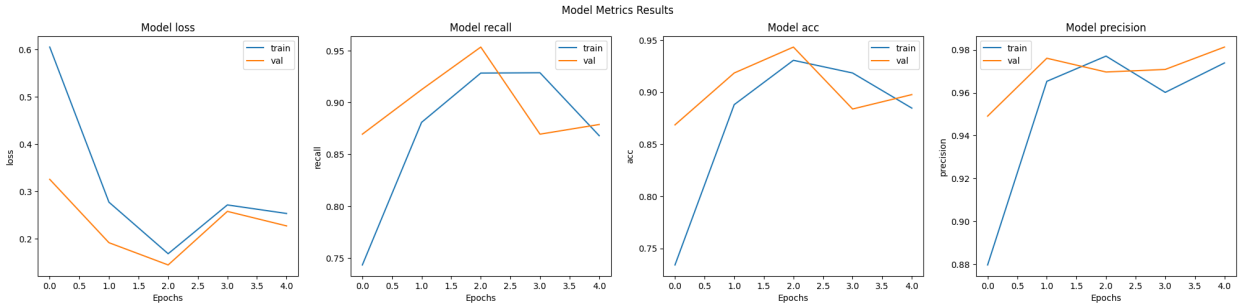
```
2023-04-06 22:14:31.469186: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-06 22:14:56.337781: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [66]:
```
#Model Summary
model_4.summary()
```

```
Model: "sequential_120"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_360 (Conv2D)         (None, 224, 224, 32)      896

 max_pooling2d_360 (MaxPooli  (None, 112, 112, 32)     0
 ng2D)

 conv2d_361 (Conv2D)         (None, 112, 112, 64)      18496

 max_pooling2d_361 (MaxPooli  (None, 56, 56, 64)       0
 ng2D)

 conv2d_362 (Conv2D)         (None, 56, 56, 96)        55392

 max_pooling2d_362 (MaxPooli  (None, 28, 28, 96)       0
 ng2D)

 flatten_120 (Flatten)       (None, 75264)             0

 dense_240 (Dense)           (None, 64)                4816960

 dropout_118 (Dropout)       (None, 64)                0

 dense_241 (Dense)           (None, 1)                 65

=================================================================
Total params: 4,891,809
Trainable params: 4,891,809
Non-trainable params: 0
_________________________________________________________________
```

In [67]:
```
print("Basline Evaluation Metrics Best Metrics :\n")
eval_metrics(history4)
```

```
Basline Evaluation Metrics Best Metrics :
```

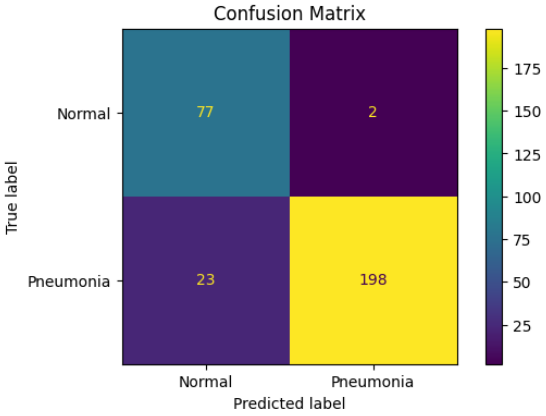


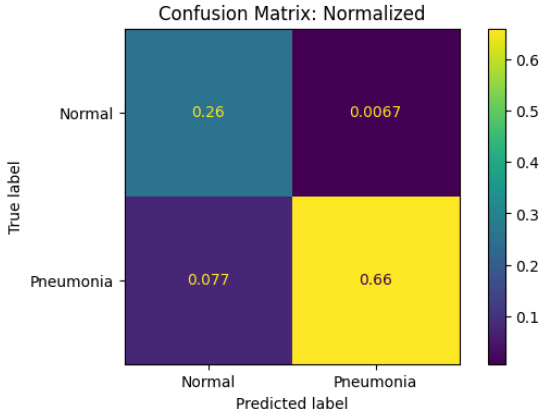
In [95]:
```
train_labels, train_predictions = predict_results(model_4,train_gen)
val_labels, val_predictions = predict_results(model_4,val_gen)
```

```
10/10 [==============================] - 0s 51ms/step
4/4 [==============================] - 0s 118ms/step
```

In [96]:
```
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

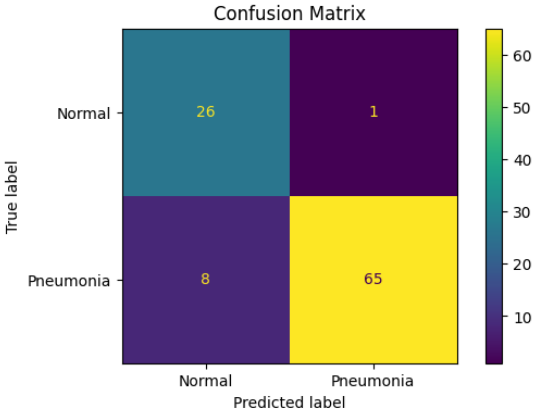
```
Train Confusion Matrices:
```

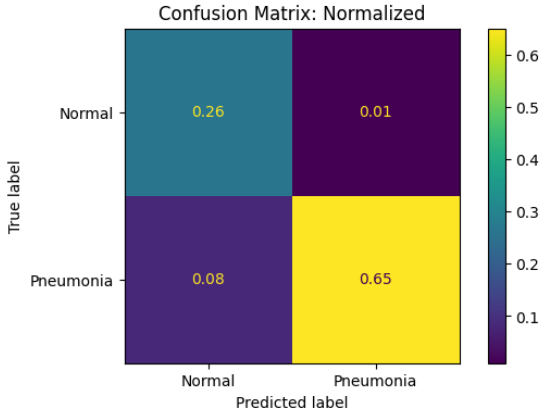




In [97]:
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```





In [98]:
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history4)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.770     0.975     0.860        79
         1.0      0.990     0.896     0.941       221

    accuracy                         0.917       300
   macro avg      0.880     0.935     0.900       300
weighted avg      0.932     0.917     0.919       300



16/16 [==============================] - 21s 1s/step - loss: 0.1936 - acc: 0.9
162 - recall: 0.8957 - precision: 0.9904
Loss: 0.1936
```

In [73]:
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history4)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.724     1.000     0.840        21
         1.0      1.000     0.899     0.947        79

    accuracy                          0.920       100
   macro avg      0.862     0.949     0.893       100
weighted avg      0.942     0.920     0.924       100


8/8 [==============================] - 4s 404ms/step - loss: 0.2267 - acc: 0.8
976 - recall: 0.8787 - precision: 0.9813
Loss: 0.2267
```

Model 4 Evaluation

- **Model Plots: The validation metrics are slightly worse than the training metrics, but not far off.**
- **Confusion Matrix: The validation model has 8% FN and less than 1% for both models.**
- **Report: The Validation Normal Recall scores are the 100!**
- **Overall the Recall and Loss are still acceptable**

Model 5

Using gridsearch and keras wrappers to do some hyperparameter optimization

- **epochs = [45,65,90]**
- **optimizer = ['adam', "SGD"]**
- **dropout_rate = [.3,.5]**
- **Will add Conv2D and a Dense layer for more complexity.**

Description of the model hyperparameters and data parameters:

- **Model***
  - **layers: 4 Conv2D each w/ MaxPooling2D, 3 Dense output layers**
  - **filters: 32,64,96,128 for Conv2D, 128, 64,1 Dense layer**
  - **activation: Relu for all except Sigmoid for output layer**
  - **dropout: .5 between the dense output layers**
- **Model Compile**
  - **loss: binary_crossentropy**
  - **optimizer:adam**
  - **learning rate: .001 (Default)**

    - metrics: Accuracy,Recall,Precision
- Fit
    - data: train_gen, val_gen
    - train batchsize: 128
    - epochs: 90
    - early stopping : True
    - class weights: {0: 1.6, 1: 0.7272727272727273}
- Additional Comments
    - Conv2D model is best for images
    - MaxPooling 2D is a downsampling strategy CNNs

In [74]:
```
#Build model for classifier with parameters in signature
def buildModel(dropout_rate, optimizer = "adam"):

    model= models.Sequential([
    Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
           padding = 'same', input_shape=(224,224,3)),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=128, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Flatten(),

    Dense(units=128, activation='relu',),
    Dropout(dropout_rate),
    Dense(units=64, activation='relu'),
    Dropout(dropout_rate),
    Dense(units=1, activation='sigmoid'),
    ])

    model.compile(loss='binary_crossentropy',
                  optimizer= optimizer,
                  metrics=["acc","Recall","Precision"]
                   )
    return model
```

In [75]:
```
#Hyperparameters for GridsearchCV
epochs = [45,65,90]
optimizer = ['adam', "SGD"]
dropout_rate = [.3,.5]

#Creating a dictionary to use as parameters
parameters_2 = dict(optimizer=optimizer, dropout_rate=dropout_rate,  epochs=epo
parameters_2
```

Out[75]:
```
{'optimizer': ['adam', 'SGD'],
 'dropout_rate': [0.3, 0.5],
 'epochs': [45, 65, 90]}
```

In [76]:
```
#Creating a wrapped classifier
classifier = KerasClassifier(build_fn=buildModel, batch_size = 128, class_weigh

#Gridsearch Creation
grid_search = GridSearchCV(estimator = classifier,
                           param_grid = parameters_2,
                           refit = "Acc",
                           scoring = "accuracy",
                           cv = 3,return_train_score=True)
#Gridsearch Fitting
gs_2 = grid_search.fit(X_train, y_train, verbose = 0)
```

```
2023-04-07 00:00:25.977642: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:00:43.496738: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 64ms/step
7/7 [==============================] - 0s 35ms/step
2023-04-07 00:00:45.301630: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:00:59.866645: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 58ms/step
7/7 [==============================] - 0s 31ms/step
2023-04-07 00:01:00.883912: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:01:15.531836: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 59ms/step
7/7 [==============================] - 0s 35ms/step
2023-04-07 00:01:16.618261: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:01:30.943724: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 60ms/step
7/7 [==============================] - 0s 37ms/step
2023-04-07 00:01:31.943845: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:01:46.986570: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 72ms/step
7/7 [==============================] - 0s 35ms/step
2023-04-07 00:01:48.504050: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:02:03.206688: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

```
4/4 [==============================] - 1s 158ms/step
7/7 [==============================] - 0s 39ms/step
2023-04-07 00:02:04.643604: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:02:25.448853: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 39ms/step
2023-04-07 00:02:26.639295: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:02:47.522984: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 74ms/step
7/7 [==============================] - 0s 35ms/step
2023-04-07 00:02:48.672276: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:03:09.306651: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 54ms/step
7/7 [==============================] - 0s 34ms/step
2023-04-07 00:03:10.387698: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:03:31.461986: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 75ms/step
7/7 [==============================] - 0s 34ms/step
2023-04-07 00:03:33.456310: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:03:53.691911: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 61ms/step
7/7 [==============================] - 0s 33ms/step
2023-04-07 00:03:54.687671: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:04:15.052165: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 79ms/step
7/7 [==============================] - 0s 37ms/step
2023-04-07 00:04:16.265813: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:04:43.529429: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 67ms/step
7/7 [==============================] - 0s 35ms/step
2023-04-07 00:04:44.658488: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:05:12.248194: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 57ms/step
7/7 [==============================] - 0s 39ms/step
```

```
2023-04-07 00:05:14.051663: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:05:41.446408: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 127ms/step
7/7 [==============================] - 1s 104ms/step
2023-04-07 00:05:43.097610: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:06:10.243083: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 56ms/step
7/7 [==============================] - 0s 39ms/step
2023-04-07 00:06:11.283523: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:06:39.302616: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 134ms/step
7/7 [==============================] - 0s 34ms/step
2023-04-07 00:06:40.546010: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:07:08.134392: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 130ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-07 00:07:09.518089: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:07:27.457465: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 63ms/step
7/7 [==============================] - 0s 62ms/step
2023-04-07 00:07:29.549766: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:07:47.173232: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 70ms/step
7/7 [==============================] - 0s 33ms/step
2023-04-07 00:07:48.364093: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:08:06.472645: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 127ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-07 00:08:08.225631: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
1/4 [======>.....................] - ETA: 0s
2023-04-07 00:08:25.824219: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 0s 55ms/step
7/7 [==============================] - 0s 45ms/step
```

```
2023-04-07 00:08:26.890849: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:08:46.057738: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 103ms/step
7/7 [==============================] - 0s 50ms/step
2023-04-07 00:08:47.455723: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:09:06.492196: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 88ms/step
7/7 [==============================] - 0s 58ms/step
2023-04-07 00:09:08.626593: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:09:32.788832: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 106ms/step
7/7 [==============================] - 0s 49ms/step
2023-04-07 00:09:34.211241: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:09:58.446820: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 128ms/step
7/7 [==============================] - 0s 52ms/step
2023-04-07 00:09:59.975005: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:10:25.490118: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 84ms/step
7/7 [==============================] - 0s 74ms/step
2023-04-07 00:10:27.016270: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:10:52.608583: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 104ms/step
7/7 [==============================] - 0s 55ms/step
2023-04-07 00:10:54.677792: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:11:19.022660: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 114ms/step
7/7 [==============================] - 0s 67ms/step
2023-04-07 00:11:20.611674: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:11:46.265412: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 113ms/step
7/7 [==============================] - 0s 64ms/step
2023-04-07 00:11:47.831712: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:12:19.028200: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 139ms/step
7/7 [==============================] - 0s 52ms/step
```

```
2023-04-07 00:12:20.590019: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:12:51.124387: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 104ms/step
7/7 [==============================] - 0s 52ms/step
2023-04-07 00:12:53.084721: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:13:23.220792: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 107ms/step
7/7 [==============================] - 0s 60ms/step
2023-04-07 00:13:24.690736: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:13:54.089794: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 132ms/step
7/7 [==============================] - 0s 58ms/step
2023-04-07 00:13:55.652306: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:14:25.953138: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 134ms/step
7/7 [==============================] - 0s 51ms/step
2023-04-07 00:14:27.466882: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:14:57.691835: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 105ms/step
7/7 [==============================] - 0s 51ms/step
2023-04-07 00:14:59.135009: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [77]:
```
best_optimizer = gs_2.best_params_["optimizer"]
best_epoch2 = gs_2.best_params_["epochs"]
best_drop = gs_2.best_params_["dropout_rate"]
print("Best: %f using %s" % (gs_2.best_score_, gs_2.best_params_))
```

```
Best: 0.933333 using {'dropout_rate': 0.3, 'epochs': 45, 'optimizer': 'adam'}
```

Model 5 with Best Parameters

In [78]:
```
model_5= models.Sequential([
Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
           padding = 'same', input_shape=(224,224,3)),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Conv2D(filters=128, kernel_size=(3, 3), activation='relu',
```

```
            padding = 'same'),
MaxPooling2D(pool_size=(2, 2), strides=2),

Flatten(),

Dense(units=128, activation='relu',),
Dropout(best_drop),
Dense(units=64, activation='relu'),
Dropout(best_drop),
Dense(units=1, activation='sigmoid'),
    ])

model_5.compile(loss='binary_crossentropy',
                   optimizer= best_optimizer,
                   metrics=["acc","Recall","Precision"]
                    )
```

In [79]:
```
#Model 5 Summary
model_5.summary()
```

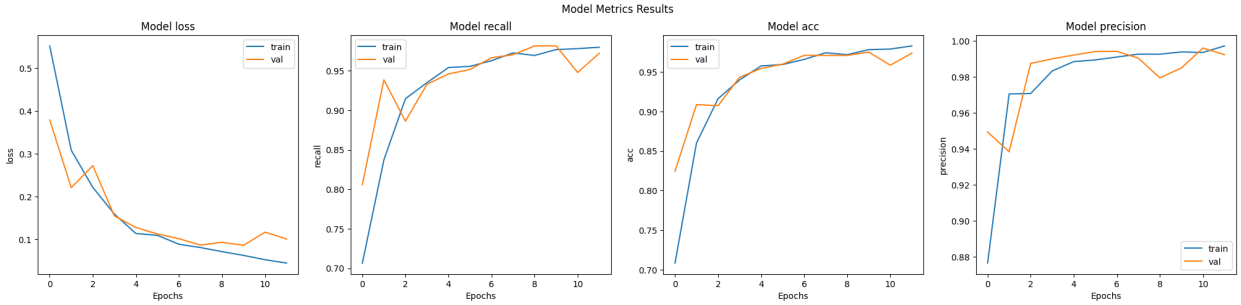
```
Model: "sequential_158"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_511 (Conv2D)         (None, 224, 224, 32)      896

 max_pooling2d_511 (MaxPooli  (None, 112, 112, 32)     0
 ng2D)

 conv2d_512 (Conv2D)         (None, 112, 112, 64)      18496

 max_pooling2d_512 (MaxPooli  (None, 56, 56, 64)       0
 ng2D)

 conv2d_513 (Conv2D)         (None, 56, 56, 96)        55392

 max_pooling2d_513 (MaxPooli  (None, 28, 28, 96)       0
 ng2D)

 conv2d_514 (Conv2D)         (None, 28, 28, 128)       110720

 max_pooling2d_514 (MaxPooli  (None, 14, 14, 128)      0
 ng2D)

 flatten_158 (Flatten)       (None, 25088)             0

 dense_353 (Dense)           (None, 128)               3211392

 dropout_193 (Dropout)       (None, 128)               0

 dense_354 (Dense)           (None, 64)                8256

 dropout_194 (Dropout)       (None, 64)                0

 dense_355 (Dense)           (None, 1)                 65

=================================================================
Total params: 3,405,217
Trainable params: 3,405,217
Non-trainable params: 0
_________________________________________________________________
```

In [80]:
```
#Modeling with best parameters
history5 = model_5.fit(train_gen,
                       epochs=best_epoch2,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es) #stopping early because of time resources
```

```
2023-04-07 00:15:27.514761: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:15:54.084900: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [81]:
```
print("Model 5 with updated hyperparamters :\n")
eval_metrics(history5)
```

```
Model 5 with updated hyperparamters :
```

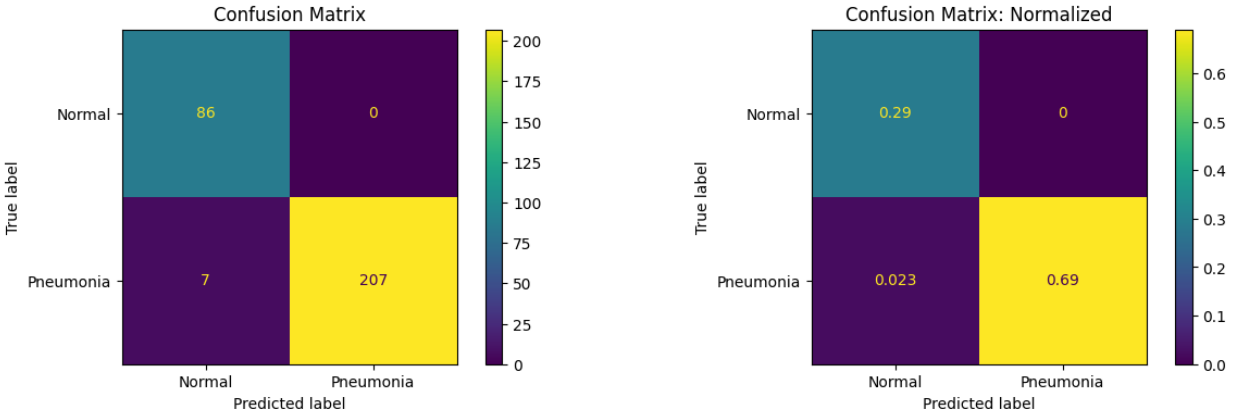


In [99]:
```
train_labels, train_predictions = predict_results(model_5,train_gen)
val_labels, val_predictions = predict_results(model_5,val_gen)
```

```
10/10 [==============================] - 0s 16ms/step
4/4 [==============================] - 0s 14ms/step
```

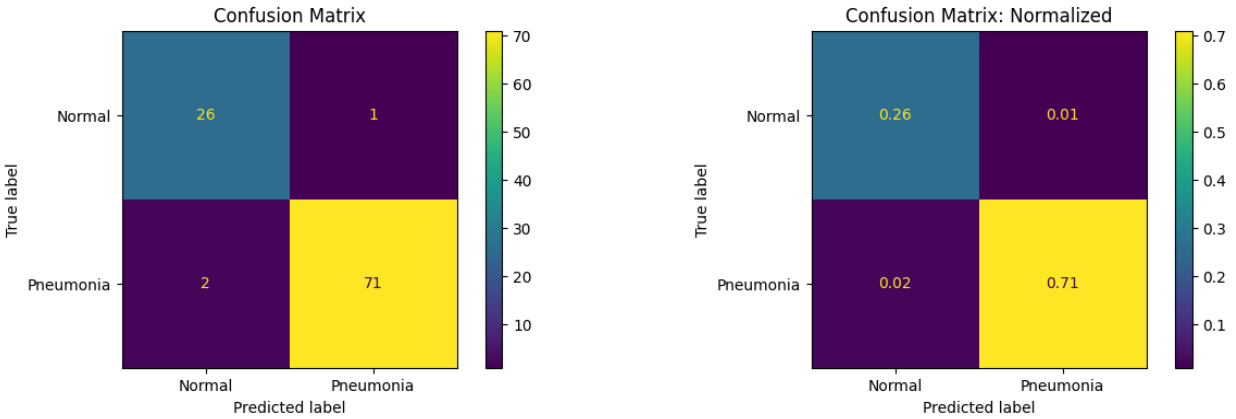
In [100…
```
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

```
Train Confusion Matrices:
```



In [101…
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```



In [102…
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history5)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.925     1.000     0.961        86
         1.0      1.000     0.967     0.983       214

    accuracy                         0.977       300
   macro avg      0.962     0.984     0.972       300
weighted avg      0.978     0.977     0.977       300
```

```
16/16 [==============================] - 22s 1s/step - loss: 0.0456 - acc: 0.9
834 - recall: 0.9779 - precision: 0.9997
Loss: 0.0456
```

In [103…

```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history5)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.929     0.963     0.945        27
         1.0      0.986     0.973     0.979        73

    accuracy                         0.970       100
   macro avg      0.957     0.968     0.962       100
weighted avg      0.971     0.970     0.970       100
```

```
8/8 [==============================] - 3s 400ms/step - loss: 0.1006 - acc: 0.9
737 - recall: 0.9720 - precision: 0.9924
Loss: 0.1006
```

Model 5 Evaluation - Needs Update

- **Model Plots: The metrics seem to be converging better.**
- **Confusion Matrix: Both models have about than 2% FN and less than 1% FP.**
- **Report: The validation recall is about .97, similar on both training and validation**
- **Ths model wasn't bad. The validation's loss increased. The recall and f1score decreased from the previous model metrics.**

Data Augmentation

Data Augmentation, YASSINE GHOUZAM, 2017

After several iterations of hyperparameter changes, the data will be augmented to see if this helps the models.

- **rotation_range=10, randomly rotate images in the range (degrees, 0 to 180)**
- **zoom_range = [.1,.2], Randomly zoom image**
- **width_shift_range=[.1,.2], randomly shift images horizontally (fraction of total width)**
- **height_shift_range=[.1,.2], Shift height**

In [104…
```
datagen = ImageDataGenerator(rescale=1/255,
        rotation_range=10,  # randomly rotate images in the range (degrees, 0 t
        zoom_range = [.1,.2], # Randomly zoom image
        width_shift_range=[.1,.2],  # randomly shift images horizontally (fract
        height_shift_range=[.1,.2],  # randomly shift images vertically (fracti
                            )
```

In [105…
```
train_gen_aug = datagen.flow_from_directory(train_directory,
                                            target_size = (224, 224),
                                            batch_size=128,
                                            classes = ["NORMAL","PNEUMON
                                            class_mode = 'binary',
                                            seed = 42)
```

```
Found 4509 images belonging to 2 classes.
```

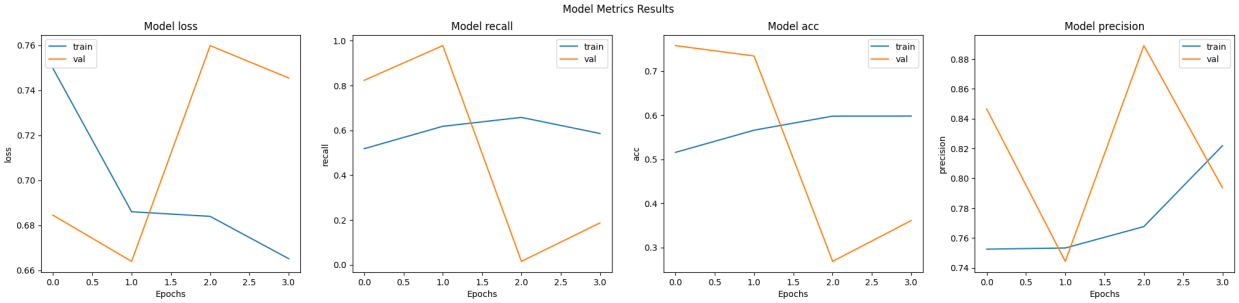
Model 6 (Model 5 with different augmented training data)

In [106…
```
#Using the same model with augmented data
model_6 = model_5
```

In [107…
```
history6 = model_5.fit(train_gen_aug,batch_size=best_batch,
                       epochs=best_epoch,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es) #stopping early because of time resources
```

In [108…
```
print("Model 6 with augmented data :\n")
eval_metrics(history6)
```

```
Model 6 with augmented data :
```

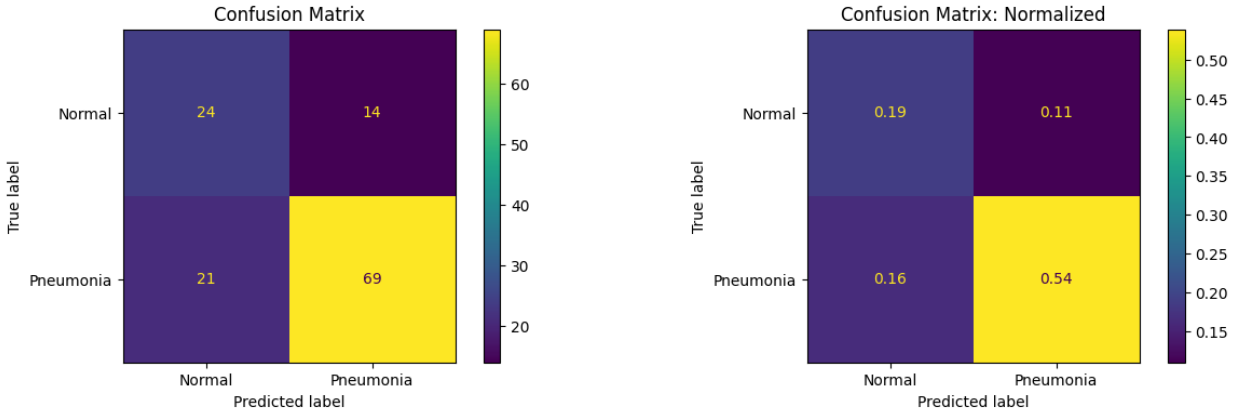


In [109…
```
train_labels, train_predictions = predict_results(model_5,train_gen_aug)
val_labels, val_predictions = predict_results(model_5,val_gen)
```

```
4/4 [==============================] - 0s 64ms/step
4/4 [==============================] - 0s 61ms/step
```

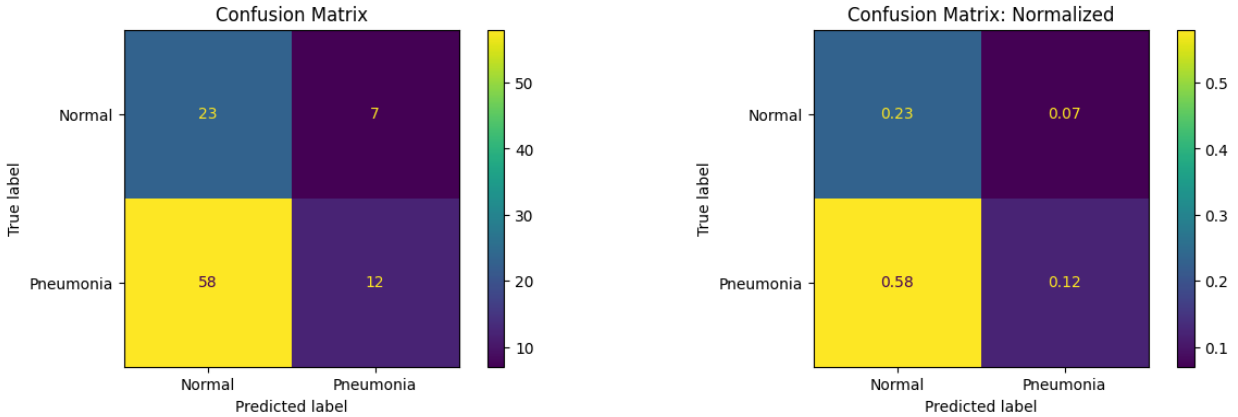
In [110…
```
#Printing regular and normalized Confusion Matrices
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

```
Train Confusion Matrices:
```



In [111…
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```



In [112…
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history6)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.533     0.632     0.578        38
         1.0      0.831     0.767     0.798        90

    accuracy                         0.727       128
   macro avg      0.682     0.699     0.688       128
weighted avg      0.743     0.727     0.733       128



16/16 [==============================] - 22s 1s/step - loss: 0.7407 - acc: 0.3
708 - recall: 0.2005 - precision: 0.8065
Loss: 0.7407
```

In [113…

```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history6)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.284     0.767     0.414        30
         1.0      0.632     0.171     0.270        70

    accuracy                         0.350       100
   macro avg      0.458     0.469     0.342       100
weighted avg      0.527     0.350     0.313       100



8/8 [==============================] - 4s 437ms/step - loss: 0.7454 - acc: 0.3
610 - recall: 0.1866 - precision: 0.7937
Loss: 0.7454
```

Model 6 Evaluation - Needs Updating

- **Using augmented training data produced the worst results so far.**
- **Epoch 2 caused the most variation. The model had almost 50% FN on validation and 18% FN on training. The loss is the highest so far in all iterations. The augmented data will not be used in the next iteration.**

Model 7 - Learning Rate Optimization

Using gridsearch and keras wrappers to do some hyperparameter optimization

- **learning_rate = [.001,.01,.1,.2]**

Description of the model hyperparameters and data parameters:

- **Model***

  - layers: 3 Conv2D each w/ MaxPooling2D, 3 Dense output layers
  - filters: 32,64,96,128 for Conv2D, 64,1 Dense layer
  - activation: Relu for all except Sigmoid for output layer
  - dropout:** .5 between the dense output layers
- Model Compile
  - loss: binary_crossentropy
  - optimizer:adam
  - learning rate: .001 (Default)
  - metrics: Accuracy,Recall,Precision
- Fit
  - data: train_gen, val_gen
  - train batchsize: 128
  - epochs: 90
  - early stopping : True
  - class weights: {0: 1.6, 1: 0.7272727272727273}
- Additional Comments
  - Conv2D model is best for images
  - MaxPooling 2D is a downsampling strategy CNNs

In [114…

```
def buildModel(learning_rate):

    model= models.Sequential([
    Conv2D(filters=32, kernel_size=(3, 3), activation='relu',
           padding = 'same', input_shape=(224,224,3)),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=64, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=96, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=128, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Conv2D(filters=128, kernel_size=(3, 3), activation='relu',
           padding = 'same'),
    MaxPooling2D(pool_size=(2, 2), strides=2),

    Flatten(),

    Dense(units=128, activation='relu',),
    Dropout(.3),
    Dense(units=64, activation='relu'),
    Dropout(.3),
    Dense(units=1, activation='sigmoid'),
```

```
    ])

    model.compile(loss='binary_crossentropy',
                  optimizer = Adam(learning_rate = learning_rate, ),
                    metrics=["acc","Recall","Precision"])


    return model
```

Learning rate determines the weights of out neural network with respect to the loss gradient.

In [115…
```
learning_rate = [.001,.01,.1,.2]
parameters_3 = dict( learning_rate=learning_rate)
parameters_3
```

Out[115]:
```
{'learning_rate': [0.001, 0.01, 0.1, 0.2]}
```

In [116…
```
classifier = KerasClassifier(build_fn=buildModel, batch_size = best_batch, epoc
#What hyperparameter we want to play with

grid_search = GridSearchCV(estimator = classifier,
                           error_score="raise",
                           param_grid = parameters_3,
                           refit = "Acc",
                           scoring = "accuracy",
                           cv = 3,return_train_score=True)
gs_3 = grid_search.fit(X_train, y_train, verbose = 0)
```

```
2023-04-07 00:36:13.797974: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:36:37.883827: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 214ms/step
7/7 [==============================] - 1s 143ms/step
2023-04-07 00:36:40.411480: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:37:02.681935: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 158ms/step
7/7 [==============================] - 1s 91ms/step
2023-04-07 00:37:04.715293: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:37:27.393308: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 178ms/step
7/7 [==============================] - 0s 66ms/step
2023-04-07 00:37:29.728439: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:37:52.058549: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 171ms/step
7/7 [==============================] - 0s 63ms/step
2023-04-07 00:37:53.901479: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:38:15.810424: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

```
4/4 [==============================] - 1s 152ms/step
7/7 [==============================] - 0s 69ms/step
2023-04-07 00:38:17.661180: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:38:40.323120: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 148ms/step
7/7 [==============================] - 1s 85ms/step
2023-04-07 00:38:42.287590: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:39:04.611612: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 152ms/step
7/7 [==============================] - 0s 74ms/step
2023-04-07 00:39:06.609164: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:39:29.158048: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 152ms/step
7/7 [==============================] - 0s 69ms/step
2023-04-07 00:39:31.754746: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:39:53.886288: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 175ms/step
7/7 [==============================] - 0s 67ms/step
2023-04-07 00:39:56.061400: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:40:18.097440: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 174ms/step
7/7 [==============================] - 0s 63ms/step
2023-04-07 00:40:19.989796: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:40:41.766724: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 183ms/step
7/7 [==============================] - 0s 69ms/step
2023-04-07 00:40:43.710349: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:41:06.131284: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
4/4 [==============================] - 1s 153ms/step
7/7 [==============================] - 1s 84ms/step
2023-04-07 00:41:08.785327: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [117…

```
best_rate = gs_3.best_params_["learning_rate"]
print("Best: %f using %s" % (gs_3.best_score_, gs_3.best_params_))
```

```
Best: 0.870000 using {'learning_rate': 0.001}
```

This .001 is the defult learning rate, nothing modifications and additons will be done. The results will be same as Model Iteration 5.

Transfer Learning

Using pretrained models from another problem and using them on another problem

- **VGG19 Pre-Trained Model**
  This method has faster training speed, fewer training samples per time, and higher accuracy
- **Feature Extraction without augmented data**
- **Feature Extraction with augmented data**
- **Fine Tuning**

In [118…
```
#The imagenet weights are downloaded from online
cnn_base = VGG19(weights="imagenet",
                 include_top=False,
                 input_shape=(224, 224, 3))
```

In [119…
```
#Creating the model layer architecture including cnn_base (19 layers)
model_8 = models.Sequential()
model_8.add(cnn_base)
model_8.add(layers.Flatten())
model_8.add(layers.Dense(132, activation='relu'))
model_8.add(layers.Dense(1, activation='sigmoid'))
```

In [120…
```
cnn_base.summary()
```

```
Model: "vgg19"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv4 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv4 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv4 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

=================================================================
Total params: 20,024,384
Trainable params: 20,024,384
Non-trainable params: 0
_________________________________________________________________
```

In [121…

```
model_8.summary()
```

```
Model: "sequential_172"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 7, 7, 512)         20024384

 flatten_172 (Flatten)       (None, 25088)             0

 dense_395 (Dense)           (None, 132)               3311748

 dense_396 (Dense)           (None, 1)                 133

=================================================================
Total params: 23,336,265
Trainable params: 23,336,265
Non-trainable params: 0
_________________________________________________________________
```

In [122…
```
# You can check whether a layer is trainable (or alter its setting) through the
for layer in model_8.layers:
    print(layer.name, layer.trainable)

# Similarly, you can check how many trainable weights are in the model
print(len(model_8.trainable_weights))
```

```
vgg19 True
flatten_172 True
dense_395 True
dense_396 True
36
```

In [123…
```
# Freeze the model
cnn_base.trainable = False
```

In [124…
```
# You can check whether a layer is trainable (or alter its setting) through the
for layer in model_8.layers:
    print(layer.name, layer.trainable)

# Similarly, you can check how many trainable weights are in the model
print(len(model_8.trainable_weights))
```

```
vgg19 False
flatten_172 True
dense_395 True
dense_396 True
4
```

In [125…
```
model_8.compile(loss='binary_crossentropy',
                    optimizer= "adam",
                    metrics=["acc","Recall","Precision"]
                     )
```

In [126…
```
#Copying model before trained for Augmented Data Fitting
model_9 = model_8
```

Created sepearate data generator to allow variation from the regular training batch.
The Kernel frequently dies with too high of a training size number

In [127…
```
#Using a generator for the images to work with.
trans_gen = train_datagen.flow_from_directory(train_directory,
                                              target_size = (224, 224),
                                              batch_size=300,
                                              classes = ["NORMAL","PNEUMON
                                              class_mode = 'binary',
                                              seed = 42)
```

```
Found 4509 images belonging to 2 classes.
```

In [128…
```
X_trans, y_trans = next(trans_gen)
```

In [129…
```
#Adding early stopping
es = EarlyStopping(monitor='val_loss', mode='min', verbose=0, patience=2)
#Calculating class weights for imbalance

class_weights = calc_weight(y_trans)
class_weights
```

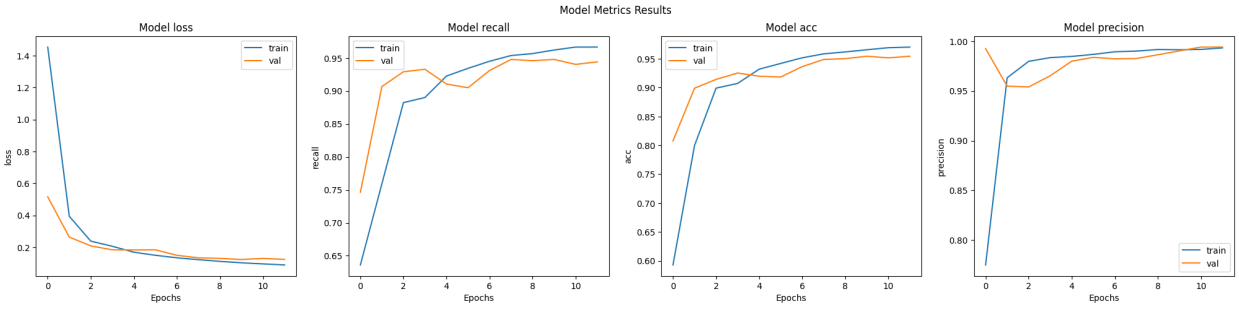
Out[129]:
```
{0: 1.8987341772151898, 1: 0.6787330316742082}
```

In [130…
```
#Fitting with regular training data
history8 = model_8.fit(trans_gen,
                       epochs=90,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es) #stopping early because of time resources
```

```
2023-04-07 00:58:05.603421: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 00:58:34.393428: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [131…
```
print("Model 8 with augmented data :\n")
eval_metrics(history8)
```

```
Model 8 with augmented data :
```

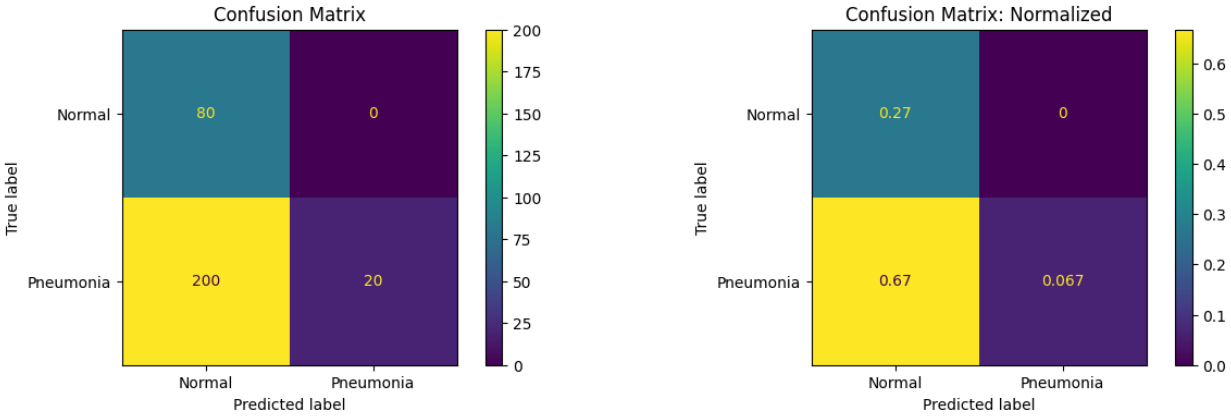


In [139…
```
train_labels, train_predictions = predict_results(model_8,trans_gen)
val_labels, val_predictions = predict_results(model_8,val_gen)
```

```
10/10 [==============================] - 2s 209ms/step
4/4 [==============================] - 1s 183ms/step
```

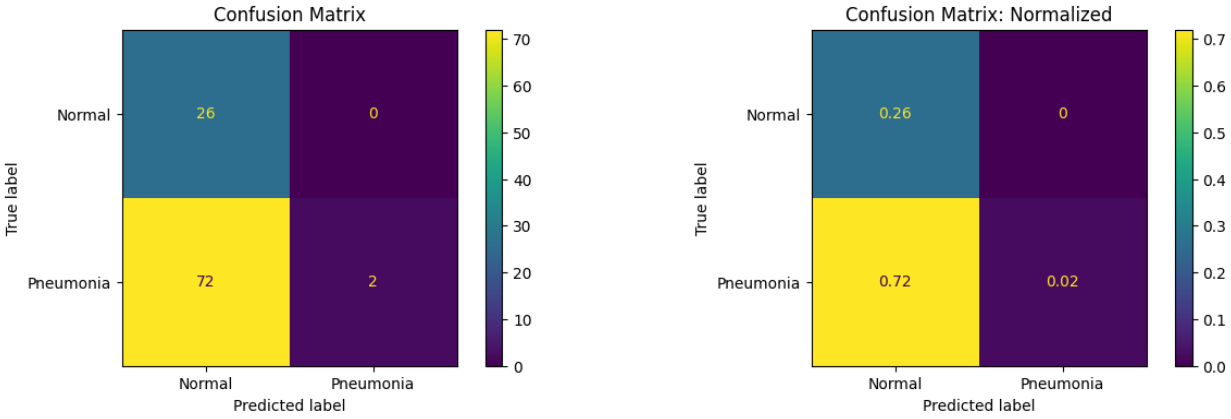
In [141…
```
#Printing regular and normalized Confusion Matrices
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

Train Confusion Matrices:



In [142…
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

Validation Confusion Matrices:



In [144…
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,trans_gen,history8)
```

Training Classificaiton Report:

	precision	recall	f1-score	support
0.0	0.286	1.000	0.444	80
1.0	1.000	0.091	0.167	220
accuracy			0.333	300
macro avg	0.643	0.545	0.306	300
weighted avg	0.810	0.333	0.241	300

```
16/16 [==============================] - 30s 2s/step - loss: 1.6054 - acc: 0.3
098 - recall: 0.0702 - precision: 1.0000
Loss: 1.6054
```

In [145…
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history8)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0       0.265     1.000     0.419        26
         1.0       1.000     0.027     0.053        74

    accuracy                          0.280       100
   macro avg       0.633     0.514     0.236       100
weighted avg       0.809     0.280     0.148       100


8/8 [==============================] - 6s 579ms/step - loss: 1.6410 - acc: 0.3
084 - recall: 0.0672 - precision: 1.0000
Loss: 1.641
```

Transfer Model 10 Evaluation

- The model is fitted well.
- The training model had 0% FN but 35% FP. The val model had less than 2% FN and FP.
- The scores are the worst of all the evaluations.
- Will not use thi model.

Transfer Model 9 w/ Augmented Data

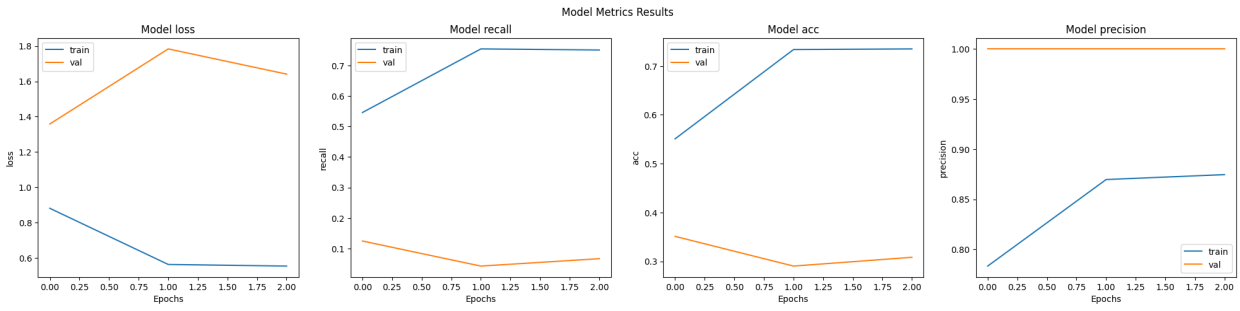
In [137…

```
#Fitting with augmented data
history9 = model_9.fit(train_gen_aug,batch_size=50,
                       epochs=90,
                       verbose = 0,
                       class_weight=class_weights,
                       validation_data=val_gen,
                      callbacks = es) #stopping early because of time resources
```

In [146…

```
print("Model 9 w augmented data :\n")
eval_metrics(history9)
```

```
Model 9 w augmented data :
```

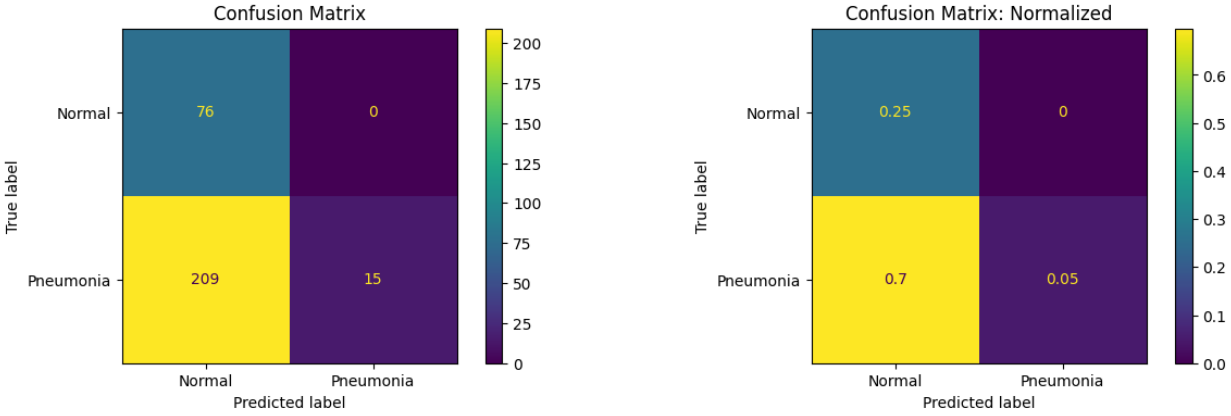


In [147…
```
train_labels, train_predictions = predict_results(model_9,train_gen)
val_labels, val_predictions = predict_results(model_9,val_gen)
```

```
10/10 [==============================] - 3s 268ms/step
4/4 [==============================] - 1s 175ms/step
```

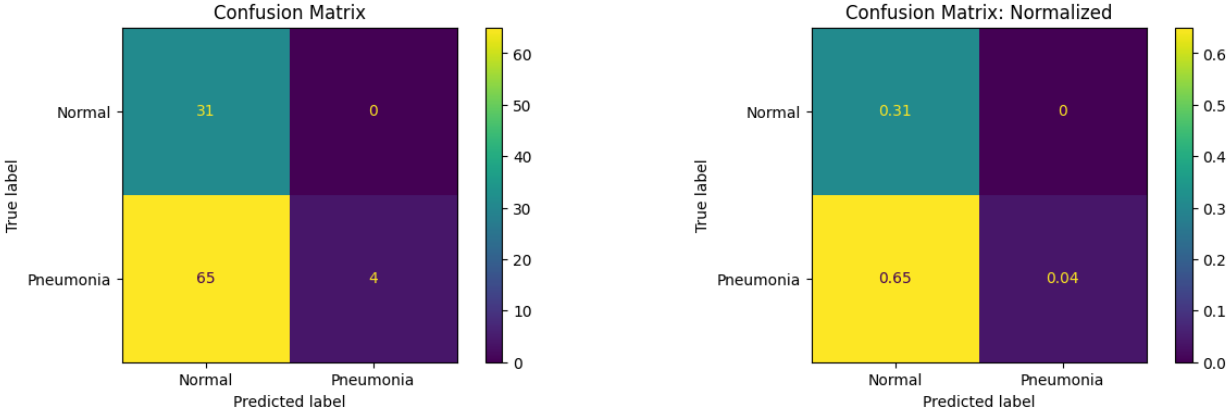
In [148…
```
#Printing regular and normalized Confusion Matrices
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

```
Train Confusion Matrices:
```



In [149…
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```



In [150…
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history9)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.267     1.000     0.421        76
         1.0      1.000     0.067     0.126       224

    accuracy                          0.303       300
   macro avg      0.633     0.533     0.273       300
weighted avg      0.814     0.303     0.200       300



16/16 [==============================] - 30s 2s/step - loss: 1.6054 - acc: 0.3
098 - recall: 0.0702 - precision: 1.0000
Loss: 1.6054
```

In [151…
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history9)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.323     1.000     0.488        31
         1.0      1.000     0.058     0.110        69

    accuracy                          0.350       100
   macro avg      0.661     0.529     0.299       100
weighted avg      0.790     0.350     0.227       100



8/8 [==============================] - 5s 492ms/step - loss: 1.6410 - acc: 0.3
084 - recall: 0.0672 - precision: 1.0000
Loss: 1.641
```

Transfer Model 9 Evaluation

The Transfer VGG19 model with augmented data was the worst so far. For the validation data there were mostly FN and and some TNs. Pneumonia was barely detected. Models trainined the augmented training data will not be used.

Transfer Model 10 w/ Fine Tuning

Removing a dense layer from the pre trained model

The fine tuning model is using the original training size

In [152…
```
#Copying the models to fine tune
model_10 = model_8
cnn_base_tuned = cnn_base
```

In [153…
```
model_10.summary()
```

```
Model: "sequential_172"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 7, 7, 512)         20024384

 flatten_172 (Flatten)       (None, 25088)             0

 dense_395 (Dense)           (None, 132)               3311748

 dense_396 (Dense)           (None, 1)                 133

=================================================================
Total params: 23,336,265
Trainable params: 3,311,881
Non-trainable params: 20,024,384
_________________________________________________________________
```

In [154…
```
cnn_base_tuned.trainable = True
```

In [155…
```
cnn_base_tuned.summary()
```

```
Model: "vgg19"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv4 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv4 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv4 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

=================================================================
Total params: 20,024,384
Trainable params: 20,024,384
Non-trainable params: 0
_________________________________________________________________
```

Removing one of the last layers from the pretrained model

In [156…

```
cnn_base_tuned.trainable = True
set_trainable = False
for layer in cnn_base_tuned.layers:
```

```
    if layer.name == 'block5_conv1':
        set_trainable = True
    if set_trainable:
        layer.trainable = True
    else:
        layer.trainable = False
```

In [157…
```
model_10.compile(loss='binary_crossentropy',
                 optimizer= "adam",
                 metrics=["acc","Recall","Precision"]
                  )
```

In [158…
```
history10 = model_10.fit(train_gen,
                         epochs=90,
                         verbose = 0,
                         class_weight=class_weights,
                         validation_data=val_gen,
                        callbacks = es) #stopping early because of time resources
```

```
2023-04-07 01:25:46.045121: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
2023-04-07 01:26:26.174943: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
```

In [159…
```
print("Model 9 w/o augmented data :\n")
eval_metrics(history10)
```

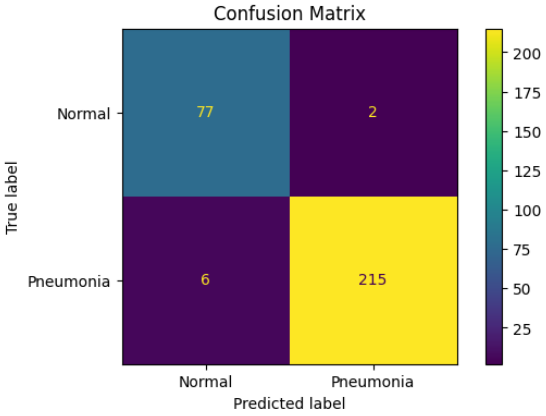
```
Model 9 w/o augmented data :
```

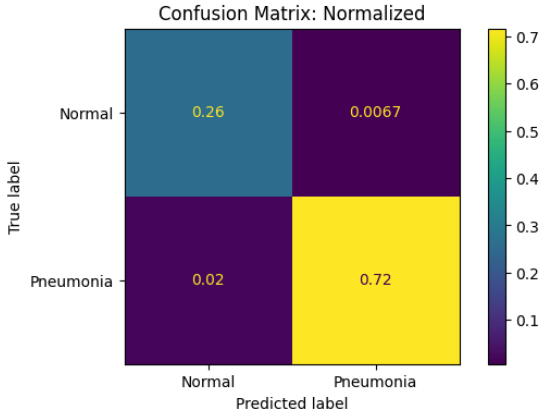
In [160…
```
train_labels, train_predictions = predict_results(model_10,train_gen)
val_labels, val_predictions = predict_results(model_10,val_gen)
```

```
2023-04-07 01:34:45.485883: I tensorflow/core/grappler/optimizers/custom_graph
_optimizer_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
10/10 [==============================] - 3s 189ms/step
1/1 [==============================] - 1s 845ms/step
```

In [162…
```
#Printing regular and normalized Confusion Matrices
print("Train Confusion Matrices:\n")
conf_matrix(train_labels, train_predictions)
```

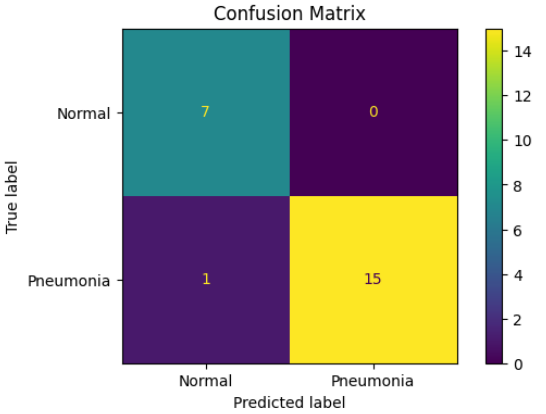
```
Train Confusion Matrices:
```

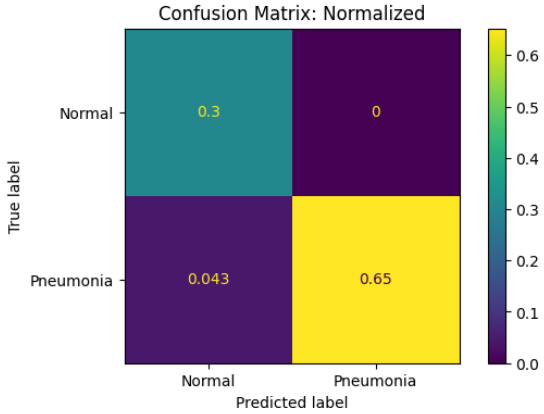




In [163…
```
print("Validation Confusion Matrices:\n")
conf_matrix(val_labels, val_predictions)
```

```
Validation Confusion Matrices:
```





In [164…
```
#Training Classification Report
print("Training Classificaiton Report:\n")
eval_report(train_labels,train_predictions,train_gen,history10)
```

```
Training Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.928     0.975     0.951        79
         1.0      0.991     0.973     0.982       221

    accuracy                         0.973       300
   macro avg      0.959     0.974     0.966       300
weighted avg      0.974     0.973     0.974       300



16/16 [==============================] - 29s 2s/step - loss: 0.0573 - acc: 0.9
783 - recall: 0.9740 - precision: 0.9966
Loss: 0.0573
```

In [165…
```
#Training Classification Report
print("Validation Classificaiton Report:\n")
eval_report(val_labels,val_predictions,val_gen,history10)
```

```
Validation Classificaiton Report:

              precision    recall  f1-score   support

         0.0      0.875     1.000     0.933         7
         1.0      1.000     0.938     0.968        16

    accuracy                          0.957        23
   macro avg      0.938     0.969     0.951        23
weighted avg      0.962     0.957     0.957        23


8/8 [==============================] - 5s 515ms/step - loss: 0.0974 - acc: 0.9
627 - recall: 0.9571 - precision: 0.9923
Loss: 0.0974
```

Transfer Model 10 Evaluation

The fine tuning drastically improved the model metrics. The loss for both are less than 10%. The metrics were as good as the initial models. The FN is both than 4% for each and about a 0 FP. This is a useable model.

Final Model

Best Model

The top 3 models listed all had comparable metrics for the recall and f1-score.

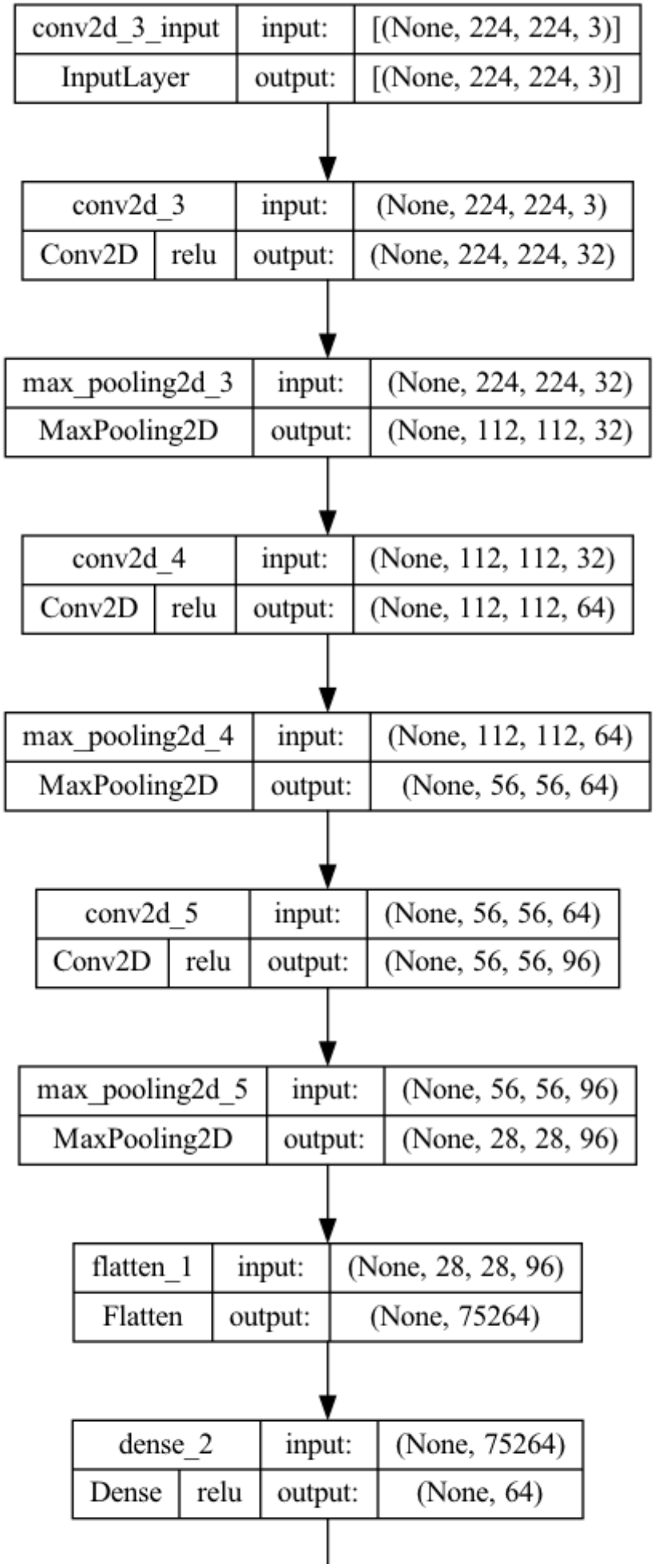
Based on this information and loss, Model will be chosen

- **Model 1 (Baseline) - Loss: .0865**
- **Model 2 (Baseline with class weight)**
  - **Normal: Recall - .93%, F1-Score - 97%**
  - **Pneumonia: Recall - 100%, F1-Score - 99%**
  - **Loss: .09**
- **Model 10 (VGG19 Model w/ Fine Tuning) - Loss: .0974**

In [167…

```
#Plotting model architecture
plot_model(baseline2, show_shapes=True, show_layer_names=True, show_layer_activ
```

Out[167]:


conv2d_3_input | input: | [(None, 224, 224, 3)]
InputLayer | output: | [(None, 224, 224, 3)]
conv2d_3 | input: | (None, 224, 224, 3)
Conv2D | relu | output: | (None, 224, 224, 32)
max_pooling2d_3 | input: | (None, 224, 224, 32)
MaxPooling2D | output: | (None, 112, 112, 32)
conv2d_4 | input: | (None, 112, 112, 32)
Conv2D | relu | output: | (None, 112, 112, 64)
max_pooling2d_4 | input: | (None, 112, 112, 64)
MaxPooling2D | output: | (None, 56, 56, 64)
conv2d_5 | input: | (None, 56, 56, 64)
Conv2D | relu | output: | (None, 56, 56, 96)
max_pooling2d_5 | input: | (None, 56, 56, 96)
MaxPooling2D | output: | (None, 28, 28, 96)
flatten_1 | input: | (None, 28, 28, 96)
Flatten | output: | (None, 75264)
dense_2 | input: | (None, 75264)
Dense | relu | output: | (None, 64)

In [168…
```
#Saving Model 5
save_file("model_2",baseline2)
```

```
WARNING:absl:Found untraced functions such as _jit_compiled_convolution_op, _jit_compiled_convolution_op, _jit_compiled_convolution_op while saving (showing 3 of 3). These functions will not be directly callable after loading.
INFO:tensorflow:Assets written to: ram://f56172d8-5045-4ab8-b627-610410d38cf0/assets
INFO:tensorflow:Assets written to: ram://f56172d8-5045-4ab8-b627-610410d38cf0/assets
```

Prediction

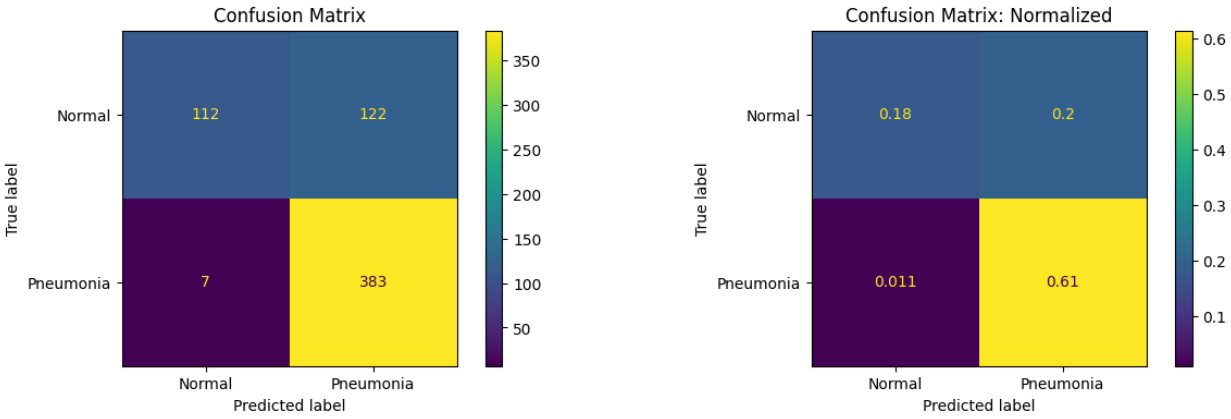
Using the chosen model and test data to predict

In [169…
```
#Test data previously generated in preprocessing step
test_labels, test_predictions = predict_results(baseline2,test_gen)
```

```
20/20 [==============================] - 1s 64ms/step
```

In [170…
```
print("Test Confusion Matrices:\n")
conf_matrix(test_labels, test_predictions)
```

```
Test Confusion Matrices:
```



In [171…
```
test_pred_round = np.round(test_predictions,0)
classification_report(test_labels,test_pred_round, output_dict=True)
```

Out[171]:
```
{'0.0': {'precision': 0.9411764705882353,
  'recall': 0.47863247863247865,
  'f1-score': 0.6345609065155807,
  'support': 234},
 '1.0': {'precision': 0.7584158415841584,
  'recall': 0.982051282051282,
  'f1-score': 0.8558659217877095,
  'support': 390},
 'accuracy': 0.7932692307692307,
 'macro avg': {'precision': 0.8497961560861969,
  'recall': 0.7303418803418803,
  'f1-score': 0.7452134141516451,
  'support': 624},
 'weighted avg': {'precision': 0.8269510774606872,
  'recall': 0.7932692307692307,
  'f1-score': 0.7728765410606613,
  'support': 624}}
```

In [172…
```
#Evaluating Model With Training Data
test_results = baseline2.evaluate(test_gen, return_dict= True)
test_results
```

```
1/1 [==============================] - 4s 4s/step - loss: 0.8950 - acc: 0.7933
- recall: 0.9821 - precision: 0.7584
```

Out[172]:
```
{'loss': 0.8949690461158752,
 'acc': 0.7932692170143127,
 'recall': 0.9820513129234314,
 'precision': 0.7584158778190613}
```

The model doesn't perform as well on the test data as it did on the training and validation data.

From the 624 Dataset:

**- .61% True Positive (383)
**
**- .018% True Negative (112)
**
**- .01% False Negative (7)
**
**- .2% False Positive (122)
**

Normal:

- **Recall:.48**
- **F1-Score:.63**
- **Precision:94**

Pneumonia:

- **Recall:.98**
- **F1-Score:.86**
- **Precision:.76**

Model Accuracy:.79

Model Explainer

The Lime Module will outline images and support explanation by highlighting important areas

In [173…

```
def img_array_pred(y_label, X_image_array,model,num):
    """ Inputing the data set and model to retrieve an image and prediciton res
    Lime Explainer lodule"""

    #num = np.random.randint(0,20) If want to later generate random pictures

    label = y_label[num]
    img = X_image_array[num]

    # Get Model Prediction
    pred = model.predict(np.array([img]))
    pred_round = np.round(pred,0)

    # Print True and Predicted Image
    if pred_round == 0:
        print('Image predicted as Normal (0)!')
    else:
        print('Image predicted as Pneumonia (1)')

    if label == 0:
        print('Image is Normal (0)!')
    else:
        print('Image is Pneumonia (1)!')

    print (f"Image Index {num} was chosen.")

    return array_to_img(img)
```

Analysis of True Positive Image

In [174…

```
#Prints Prediction, True Label, Associate Predicted Image and Index
img = img_array_pred(y_test, X_test,baseline2,300)
img
```

```
1/1 [==============================] - 1s 642ms/step
Image predicted as Pneumonia (1)
Image is Pneumonia (1)!
Image Index 300 was chosen.
```

Out[174]:

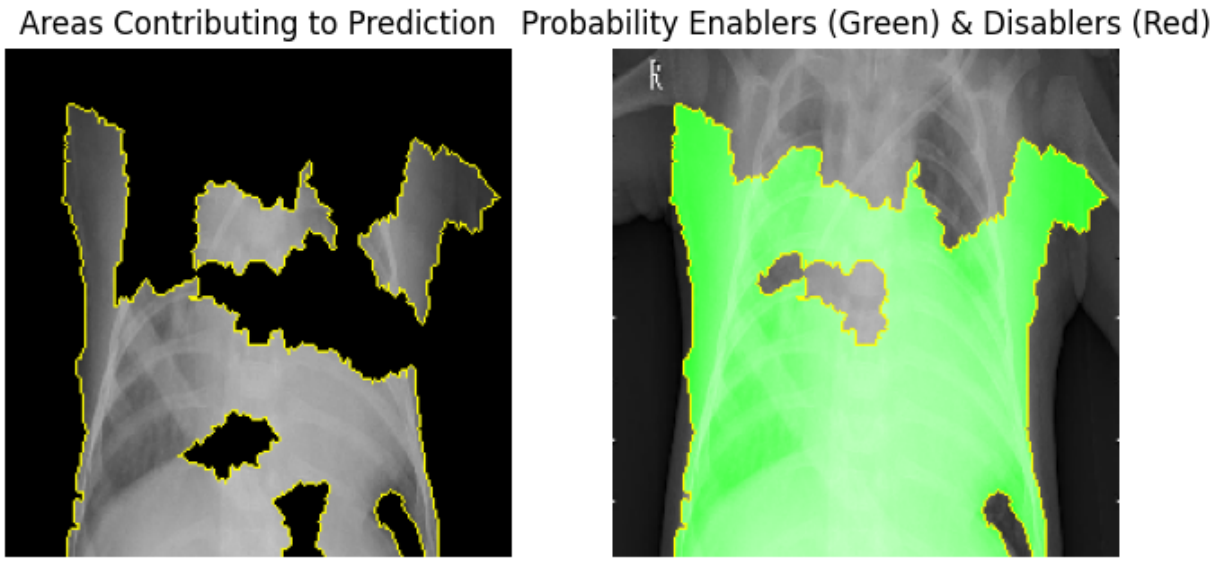
In [175…
```
explainer = lime_image.LimeImageExplainer()
```

In [176…
```
explanation = explainer.explain_instance(X_test[300].astype('double'), baseline
                                         top_labels=3, hide_color=0, num_sample
```

```
  0%|          | 0/1000 [00:00<?, ?it/s]
20/20 [==============================] - 2s 63ms/step
12/12 [==============================] - 1s 47ms/step
```

In [177…
```
temp_1, mask_1 = explanation.get_image_and_mask(explanation.top_labels[0], posi
temp_2, mask_2 = explanation.get_image_and_mask(explanation.top_labels[0], posi
```

In [178…
```
fig, axes = plt.subplots(1,2,figsize=(8,8))
axes[0].imshow(mark_boundaries(temp_1, mask_1))
axes[0].set_title("Areas Contributing to Prediction")
axes[1].imshow(mark_boundaries(temp_2, mask_2))
axes[1].set_title("Probability Enablers (Green) & Disablers (Red)")
axes[0].axis('off')
axes[1].axis('off')
plt.savefig('images/Lime_TP.png')
```



Analysis of False Negative Image

In [217…
```
img2 = img_array_pred(y_test, X_test,baseline2,599)
img2
```

```
1/1 [==============================] - 0s 10ms/step
Image predicted as Normal (0)!
Image is Pneumonia (1)!
Image Index 599 was chosen.
```

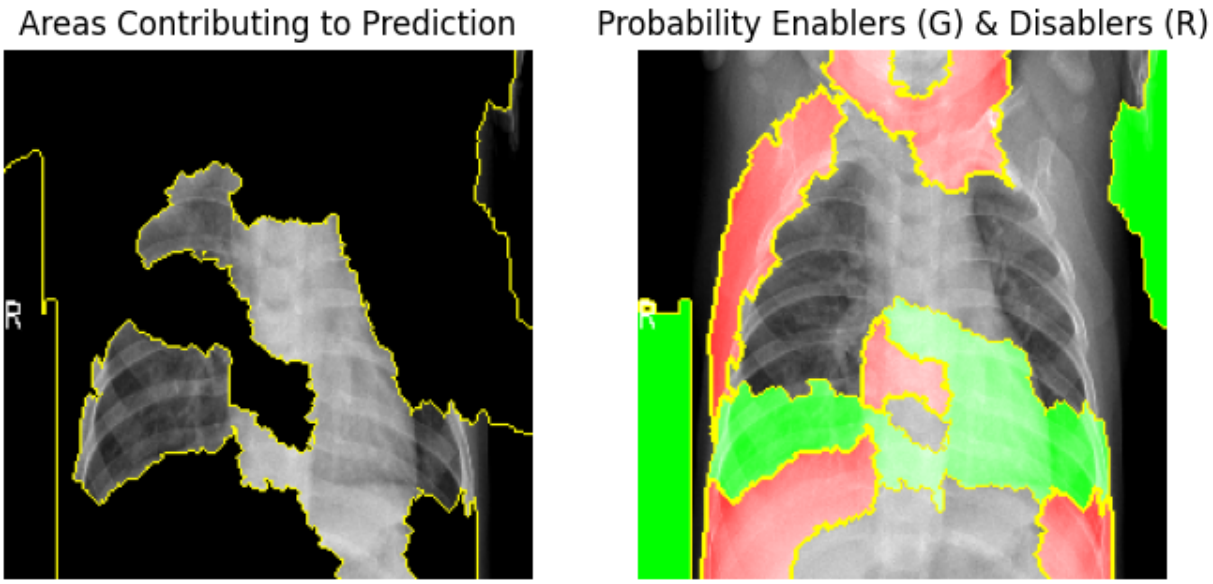
Out[217]:

In [220…
```
explainer = lime_image.LimeImageExplainer()
explanation2 = explainer.explain_instance(X_test[599].astype('double'), baselir
                                          top_labels=3, hide_color=0, num_sample
```

```
  0%|          | 0/1000 [00:00<?, ?it/s]
20/20 [==============================] - 1s 52ms/step
12/12 [==============================] - 1s 53ms/step
```

In [221…
```
temp_12, mask_12 = explanation2.get_image_and_mask(explanation.top_labels[0], p
temp_22, mask_22 = explanation2.get_image_and_mask(explanation.top_labels[0], p
```

In [222…
```
fig, axes = plt.subplots(1,2,figsize=(8,8))
axes[0].imshow(mark_boundaries(temp_12, mask_12))
axes[0].set_title("Areas Contributing to Prediction")
axes[1].imshow(mark_boundaries(temp_22, mask_22))
axes[1].set_title("Probability Enablers (G) & Disablers (R)")
axes[0].axis('off')
axes[1].axis('off')
plt.savefig('images/Lime_FN.png')
```



The model seems like it best with edges, the center areas decrease the percentage of guessing correctly.

In [223…

```
#Calculating end time
end = datetime.datetime.now()
duration = end - original_start
print('Notebook model execution: {}'.format(duration))
```

```
Notebook model execution: 5:01:27.089300
```

Conclusion

Reccommmendation

The intended use was to correctly identify Normal or Pneumonia diagnosis. The primary goal was to minimize False Negative diagnosis (identifying Normal when Pneumonia is present).

Hospital:

- **Usage: The model is best as a learning tool and not an official diagnosis.**
- **Strategy: Use the model as an initial reviewer of the images.**
- **Staffing: The model is best used with a doctor, not standalone.**

Technical:

- **Scope, review and process images more beforehand**
- **Visualize the activation functions to see better what areas the model layers are diagnosing**
- **Iterate model improvement with with augmented data**
- **Visually inspect the images that were FN and FP.**

Limitations

Hospital:

- **Important image areas**
- **Radiologist SME knowledge**

Technical:

- **Hardware and Software Compatibility (Modeling on M2 GPU Laptop)**
- **Hyperparameter optimization limits**
- **Blackbox of Hidden Layers**