COMPUTER VISION

PNEUMONIA DETECTION CHALLENGE

Interim Report Submission for Post Graduate Program in

Artificial Intelligence and Machine Learning (PGP - AIML)

By

Nov 21B CV1 Group 1

1. Ramesh Gopinath
2. Thrilok Madduru
3. Arabinda
4. Amol Pimpale
5. Kishore Bhagat
6. Amit Kumar Singhall

MENTOR: DR. AAMIT LAKHANI

Submission date: Oct 30, 2022

TABLE OF CONTENTS:

1. PROBLEM STATEMENT, OBJECTIVE AND DATA SET SUMMARY

✧ 1.1. Problem Statement Summary:

In various healthcare applications, CNN is widely used in assisting the medical professionals while treating the patients for better decision making during the treatment of various diseases. One of such disease is pneumonia which can be detected through X-ray by detecting lungs inflammation which generates immense amount of data. In case there is a presence of opacity in the lungs, then it is classified as **"**Lung Opacity**"** and if there is no opacity then it is classified as "Normal". We have new rows in the data that are classified under "Not Normal No Lung Opacity" which says there is abnormality in the lungs even though there is no opacity. We have a set of medical images which are stored in DCM format which contains metadata and image arrays for pixel data.

✧ 1.2. Data Sets:

Stage_2_train_images.zip - Contains 26684 training images in .dcm format
Stage_2_test_images.zip - Contains 3000 test images in .dcm format
Stage_2_train_labels.csv - CSV file containing training labels x,y,width,height,target
Stage_2_detailed_class_info.csv - Contains class info of training images like whether lung opacity is normal, Lung Opacity, No Lung Opacity / Not Normal
Stage_2_sample_submission.csv - Contains 3000 patient id's which are to be predicted using the Trained model.
* train and test images contains different metadata stored along with the images for a patient.

✧ 1.3. Objective:

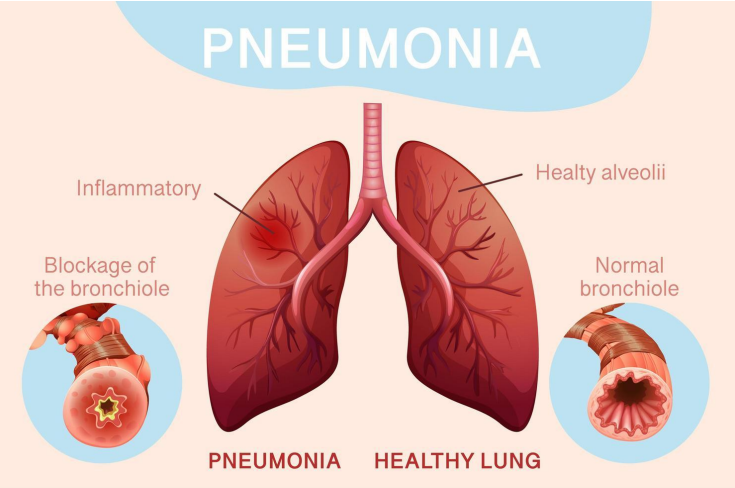
Design a deep learning-based algorithm for detecting pneumonia. We are making use of Deep Learning algorithms like Convolutional Neural Network models to properly predict the training images. In the later part, we will make use of transfer learning to train the model on given dataset and predict the test images.

2. BRIEF INTRODUCTION

✧ 2.1. What is Pneumonia disease?

Pneumonia is an infection that affects one or both lungs. It causes the air sacs, or alveoli, of the lungs to fill up with fluid or pus. Bacteria, viruses, or fungi may cause pneumonia. Symptoms can range from mild to serious and may include a cough with or without mucus (a slimy substance), fever, chills, and troubled breathing. How serious pneumonia is, depends on patients age, overall health, and what caused the infection.

To diagnose pneumonia, healthcare provider will review patients medical history, perform a physical exam, and order diagnostic tests such as a chest X-ray. This information can help determine what type of pneumonia the patient has.

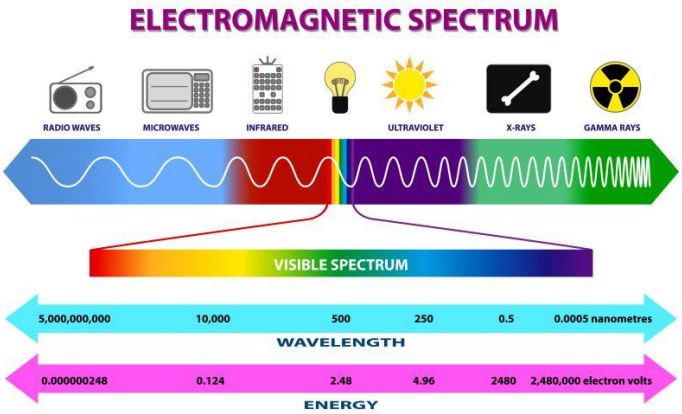


Pic Credits: Poster for Pneumonia with Human Lungs 1109633 Vector Art at Vecteezy

✧ 2.2. What are medical x-rays?

X-rays are a form of electromagnetic radiation, similar to visible light. Unlike light, however, x-rays have higher energy and can pass through most objects, including the body. Medical x-rays are used to generate images of tissues and structures inside the body. If x-rays traveling through the body also pass through an x-ray detector on the other side of the patient, an image will be formed that represents the "shadows" formed by the objects inside of the body.

One type of x-ray detector is photographic film, but there are many other types of detectors that are used to produce digital images. The x-ray images that result from this process are called radio graphs.



Pic Credits: https://www.nibib.nih.gov/science-education/science-topics/x-rays

✧ 2.3. AP and PA view of x ray:

2.3.1. Posterior-Anterior (PA) projection:

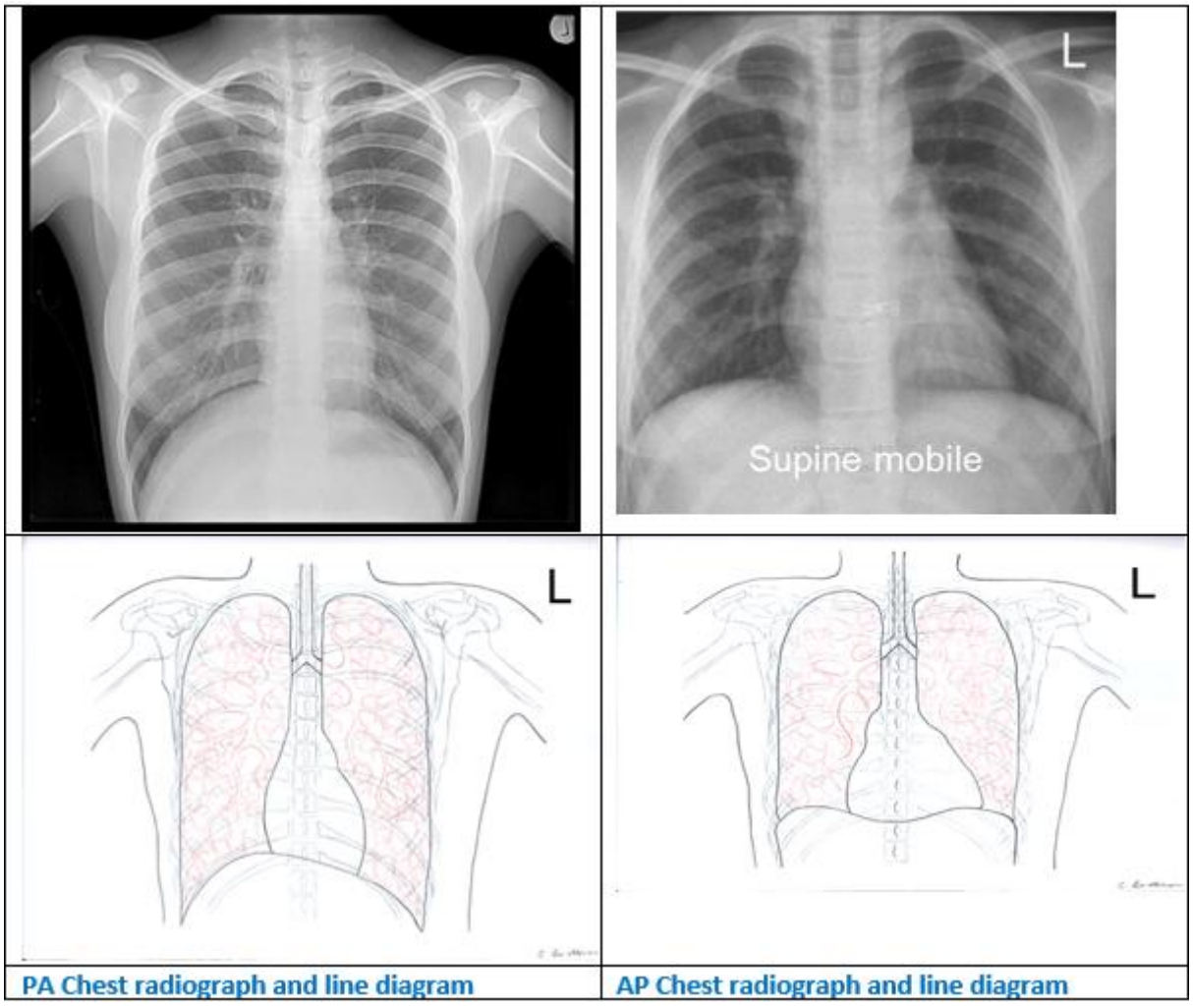
The standard chest radiograph is acquired with the patient standing up, and with the X-ray beam passing through the patient from Posterior to Anterior (PA).

The chest X-ray image produced is viewed as if looking at the patient from the front, face-to-face. The heart is on the right side of the image as you look at it.

2.3.2. Anterior-Posterior (AP) projection:

Sometimes it is not possible for radiographers to acquire a PA chest X-ray. This is usually because the patient is too unwell to stand.

The chest X-ray image is still viewed as if looking at the patient face-to-face.



Pic Credits: https://www.elearning.isrrt.org

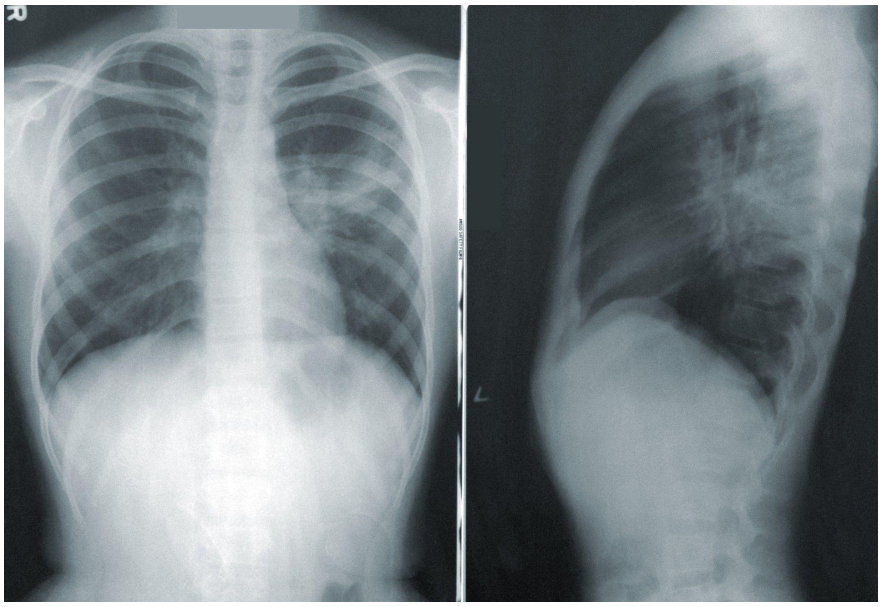
2.3.3. AP v PA projection:

In AP Projection, Heart size is exaggerated because the heart is relatively farther from the detector, and also because the X-ray beam is more divergent as the source is nearer the patient.

In PA Projection, heart size is nearer to the real size, as the heart is relatively nearer the detector. Magnification of the heart is also minimized by use of a narrower beam, produced by the increased distance between the source and the patient.

✧ 2.4 What is DICOM?

DICOM (Digital Imaging and Communications in Medicine) is a standard format that enables medical professionals to view, store, and share medical images irrespective of their geographic location or the devices they use, as long as those devices support the format. DICOM images need to be viewed through specific software called DICOM viewers that can read and display the format. The images, along with the corresponding patient data, are often stored in a large database called the Picture Archiving and Communication System (PACS). The purpose of a DICOM application is to store information in the PACS about the imaging examination, along with patient details, and then when required, to view and interpret (and possibly edit) medical images that are retrieved from the PACS. DICOM images are unique in the fact that they contain patient information in addition to the image data.



Pic Credits: https://towardsdatascience.com/how-to-use-fastai-to-evaluate-dicom-medical-files-738d7f7bc14d

3. METADATA SUMMARY

✧ 3.1. Metadata from dicom images:

COLUMN NAME	BRIEF INFORMATION
PATIENT ID	Primary identifier for the patient
X	Co-ordinate of image on X - Axis
Y	Co-ordinate of image on Y - Axis
WIDTH	Width of the image
HEIGHT	Height of the image
TARGET	1 - Pneumonia / 0 - No Pneumonia
CLASS	Normal, Lung Opacity, Not Normal / No Lung Opacity
SPECIFIC CHARACTER SET	ISO_IR 100, Character set description
SOP CLASS UID	'1.2.840.10008.5.1.4.1.1.7', The Secondary Capture (SC) Image Information Object Definition (IOD) specifies images that are converted from a non-DICOM format to a modality independent DICOM format.
SOP INSTANCE UID	Uniquely identifies the SOP Instance.
STUDY DATE	Date of the study started
STUDY TIME	Time of the study started
ACCESSION NUMBER	A RIS generated number that identifies the order for Th study
MODALITY	Type of equipment that originally acquired the data used to create the images in this Series.
CONVERSION TYPE	Describes the type of image conversion
REFERRING PHYSICIAN NAME	Name of the patients referring physician
SERIES DESCRIPTION	Description of the series - Posterior - Anteriror (PA) or Anterior - Posterior
PATIENT NAME	Full name of the patient
PATIENT BIRTH DATE	Birth date of the patient
PATIENT SEX	Sex of the patient. (Male / Female)
PATIENT AGE	Age of the patient
BODY PART EXAMINED	Description of the part of body examined
VIEW POSITION	Radiographic view of the image
STUDY INSTANCE UID	Unique identifier assigned to the study
SERIES INSTANCE UID	Unique identifier assigned to the series
STUDY ID	Equipment generated study identifier
SERIES NUMBER	Number to identify this series
INSTANCE NUMBER	Number that identifies this image
PATIENT ORIENTATION	Patient direction of the rows and columns of image
SAMPLES PER PIXEL	Number of samples in an image
PHOTOMETRIC INTERPRETATION	Specifies intended interpretation of the pixel data.
ROWS	No of rows in the Image
COLUMNS	No of columns in the image
MULTI PIXEL SPACING	Space between multiple pixels of an image
PIXEL SPACING / PIXEL SPACING 1	physical distance between the centers of each Two -dimensional pixel, specified by two numeric values
BITS ALLOCATED	No of bits allotted to each pixel sample
BITS STORED	No of bits stored for each pixel sample
HIGH BIT	Most significant bit for pixel sample data
PIXEL REPRESENTATION	Data representation of pixel samples
LOSSY IMAGE COMPRESSION	Specifies whether an image has undergone any image compression

LOSSY IMAGE COMPRESSION METHOD	Lossy Compression Method applied on image
FNAME	Name of each image file
IMG_MIN	Minimum pixel value in the image
IMG_MAX	Maximum pixel value in the image
IMG_MEAN	Average pixel value in the image
IMG_STD	Standard deviation of pixel values in the image
IMG_PCT_WINDOW	Image saved in Macintosh PICT format

4. EDA & VISUALIZATIONS

✧ What is EDA?

EDA - Exploratory Data Analysis is an important activity applied to investigate the data in depth and learn different data characteristics, summarize key inputs, and better understand the features of the data often with visual means and find useful patterns in the data.

✧ What are the steps involved?

1. Data Collection: This is the essential process where we find and load the data into system.

In the capstone project we are provided with below data:

A. 26684 Train images in Dicom Format.
B. 3000 Test images in Dicom Format.
C. Class information of training images.
D. Label information of training images.
E. Sample submission file for test images.

We have unzipped the images file and saved them in respective folders. Class and Label information was read into a Pandas Data Frame. Apart from the image files and data csv files provided, we have metadata in the dicom images, which is extracted and stored in the Data Frame. After loading all the required data, we displayed basic information of the data loaded using various methods like .info(), head(), .tail()…etc.

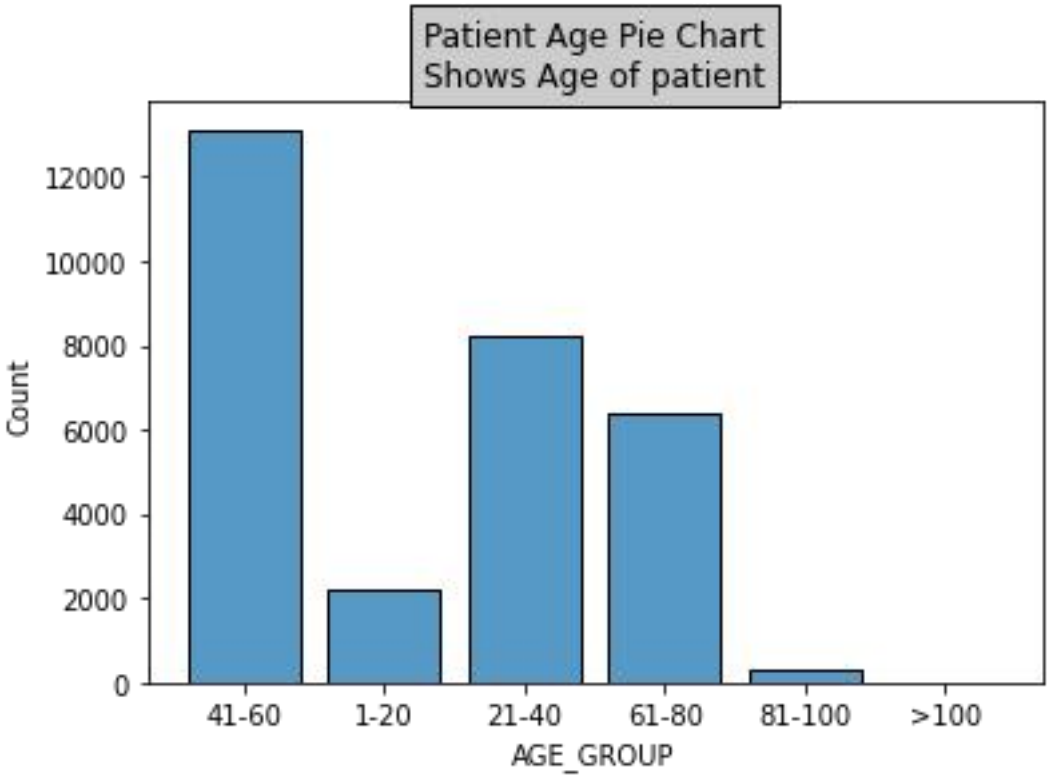
2. Data Cleaning:

Here we look for various unwanted variables and values, getting rid of the irregularities, check for missing values and outliers and take necessary actions like dropping the rows / columns having unwanted / missing data. Deal with outliers and reduce the number of outliers in the data, replacing the missing values with significant values.

In the metadata of images extracted, we are having 25 columns where there is only single value and 5 out of those are having NaN value which doesn't have meaningful information that affects model performance and dropped. Hence we haven't considered these features while doing Uni-variate, Bi-variate analysis.

There seems to be some issues while collecting Age information, there are quite a few age report ed above 100 which might not be true, there was one number even reported as 155. I guess those are mistakes hence converting Age into bins of

- 01 - 20
- 21 - 40
- 41 - 60
- 61 - 80
- 81 - 100
- > 100



when observed there are large number of patients between 41-60 followed by 21-40, 61-80 and relatively lesser number of patients between 1 - 20, 81 - 100 and > 100

this observation looks logically correct since there will be very less patients in the largest age category 81-100 and > 100 and in the smallest age group of 1 -
20 (since there are greater probabilities of young people being hale and healthy)

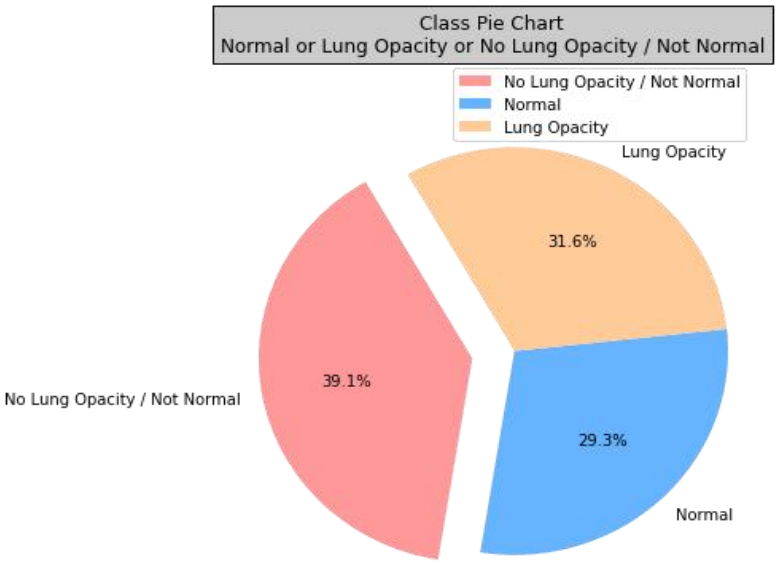
We have mapped the training class and labels information to the images where there 7402 duplicate records in the merged data frame which were dropped from the data frame to get back the original 30227 rows after the merge

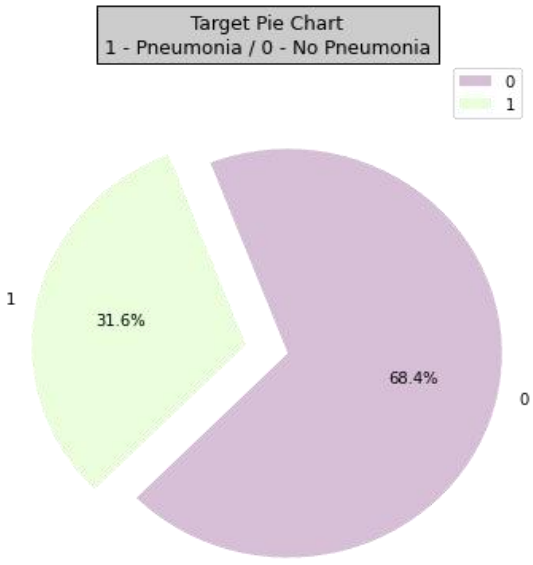
3. <u>Uni-variate Analysis:</u>

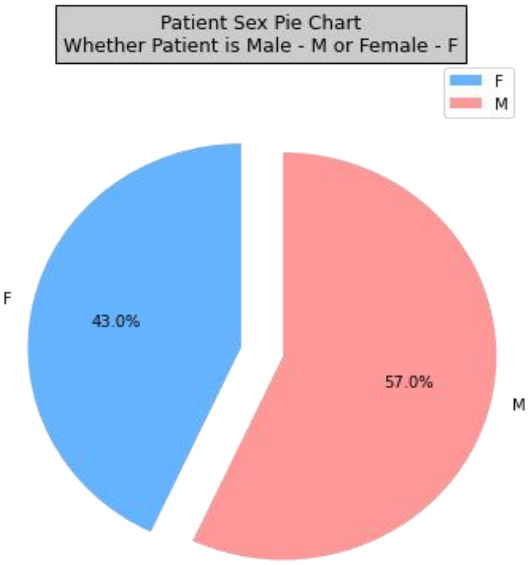
As the name represents, in this analysis, we analyze the data of just a single feature or column. This can be done by using graphical methods using various visualization techniques or non-graphical methods by finding specific mathematical values in the data. We used graphical approach for this analysis on below features:

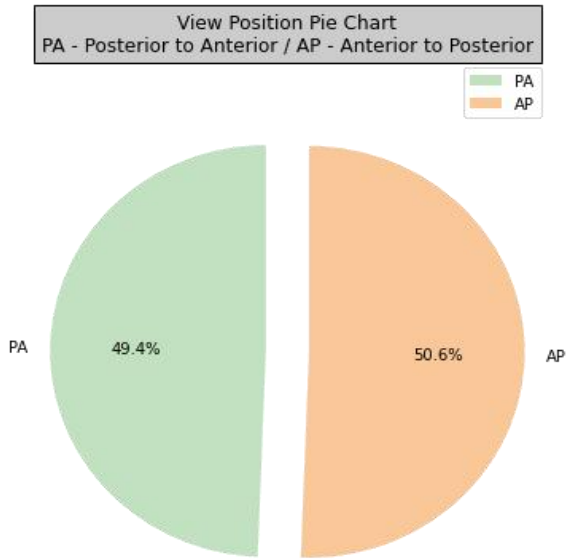
A. CLASS - Normal, Lung Opacity, No Lung Opacity / Not Normal
B. TARGET - Either 0 - No Pneumonia / 1 - Pneumonia
C. PATIENT SEX - M - Male / F - Female
D. VIEW POSITION - AP - Anterior to Posterior / PA - Posterior to Anterior

We used pie-charts for the graphical representation of the above features individually.









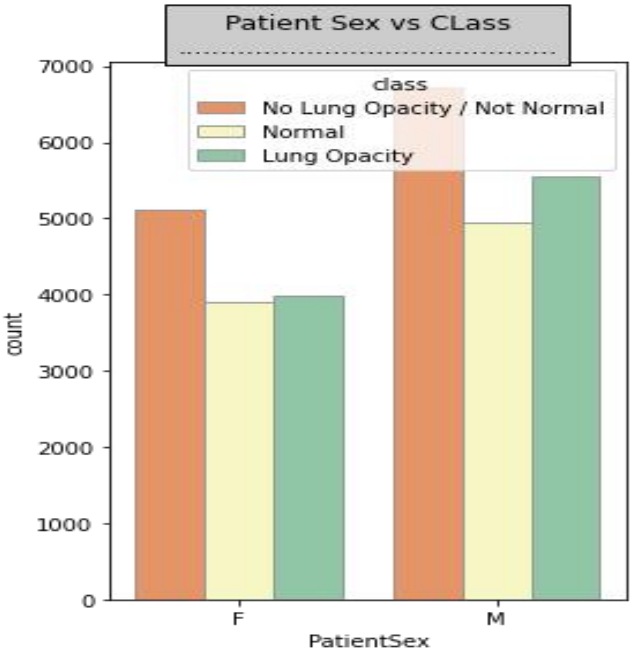
Observations:

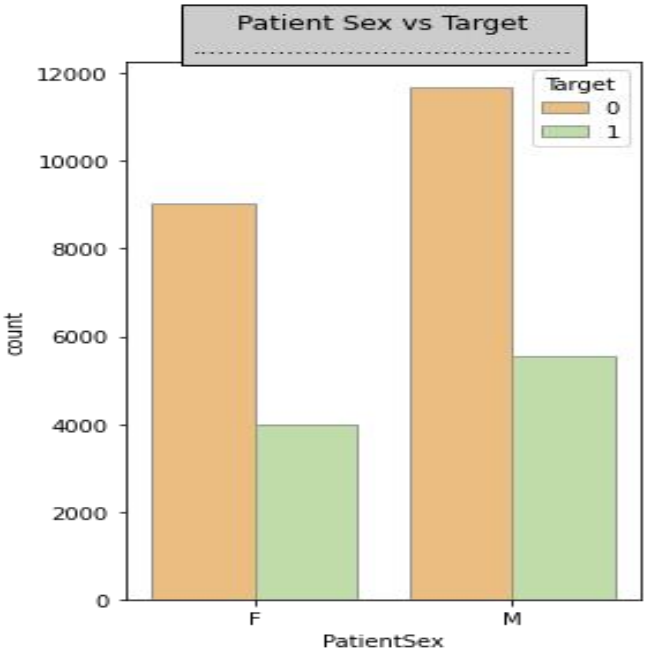
- Pneumonia is about 1/3 of the cases and non-Pneumonia are 2/3 of the cases
- More Males have got themselves checked than Females
- There are almost equal proportions of View Positions

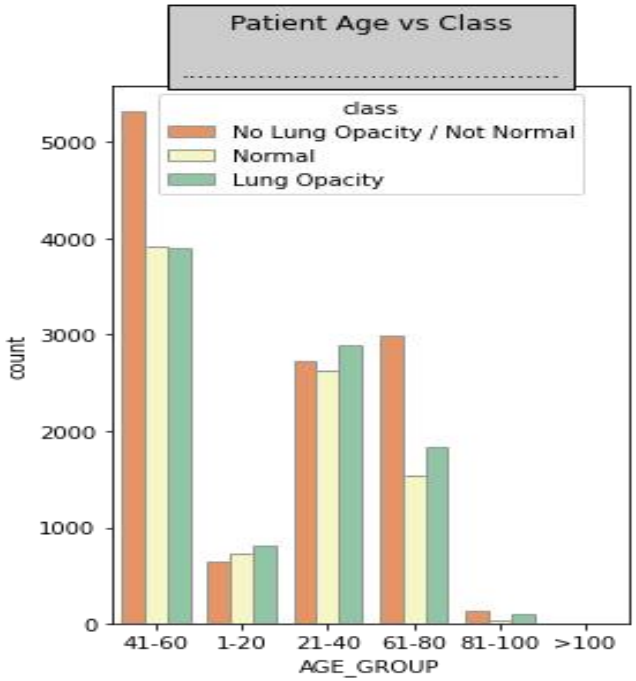
4. <u>Bi-Variate Analysis:</u>

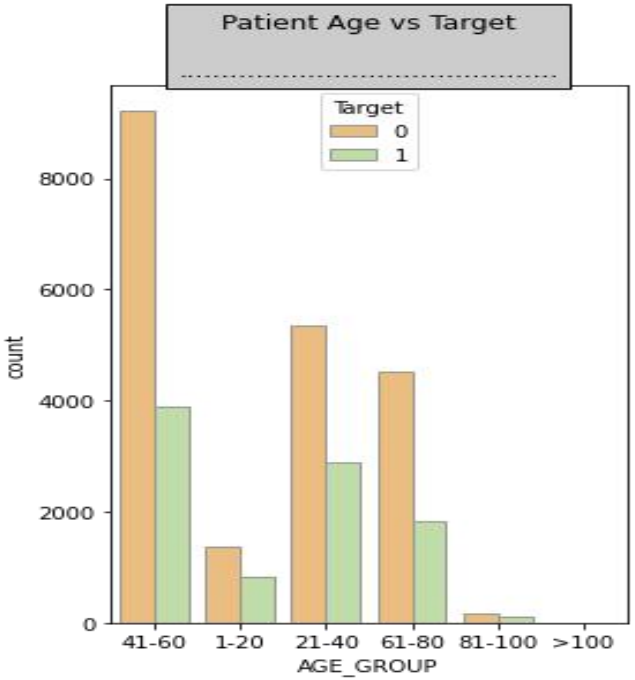
Here we use two features and compare them. In this we can find how one feature affect the other feature. Here we are doing below Bi-Variate analysis:

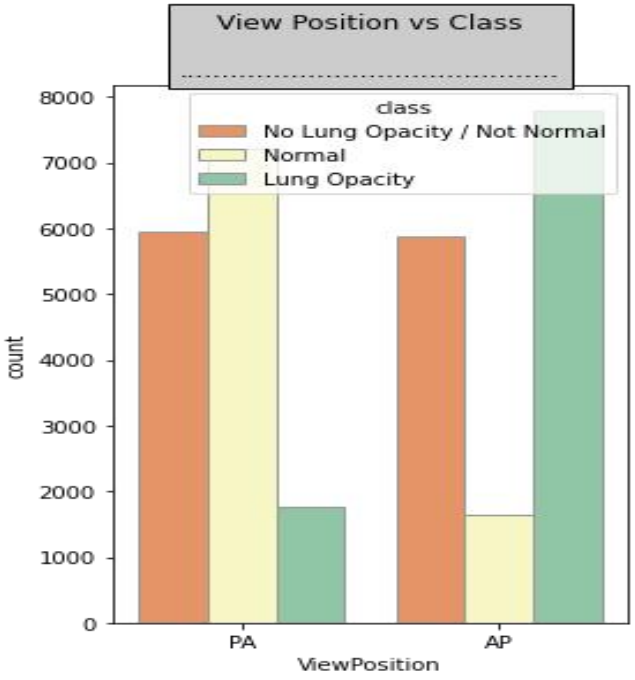
A. Patient Sex Vs Class
B. Patient Sex Vs Target
C. Patient Age Vs Class
D. Patient Age Vs Target
E. View Position Vs Class
F. View Position Vs Target
G. Class Vs Target
H. Centers of Lung Opacity Rectangles over Rectangle

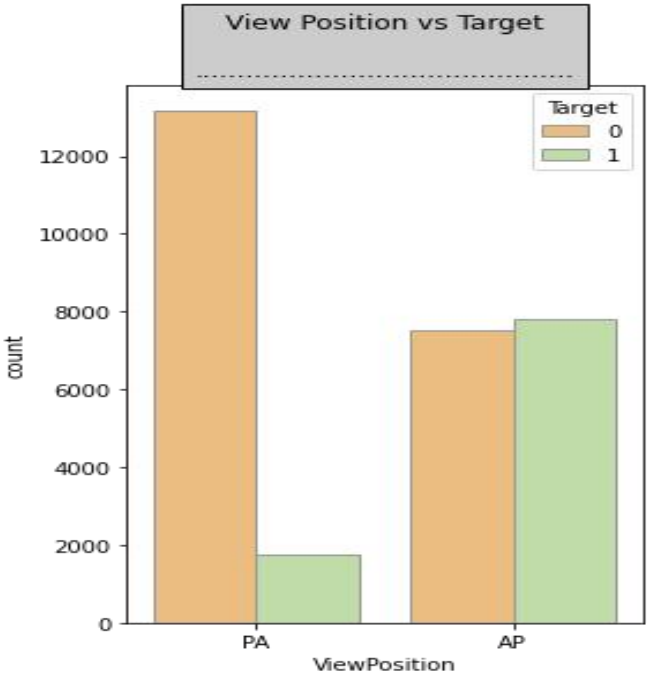

Patient Sex vs CLass
class
No Lung Opacity / Not Normal
Normal
Lung Opacity
count
PatientSex
F
M


Patient Sex vs Target
Target
0
1
count
PatientSex
F
M


Patient Age vs Class
class
No Lung Opacity / Not Normal
Normal
Lung Opacity
count
AGE_GROUP
41-60
1-20
21-40
61-80
81-100
>100


Patient Age vs Target
Target
0
1
count
AGE_GROUP
41-60
1-20
21-40
61-80
81-100
>100


View Position vs Class
class
No Lung Opacity / Not Normal
Normal
Lung Opacity
count
ViewPosition
PA
AP


View Position vs Target
Target
0
1
count
ViewPosition
PA
AP

Observations:

1. Sex Vs Class

    Since there are Male patients than Female patients as observed earlier the proportion of Classes also seem to be almost similar between the genders

2. Sex Vs Target

    Since there are more Male patients, the number of Pneumonia cases are also higher in them than in the Female patients. But more or less the proportions seem to be almost similar between the genders

3. Age group Vs Class

    As observed earlier since the number of patients are very high in the 41-60 age category, the classes are also very high in that category. Normal cases and Lung opacity cases are almost the same in the 41-60 category with the Not normal cases being very high

    Since 21-40 age category patients are next highest after 41-60, the number of classes are also high there. Interestingly all the 3 classes are almost similar in this age category

    61-80 age has higher Not Normal cases than 21-40 Not normal cases

4. Age group Vs Target

    Age group 41-60 has the highest Pneumonia cases followed by 21-40 then 61-80 and then 1-20

5. View Position Vs Class

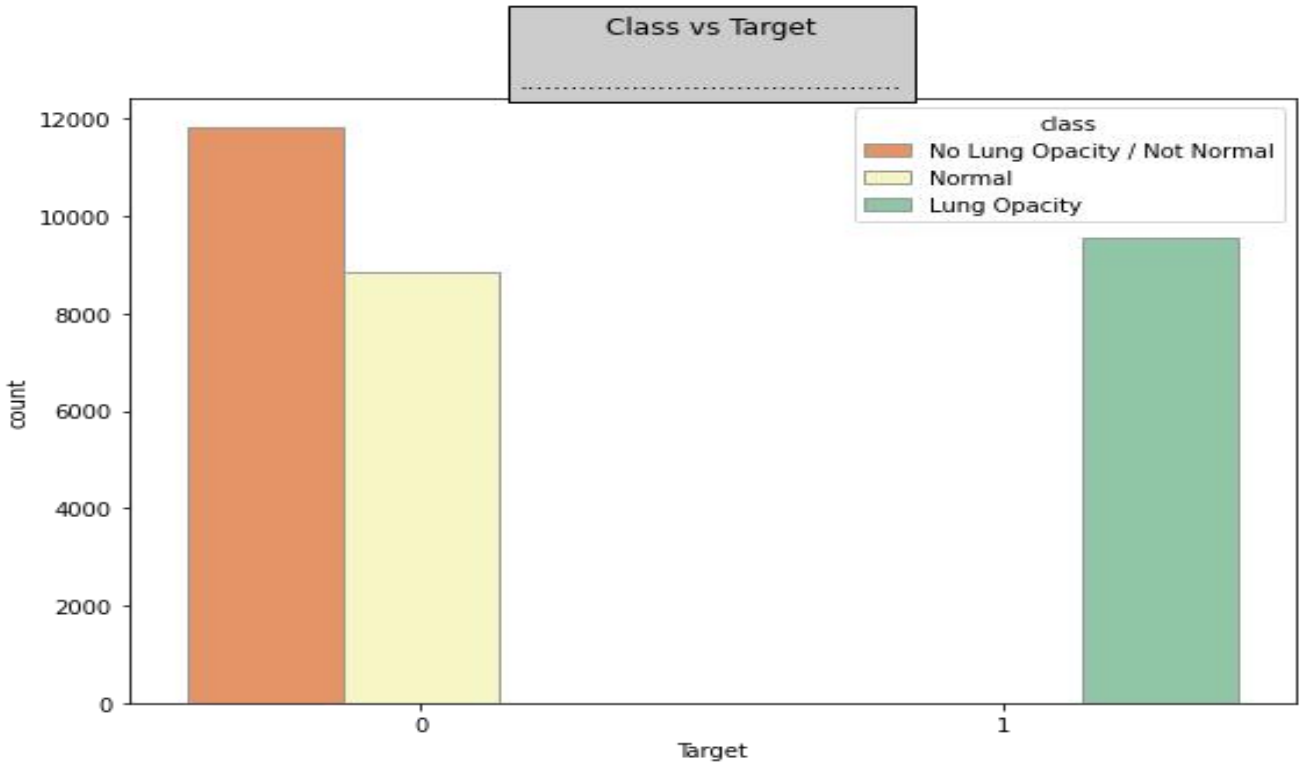
    PA - Posterior to Anterior means X-ray source passes through the back of the patient with the chest facing the Film

    AP - Anterior to Posterior means X-ray source passes through the front of the patient with the back facing the Film

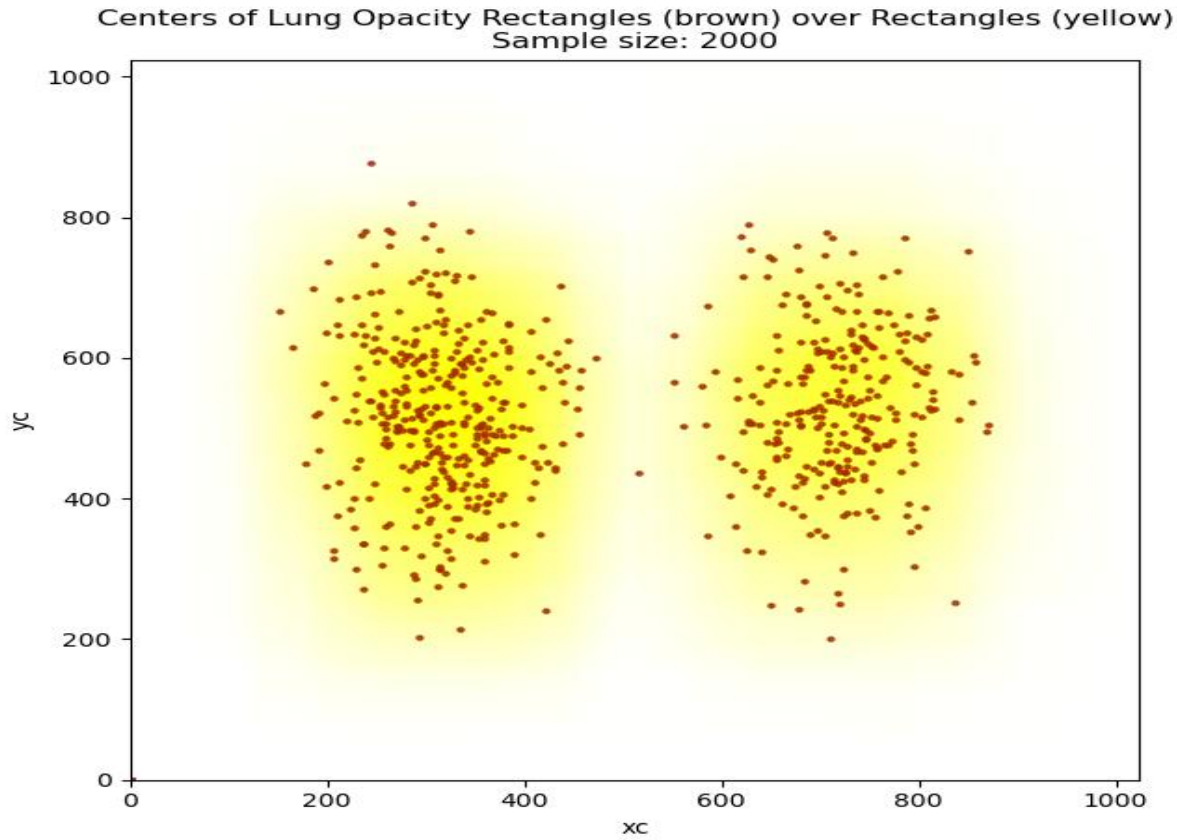
    More Normal Classes are found in PA than in AP view
    Nor normal / No Lung opacity classes are almost similar in both PA and AP views

6. View Position Vs Target

    Huge number of Pneumonia cases found in AP views than in PA views



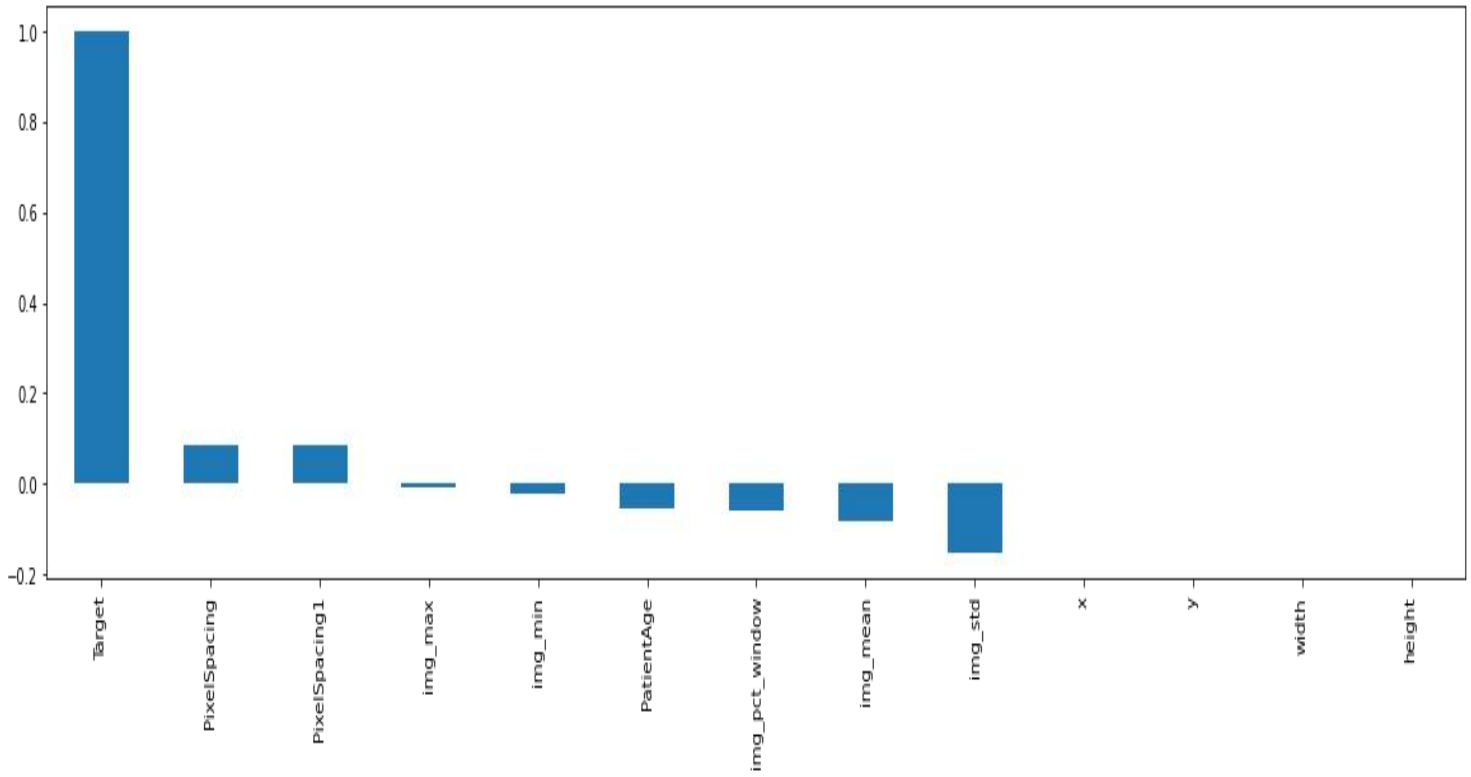
Observation: It is quite obvious that Pneumonia (Lung Opacity) shows up only in the Target = 1 and not in the Target = 0

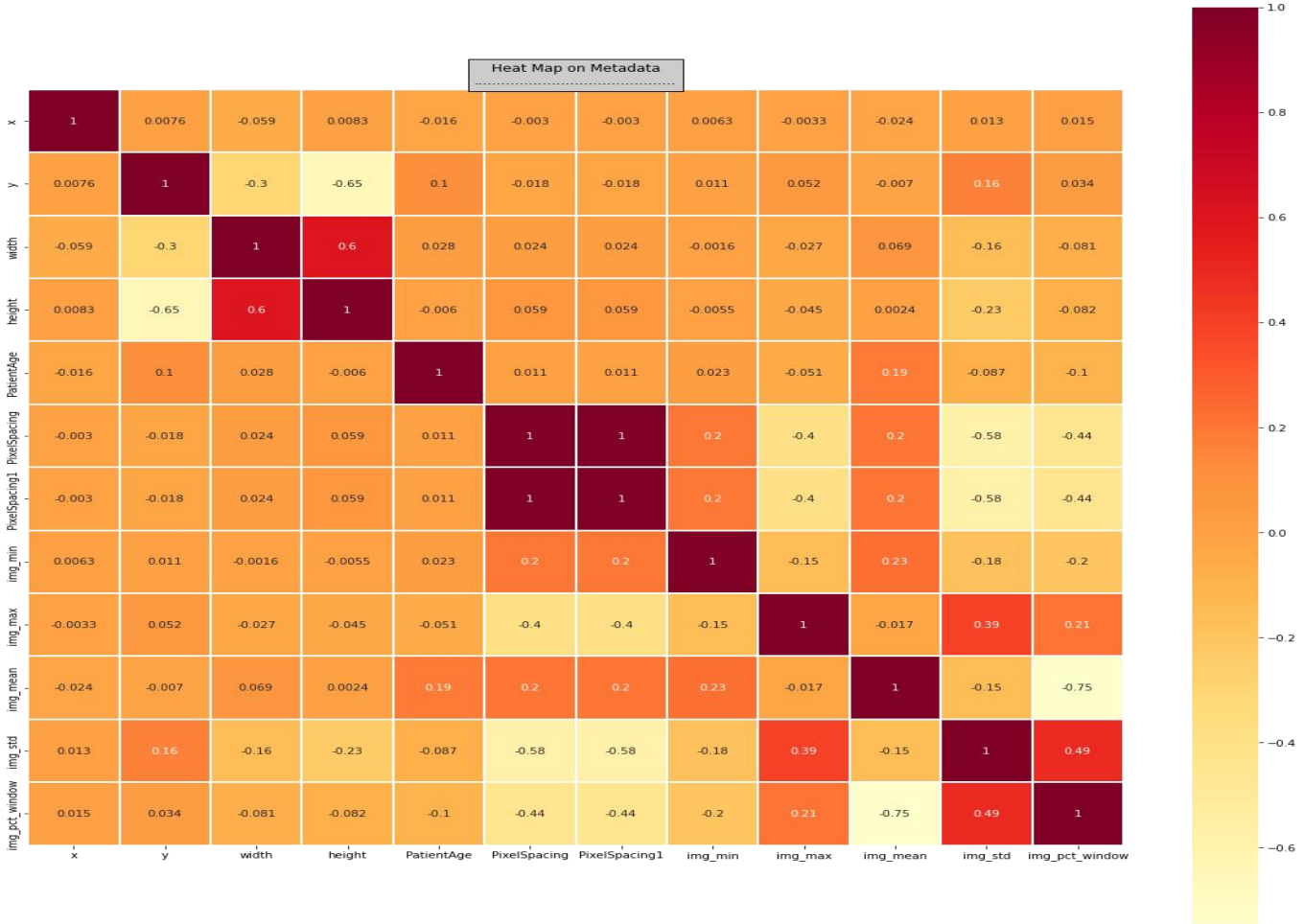


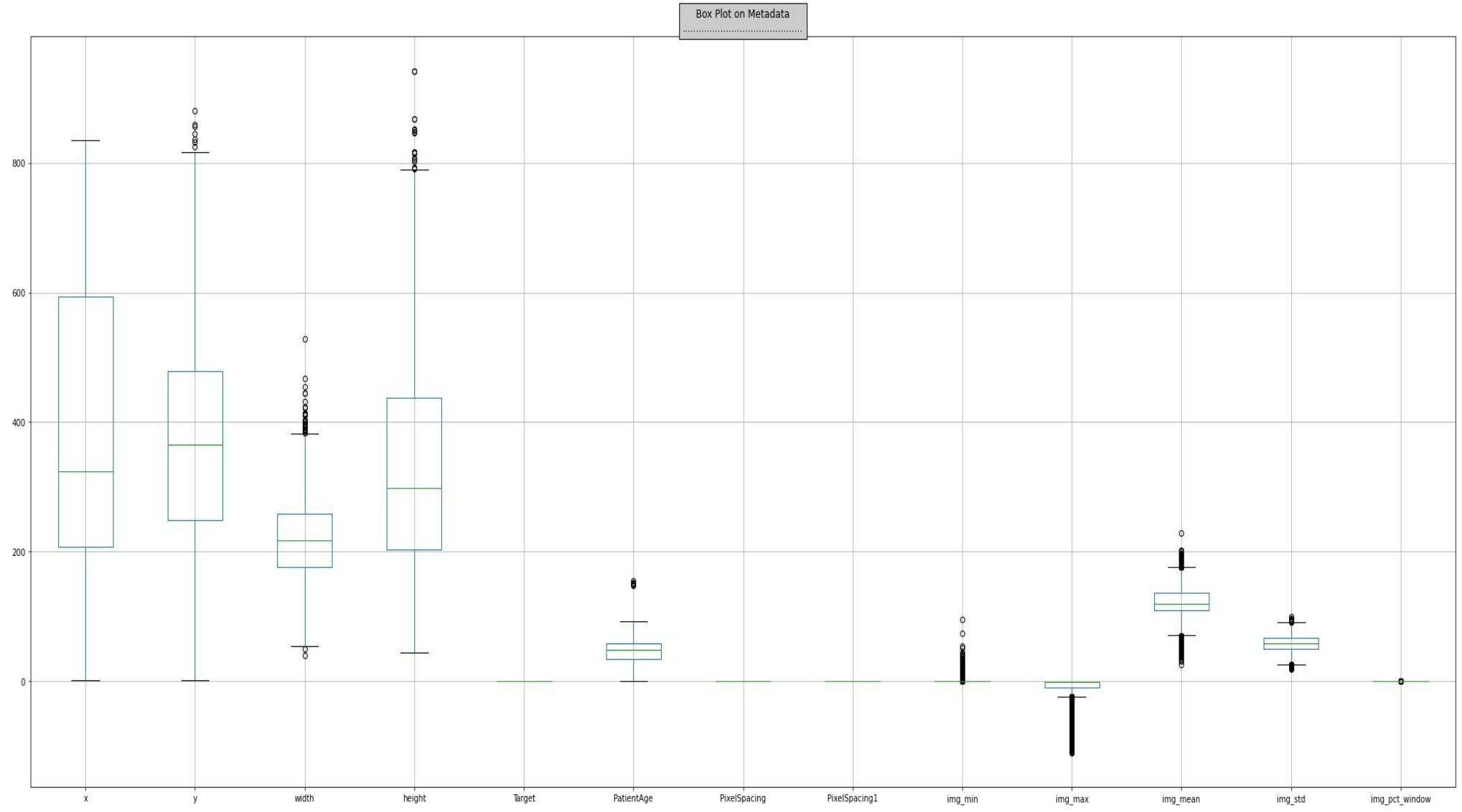
Observation: I guess this graph picks up a sample size of 2000 Pneumonia X-rays and shows the centres of Lung Opacity. This graph could possibly useful to visually see if there any patterns which suggest a huge number of Lesions / Inflammations / Lung Opacities observed in a certain part of the Chest. But in this particular graph it looks more or less equally spread out with slightly an increase in density on the left side maybe. **These kind of graphs could probably come to use during Pandemics and Outbreaks.**

5. Multi Variate Analysis:

In this analysis, we use more than 2 features from the data set and reveal the relationship among several features simultaneously. We have used pair plot, correlation plot, heat map and box plot to visualize the multi variables.






Box Plot on Metadata
800
600
400
200
0
x
y
width
height
Target
PatientAge
PixelSpacing
PixelSpacing1
img_min
img_max
img_mean
img_std
img_pct_window

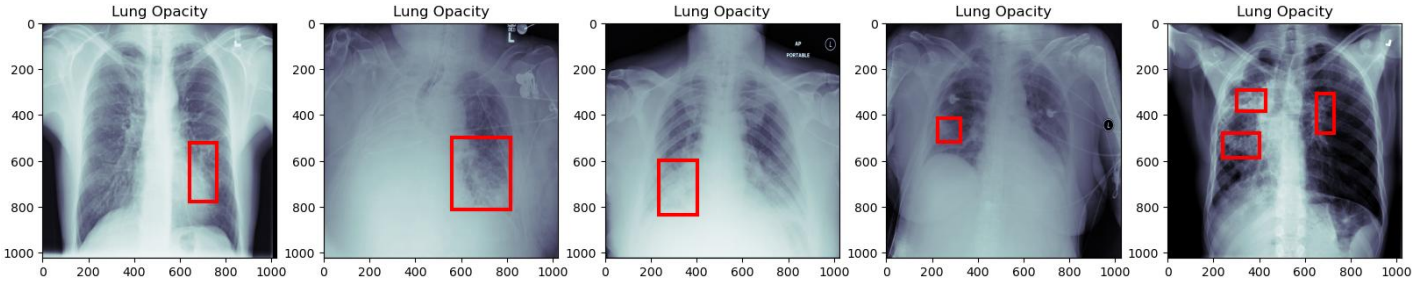
6. Image Pre-Processing:

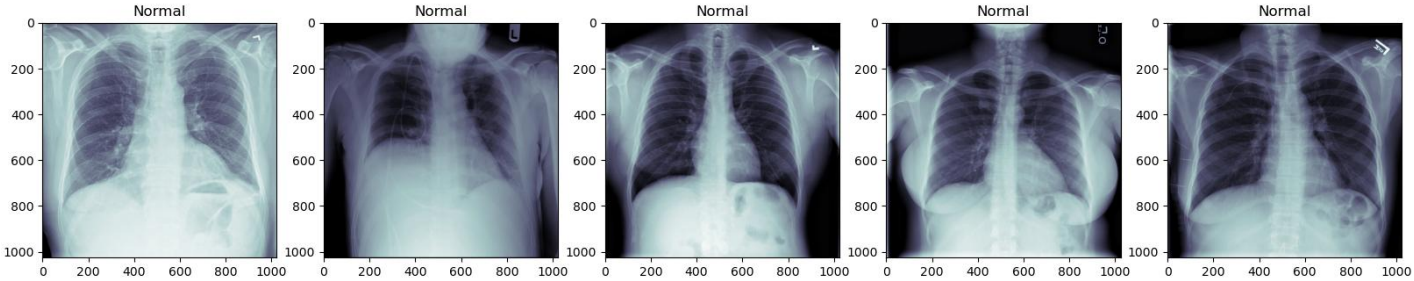
Data pre-processing / Cleaning is the crucial step in machine learning where we will deal with outliers, treating the missing values and remove any unwanted or noisy data. When we are dealing with data such as images, then we need to take care of additional techniques like dealing with pixel brightness - increase or decrease the size, brightness, transforming the geometry of the images, filtering the images, segmentation of the images, saturation / re-saturation.

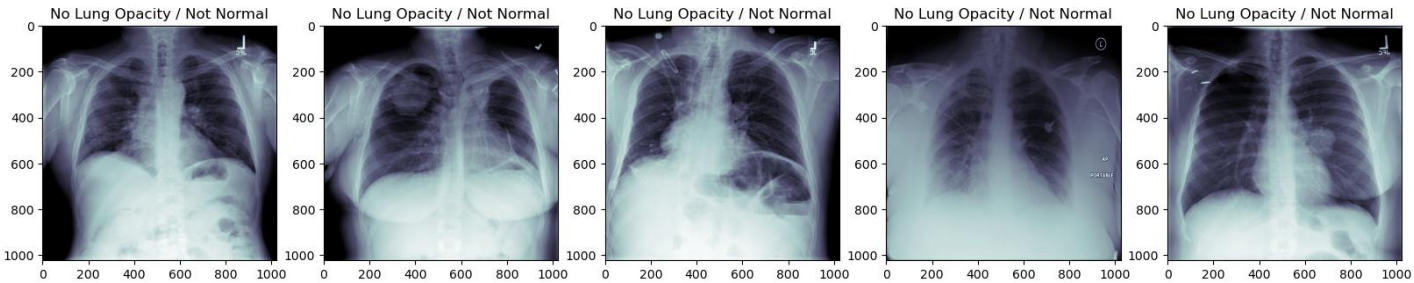
In the current project, we are provided with 26684 training images and 3000 testing images which are dicom images.

We are extracting the image zip files provided into respective folders. For ease of reference, we have segregated the images into different folders based on their class. A new data frame is created containing the image file data like file name, path, class and target details. Below are steps taken to pre-process the image.

A. Convert the images into gray scale
B. Resize the image into width: 224 and height = 224
C. Display the images with the bounding box rectangles.







5. MODEL BUILDING

1. Data related information:

There are 26684 Images which need to be fitted into a CNN model and Trained. Since picking up all the 26684 Dicom images, converting them into Pixel_arrays of 224 x 224 x 3 (RGB layers) would mean a huge of amount of Data for Training, Validation and Testing.
We decided to hand pick 6000 sample Images with the right proportion of 3000 Pneumonia

Images (Target = 1) and 3000 Not Pneumonia Images (Target = 0) to ensure there is no Target Bias while training and Validating the Model

Used from **pydicom.pixel_data_handlers.util import apply_color_lut** libraries and **rgb = apply_color_lut(img_3, palette='PET')**function to get the RGB layers of dimension 224 x 224

So the final X array shape is 6000 x 224 x 224 x 3

Y variable was the Targets (0 & 1) with shape **6000 x 1**

Converted y to_categorical [1,0] array so the shape changed to **6000 x 2**

Split X into X_train and X_test in 80% and 20% ratio with **stratify=y** (to ensure we maintain the y proportion) this resulted in the following

X_train	(4800, 224, 224, 3)	–	Master Training dataset
y_train	(4800, 2)	–	Master Training dataset
X_test	(1200, 224, 224, 3)	–	Test dataset
y_test	(1200, 2)	–	Test dataset

we further split the X_train & y_train into X_train1 – Training dataset and X_val and y_val (Validation dataset) with the same 80% and 20% ratio and again with **stratify = "y_train"** and got the following

X_train1	(3840, 224, 224, 3)
y_train1	(3840, 2)
X_val	(960, 224, 224, 3)
y_val	(960, 2)

Model:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 109, 109, 32)	4736
max_pooling2d (MaxPooling2D)	(None, 54, 54, 32)	0
batch_normalization (BatchNormalization)	(None, 54, 54, 32)	128
conv2d_1 (Conv2D)	(None, 50, 50, 64)	51264
batch_normalization_1 (BatchNormalization)	(None, 50, 50, 64)	56
max_pooling2d_1 (MaxPooling 2D)	(None, 25, 25, 64)	0
flatten (Flatten)	(None, 40000)	0
dense (Dense)	(None, 128)	5120128
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 2)	130

Total params:	5,184,898
Trainable params:	5,184,706
Non-trainable params:	192

Model2:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 109, 109, 32)	4736
max_pooling2d (MaxPooling2D)	(None, 54, 54, 32)	0
batch_normalization (BatchNormalization)	(None, 54, 54, 32)	128
conv2d_1 (Conv2D)	(None, 50, 50, 64)	51264
max_pooling2d_1 (MaxPooling 2D)	(None, 25, 25, 64)	0
batch_normalization_1 (BatchNormalization)	(None, 25, 25, 64)	256
conv2d_2 (Conv2D)	(None, 23, 23, 128)	73856
max_pooling2d_2 (MaxPooling 2D)	(None, 11, 11, 128)	0
batch_normalization_2 (BatchNormalization)	(None, 11, 11, 128)	512
flatten (Flatten)	(None, 15488)	0
dense (Dense)	(None, 128)	1982592
dense_1 (Dense)	(None, 84)	10836
dense_2 (Dense)	(None, 42)	3570
dense_3 (Dense)	(None, 2)	86

Total params: 2,127,836
Trainable params: 2,127,388
Non-trainable params: 448

Model 2a:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 220, 220, 32)	2432
max_pooling2d (MaxPooling2D)	(None, 110, 110, 32)	0
batch_normalization (BatchNormalization)	(None, 110, 110, 32)	128
conv2d_1 (Conv2D)	(None, 106, 106, 64)	51264
max_pooling2d_1 (MaxPooling2D)	(None, 53, 53, 64)	0
batch_normalization_1 (BatchNormalization)	(None, 53, 53, 64)	256
conv2d_2 (Conv2D)	(None, 51, 51, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 25, 25, 128)	0
batch_normalization_2 (BatchNormalization)	(None, 25, 25, 128)	512
dropout (Dropout)	(None, 25, 25, 128)	0
conv2d_3 (Conv2D)	(None, 23, 23, 256)	295168
max_pooling2d_3 (MaxPooling2D)	(None, 11, 11, 256)	0
batch_normalization_3 (BatchNormalization)	(None, 11, 11, 256)	1024
dropout_1 (Dropout)	(None, 11, 11, 256)	0
flatten (Flatten)	(None, 30976)	0
dense (Dense)	(None, 256)	7930112
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 128)	32896
dense_2 (Dense)	(None, 64)	8256
dense_3 (Dense)	(None, 32)	2080
dense_4 (Dense)	(None, 2)	66

Total params: 8,398,050
Trainable params: 8,397,090
Non-trainable params: 960

Model 3:

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 220, 220, 32)	2432
max_pooling2d (MaxPooling2D)	(None, 110, 110, 32)	0
conv2d_1 (Conv2D)	(None, 106, 106, 64)	51264
max_pooling2d_1 (MaxPooling2D)	(None, 53, 53, 64)	0
conv2d_2 (Conv2D)	(None, 51, 51, 128)	73856
max_pooling2d_2 (MaxPooling2D)	(None, 25, 25, 128)	0
batch_normalization (BatchNormalization)	(None, 25, 25, 128)	512
conv2d_3 (Conv2D)	(None, 23, 23, 256)	295168
max_pooling2d_3 (MaxPooling2D)	(None, 11, 11, 256)	0
dropout (Dropout)	(None, 11, 11, 256)	0
flatten (Flatten)	(None, 30976)	0
dense (Dense)	(None, 128)	3965056
dense_1 (Dense)	(None, 84)	10836
dense_2 (Dense)	(None, 42)	3570
dense_3 (Dense)	(None, 2)	86

Total params: 4,402,780
Trainable params: 4,402,524
Non-trainable params: 256

Above are the 4 models, we have designed to train the model. There are various layers in the sequential model as below:

Conv2D:
It is a 2D convolution layer which creates a kernel with layers input that produces the tensor of outputs.

Maxpooling 2D:
It downsamples the input along with its spatial dimensions like height and width by taking maximum value over an input for each channel of input

Batch Normalization:
It is a technique for training deep neural networks which standardizes the inputs to a layer for each batch by maintaining the mean output close to 0.

Dropout:
This randomly sets the input to units 0 with a frequency of rate at each step during training time, which helps with preventing the data overfitting problem.

Flatten:
It flattens the input without affecting the batch size. If inputs are shaped (batch,) without a feature axis, then flattening adds an extra channel dimension (batch, 1)

Dense:
Dense layer is the regular deeply connected neural network layer. It is most common and frequently used layer. Dense layer does the below operation on the input and return the output.

Activation Functions:
These functions are used to introduce non-linearity to the output of the neuron given an input or set of inputs which basically decides whether to activate a neuron or not. We used Relu and Sigmoid functions in our model designing.

ReLU:
Rectified Lienar Unit is a piecewise linear function that will output the input directly if it is positive otherwise it will output zero.

Sigmoid:
Non linear activation function where the output of the unit will be always between 0 and 1

Optimizer:
An optimizer is the function that modifies the attributes of the neural network such as weights and learning rate helping in reducing the overall loss and increasing the accuracy of the model. We have used Adam optimizer in the model.

Adam Optimizer:
Derived from Adaptive Moment Estimation It is a adaptive learning rate optimization algorithm that's been designed for training deep neural networks. It is a combination of gradient descent with momentum algorithm RMSP algorithm. It requires less memory and is efficient when working with large problems involving lots of data.

Loss Function:
It is a function that compares the target and predicted output values, measures how well the the neural network models the training data. While training the model, our priority is to reduce the loss between the predicted and target output thus increasing the accuracy. We used binary cross entropy, categorical cross entropy loss functions in the model building.

Binary Cross Entropy:
It is the loss function used in binary classification tasks where we have only 2 choices 0 or 1, Yes or No etc just as in our project where we have 2 targets either 0 - No Pneumonia or 1 - Pneumonia.

Categorical Cross Entropy:
It is the loss function used in multi class classification problems where the output can be any of the many possible categories and model decides the which category data belongs to. We can say binary cross entropy loss is special case of categorical cross entropy where the targets are only 2 classes.

Metrics:
Metrics are used to monitor and measure the performance of a model during the training and testing phase. These values are recorded at end of each epoch. We used accuracy as metrics while training the model.

Batch & Epochs:
- The batch size is a hyperparameter of gradient descent that controls the number of training samples to work through before the model's internal parameters are updated.
- The number of epochs is a hyperparameter of gradient descent that controls the number of complete passes through the training dataset.

Callbacks:
You define and use a callback when you want to automate some tasks after every training/epoch that help you have controls over the training process. This includes stopping training when you reach a certain accuracy/loss score, saving your model as a checkpoint after each successful epoch, adjusting the learning rates over time, and more. We used callback to early stop the model after reaching certain loss score.

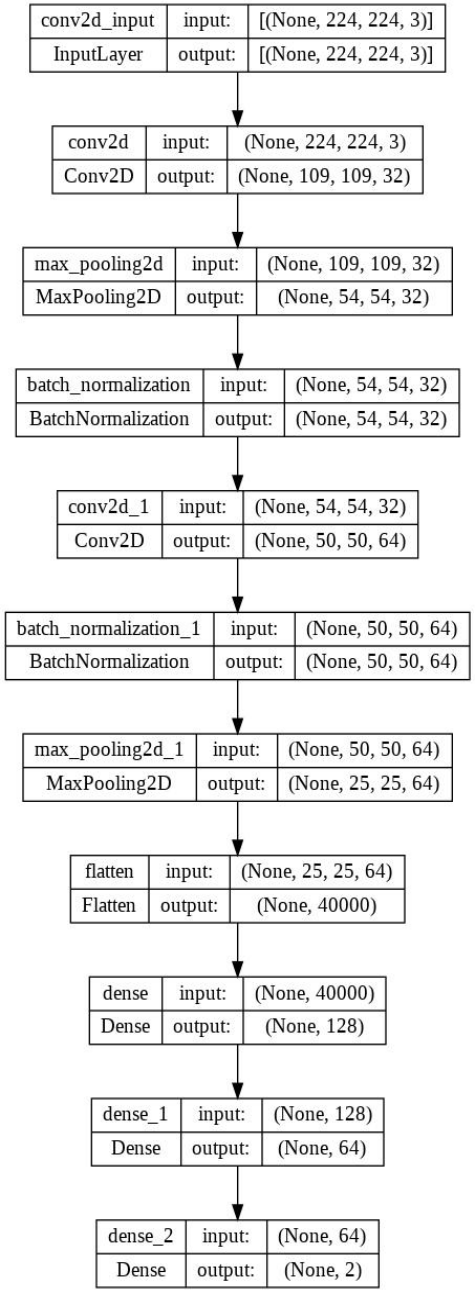
Evaluate:
Evaluation is a process during development of the model to check whether the model is best fit for the given problem and corresponding data. Keras model provides a function, evaluate which does the evaluation of the model.

Predict:
It is used to predict the class for test data / new data instances using our final classification model. It returns the final output of the model which are actual predictions for all the test samples.

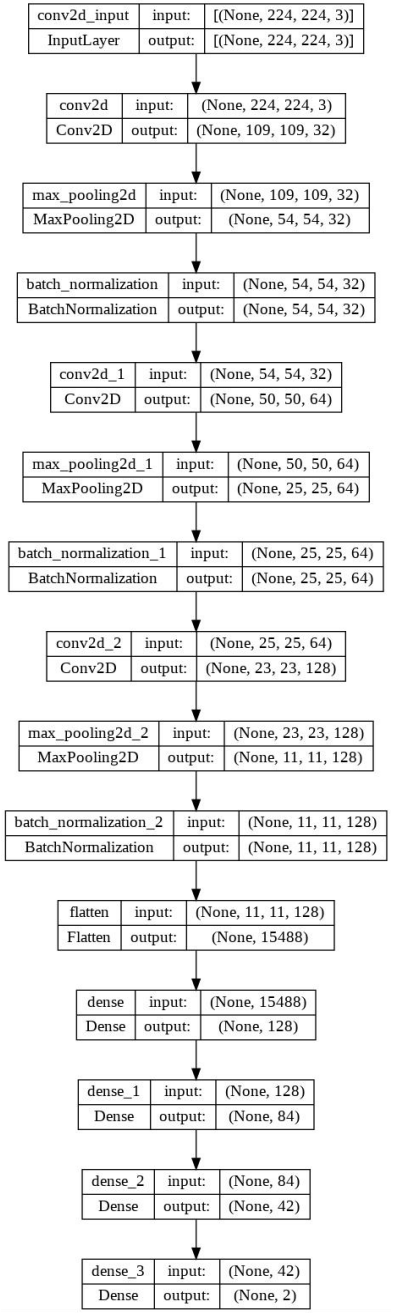
Compile:
It is the final step in designing the model where we compile the model using arguments like optimizer, loss and metrics thus finally making it completely ready to use.

Below are the visual representation of the flow of the layers in the four models we have designed.

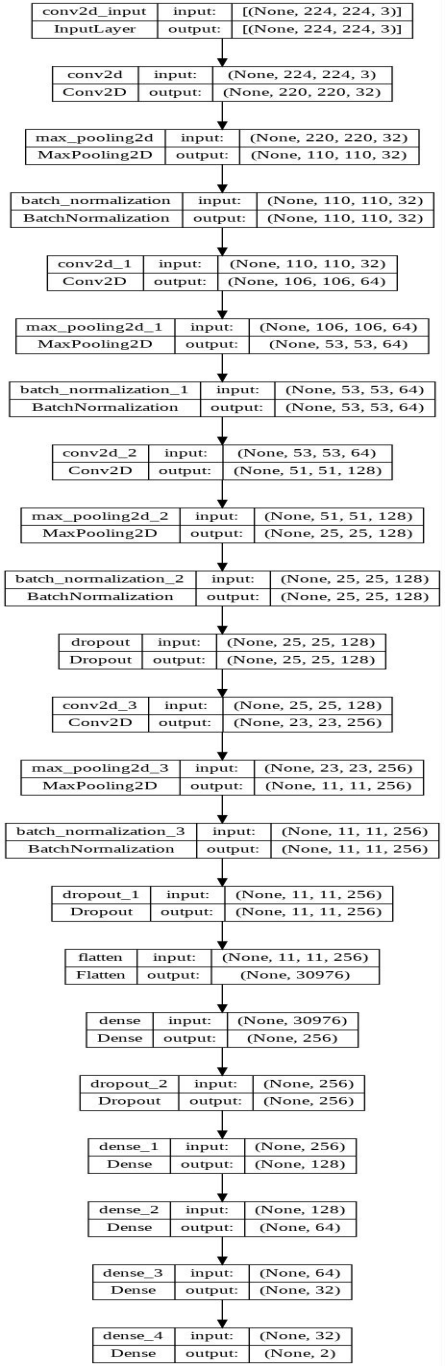
model1



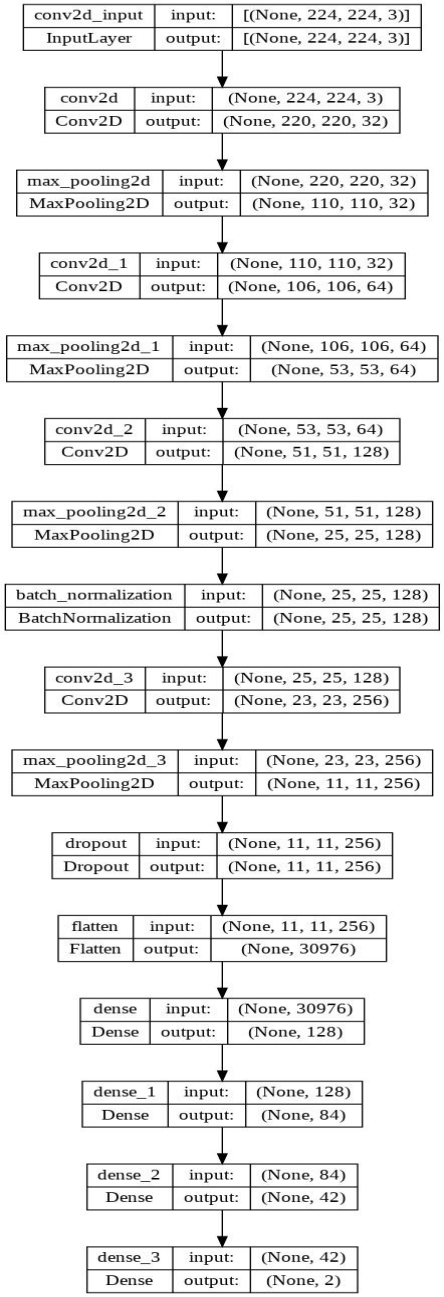
model2



model2a

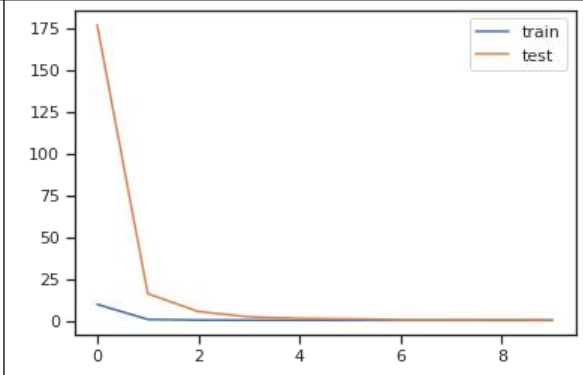
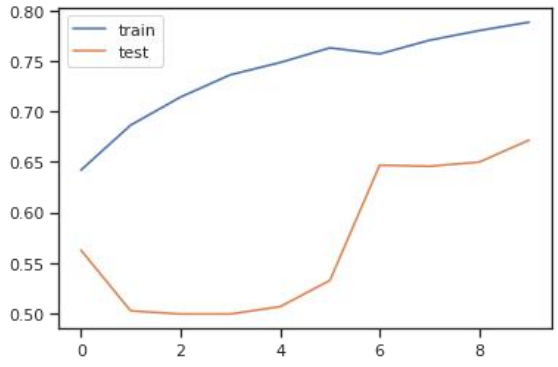
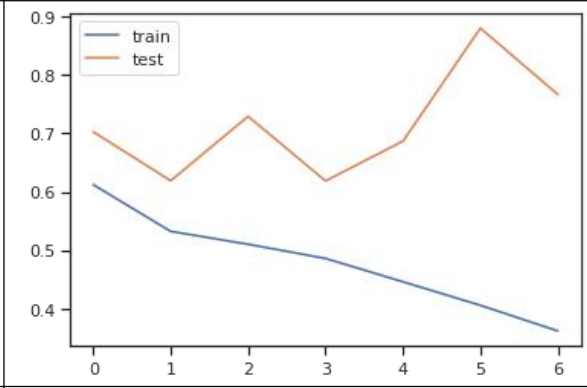
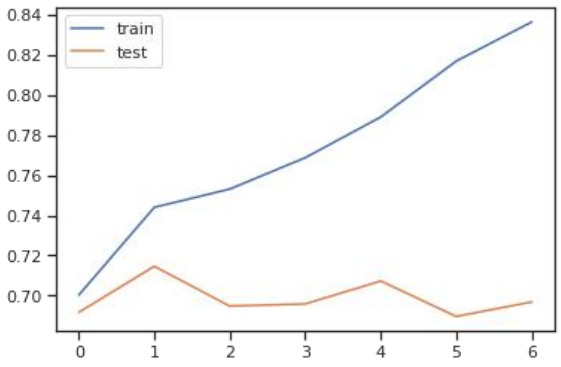
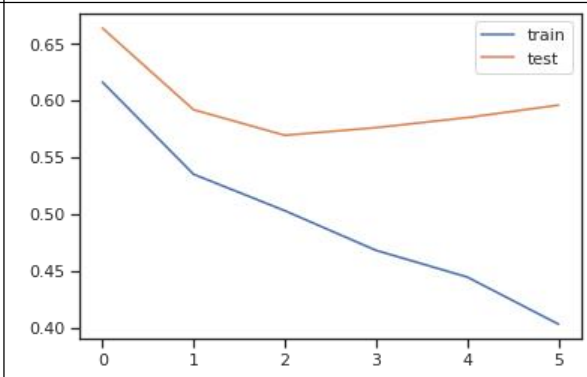
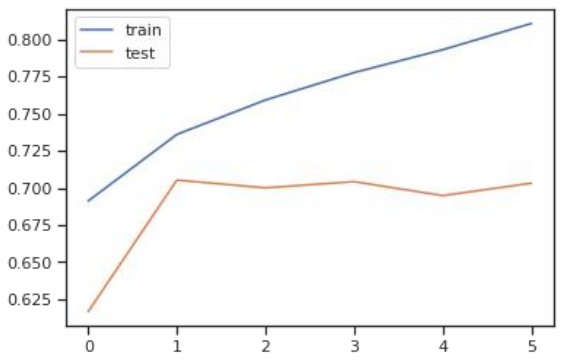
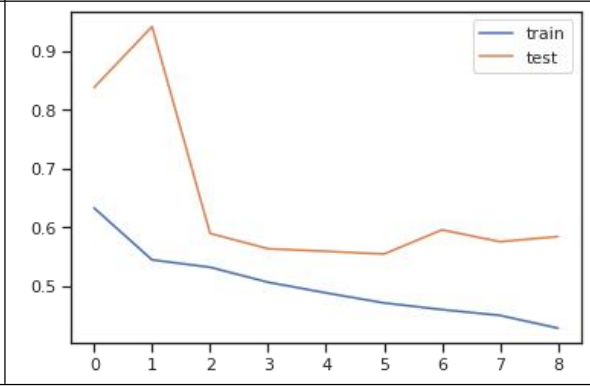
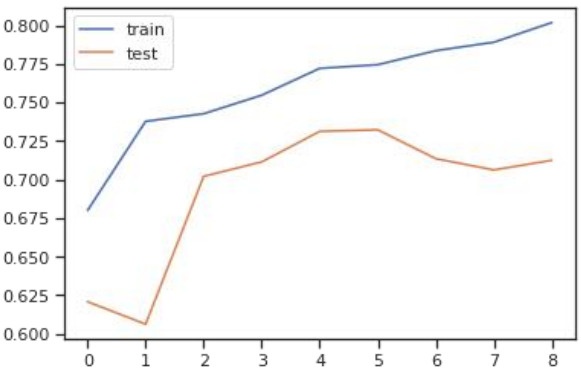


model3



6. MODEL TRAINING

After building and compiling the model, we have to train the model by fitting the designed model to training and validation data set. We used batch size of 128 with 10 epochs. Using below plot figures, we can compare the loss and accuracy across training and validation data. This gives us the loss values and metrics for the model. It gives us the how well the model is generalizing the similar data on which it is trained. When the model is well fitted, it produces more accurate outcomes on the test data / new instances. In below table let us compare the loss and accuracy graphs for each of the 4 models trained. It shows how the loss and accuracies are varied across various epochs.

	Train / Validation Loss Graph	Train / Validation Accuracy Graph
Model		
Model2		
Model_2a		
Model3		

Once the model is trained on fitting the training and validation data set, we evaluated and predicted for test data to see how the 4 models are performed. This can be achieved through comparing the performance metrics like precision, recall, f1-score, accuracy.

Accuracy: Accuracy represents the number of correctly classified data instances over the total number of data instances.

Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positive observations.

Recall (Sensitivity): Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

F1-Score: The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean.

Let us compare the above performance metrics for the 4 models designed and trained to see which model gives the best performance in comparison to other.

Classification Report Comparison					
Model		**precision**	**recall**	**f1-score**	**support**
	0	0.73	0.62	0.67	600
	1	0.67	0.77	0.72	600
	accuracy			0.69	1200
	macro avg	0.70	0.69	0.69	1200
	weighted avg	0.70	0.69	0.69	1200
Model2		**precision**	**recall**	**f1-score**	**support**
	0	0.69	0.71	0.70	600
	1	0.70	0.68	0.69	600
	accuracy			0.70	1200
	macro avg	0.70	0.70	0.70	1200
	weighted avg	0.70	0.70	0.70	1200
Model2a		**precision**	**recall**	**f1-score**	**support**
	0	0.73	0.66	0.69	600
	1	0.69	0.75	0.72	600
	accuracy			0.70	1200
	macro avg	0.71	0.71	0.70	1200
	weighted avg	0.71	0.70	0.70	1200
Model3		**precision**	**recall**	**f1-score**	**support**
	0	0.74	0.69	0.71	600
	1	0.71	0.76	0.73	600
	accuracy			0.72	1200
	macro avg	0.73	0.72	0.72	1200
	weighted avg	0.73	0.72	0.72	1200

Model	**Training Accuracy**	**Validation Accuracy**	**Testing Accuracy**
Model	79%	67%	69%
Model 2	83%	70%	69%
Model 2a	81%	70%	70%
Model 3	80%	73%	72%

7. IMPROVING MODEL PERFORMANCE

In the initial run, all the models are performing around 70% accuracy against the training and validation data set.

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
Model	79%	67%	69%
Model 2	83%	70%	69%
Model 2a	81%	70%	70%
Model 3	80%	73%	72%

All the Models seem to be high bias and high variance but when compared to the bias the variances are relatively lower.

There is almost 17-20% of bias in the models while the variances are 7 – 12%

The Validation accuracies are almost matching with the Final Testing accuracies and there isn't much difference there, not sure if it is due to the similar volume of data used for Validation set and Test set.

This performance can be improved by tweaking some parameters, adding / deleting layers, using other activation functions and so on. Below are some steps that can be taken to improve the model performance:

1. **Increase the size of datasets used to train the model** - The more the data, the better the chances for model to understand the patterns of the dataset in classifying them. We have considered only a sample of 6000 images for the Milestone 1.

2. **Image Augmenting:** All the Images might not be in the same exact shape, there could be some tilting, stretching and other orientational issues. We need to ensure the algorithm is Trained on all these so that when it encounters such alignment issues model can accordingly take note of it.

3. **Overfitting of the model** - When the model is really performing well against the Training data but underperforming on the Validation datasets or test data set, then the model is said to over fit. Our models seem to be having a high bias so first of all needs to be improved on the Training dataset but slightly under performing on the Validation and Test data sets. we can probably use more dropout to overcome the problem. Dropout helps in randomly switching off the neurons and reduces the complexity of the architecture.

4. **Lowering the learning rate** - Lowering the learning rate can help in finding an epoch where the model is performing better before over fitting. We have tried a few learning rates but probably there are some more to be explored

5. **Keras Tuner:** We could use Keras Tuner to figure out the best number of Convolutional layers, Neurons, kernels, Dense layers and Neurons along with the best optimizer, best learning rate etc and finally use that to get the best accuracies.

6. **Transfer Learning** - Transfer learning is improving of learning in a new task by transfer of knowledge from a related task that was already trained on similar tasks, in this case image classification.

8. CONCLUSION

Let us summarize the tasks we have done through out in order.

1. Extracted all the required datasets and image data provided.
2. EDA - Where pre-processed the data provided along with image data and visualizing them
3. Prepare train, validation and test datasets
4. Designed basic CNN models to train the data.
5. Compiled and fitted the data into 4 models designed.
6. Compared the flow of loss and accuracy across the epochs for each model.
7. Evaluated the model against the test data and predicted the targets.
8. Print the classification reports for models to understand the performance metrics and compare.
9. Briefed on how the model can be improved to achieve higher accuracy.

With this we are concluding the interim report.

Thank You