# Pneumonia Detection Challenge

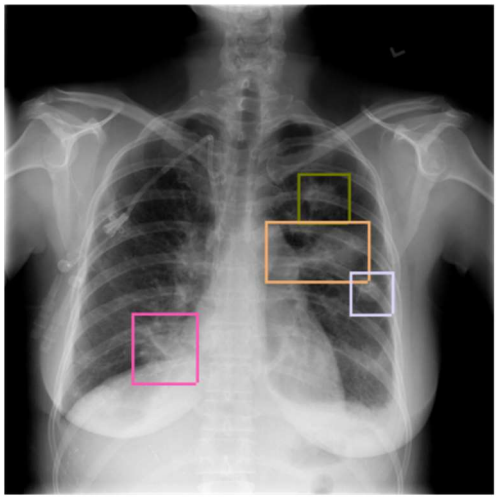
Maddipalli Gopalakrishna, Venkatesh Nagilla, Roshan Patel,
Pavan Krishna Rampal

**Capstone Project Interim Report | 22-Nov-2020**

# Content

# Introduction

# What is Pneumonia?



**Pneumonia** is an infection in one or both lungs. Bacteria, viruses, and fungi cause it. The infection causes inflammation in the air sacs in your lungs, which are called alveoli.

Pneumonia accounts for over 15% of all deaths of children under 5 years old internationally. In 2017, 920,000 children under the age of 5 died from the disease. It requires review of a chest radiograph (CXR) by highly trained specialists and confirmation through clinical history, vital signs, and laboratory exams. Pneumonia usually manifests as an area or areas of increased opacity on CXR. However, the diagnosis of pneumonia on CXR is complicated because of several other conditions in the lungs such as fluid overload (pulmonary edema), bleeding, volume loss (atelectasis or collapse), lung cancer, or post-radiation or surgical changes. Outside of the lungs, fluid in the pleural space (pleural effusion) also appears as increased opacity on CXR. When available, comparison of CXRs of the patient taken at different time points and correlation with clinical symptoms and history are helpful in making the diagnosis.

**CXRs are the most performed diagnostic imaging study. Several factors such as positioning of the patient and depth of inspiration can alter the appearance of the CXR, complicating interpretation further. In addition, clinicians are faced with reading high volumes of images every shift.**

Details about the data and dataset files are given in below link,

https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data

The goal is to build a pneumonia detection system, to locate the position of inflammation in an image.

Tissues with sparse material, such as lungs which are full of air, do not absorb the X-rays and appear black in the image. Dense tissues such as bones absorb X-rays and appear white in the image.

While we are theoretically detecting “lung opacities”, there are lung opacities that are not pneumonia related.

In the data, some of these are labeled “Not Normal No Lung Opacity”. This extra third class indicates that while pneumonia was determined not to be present, there was nonetheless some type of abnormality on the image and oftentimes this finding may mimic the appearance of true pneumonia.

Dicom original images: - Medical images are stored in a special format called DICOM files (*.dcm). They contain a combination of header metadata as well as underlying raw image arrays for pixel data.

# The objective of the project is

- Learn to how to do build an Object Detection Model
- Use transfer learning to fine-tune a model.
- Learn to set the optimizers, loss functions, epochs, learning rate, batch size, check pointing, early stopping etc.
- Read different research papers of given domain to obtain the knowledge of advanced models for the given problem.

# Exploratory Data Analysis

## 1. Data Preparation

Available data

stage_2_train_labels.csv

| | patientId | x | y | width | height | Target |
|---|---|---|---|---|---|---|
| 0 | 0004cfab-14fd-4e49-80ba-63a80b6bddd6 | NaN | NaN | NaN | NaN | 0 |
| 1 | 00313ee0-9eaa-42f4-b0ab-c148ed3241cd | NaN | NaN | NaN | NaN | 0 |
| 2 | 00322d4d-1c29-4943-afc9-b6754be640eb | NaN | NaN | NaN | NaN | 0 |
| 3 | 003d8fa0-6bf1-40ed-b54c-ac657f8495c5 | NaN | NaN | NaN | NaN | 0 |
| 4 | 00436515-870c-4b36-a041-de91049b9ab4 | 264.0 | 152.0 | 213.0 | 379.0 | 1 |
| 5 | 00436515-870c-4b36-a041-de91049b9ab4 | 562.0 | 152.0 | 256.0 | 453.0 | 1 |
| 6 | 00569f44-917d-4c86-a842-81832af98c30 | NaN | NaN | NaN | NaN | 0 |
| 7 | 006cec2e-6ce2-4549-bffa-eadfcd1e9970 | NaN | NaN | NaN | NaN | 0 |
| 8 | 00704310-78a8-4b38-8475-49f4573b2dbb | 323.0 | 577.0 | 160.0 | 104.0 | 1 |
| 9 | 00704310-78a8-4b38-8475-49f4573b2dbb | 695.0 | 575.0 | 162.0 | 137.0 | 1 |

stage_2_detailed_class_info.csv

| | patientId | class |
|---|---|---|
| 0 | 0004cfab-14fd-4e49-80ba-63a80b6bddd6 | No Lung Opacity / Not Normal |
| 1 | 00313ee0-9eaa-42f4-b0ab-c148ed3241cd | No Lung Opacity / Not Normal |
| 2 | 00322d4d-1c29-4943-afc9-b6754be640eb | No Lung Opacity / Not Normal |
| 3 | 003d8fa0-6bf1-40ed-b54c-ac657f8495c5 | Normal |
| 4 | 00436515-870c-4b36-a041-de91049b9ab4 | Lung Opacity |
| 5 | 00436515-870c-4b36-a041-de91049b9ab4 | Lung Opacity |
| 6 | 00569f44-917d-4c86-a842-81832af98c30 | No Lung Opacity / Not Normal |
| 7 | 006cec2e-6ce2-4549-bffa-eadfcd1e9970 | No Lung Opacity / Not Normal |
| 8 | 00704310-78a8-4b38-8475-49f4573b2dbb | Lung Opacity |
| 9 | 00704310-78a8-4b38-8475-49f4573b2dbb | Lung Opacity |

There is Data available of 26684 Patient.

# Merged Data for Detailed info

| | patientId | x | y | width | height | Target | aspect_ratio | area | age | gender |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0004cfab-14fd-4e49-80ba-63a80b6bddd6 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 51 | F |
| 1 | 00313ee0-9eaa-42f4-b0ab-c148ed3241cd | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 48 | F |
| 2 | 00322d4d-1c29-4943-afc9-b6754be640eb | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 19 | M |
| 3 | 003d8fa0-6bf1-40ed-b54c-ac657f8495c5 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 28 | M |
| 4 | 00436515-870c-4b36-a041-de91049b9ab4 | 264.0 | 152.0 | 213.0 | 379.0 | 1 | 0.562005 | 80727.0 | 32 | F |

Shape of train data: - (30227, 10)

| | |
|---|---|
| No Lung Opacity / Not Normal | 11821 |
| Lung Opacity | 9555 |
| Normal | 8851 |
| Name: class | Dtype: int64 |

| | x | y | width | height | Target | aspect_ratio | area | age | gender | class | patientId |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 51 | F | No Lung Opacity / Not Normal | 0004cfab-14fd-4e49-80ba-63a80b6bddd6 |
| 1 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 48 | F | No Lung Opacity / Not Normal | 00313ee0-9eaa-42f4-b0ab-c148ed3241cd |
| 2 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 19 | M | No Lung Opacity / Not Normal | 00322d4d-1c29-4943-afc9-b6754be640eb |
| 3 | NaN | NaN | NaN | NaN | 0 | NaN | NaN | 28 | M | Normal | 003d8fa0-6bf1-40ed-b54c-ac657f8495c5 |
| 4 | 264.0 | 152.0 | 213.0 | 379.0 | 1 | 0.562005 | 80727.0 | 32 | F | Lung Opacity | 00436515-870c-4b36-a041-de91049b9ab4 |

# 2. Data Overview

## 2.1 Sample DICOM file

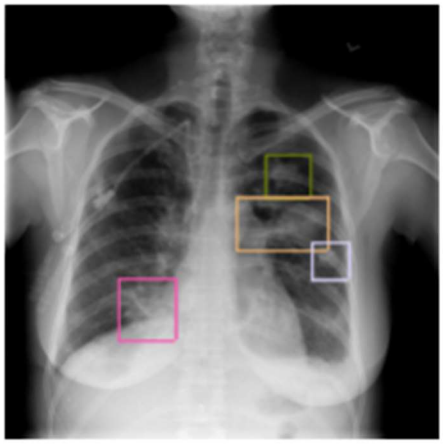
```
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 202
(0002, 0001) File Meta Information Version       OB: b'\x00\x01'
(0002, 0002) Media Storage SOP Class UID         UI: Secondary Capture Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 1.2.276.0.7230010.3.1.4.8323329.28530.1517874485.775526
(0002, 0010) Transfer Syntax UID                 UI: JPEG Baseline (Process 1)
(0002, 0012) Implementation Class UID            UI: 1.2.276.0.7230010.3.0.3.6.0
(0002, 0013) Implementation Version Name         SH: 'OFFIS_DCMTK_360'
-------------------------------------------------
(0008, 0005) Specific Character Set              CS: 'ISO_IR 100'
(0008, 0016) SOP Class UID                       UI: Secondary Capture Image Storage
(0008, 0018) SOP Instance UID                    UI: 1.2.276.0.7230010.3.1.4.8323329.28530.1517874485.775526
(0008, 0020) Study Date                          DA: '19010101'
(0008, 0030) Study Time                          TM: '000000.00'
(0008, 0050) Accession Number                    SH: ''
(0008, 0060) Modality                            CS: 'CR'
(0008, 0064) Conversion Type                     CS: 'WSD'
(0008, 0090) Referring Physician's Name          PN: ''
(0008, 103e) Series Description                  LO: 'view: PA'
(0010, 0010) Patient's Name                      PN: '0004cfab-14fd-4e49-80ba-63a80b6bddd6'
(0010, 0020) Patient ID                          LO: '0004cfab-14fd-4e49-80ba-63a80b6bddd6'
(0010, 0030) Patient's Birth Date                DA: ''
(0010, 0040) Patient's Sex                       CS: 'F'
(0010, 1010) Patient's Age                       AS: '51'
(0018, 0015) Body Part Examined                  CS: 'CHEST'
(0018, 5101) View Position                       CS: 'PA'
(0020, 000d) Study Instance UID                  UI: 1.2.276.0.7230010.3.1.2.8323329.28530.1517874485.775525
(0020, 000e) Series Instance UID                 UI: 1.2.276.0.7230010.3.1.3.8323329.28530.1517874485.775524
(0020, 0010) Study ID                            SH: ''
(0020, 0011) Series Number                       IS: "1"
(0020, 0013) Instance Number                     IS: "1"
(0020, 0020) Patient Orientation                 CS: ''
(0028, 0002) Samples per Pixel                   US: 1
(0028, 0004) Photometric Interpretation          CS: 'MONOCHROME2'
(0028, 0010) Rows                                US: 1024
(0028, 0011) Columns                             US: 1024
(0028, 0030) Pixel Spacing                       DS: [0.14300000000000002, 0.14300000000000002]
(0028, 0100) Bits Allocated                      US: 8
(0028, 0101) Bits Stored                         US: 8
(0028, 0102) High Bit                            US: 7
(0028, 0103) Pixel Representation                US: 0
(0028, 2110) Lossy Image Compression             CS: '01'
(0028, 2114) Lossy Image Compression Method      CS: 'ISO_10918_1'
(7fe0, 0010) Pixel Data                          OB: Array of 142006 elements
```

## 2.2 Sample CXR image (without opacities)

## 2.3 Locating CXR image with bounding box around opacities

| | 0004cfab-14fd-4e49-80ba-63a80b6bddd6 | 00313ee0-9eaa-42f4-b0ab-c148ed3241cd | 00322d4d-1c29-4943-afc9-b6754be640eb | 003d8fa0-6bf1-40ed-b54c-ac657f8495c5 | 00436515-870c-4b36-a041-de91049b9ab4 | 00569f44-917d-4c86-a842-81832af98c30 | 006cec2e-6ce2-4549-bffa-eadfcd1e9970 | 00704310-78a8-4b38-8475-49f4573b2dbb | 008c19e8-a820-403a-930a-bc74a4053664 | 009482dc-3db5-48d4-8580-5c89c4f01334 | 009eb222-eabc-4150-8121-d5a6d06b8ebf |
|---|---|---|---|---|---|---|---|---|---|---|---|
| dicom | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... | ../content/drive/My Drive/Colab Notebooks/caps... |
| label | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| boxes | [] | [] | [] | [] | [[152.0, 264.0, 379.0, 213.0], [152.0, 562.0, ... | [] | [] | [[577.0, 323.0, 104.0, 160.0], [575.0, 695.0, ... | [] | [] | [] |

3 rows × 26684 columns

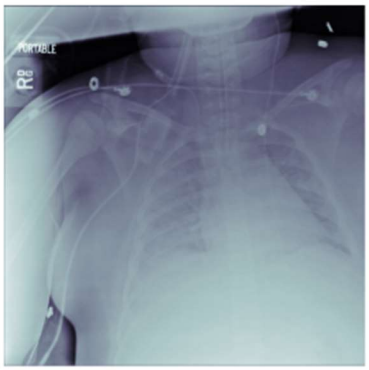


**Observation:-**

1. The minimum number of opacities is 1.
2. the maximum number of opacities is 4.

Few samples of Patients with Lung Opacity

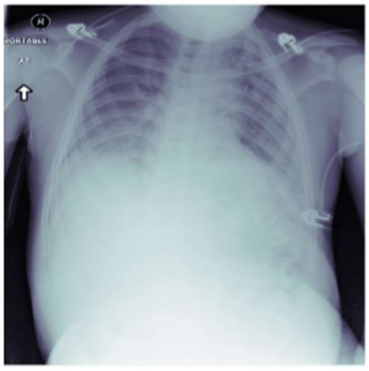
ID: b2592a14-6876-4e33-a1c3-ab09cd7c774f
Modality: CR Age: 11 Sex: F Target: 1
Class: Lung Opacity
Window: 375.0:375.0:198.0:338.0



ID: 988a4bbd-3a14-4b4b-b846-6729e35fcd91
Modality: CR Age: 46 Sex: F Target: 1
Class: Lung Opacity
Window: 183.0:569.0:230.0:185.0

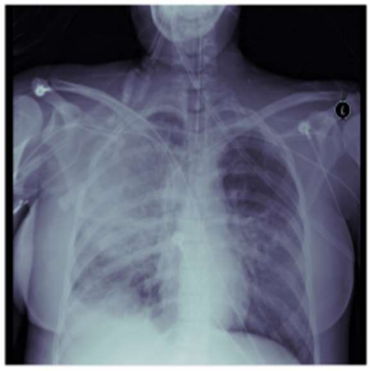
ID: f5dadf63-c828-441a-b595-ea7698087291
Modality: CR Age: 42 Sex: F Target: 1
Class: Lung Opacity
Window: 593.0:533.0:189.0:237.0

ID: 0f1fced1-4c33-44b2-9009-1370aa6287f3
Modality: CR Age: 16 Sex: M Target: 1
Class: Lung Opacity
Window: 255.0:322.0:159.0:124.0



ID: b54c5581-bb40-4c19-9d5a-3be944ef8132
Modality: CR Age: 64 Sex: M Target: 1
Class: Lung Opacity
Window: 133.0:524.0:172.0:139.0

ID: bd7b4e6e-5384-4cce-9732-0dd7f9d15da8
Modality: CR Age: 33 Sex: F Target: 1
Class: Lung Opacity
Window: 597.0:399.0:262.0:603.0



ID: 4a5bc00d-f353-47dd-b014-cd1e7b28bed7
Modality: CR Age: 25 Sex: M Target: 1
Class: Lung Opacity
Window: 616.0:368.0:212.0:456.0

ID: 93986f45-f0ed-44c0-bbc4-15bfe78ec6da
Modality: CR Age: 69 Sex: M Target: 1
Class: Lung Opacity
Window: 642.0:457.0:316.0:408.0

ID: 09b2d54e-3a85-4efd-8204-ea73bc70c405
Modality: CR Age: 50 Sex: F Target: 1
Class: Lung Opacity
Window: 582.0:310.0:276.0:649.0

Few samples of Patients with Lung Opacity (overlay boxes superposed)

ID: 36ca4185-52e1-4cf3-b007-5c7680d2ae74
Modality: CR Age: 35 Sex: M Target: 1
Class: Lung Opacity

ID: 82b9a505-f0ed-4bdd-a3d8-be9b13d720d7
Modality: CR Age: 50 Sex: F Target: 1
Class: Lung Opacity

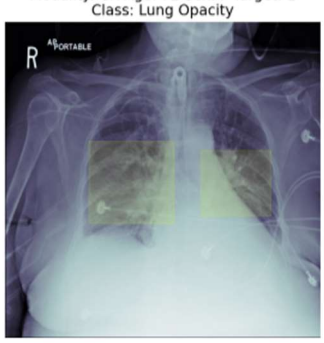
ID: 334c4445-63ce-401a-a1a2-1a467901a1ae
Modality: CR Age: 59 Sex: F Target: 1
Class: Lung Opacity

ID: 8a233483-8ab2-4cc5-b32b-20fe40cc5d77
Modality: CR Age: 24 Sex: M Target: 1
Class: Lung Opacity

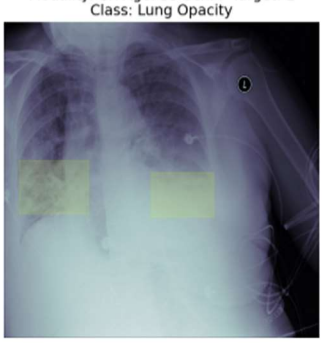
ID: b724406f-51b8-4134-a9eb-2185696cbbfb
Modality: CR Age: 52 Sex: M Target: 1
Class: Lung Opacity

ID: b76086b3-9550-44f9-a2a8-394c7793bac8
Modality: CR Age: 38 Sex: F Target: 1
Class: Lung Opacity

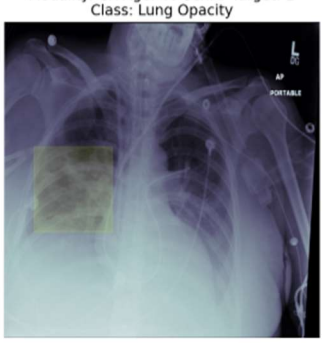
ID: 453a9434-2ca7-4a28-b6f3-833b994e6e73
Modality: CR Age: 72 Sex: F Target: 1
Class: Lung Opacity



ID: 401afe9b-4c20-41cc-b52d-21995c71e872
Modality: CR Age: 55 Sex: F Target: 1
Class: Lung Opacity

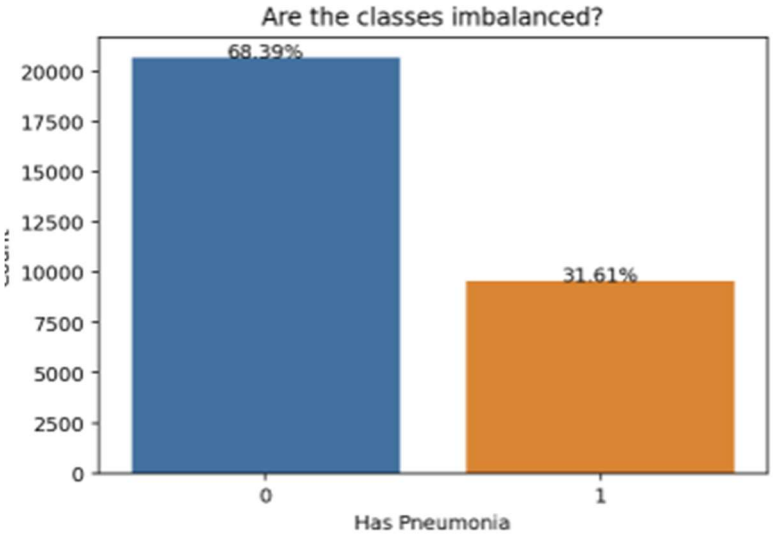


ID: af051401-ddb1-4ef7-be58-4927387ddf51
Modality: CR Age: 40 Sex: F Target: 1
Class: Lung Opacity



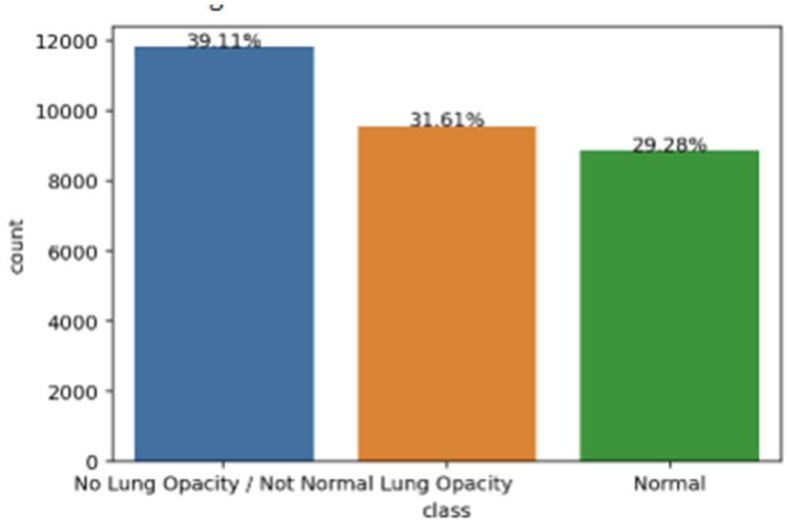
# 3. Univariate Analysis

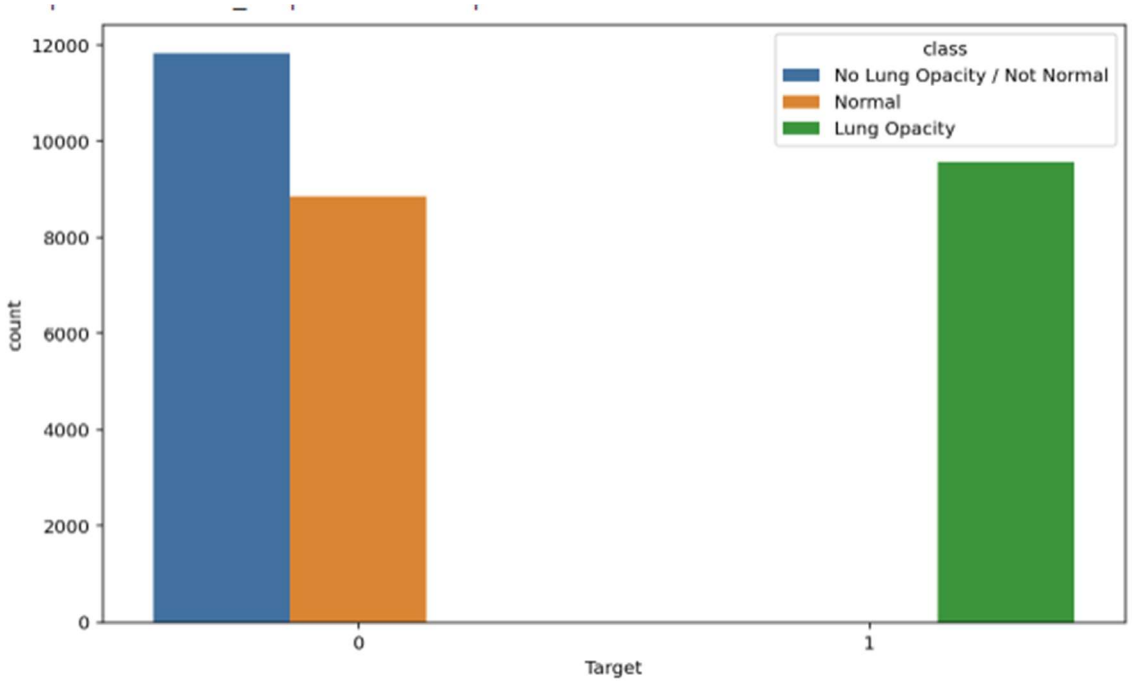
## 3.1 Class Imbalance



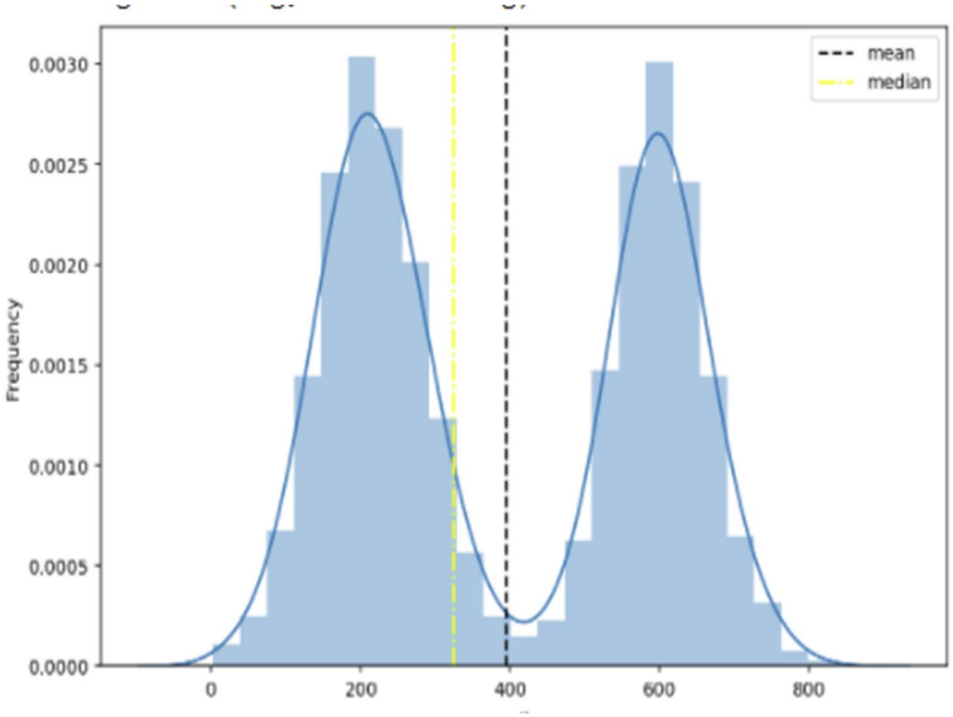

**Observation: -**

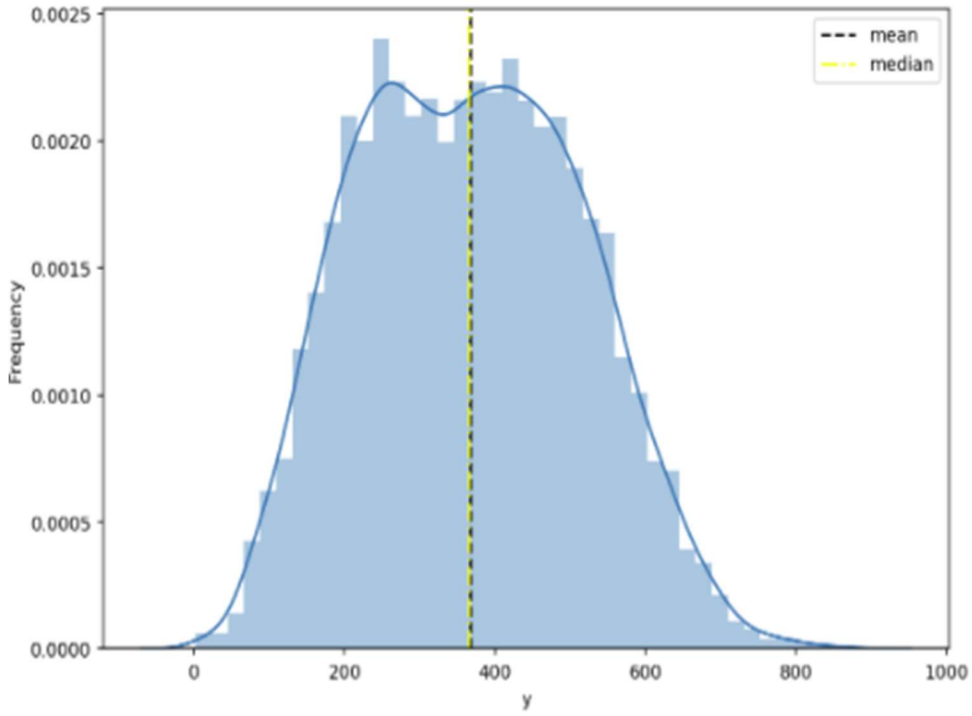
1. From the plot, the data is highly imbalanced.
2. There is almost 5000 positive & 20000 negative points.
3. The imbalance ratio is almost 4:1 (negative: positive)
4. The reason behind the imbalance is that there are almost 11000 data points which are classified as No Lung Opacity / Not Normal, but those points are also considered as negative.

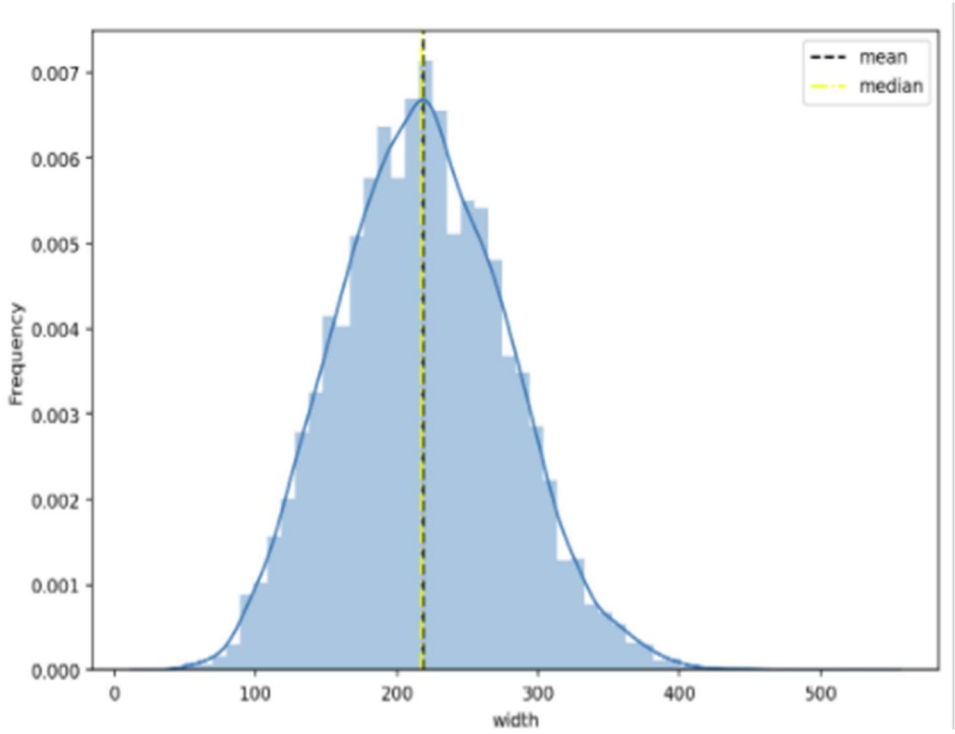


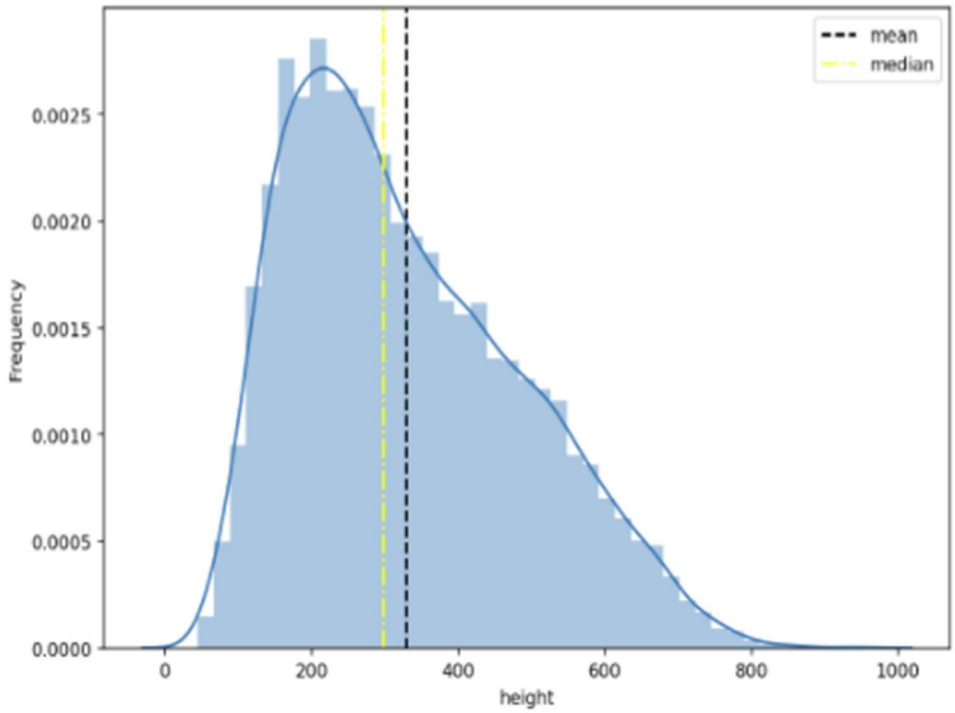



Bounding Box Plots, Coordinates

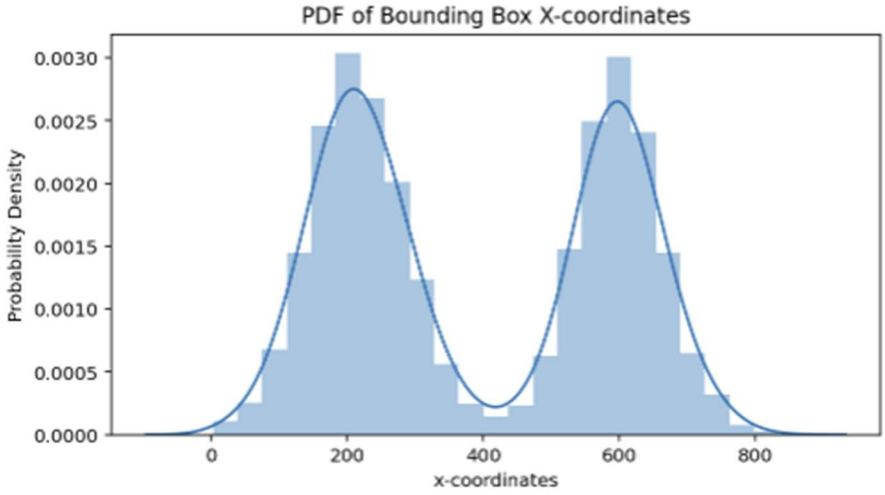


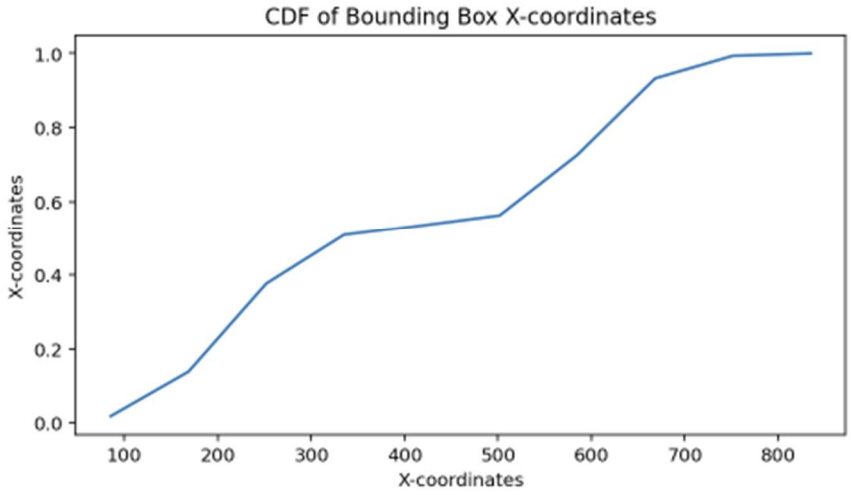

mean
median
Frequency
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
200
400
600
800
1000
y

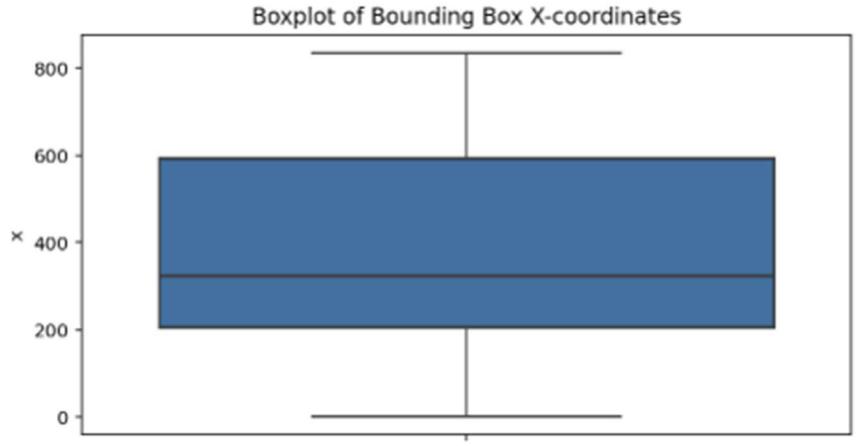

mean
median
Frequency
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
0
100
200
300
400
500
width


mean
median
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
Frequency
0
200
400
600
800
1000
height

# 3.2 Bounding Box X-Coordinates



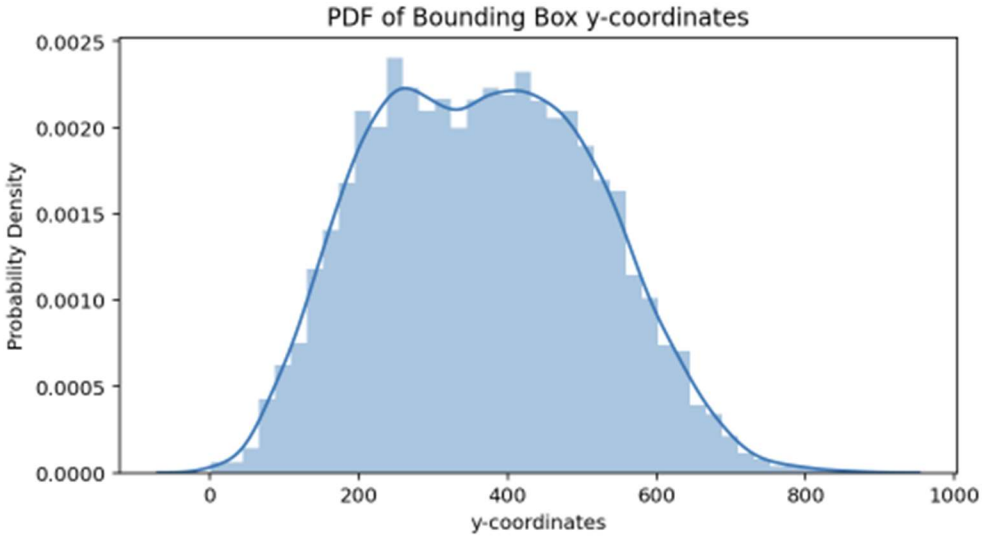



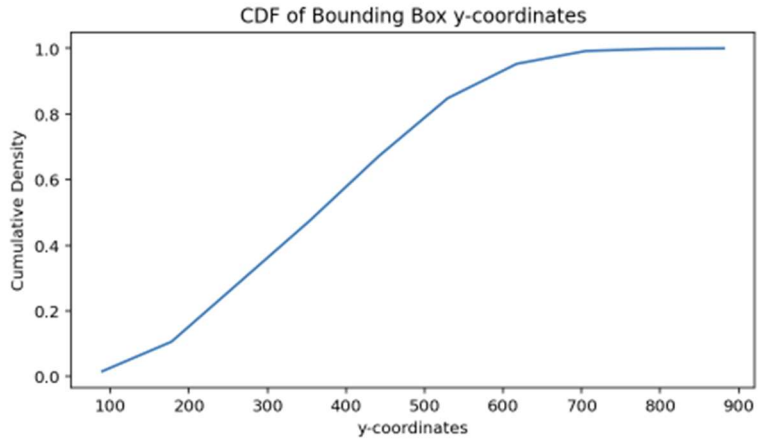



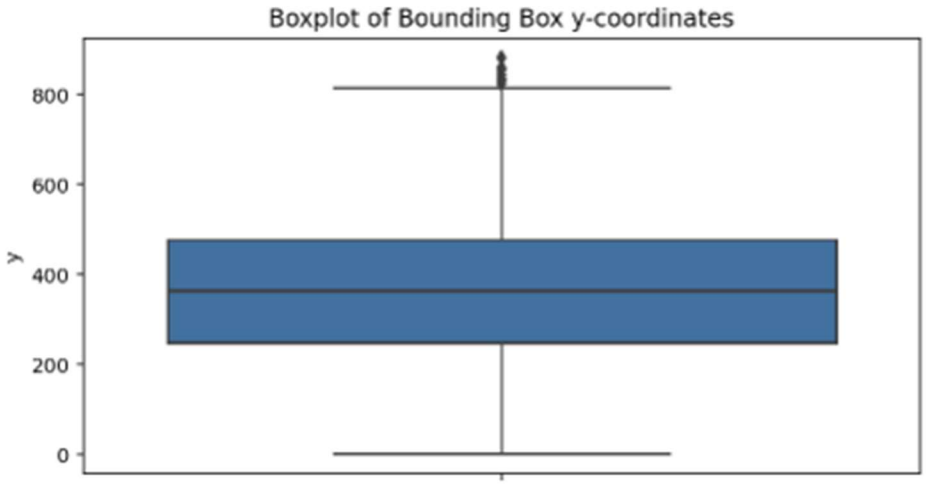
**Observation: -**

- The double-bell curve that we got in PDF of x-coordinates is due to the location of the lungs (as the opacity bounding boxes are on lungs only).
- The x-coordinates passing over the lungs has high values in comparison to the area which do not have lungs.
- The range that we got for x-coordinates is from 0 to 800.
- Almost 99% of the x-coordinate values are less than 750.
- The IQR (interquartile range) that we got is from 200 (25th percentile) & 600(75th percentile).
- The median value that we got is 300(approx.)

## 3.3 Bounding Box y-Coordinates



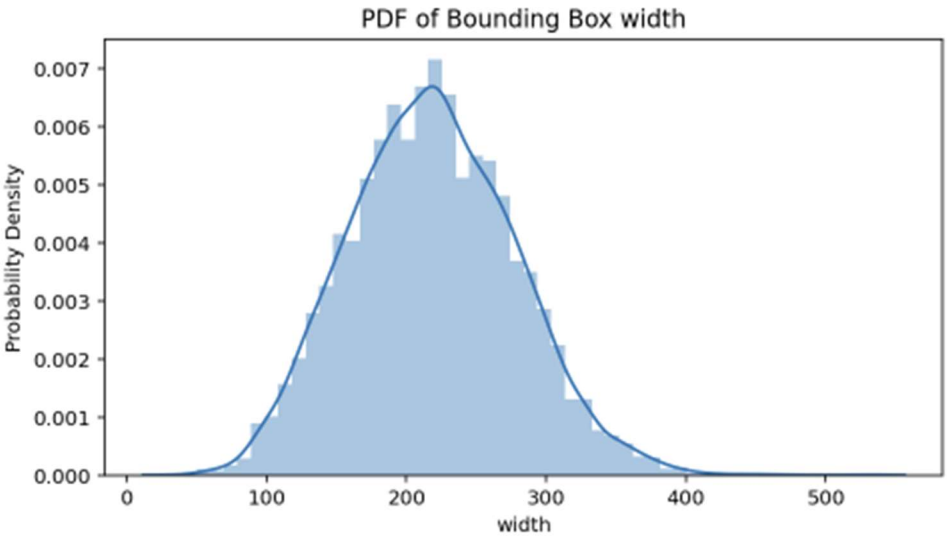



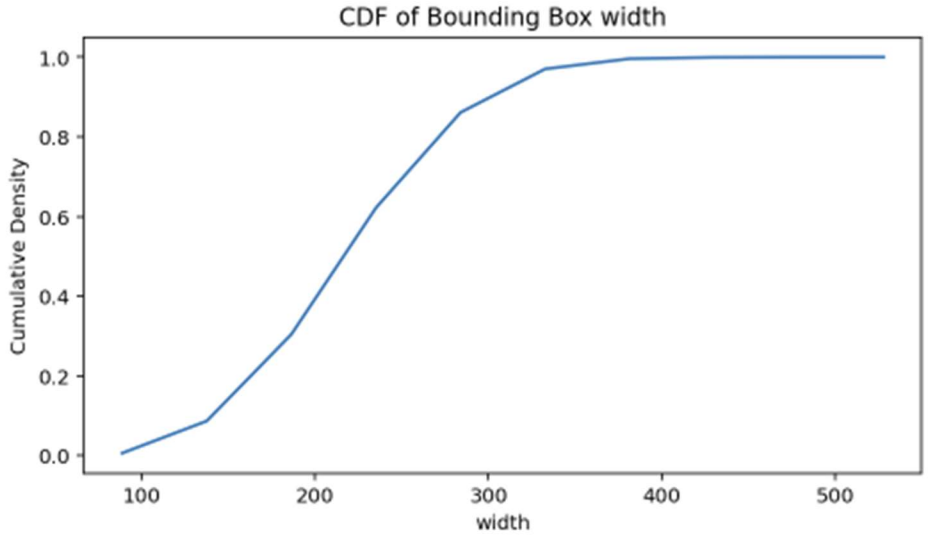


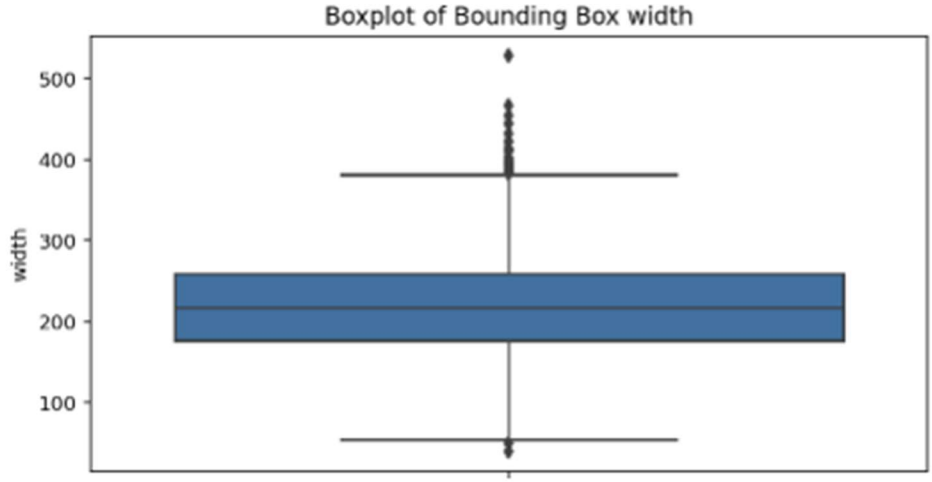

**Observation: -**

- The decline that we can observe at the middle of the curve is due to the center of the chest as there is no lungs at the center (and bounding boxes are only on lungs).
- The minimum & maximum value that we got is 0 and 800, respectively.
- Almost 99% values are between 100 & 700.
- The IQR (interquartile range) we got is from 250 (25th percentile) & 500 (75th percentile).
- The median value that we got is 350 (approx.).
- There are certain outlier values also above 800.

## 3.4 Bounding Box width



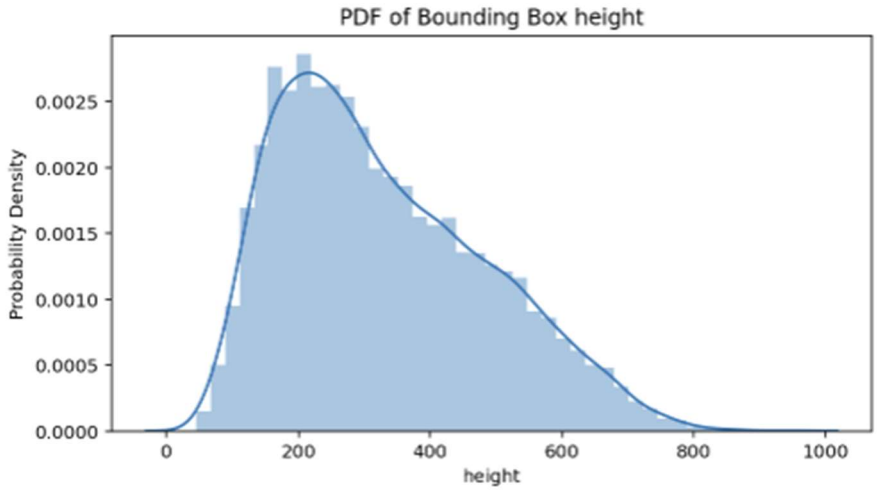


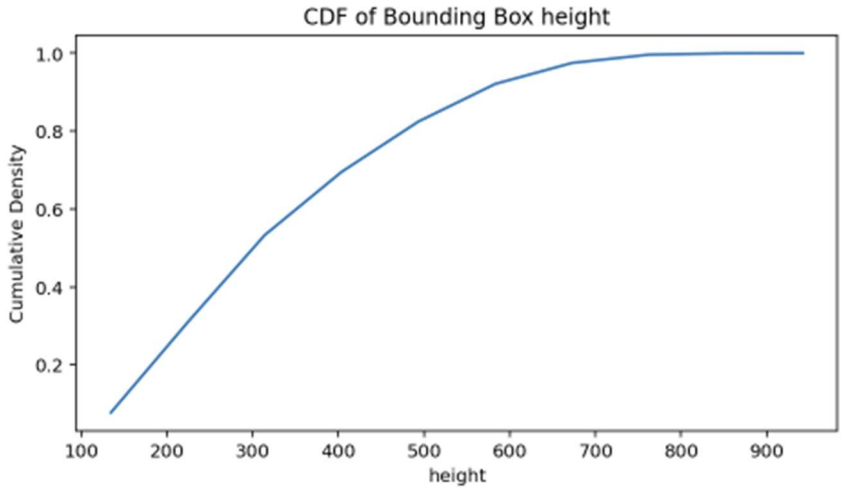



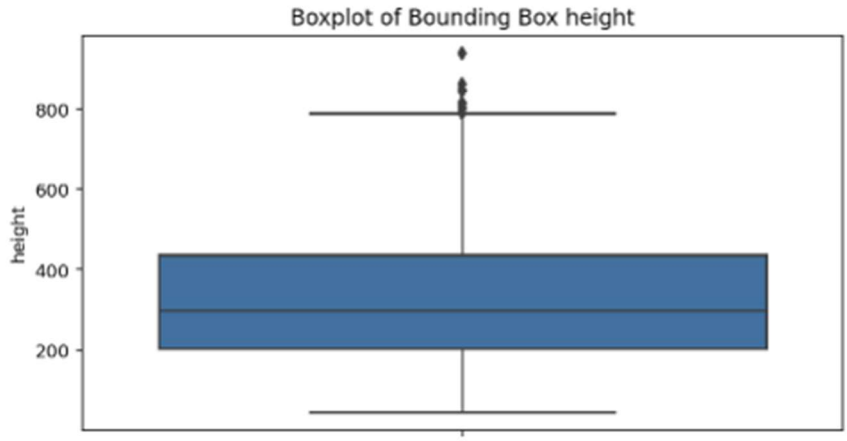

**Observation: -**

- The PDF of the bounding box width that we got is approx. normally distributed.
- The minimum & maximum values that we got is 50 & 400 with certain outliers above 400.
- Almost 99% values are less than 350.
- The IQR (interquartile range) that we got is from 175 to 275, respectively.
- The median value that we got is 225 (approx.)

## 3.5 Bounding Box Height



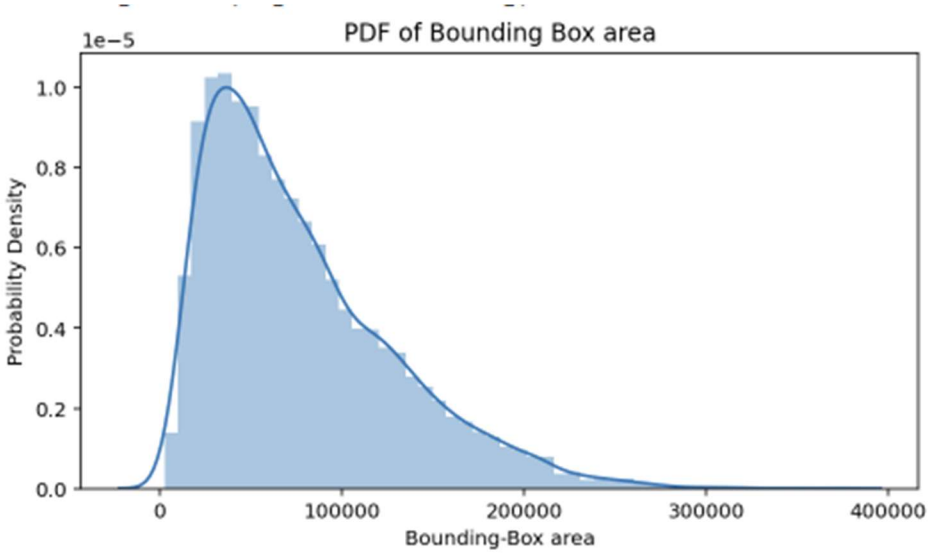



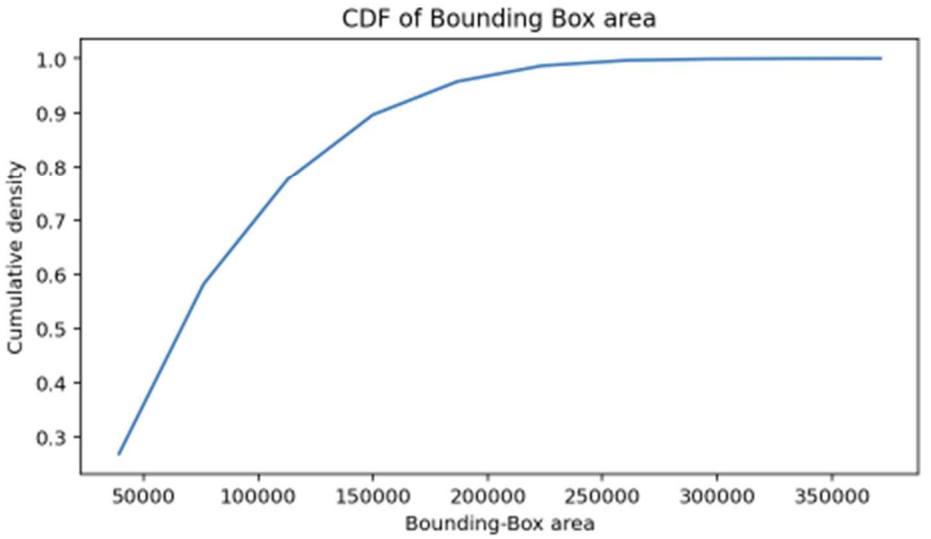



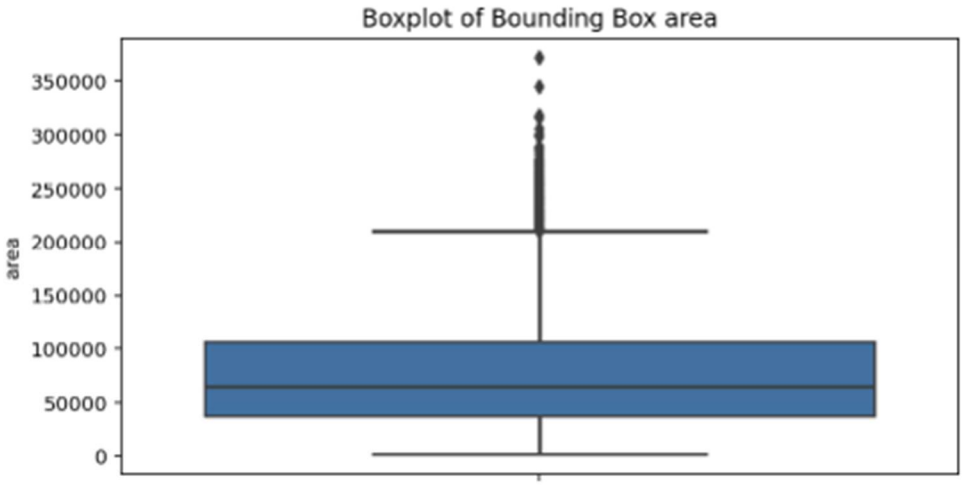
**Observation: -**

- The PDF of the bounding box height that we got is positively skewed.
- The minimum & maximum values that we got is 0 & 800 with certain outliers above 800.
- Almost 99% values are less than 700.
- The IQR (interquartile range) that we got is from 200 to 450, respectively.
- The median value that we got is 300 (approx.).

## 3.6 Bounding Box Area



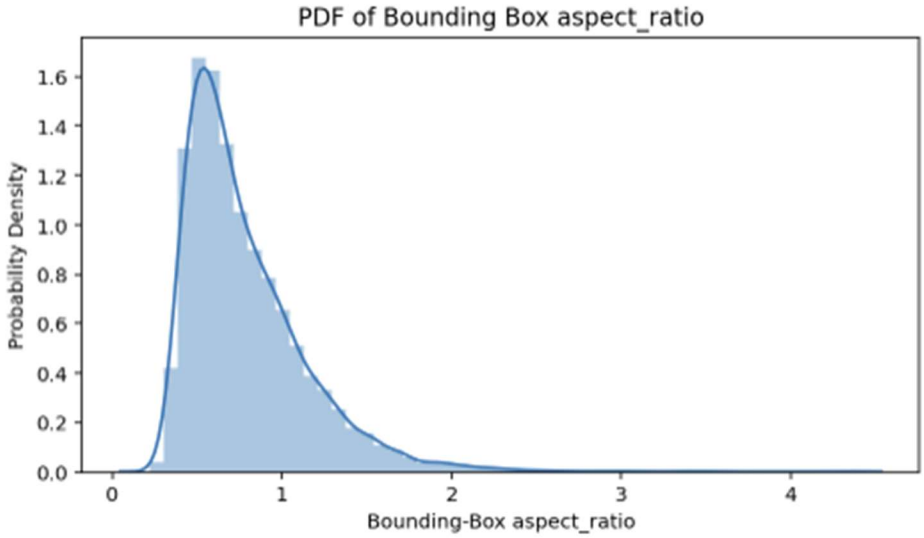



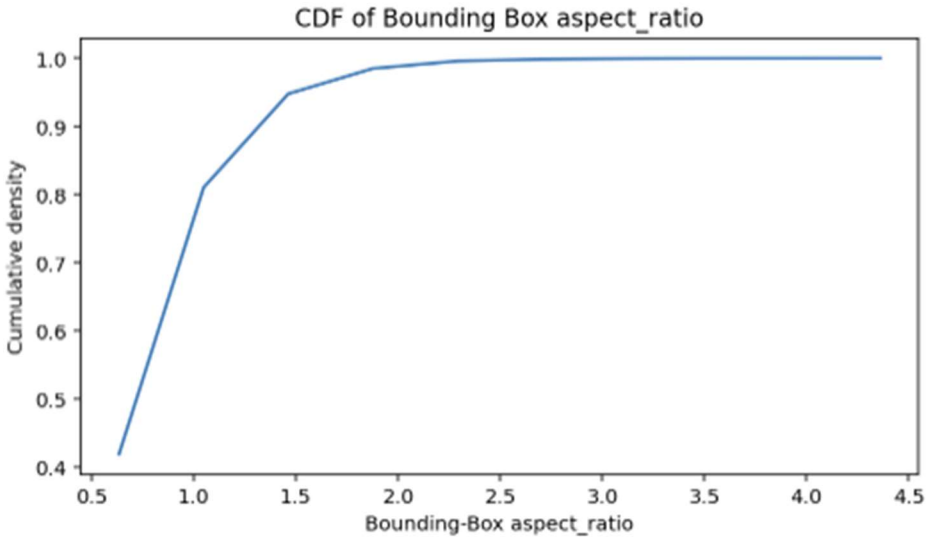


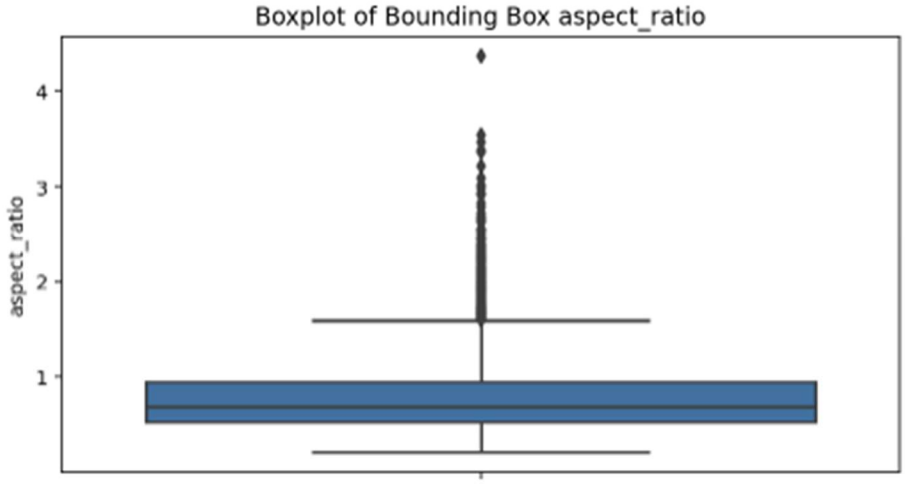

**Observation: -**

- The PDF of the bounding box bounding-box area that we got is positively skewed.
- The general minimum & maximum values that we got is 0 & 200000px (pixels) respectively.
- Almost 99% values are less than 250000.
- The IQR (interquartile range) that we got is from 25000 to 100000, respectively.
- There are certain outliers above 200000.

## 3.7 Bounding Box Aspect Ratio



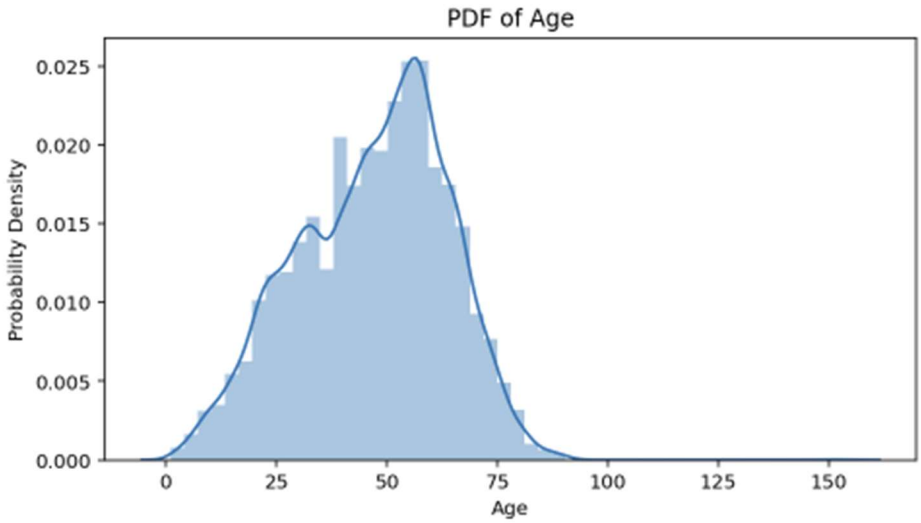


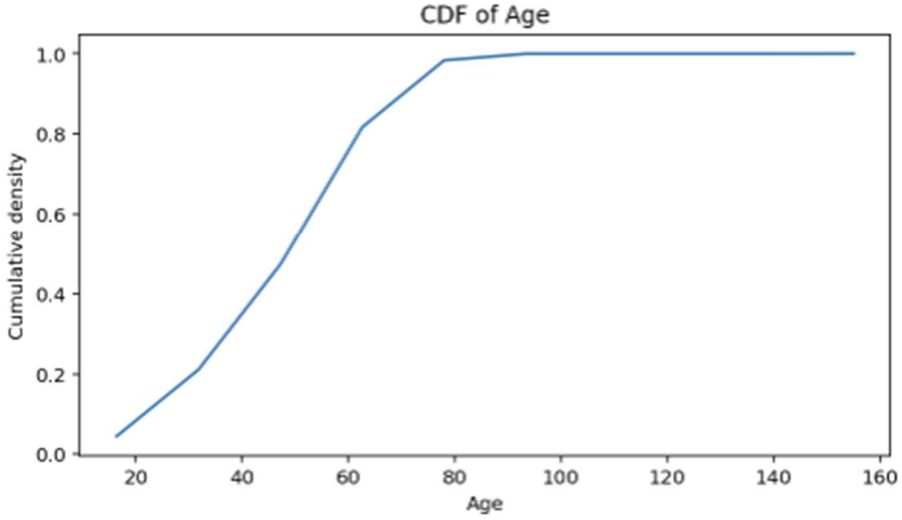



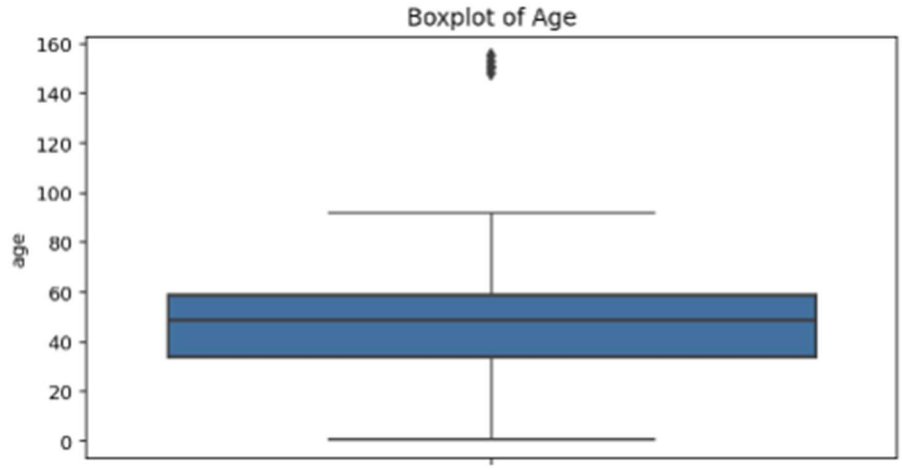

**Observation: -**

- The PDF of the bounding box bounding-box area that we got is positively skewed.
- The general minimum & maximum values that we got is 0 & 2, respectively.
- Almost 99% values are less than 2.
- The IQR (interquartile range) that we got is from 0.5 to 1, respectively.
- There are certain outliers above 1.75.

## 3.8 Patient's Age



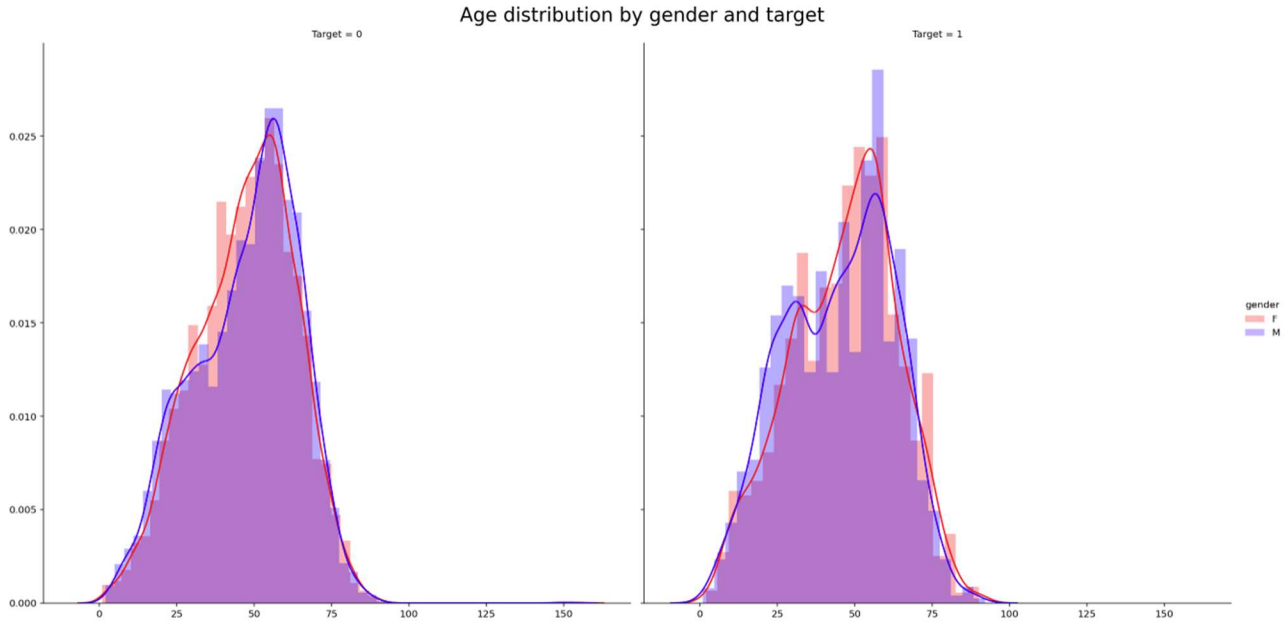



**Observation: -**

- The range of patient's age is from 15 to 95 years.
- There are certain values between 140 and 160 years which are certainly outliers.
- The age group from 30 to 95 is highly prone to the disease.
- The IQR range is from 30 to 60 years, respectively.
- The median age of the patients is 50 years.

## 3.9 Age distribution by gender and target

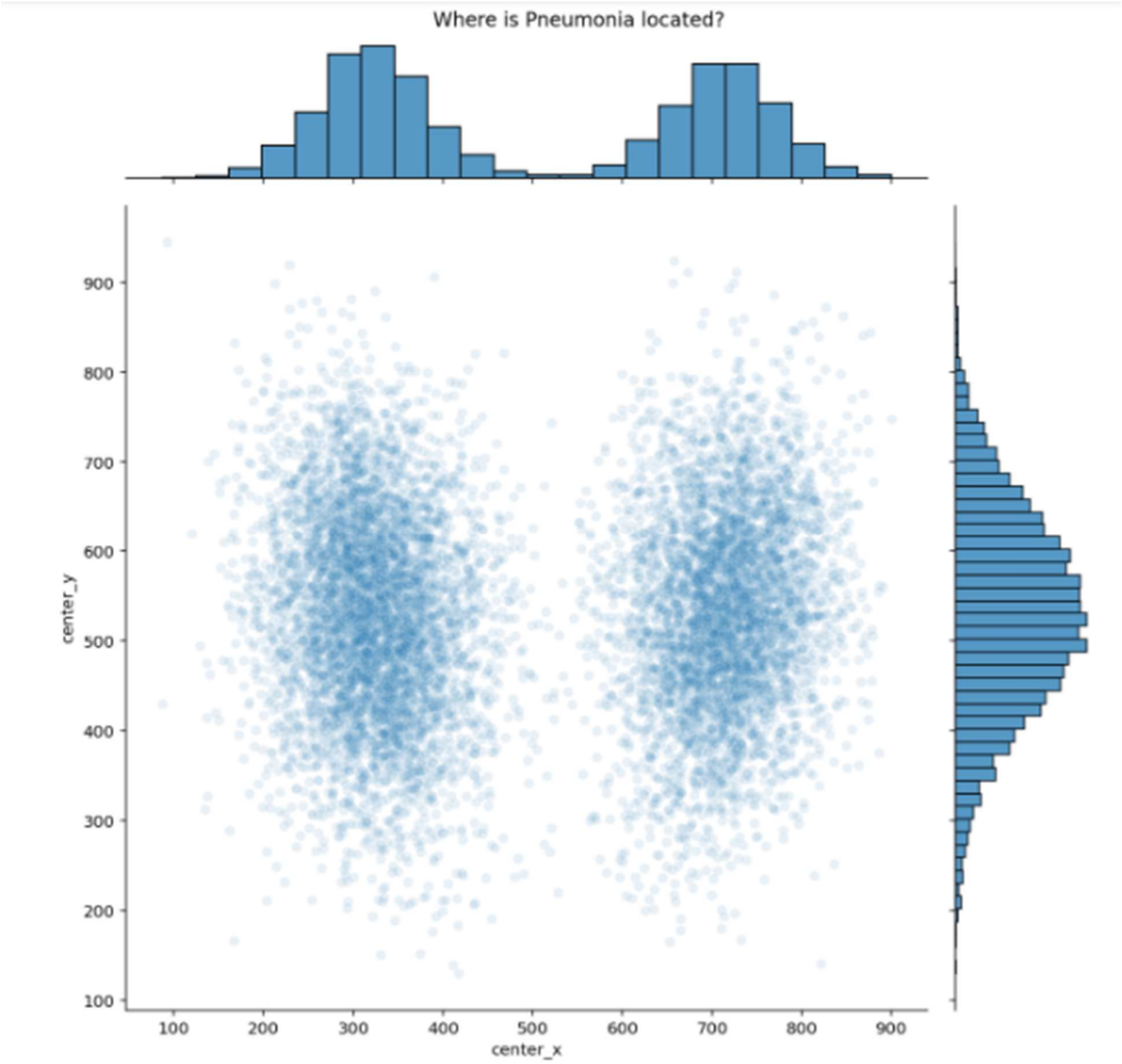


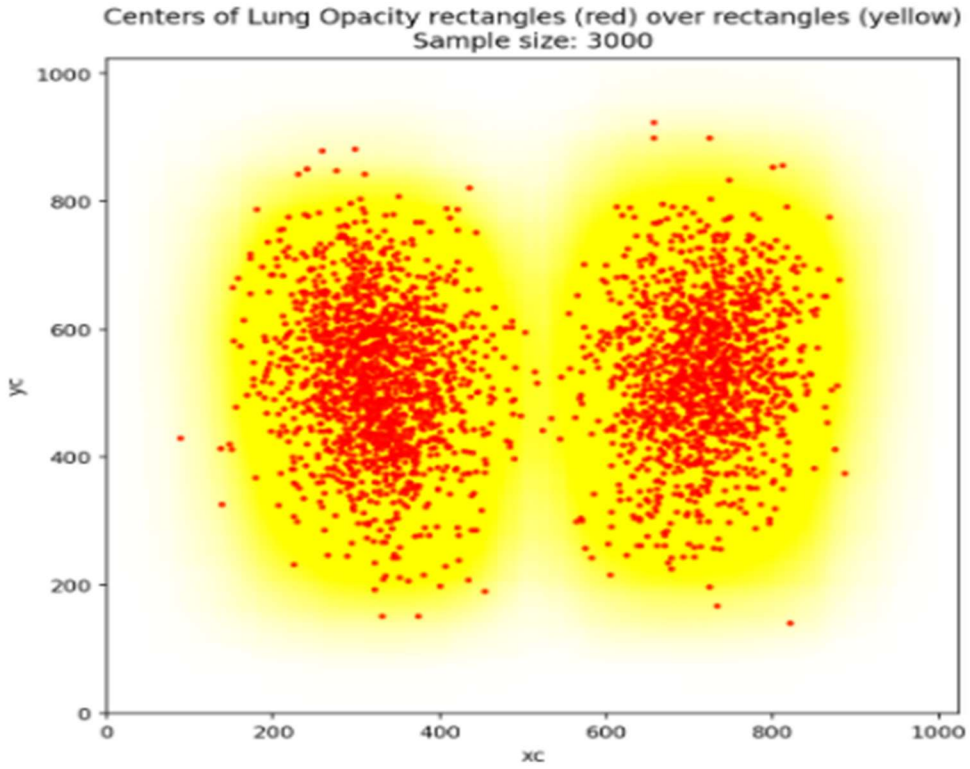

**Observation**: -

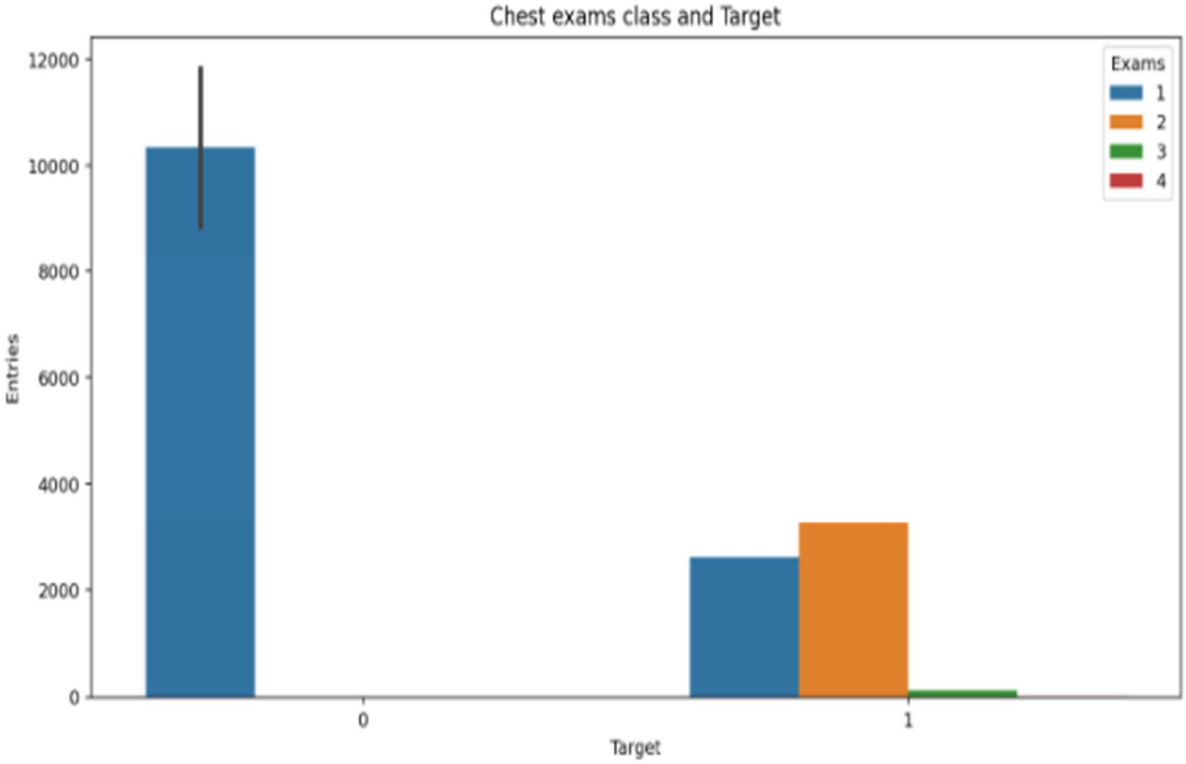
- There is perfect overlap between the distribution of both the genders for both Target=0 & Target=1.
- Both the relations look almost same, it explains fact that both Target = 0 & Target = 1 follows same trend in terms of age.
- It explains the fact that there is not any specific relation between the disease & the gender.
- Even when we compare this target & gender-based plot with the above explained overall age-based plot we can easily observe that both are following same trend regardless of gender & target.

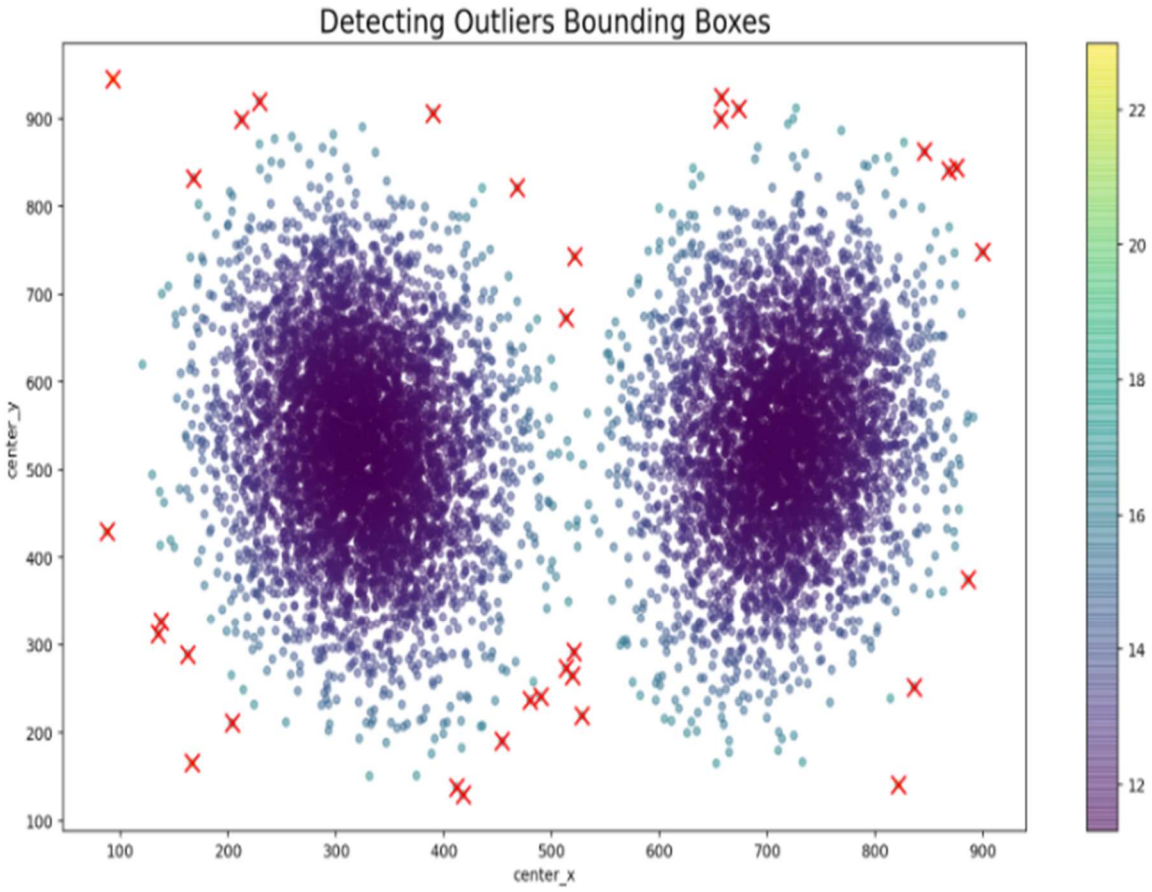
# 4. Bivariate Analysis

## 4.1 Bounding Box Centroid plot (to detect outliers bounding boxes)




Centers of Lung Opacity rectangles (red) over rectangles (yellow)
Sample size: 3000
yc
xc
0
200
400
600
800
1000

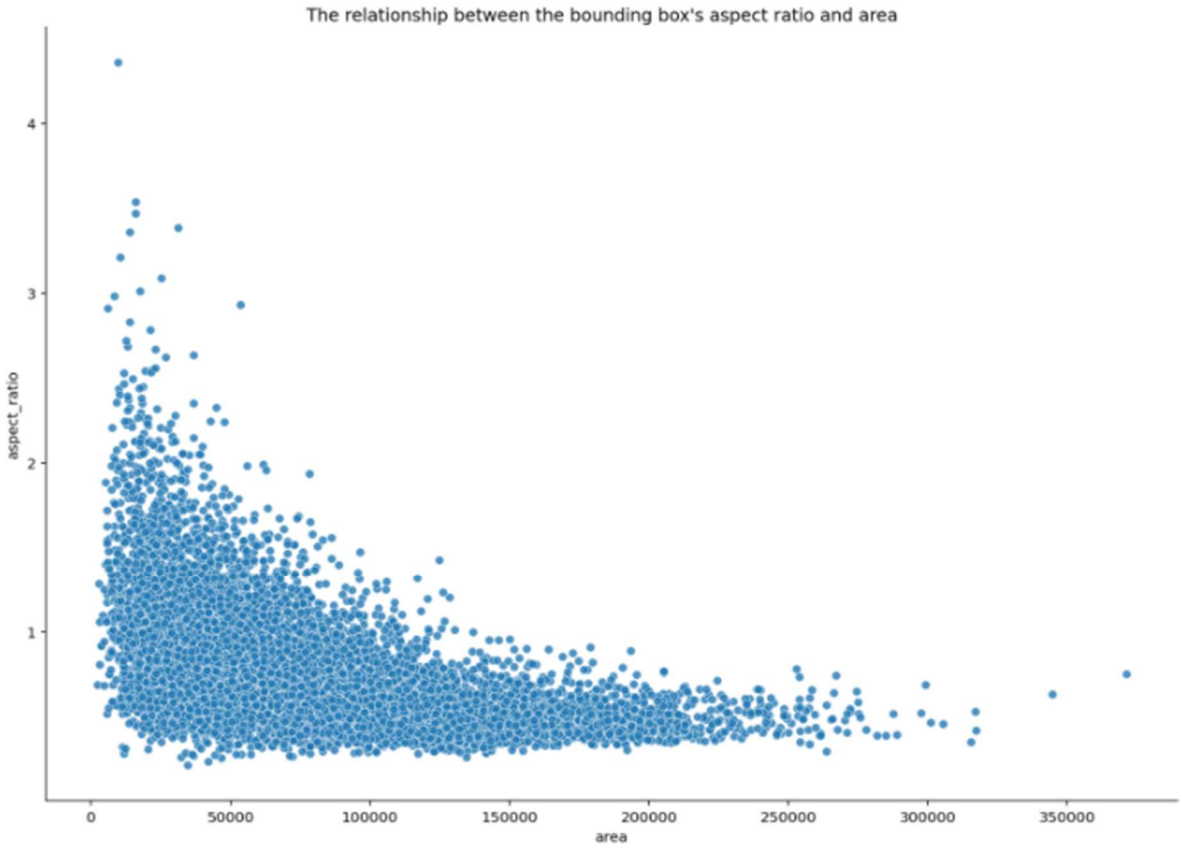

Chest exams class and Target
Exams
1
2
3
4
Entries
Target
0
2000
4000
6000
8000
10000
12000
0
1




**Observation: -**

- We can observe that the central area of both the lungs are highly dense, having maximum numbers of bounding box centroids.
- As soon as we are moving away from the center, the bounding box density is gradually decreasing.
- The red crosses that we can see on the outskirts of the lungs are outliers.
- By removing those outlier values, we can get rid of the effect of outliers.

## 4.2 Aspect ratio vs Area




**Observation: -**

- The plot depicts an inversely proportional relation between the bounding aspect ratio and area.
- As soon as the aspect ratio is increasing the area is decreasing and vice-versa.
- The bounding boxes having maximum heights have small width & vise-versa.
- There are certain bounding boxes having very high aspect ratios.

# 5. Feature Engineering & Modelling

## 5.1 PREPROCESSING

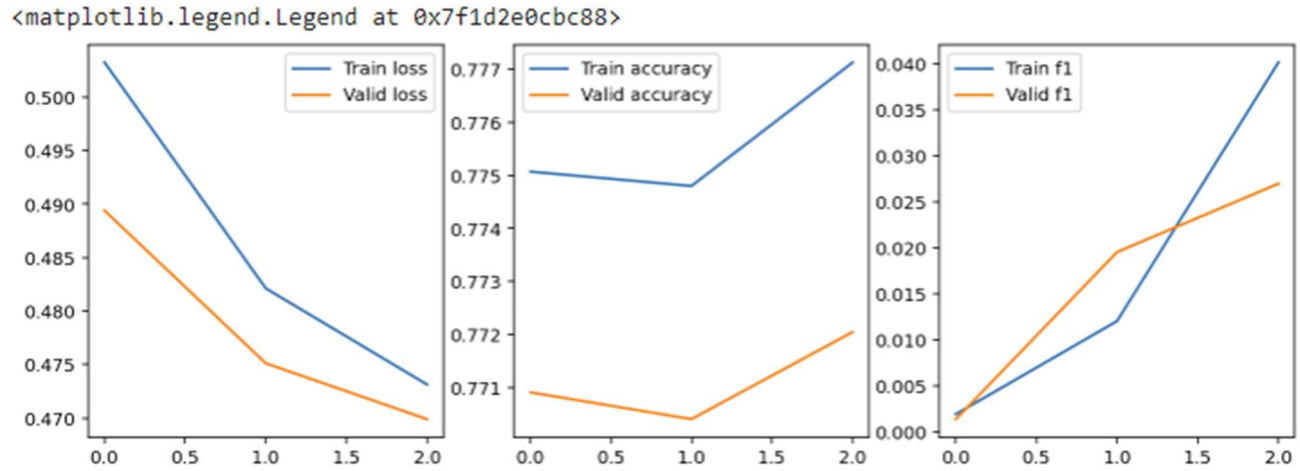
- All X-ray Image which are in the Digital Imaging and Communications in Medicine (**DICOM**) format have converted into jpg with the defined diocm_to_jpg Function.
- dicom_to_jpg Function fetch the dicom data from the source folder and convert the DICOM to jpg into the destination folder.
- We are also looking forward to balancing the data in further.
- We have defined some model performance metrics are recall, precision, f1-score.
- Train and validation data have generated with the help of ImageDataGenerator which is in the keras preprocessing tool.
- Data has splitted into to two parts i.e, train, and validation with the ratio of 70:30.
- Parameters in the Data Augmentation:
    - Rescale = 1. /255
    - Rotation Range = 40
    - Shear Range = 0.2
    - Zoom Range = 0.2
    - Horizontal Flip = True
    - Brightness Range = (0.5, 1.5)

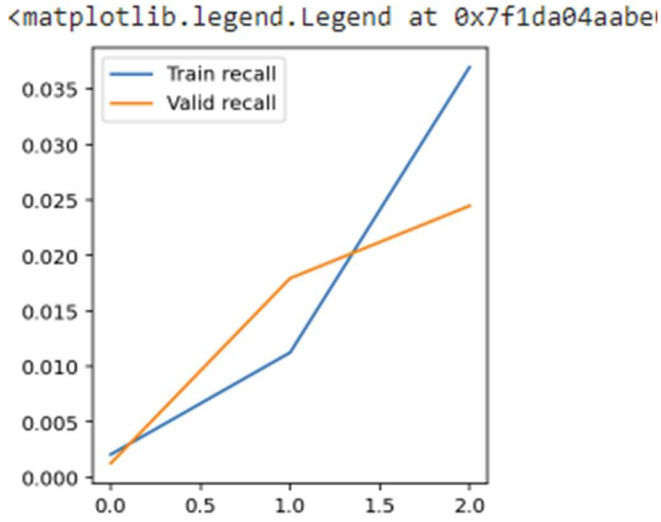
## 5.2 Model Building:

### 5.2.1 VGG19:

- We have defined the VGG 19 model preloaded with the vgg19_weights
- Input shape = 244,244,3
- We use the following methods in the model:
    - Polling = Avg
    - Optimizer – Adam(lr=0.001)
    - Loss = Binary_CrossEntropy
    - Metrics = defined Accuracy metrics i.e, Accuracy, Precision, Recall & f1
    - Epochs = 3

## Results:



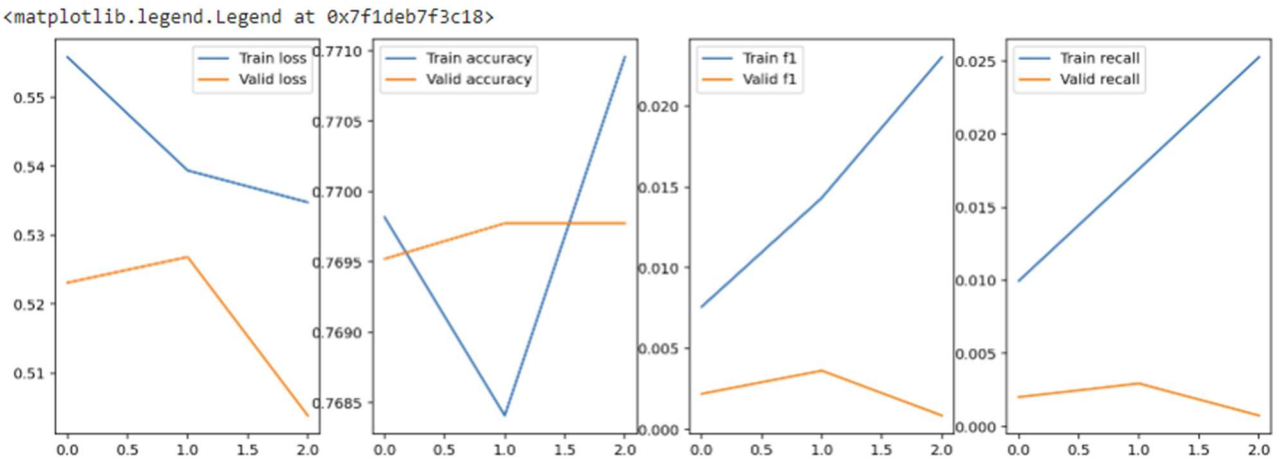

- Vgg19 has very low f1 score in training. It is a Basic Image classification model. So, we ignore this model for further development.

## 5.2.2 ResNet50:

- We have defined the ResNet50 model preloaded with Image weights.
- Input shape = 244,244,3
- We use the following methods in the model:
  - Polling = Avg

- Optimizers – Adam(lr=0.001)
- Loss = Binary_CrossEntropy
- Metrics = defined Accuracy metrics i.e, Accuracy, Precision, Recall & f1
- We use sigmoid Activation function for the output layer.
- Epochs = 3

## Results:

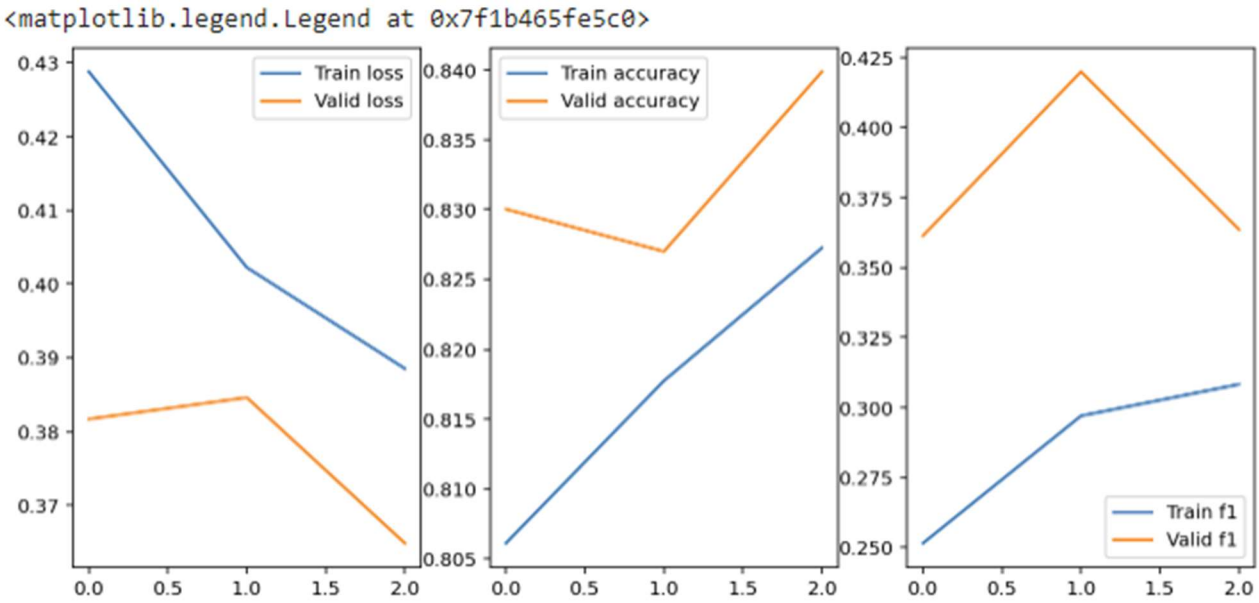



- ResNet50 has also low f1 score.

## 5.2.3 ChexNet i.e., DenseNet 121:

- We have defined the DenseNet model preloaded with Dense121 Weights.
- Input shape = 244,244,3
- We use the following methods in the model:
  - Polling = Avg
  - Optimizer – Adam(lr=0.001)
  - Loss = Binary_CrossEntropy
  - Metrics = defined Accuracy metrics i.e, Accuracy, Precision, Recall & f1
  - We use sigmoid Activation function for the output layer.
- Model loaded with the pre-trained Chex net weights available from the git-hub "brucechou1983_CheXNet_Keras_0.3.0_weights.h5"
- we used pretrained Chex Net architecture initialized with trained weights by removing top layers to make it binary classifier.
- We removed the last Dense layer with 14 classes and replaced with the Dense layer of 1 class.

## Results:




- Chex Net provides an average F1 score which can be considered for further evaluation.

## 5.2.4 Retina Net

## Architecture:

FAIR has released two papers in 2017 and 2018 respectively on their state-of-the-art object detection frameworks. We will see how various layers come together to form a robust object detection pipeline. As usual I will take the math approach to explain everything. In short, we will discuss the following two papers

1. Feature Pyramid Networks for Object detection[Paper]
2. Focal loss for Dense object detection [Paper]

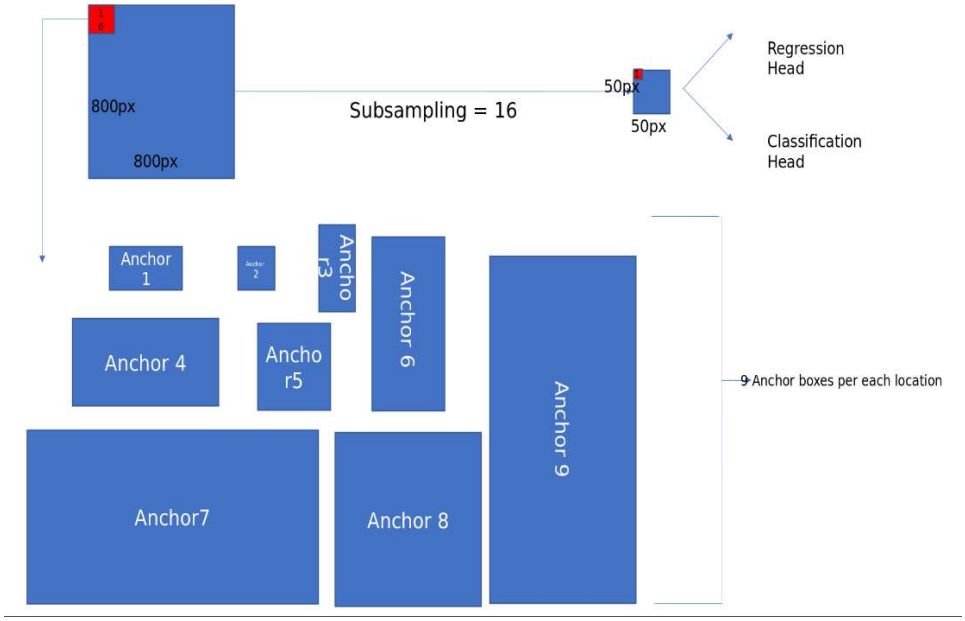
## THE FOLLOWING TOPICS WILL BE DISCUSSED

1. Anchor boxes
2. How RPN works?

3. Problems with RPN
4. Building Feature Pyramid networks
5. Focal Loss
6. RetinaNet for Object detection
7. Training RetinaNet
8. Inference on RetinaNet

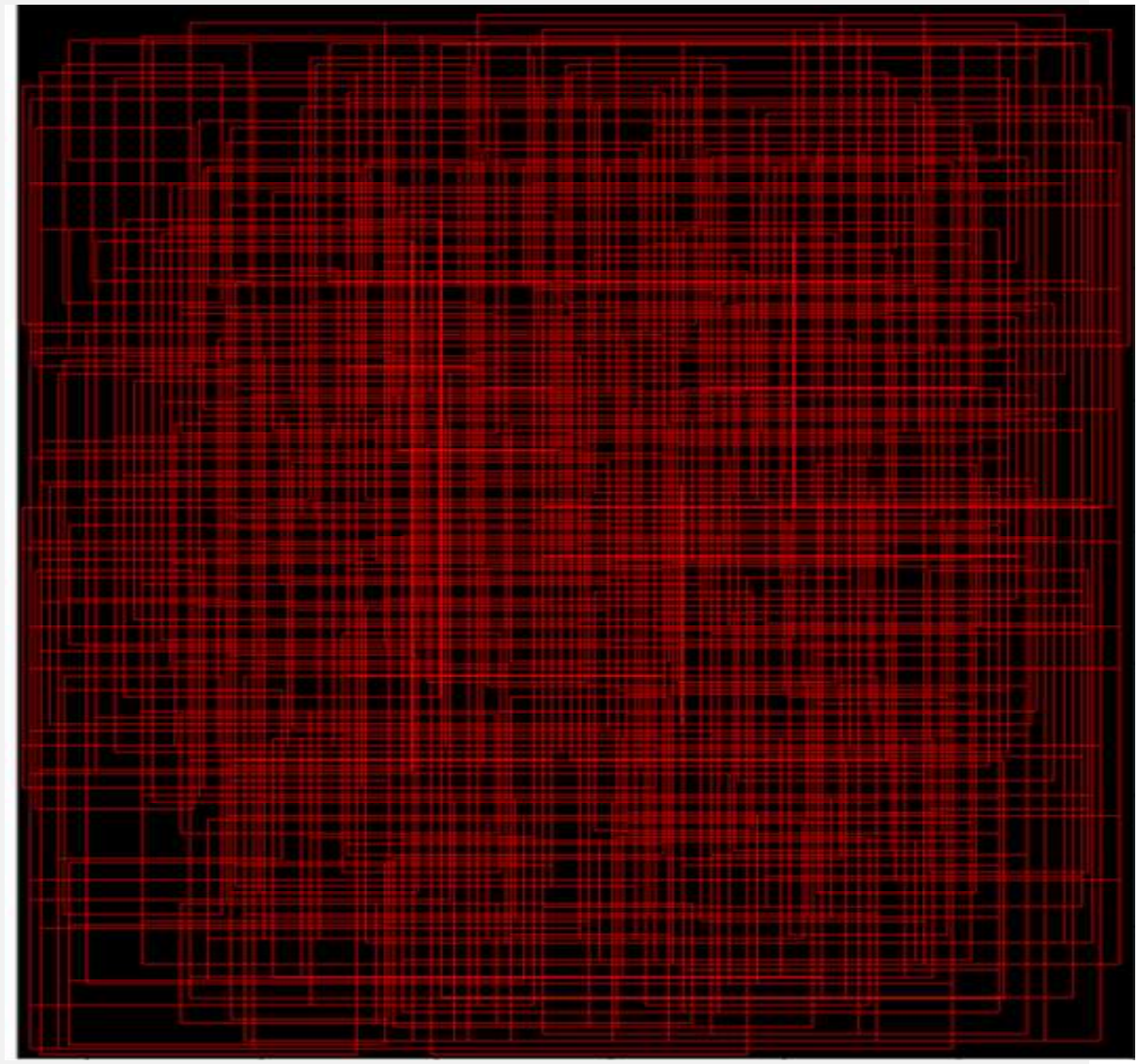
# ANCHOR BOXES:

Anchor boxes were first introduced in Faster RCNN paper and later became a common element in all the following papers like yolov2, ssd and RetinaNet. Previously selective search and edge boxes used to generate region proposals of various sizes and shapes depending on the objects in the image, with standard convolutions it is highly impossible to generate region proposals of varied shapes, so anchor boxes come to our rescue.




Anchor boxes mapping to Image from feature map

The below diagram helps in viewing the valid anchor boxes on an Image,

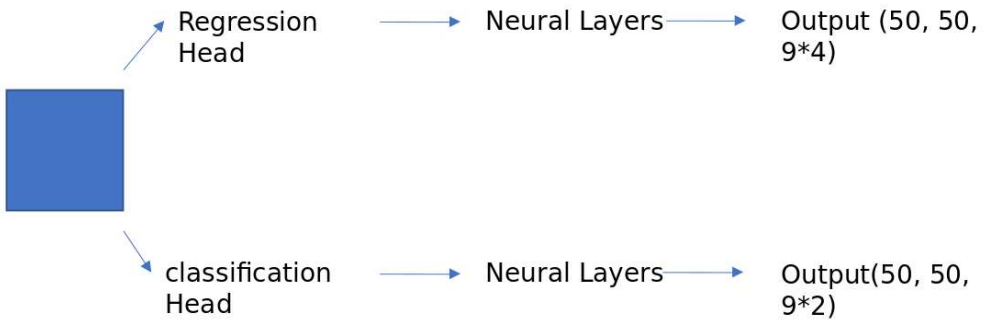


## HOW RPN WORKS?

To the above diagram 1 attach two heads one for regression and the other for classification as shown below. We will discuss how these heads are designed below.

**Regression head:** The output of the Faster RPN network as discussed and shown in the image above is a 50*50 feature map. A conv layer [kernal 3*3] strides through this image, at each location it predicts the 5 [x1, y1, h1, w1] values for each anchor boxes (9). In total, the output layer has 50*50*9*4 output probability scores. Usually this is represented in NumPy as np. array (2500, 36).

**Classification head:** Like the Regression head, this will predict the probability of an object present or not at each location for each anchor boss. This is represented in NumPy array as np. array (2500, 9)




RPN network head

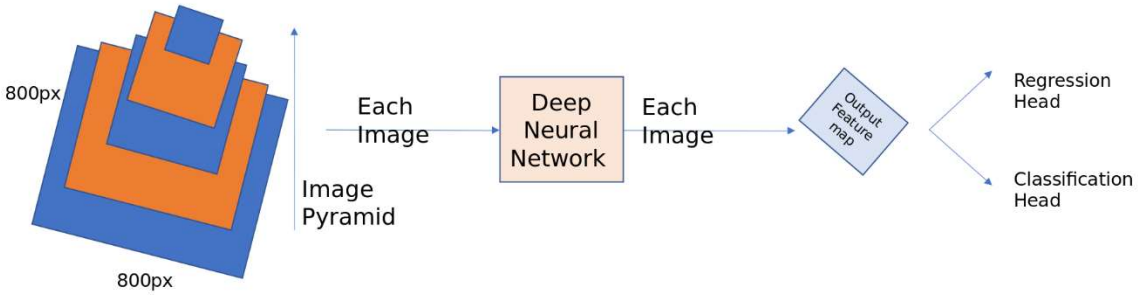
## Problems with RPN

- The Feature map created after a lot of subsampling losses a lot of semantic information at low level, thus unable to detect small objects in the image. [Feature Pyramid networks solves this]
- The loss functions use negative hard-mining by taking 128 +ve samples, 128 -ve samples because using all the labels hampers training as it is highly imbalanced and there will be many easily classified examples. [Focal loss solves this]

## HOW FEATURE PYRAMID NETWORKS WORK?

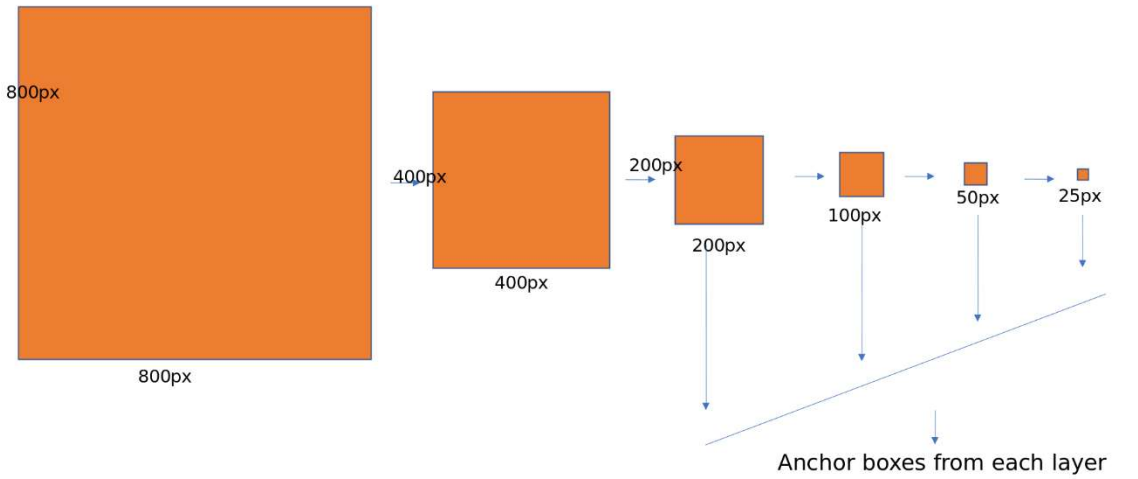
In RPN, we have built anchor boxes only using the top high level feature map. Though convnets are robust to variance in scale, all the top entries in ImageNet or COCO have used multi-scale testing on featured image pyramids. Imagine taking an 800 * 800 image and detecting bounding boxes on it. Now if you are using image pyramids, we must take images at different sizes say 256*256, 300*300, 500*500, and 800*800 etc., calculate feature maps for each of this image and then apply non-maxima suppression over all these

detected positive anchor boxes. This is a very costly operation and inference times gets high.



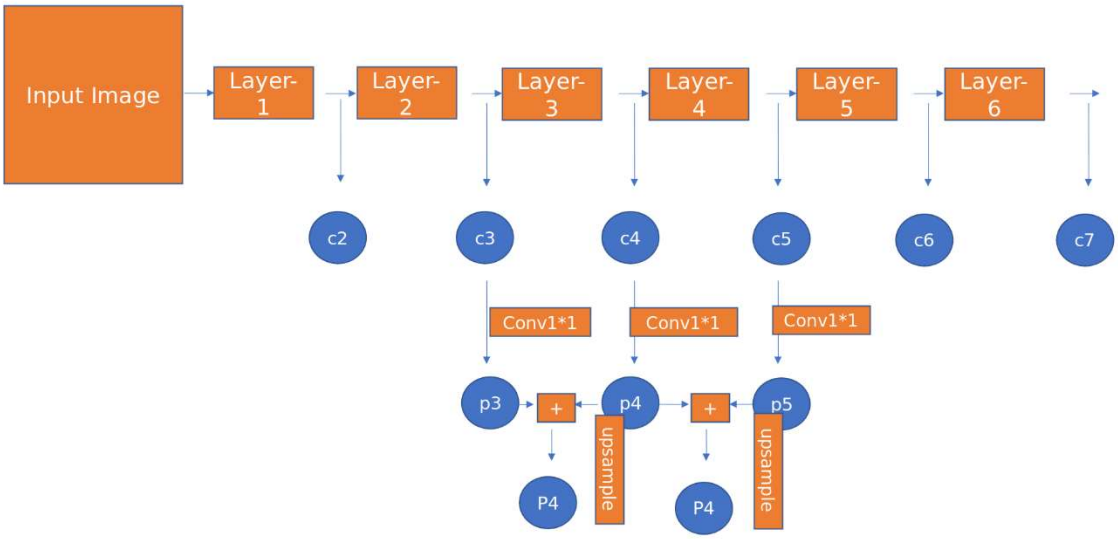

Deep learning using Image Pyramids

The authors of this paper observed that deep convnet computes a feature hierarchy layer by layer, and with subsampling layers the feature hierarchy has an inherent multi-scale, pyramidal shape. For example, take a Resnet architecture and instead of just using the final feature map as shown in RPN network, take feature maps before every pooling (subsampling) layer. Perform the same operations as for RPN on each of these feature maps and finally combine them using non-maxima suppression. This is the crude way of building the feature pyramid networks. But there is one of the problems with this approach, there are large semantic gaps caused by different depths. The high-resolution maps (earlier layers) have low-level features that harm their representational capacity for object detection.

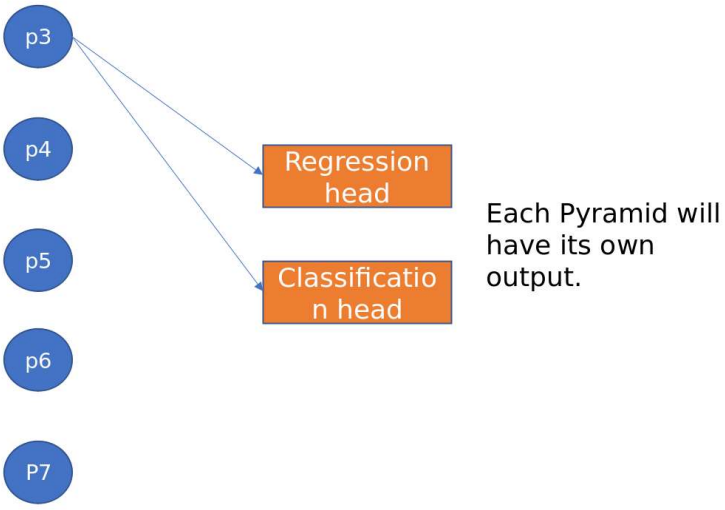



Feature Pyramid Network Rough

The goal of the authors is to naturally leverage the pyramidal shape of a Convnet feature hierarchy while creating a feature pyramid that has strong semantics at all scales. To achieve this goal, the authors relayed on an architecture that combines low-resolution, semantically strong features with high-resolution, semantically strong features via top-down pathway and lateral connection as shown in the diagram below.




Feature Pyramid Network




Feature Pyramid network output

We can design these blocks with more sophistication, but the authors have found marginal improvements, so they have chosen to keep the network simple.

The predictions are made on each level independently.

**Important points while designing anchor boxes:**

1. Since the pyramids are of different scales, no need to have multi-scale anchors on a specific level. We define the anchors to have size of [32, 54, 128, 256, 512] on P3, P4, P5, P6, P7, respectively. We use anchors of multiple aspect ratio [1:1, 1:2, 2:1]. so in-total there will be 15 anchors over the pyramid at each location.

2. All the anchor boxes outside image dimensions were ignored.

3. positive if the given anchor box has highest IoU with the ground truth box or if the IoU is greater than 0.7. negative if the IoU is less than 0.3.

4. The scales of the ground truth boxes are not used to assign them to levels of the pyramid. Instead, ground-truth boxes are associated with anchors, which have been assigned to pyramid levels. This above statement is very important to understand. I had two confusions here, weather we need to assign ground truth boxes to each level separately or compute all the anchor boxes and then assign label to the anchor box with which it has max IoU or IoU greater than 0.7. Finally, I have chosen the second option to assign labels.

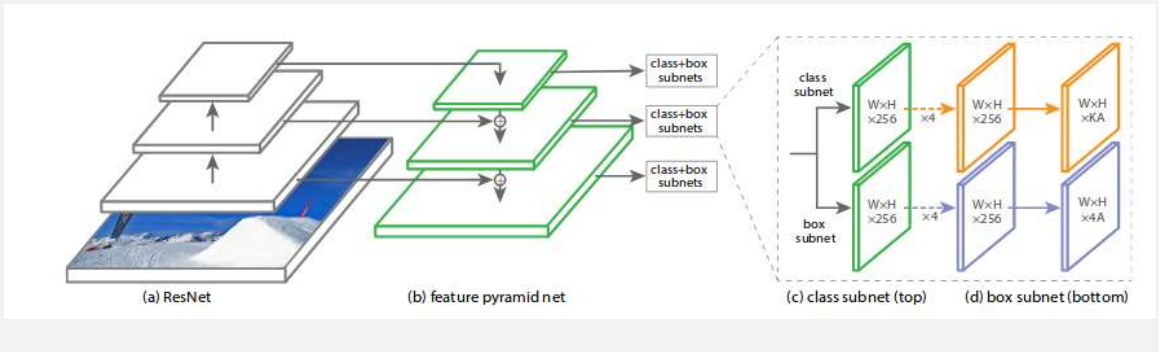
# RetinaNet for object detection

RetinaNet is a single, unified network composed of a backbone network and two task-specific subnetworks. The backbone is responsible for computing a conv feature map over an entire input image and is an off-the-self convolution network. The first subnet performs classification on the backbones output; the second subnet performs convolution bounding box regression.

**Backbone:** Feature Pyramid network built on top of ResNet50 or ResNet101. However, we can use any classifier of your choice; just follow the instructions given in FPN section when designing the network.

**Classification subnet:** It predicts the probability of object presence at each spatial position for each of the A anchors and K object classes. Takes an input feature map with C channels from a pyramid level, the subnet applies four 3x3 conv layers, each with C filters and each followed by ReLU activations. Finally, sigmoid activations are attached to the outputs. Focal loss is applied as the loss function.

**Box Regression Subnet:** Like classification net used but the parameters are not shared. Outputs the object location with respect to anchor box if an object exists. smooth_l1_loss with sigma equal to 3 is applied as the loss function to this part of the sub-network.




RetinaNet

**TRAINING RETINANET**

- Initialization of the network is very important. The authors have assumed a prior probability of 0.01 for all the anchor boxes and assigned this to the bias of last conv layer of classification sub net. I have overlooked this when implementing the net, the loss function blows up if you do not take care of this. The intuition behind attaching this prior probability is that the foreground (All positive anchor boxes) to background objects (All negative anchor boxes) in image is 1000/100000 = 0.01.

- Weight decay of 0.0001 and momentum of 0.9 with initial learning rate of 0.01 is used for first 60k Iterations. learning rate is reduced by 10 after 60k iterations and 80k iterations.

- Achieved a mAP of 40.8 using ResNeXt-101-FPN backend on MS-coco dataset

**INFERENCE ON RETINANET**

- To improve speed, Decode box predictions from at most 1k top-scoring predictions per FPN level, after thresholding the detector confidence at 0.05.
- The top predictions from all levels are merged and non-maximum suppression with a threshold of 0.5 is applied to yield the final decisions.

## Model:

- We have defined the Retina Net Model with backbone of ResNet50 Weights
- Parsed the Bounding Boxes data from the train data Dataframe
- We have created the Annotation file for the images of the format image path, x1, y1, x2, y2, 'Pneumonia'.
- We have created the class label file of the format 'Pneumonia', '0'.
- Model Parameters:
  - Batch Size = 32
  - Epochs = 10
  - Loss = loss, Regression, Classification
  - Data Augmentation: Random Transform.

```
transform_generator = random_transform_generator(
    min_rotation=-0.1,
    max_rotation=0.1,
    min_translation=(-0.1, -0.1),
    max_translation=(0.1, 0.1),
    min_shear=-0.1,
    max_shear=0.1,
    min_scaling=(0.9, 0.9),
    max_scaling=(1.1, 1.1),
    flip_x_chance=0.5,
    flip_y_chance=0.5,
)
```

  - Steps = Train_sample_size / Batch Size

# Iteration 1:

Learning Rate = default=1e-5

No of Epochs = 10

# Results:

- Retina Net Model Gives Good Mean_Iou. There are no False Positives.
- Even tough after 10 Epochs still loss is going to degrade.
- As per our mentor, we are looking to improve the model performance with increasing the number of Epochs and the learning rate.
- Retina Net Model has considered to further development.
- After the 10 Epochs, **Classification Loss Value reaches to 0.4209**. Gave the **MAP Value is 0.2723.**
- This Iteration gave the Values between **0.2 to 0.7.**

```
2020-11-17 02:24:13.459000: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] Successfully opened dynamic library
79/79 [==============================] - ETA: 0s - loss: 3.3859 - regression_loss: 2.4668 - classification_loss: 0.9191
Epoch 00001: saving model to ./snapshots/resnet50_csv_01.h5
79/79 [==============================] - 391s 5s/step - loss: 3.3859 - regression_loss: 2.4668 - classification_loss: 0.9191
Epoch 2/10
79/79 [==============================] - ETA: 0s - loss: 2.7843 - regression_loss: 2.2477 - classification_loss: 0.5366
Epoch 00002: saving model to ./snapshots/resnet50_csv_02.h5
79/79 [==============================] - 392s 5s/step - loss: 2.7843 - regression_loss: 2.2477 - classification_loss: 0.5366
Epoch 3/10
79/79 [==============================] - ETA: 0s - loss: 2.6662 - regression_loss: 2.1851 - classification_loss: 0.4811
Epoch 00003: saving model to ./snapshots/resnet50_csv_03.h5
79/79 [==============================] - 392s 5s/step - loss: 2.6662 - regression_loss: 2.1851 - classification_loss: 0.4811
Epoch 4/10
79/79 [==============================] - ETA: 0s - loss: 2.6074 - regression_loss: 2.1466 - classification_loss: 0.4607
Epoch 00004: saving model to ./snapshots/resnet50_csv_04.h5
79/79 [==============================] - 389s 5s/step - loss: 2.6074 - regression_loss: 2.1466 - classification_loss: 0.4607
Epoch 5/10
79/79 [==============================] - ETA: 0s - loss: 2.5686 - regression_loss: 2.1201 - classification_loss: 0.4486
Epoch 00005: saving model to ./snapshots/resnet50_csv_05.h5
79/79 [==============================] - 383s 5s/step - loss: 2.5686 - regression_loss: 2.1201 - classification_loss: 0.4486
Epoch 6/10
79/79 [==============================] - ETA: 0s - loss: 2.5419 - regression_loss: 2.1007 - classification_loss: 0.4412
Epoch 00006: saving model to ./snapshots/resnet50_csv_06.h5
79/79 [==============================] - 381s 5s/step - loss: 2.5419 - regression_loss: 2.1007 - classification_loss: 0.4412
Epoch 7/10
79/79 [==============================] - ETA: 0s - loss: 2.5216 - regression_loss: 2.0868 - classification_loss: 0.4348
Epoch 00007: saving model to ./snapshots/resnet50_csv_07.h5
79/79 [==============================] - 380s 5s/step - loss: 2.5216 - regression_loss: 2.0868 - classification_loss: 0.4348
Epoch 8/10
79/79 [==============================] - ETA: 0s - loss: 2.4955 - regression_loss: 2.0676 - classification_loss: 0.4280
Epoch 00008: saving model to ./snapshots/resnet50_csv_08.h5
79/79 [==============================] - 385s 5s/step - loss: 2.4955 - regression_loss: 2.0676 - classification_loss: 0.4280
Epoch 9/10
79/79 [==============================] - ETA: 0s - loss: 2.4823 - regression_loss: 2.0585 - classification_loss: 0.4238
Epoch 00009: saving model to ./snapshots/resnet50_csv_09.h5
79/79 [==============================] - 385s 5s/step - loss: 2.4823 - regression_loss: 2.0585 - classification_loss: 0.4238
Epoch 10/10
79/79 [==============================] - ETA: 0s - loss: 2.4636 - regression_loss: 2.0427 - classification_loss: 0.4209
Epoch 00010: saving model to ./snapshots/resnet50_csv_10.h5
79/79 [==============================] - 384s 5s/step - loss: 2.4636 - regression_loss: 2.0427 - classification_loss: 0.4209
```

# Iteration 2:

Learning Rate = 1e-3

No of Epochs = 50

## Results:

- In this Iteration, We Increase the Learning rate from 1e-5 to 1e-3 and increased the no of epochs to 50.
- After the 34th Epoch the Classification Loss value reaches to 0.3385
- Again, it starts to Increase the Loss Value to 0.3411, Likewise It Fluctuates the loss values between 0.3385 to 0.3422,
- At last, Finally Classification Value reaches to the **0.3396** and Given the MAP Value is **0.4715**
- This Iteration gave the Best IOU Values between **0.4 to 0.6.**

```
Epoch 00041: ReduceLROnPlateau reducing learning rate to 1.000000082740371e-09.
79/79 [==============================] - 365s 5s/step - loss: 2.1202 - regression_loss: 1.7780 - classification_loss: 0.3422
Epoch 42/50
79/79 [==============================] - ETA: 0s - loss: 2.1350 - regression_loss: 1.7936 - classification_loss: 0.3413
Epoch 00042: saving model to ./snapshots/resnet50_csv_42.h5
79/79 [==============================] - 367s 5s/step - loss: 2.1350 - regression_loss: 1.7936 - classification_loss: 0.3413
Epoch 43/50
79/79 [==============================] - ETA: 0s - loss: 2.1247 - regression_loss: 1.7857 - classification_loss: 0.3390
Epoch 00043: saving model to ./snapshots/resnet50_csv_43.h5

Epoch 00043: ReduceLROnPlateau reducing learning rate to 1.000000082740371e-10.
79/79 [==============================] - 368s 5s/step - loss: 2.1247 - regression_loss: 1.7857 - classification_loss: 0.3390
Epoch 44/50
79/79 [==============================] - ETA: 0s - loss: 2.1176 - regression_loss: 1.7790 - classification_loss: 0.3386
Epoch 00044: saving model to ./snapshots/resnet50_csv_44.h5
79/79 [==============================] - 369s 5s/step - loss: 2.1176 - regression_loss: 1.7790 - classification_loss: 0.3386
Epoch 45/50
79/79 [==============================] - ETA: 0s - loss: 2.1316 - regression_loss: 1.7917 - classification_loss: 0.3399
Epoch 00045: saving model to ./snapshots/resnet50_csv_45.h5

Epoch 00045: ReduceLROnPlateau reducing learning rate to 1.000000082740371e-11.
79/79 [==============================] - 365s 5s/step - loss: 2.1316 - regression_loss: 1.7917 - classification_loss: 0.3399
Epoch 46/50
79/79 [==============================] - ETA: 0s - loss: 2.1166 - regression_loss: 1.7786 - classification_loss: 0.3381
Epoch 00046: saving model to ./snapshots/resnet50_csv_46.h5
79/79 [==============================] - 366s 5s/step - loss: 2.1166 - regression_loss: 1.7786 - classification_loss: 0.3381
Epoch 47/50
79/79 [==============================] - ETA: 0s - loss: 2.1222 - regression_loss: 1.7832 - classification_loss: 0.3390
Epoch 00047: saving model to ./snapshots/resnet50_csv_47.h5

Epoch 00047: ReduceLROnPlateau reducing learning rate to 1.000000082740371e-12.
79/79 [==============================] - 366s 5s/step - loss: 2.1222 - regression_loss: 1.7832 - classification_loss: 0.3390
Epoch 48/50
79/79 [==============================] - ETA: 0s - loss: 2.1215 - regression_loss: 1.7826 - classification_loss: 0.3389
Epoch 00048: saving model to ./snapshots/resnet50_csv_48.h5
79/79 [==============================] - 365s 5s/step - loss: 2.1215 - regression_loss: 1.7826 - classification_loss: 0.3389
Epoch 49/50
79/79 [==============================] - ETA: 0s - loss: 2.1191 - regression_loss: 1.7802 - classification_loss: 0.3389
Epoch 00049: saving model to ./snapshots/resnet50_csv_49.h5

Epoch 00049: ReduceLROnPlateau reducing learning rate to 1.0000001044244145e-13.
79/79 [==============================] - 369s 5s/step - loss: 2.1191 - regression_loss: 1.7802 - classification_loss: 0.3389
Epoch 50/50
79/79 [==============================] - ETA: 0s - loss: 2.1117 - regression_loss: 1.7721 - classification_loss: 0.3396
Epoch 00050: saving model to ./snapshots/resnet50_csv_50.h5
79/79 [==============================] - 365s 5s/step - loss: 2.1117 - regression_loss: 1.7721 - classification_loss: 0.3396
```

```
2020-12-17 18:51:37.071568: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] succe
2020-12-17 18:51:37.072208: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] succe
2020-12-17 18:51:37.072752: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Adding v
2020-12-17 18:51:37.072803: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] S
2020-12-17 18:51:37.666806: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1257] Device i
2020-12-17 18:51:37.666866: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1263]      0
2020-12-17 18:51:37.666874: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1276] 0:   N
2020-12-17 18:51:37.667067: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] succe
2020-12-17 18:51:37.667666: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] succe
2020-12-17 18:51:37.668209: W tensorflow/core/common_runtime/gpu/gpu_bfc_allocator.cc:39] Ove
2020-12-17 18:51:37.668254: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1402] Created
Running network: N/A% (0 of 2519) |       | Elapsed Time: 0:00:00 ETA:  --:--:--2020-12-17 18
2020-12-17 18:51:44.430182: I tensorflow/stream_executor/platform/default/dso_loader.cc:48] S
Running network: 100% (2519 of 2519) |####| Elapsed Time: 0:03:07 Time:  0:03:07
Parsing annotations: 100% (2519 of 2519) || Elapsed Time: 0:00:00 Time:  0:00:00
4000 instances of class Pneumonia with average precision: 0.4715
Inference time for 2519 images: 0.0376
mAP using the weighted average of precisions among classes: 0.4715
mAP: 0.4715
```

# Iteration 3:

Learning Rate = 1e-4

No of Epochs = 50

# Results:

- In this Iteration, we decrease the Learning rate from 1e-3 to 1e-4.
- After the 43rd Epoch the Classification Loss value reaches to 0.3355
- Again, it starts to Increase the Loss Value to 0.3374, Likewise It fluctuates the loss values between 0.3355 to 0.3394.
- At last, Finally Classification Value reaches to the 0.335 and Given the MAP Value is **0.4814**
- This Iteration gave the Best IOU Values between **0.5 to 0.7.**

```
Epoch 00043: saving model to ./snapshots/resnet50_csv_43.h5
79/79 [==============================] - 341s 4s/step - loss: 2.0776 - regression_loss: 1.7421 - classification_loss: 0.3355
Epoch 44/50
79/79 [==============================] - ETA: 0s - loss: 2.0792 - regression_loss: 1.7417 - classification_loss: 0.3374
Epoch 00044: saving model to ./snapshots/resnet50_csv_44.h5
79/79 [==============================] - 339s 4s/step - loss: 2.0792 - regression_loss: 1.7417 - classification_loss: 0.3374
Epoch 45/50
79/79 [==============================] - ETA: 0s - loss: 2.0868 - regression_loss: 1.7491 - classification_loss: 0.3377
Epoch 00045: saving model to ./snapshots/resnet50_csv_45.h5

Epoch 00045: ReduceLROnPlateau reducing learning rate to 9.999999092680235e-13.
79/79 [==============================] - 336s 4s/step - loss: 2.0868 - regression_loss: 1.7491 - classification_loss: 0.3377
Epoch 46/50
79/79 [==============================] - ETA: 0s - loss: 2.0871 - regression_loss: 1.7478 - classification_loss: 0.3393
Epoch 00046: saving model to ./snapshots/resnet50_csv_46.h5
79/79 [==============================] - 345s 4s/step - loss: 2.0871 - regression_loss: 1.7478 - classification_loss: 0.3393
Epoch 47/50
79/79 [==============================] - ETA: 0s - loss: 2.0867 - regression_loss: 1.7487 - classification_loss: 0.3380
Epoch 00047: saving model to ./snapshots/resnet50_csv_47.h5

Epoch 00047: ReduceLROnPlateau reducing learning rate to 9.9999988758398e-14.
79/79 [==============================] - 342s 4s/step - loss: 2.0867 - regression_loss: 1.7487 - classification_loss: 0.3380
Epoch 48/50
79/79 [==============================] - ETA: 0s - loss: 2.1018 - regression_loss: 1.7635 - classification_loss: 0.3383
Epoch 00048: saving model to ./snapshots/resnet50_csv_48.h5
79/79 [==============================] - 338s 4s/step - loss: 2.1018 - regression_loss: 1.7635 - classification_loss: 0.3383
Epoch 49/50
79/79 [==============================] - ETA: 0s - loss: 2.0916 - regression_loss: 1.7522 - classification_loss: 0.3394
Epoch 00049: saving model to ./snapshots/resnet50_csv_49.h5

Epoch 00049: ReduceLROnPlateau reducing learning rate to 9.999999146890344e-15.
79/79 [==============================] - 341s 4s/step - loss: 2.0916 - regression_loss: 1.7522 - classification_loss: 0.3394
Epoch 50/50
79/79 [==============================] - ETA: 0s - loss: 2.0823 - regression_loss: 1.7463 - classification_loss: 0.3359
Epoch 00050: saving model to ./snapshots/resnet50_csv_50.h5
79/79 [==============================] - 340s 4s/step - loss: 2.0823 - regression_loss: 1.7463 - classification_loss: 0.3359
```

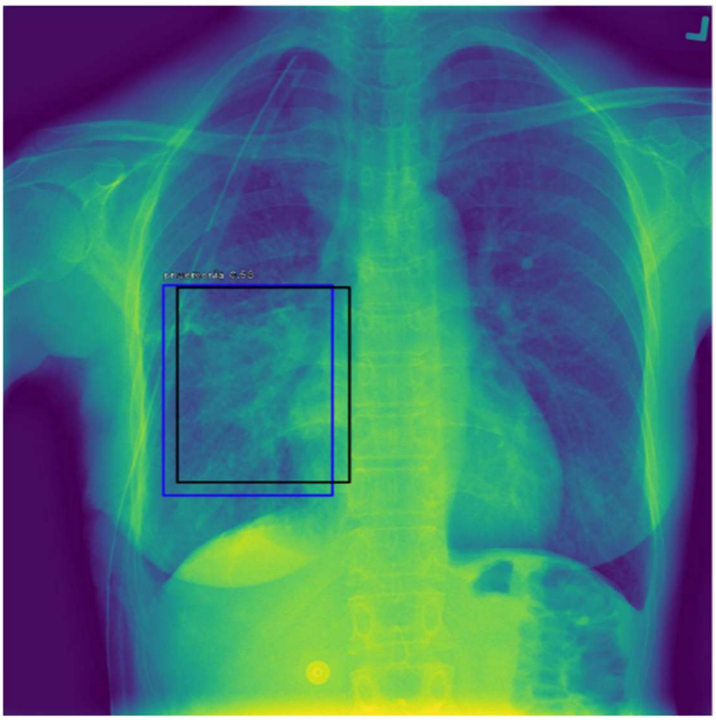
```
2020-12-17 02:09:59.806770: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] su
2020-12-17 02:09:59.807151: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1858] Addi
2020-12-17 02:09:59.807187: I tensorflow/stream_executor/platform/default/dso_loader.cc:48
2020-12-17 02:10:00.414097: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1257] Devi
2020-12-17 02:10:00.414155: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1263]
2020-12-17 02:10:00.414166: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1276] 0:
2020-12-17 02:10:00.414344: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] su
2020-12-17 02:10:00.414851: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] su
2020-12-17 02:10:00.415264: W tensorflow/core/common_runtime/gpu/gpu_bfc_allocator.cc:39]
2020-12-17 02:10:00.415298: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1402] Creat
Running network: N/A% (0 of 2519) |       | Elapsed Time: 0:00:00 ETA:  --:--:--2020-12-17
2020-12-17 02:10:06.296181: I tensorflow/stream_executor/platform/default/dso_loader.cc:48
Running network: 100% (2519 of 2519) |####| Elapsed Time: 0:03:51 Time:  0:03:51
Parsing annotations: 100% (2519 of 2519) || Elapsed Time: 0:00:00 Time:  0:00:00
4000 instances of class Pneumonia with average precision: 0.4814
Inference time for 2519 images: 0.0552
mAP using the weighted average of precisions among classes: 0.4814
mAP: 0.4814
```

```
1  THRES_SCORE = 0.4
2  patient_id='8192a218-3ed8-450a-bb3e-052e36567763'
3  draw_image(patient_id)
```

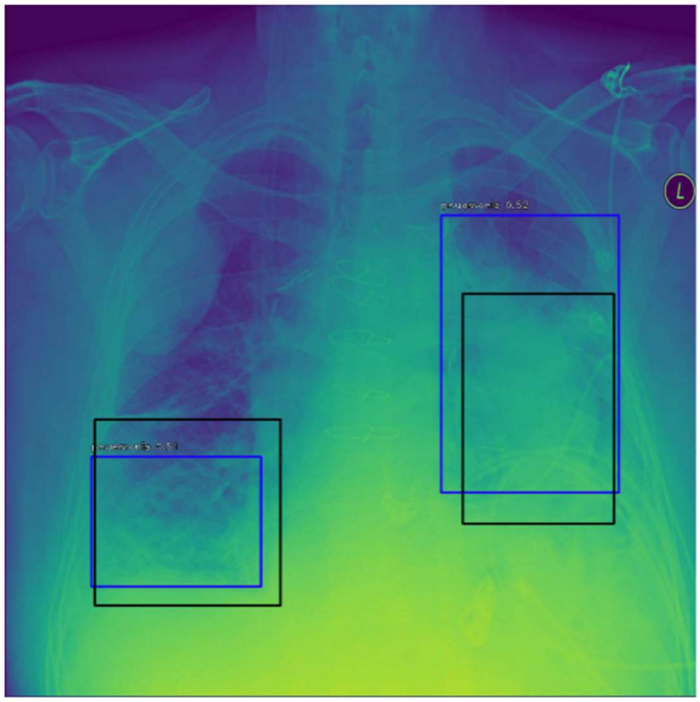
```
Mean_IOU::0.5465623167716114
```

```
THRES_SCORE =0.4
patient_id='815dd1c1-3e01-4038-9723-39d05e8b3cd3'
draw_image(patient_id)
```

```
Mean_IOU::0.7782411382313259
```



```
Mean_IOU::0.5882152805294634
```



# Iteration 4:

Learning Rate = 1e-05

No of Epochs = 50

# Results:

- In this Iteration, we decrease the Learning rate from 1e-4 to 1e-5.
- After 47th Epoch, Gradient Descent fail to reach the local minima, The Classification Value rebounding to the 0.44 to 0.40
- At last, Finally Classification Value reaches to the 0.3853 and Given the MAP Value is **0.3544**
- This Iteration gave the Best IOU Values between **0.2 to 0.6.**

```
Epoch 00043: ReduceLROnPlateau reducing learning rate to 9.99999943962493e-12.
79/79 [==============================] - 361s 5s/step - loss: 2.3140 - regression_loss: 1.9300 - classification_loss: 0.3841
Epoch 44/50
79/79 [==============================] - ETA: 0s - loss: 2.3098 - regression_loss: 1.9264 - classification_loss: 0.3834
Epoch 00044: saving model to ./snapshots/resnet50_csv_44.h5
79/79 [==============================] - 359s 5s/step - loss: 2.3098 - regression_loss: 1.9264 - classification_loss: 0.3834
Epoch 45/50
79/79 [==============================] - ETA: 0s - loss: 2.3056 - regression_loss: 1.9214 - classification_loss: 0.3842
Epoch 00045: saving model to ./snapshots/resnet50_csv_45.h5
79/79 [==============================] - 360s 5s/step - loss: 2.3056 - regression_loss: 1.9214 - classification_loss: 0.3842
Epoch 46/50
79/79 [==============================] - ETA: 0s - loss: 2.3144 - regression_loss: 1.9284 - classification_loss: 0.3859
Epoch 00046: saving model to ./snapshots/resnet50_csv_46.h5
79/79 [==============================] - 364s 5s/step - loss: 2.3144 - regression_loss: 1.9284 - classification_loss: 0.3859
Epoch 47/50
79/79 [==============================] - ETA: 0s - loss: 2.3099 - regression_loss: 1.9260 - classification_loss: 0.3839
Epoch 00047: saving model to ./snapshots/resnet50_csv_47.h5

Epoch 00047: ReduceLROnPlateau reducing learning rate to 9.999999092680235e-13.
79/79 [==============================] - 365s 5s/step - loss: 2.3099 - regression_loss: 1.9260 - classification_loss: 0.3839
Epoch 48/50
79/79 [==============================] - ETA: 0s - loss: 2.3218 - regression_loss: 1.9365 - classification_loss: 0.3853
Epoch 00048: saving model to ./snapshots/resnet50_csv_48.h5
79/79 [==============================] - 363s 5s/step - loss: 2.3218 - regression_loss: 1.9365 - classification_loss: 0.3853
Epoch 49/50
79/79 [==============================] - ETA: 0s - loss: 2.3046 - regression_loss: 1.9213 - classification_loss: 0.3833
Epoch 00049: saving model to ./snapshots/resnet50_csv_49.h5
79/79 [==============================] - 357s 5s/step - loss: 2.3046 - regression_loss: 1.9213 - classification_loss: 0.3833
Epoch 50/50
79/79 [==============================] - ETA: 0s - loss: 2.3163 - regression_loss: 1.9310 - classification_loss: 0.3853
Epoch 00050: saving model to ./snapshots/resnet50_csv_50.h5
79/79 [==============================] - 360s 5s/step - loss: 2.3163 - regression_loss: 1.9310 - classification_loss: 0.3853
```

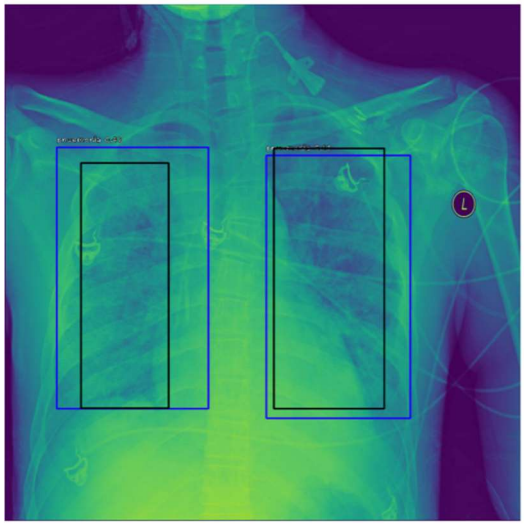
```
2020-12-17 01:22:34.539392: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1263]
2020-12-17 01:22:34.539401: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1276] 0:
2020-12-17 01:22:34.539640: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982]
2020-12-17 01:22:34.540352: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982]
2020-12-17 01:22:34.540879: W tensorflow/core/common_runtime/gpu/gpu_bfc_allocator.cc:39
2020-12-17 01:22:34.540932: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1402] Cre
Running network: N/A% (0 of 2519) |       | Elapsed Time: 0:00:00 ETA:  --:--:--2020-12-
2020-12-17 01:22:41.316659: I tensorflow/stream_executor/platform/default/dso_loader.cc:
Running network: 100% (2519 of 2519) |####| Elapsed Time: 0:03:08 Time:  0:03:08
Parsing annotations: 100% (2519 of 2519) || Elapsed Time: 0:00:00 Time:  0:00:00
4000 instances of class Pneumonia with average precision: 0.3544
Inference time for 2519 images: 0.0375
mAP using the weighted average of precisions among classes: 0.3544
mAP: 0.3544
```

```
1 THRES_SCORE = 0.4
2 patient_id='8047b765-c93a-4059-8a09-9b1e809809e7'
3 draw_image(patient_id)
```

Mean_IOU:.0.11820087218146784


```
1 THRES_SCORE = 0.4
2 patient_id='41b00808-a610-492c-b2f5-82de09d84366'
3 draw_image(patient_id)
```

Mean_IOU:.0.633755597263274




## Iteration 5:

Learning Rate = default=1e-06

No of Epochs = 150

## Results:

- In this Iteration, we decrease the Learning rate from 1e-5 to 1e-6.
- After 47th Epoch, Gradient Descent fail to reach the local minima, The Classification Value rebounding to the 0.44 to 0.40
- I think, due to the less learning rate, the gradient unable to jump to the to another plateau to reach the global minima.

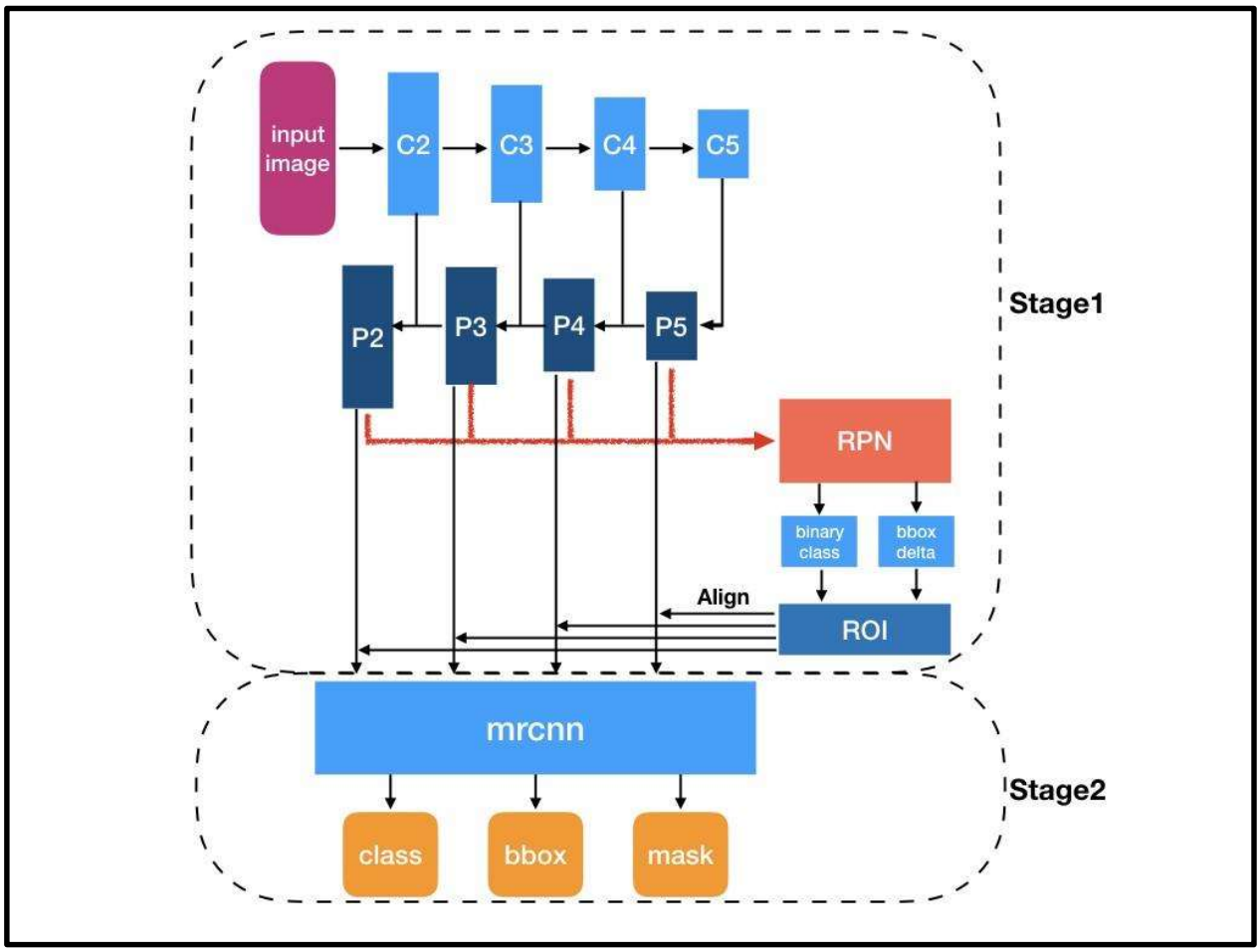
- At last, Finally Classification Value reaches to the 0.4425 and Given the MAP Value is **0.2102**
- This Iteration gave the Best IOU Values between **0 to 0.2.**

```
79/79 [==============================] - 395s 5s/step - loss: 2.5355 - regression_loss: 2.0964 - classification_loss: 0.4391
Epoch 144/150
79/79 [==============================] - ETA: 0s - loss: 2.5490 - regression_loss: 2.1069 - classification_loss: 0.4420
Epoch 00144: saving model to ./snapshots/resnet50_csv_144.h5
79/79 [==============================] - 393s 5s/step - loss: 2.5490 - regression_loss: 2.1069 - classification_loss: 0.4420
Epoch 145/150
79/79 [==============================] - ETA: 0s - loss: 2.5349 - regression_loss: 2.0936 - classification_loss: 0.4412
Epoch 00145: saving model to ./snapshots/resnet50_csv_145.h5
79/79 [==============================] - 396s 5s/step - loss: 2.5349 - regression_loss: 2.0936 - classification_loss: 0.4412
Epoch 146/150
79/79 [==============================] - ETA: 0s - loss: 2.5493 - regression_loss: 2.1086 - classification_loss: 0.4407
Epoch 00146: saving model to ./snapshots/resnet50_csv_146.h5
79/79 [==============================] - 400s 5s/step - loss: 2.5493 - regression_loss: 2.1086 - classification_loss: 0.4407
Epoch 147/150
79/79 [==============================] - ETA: 0s - loss: 2.5397 - regression_loss: 2.0992 - classification_loss: 0.4405
Epoch 00147: saving model to ./snapshots/resnet50_csv_147.h5
79/79 [==============================] - 399s 5s/step - loss: 2.5397 - regression_loss: 2.0992 - classification_loss: 0.4405
Epoch 148/150
79/79 [==============================] - ETA: 0s - loss: 2.5404 - regression_loss: 2.0990 - classification_loss: 0.4413
Epoch 00148: saving model to ./snapshots/resnet50_csv_148.h5
79/79 [==============================] - 404s 5s/step - loss: 2.5404 - regression_loss: 2.0990 - classification_loss: 0.4413
Epoch 149/150
79/79 [==============================] - ETA: 0s - loss: 2.5434 - regression_loss: 2.1022 - classification_loss: 0.4413
Epoch 00149: saving model to ./snapshots/resnet50_csv_149.h5
79/79 [==============================] - 401s 5s/step - loss: 2.5434 - regression_loss: 2.1022 - classification_loss: 0.4413
Epoch 150/150
79/79 [==============================] - ETA: 0s - loss: 2.5472 - regression_loss: 2.1047 - classification_loss: 0.4425
Epoch 00150: saving model to ./snapshots/resnet50_csv_150.h5
79/79 [==============================] - 399s 5s/step - loss: 2.5472 - regression_loss: 2.1047 - classification_loss: 0.4425
```

```
2020-12-18 06:23:21.031037: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1276] 0:
2020-12-18 06:23:21.031237: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] s
2020-12-18 06:23:21.031925: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:982] s
2020-12-18 06:23:21.032533: W tensorflow/core/common_runtime/gpu/gpu_bfc_allocator.cc:39]
2020-12-18 06:23:21.032578: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1402] Crea
Running network: N/A% (0 of 2519) |       | Elapsed Time: 0:00:00 ETA:  --:--:--2020-12-1
2020-12-18 06:23:28.909444: I tensorflow/stream_executor/platform/default/dso_loader.cc:4
Running network: 100% (2519 of 2519) |####| Elapsed Time: 0:03:36 Time:  0:03:36
Parsing annotations: 100% (2519 of 2519) || Elapsed Time: 0:00:00 Time:  0:00:00
4000 instances of class Pneumonia with average precision: 0.2102
Inference time for 2519 images: 0.0445
mAP using the weighted average of precisions among classes: 0.2102
mAP: 0.2102
```

## 5.2.5 Mask RCNN:

**Simple understanding of the model structure**:



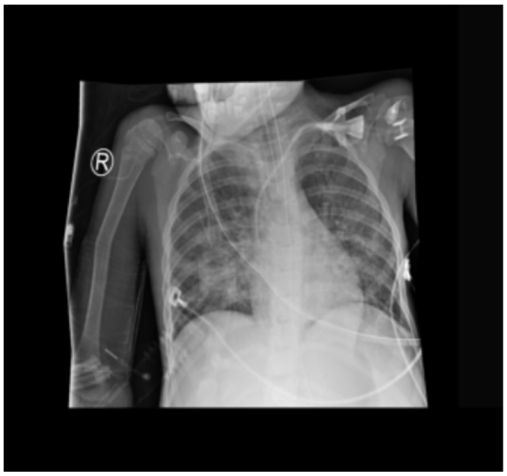
**Implementation details**:

- COCO Pre-trained weights are used as the initial weights for mask rcnn model.

- Overridden the mask rcnn configuration according to the use-case

```
class DetectorConfig(Config):
    """Configuration for training pneumonia detection on the RSNA pneumonia dataset.
    Overrides values in the base Config class.
    """

    # Give the configuration a recognizable name
    NAME = 'pneumonia'

    # Train on 1 GPU and 8 images per GPU. We can put multiple images on each
    # GPU because the images are small. Batch size is 8 (GPUs * images/GPU).
    GPU_COUNT = 1
    IMAGES_PER_GPU = 8

    BACKBONE = 'resnet50'

    NUM_CLASSES = 2  # background + 1 pneumonia classes

    IMAGE_MIN_DIM = 256
    IMAGE_MAX_DIM = 256
    RPN_ANCHOR_SCALES = (16, 32, 64, 128)
    TRAIN_ROIS_PER_IMAGE = 32
    MAX_GT_INSTANCES = 4
    DETECTION_MAX_INSTANCES = 3
    DETECTION_MIN_CONFIDENCE = 0.78  ## match target distribution
    DETECTION_NMS_THRESHOLD = 0.01

    STEPS_PER_EPOCH = 200
```

- Implemented image augmentation techniques such as Contrast, Sharpen etc.
    - Image after augmentation

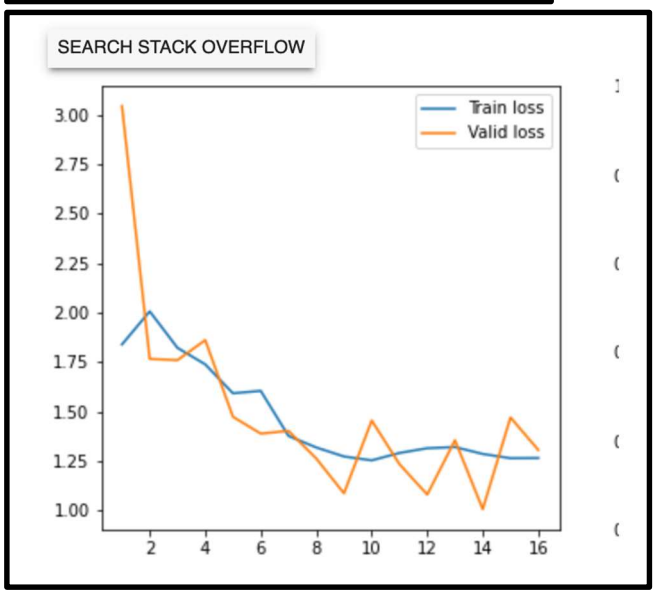


- Parameters used in training:
  - Epochs: 16
  - Initial learning rate: 0.012
  - Optimizer: SGT

**Loss and validation loss through epochs**:

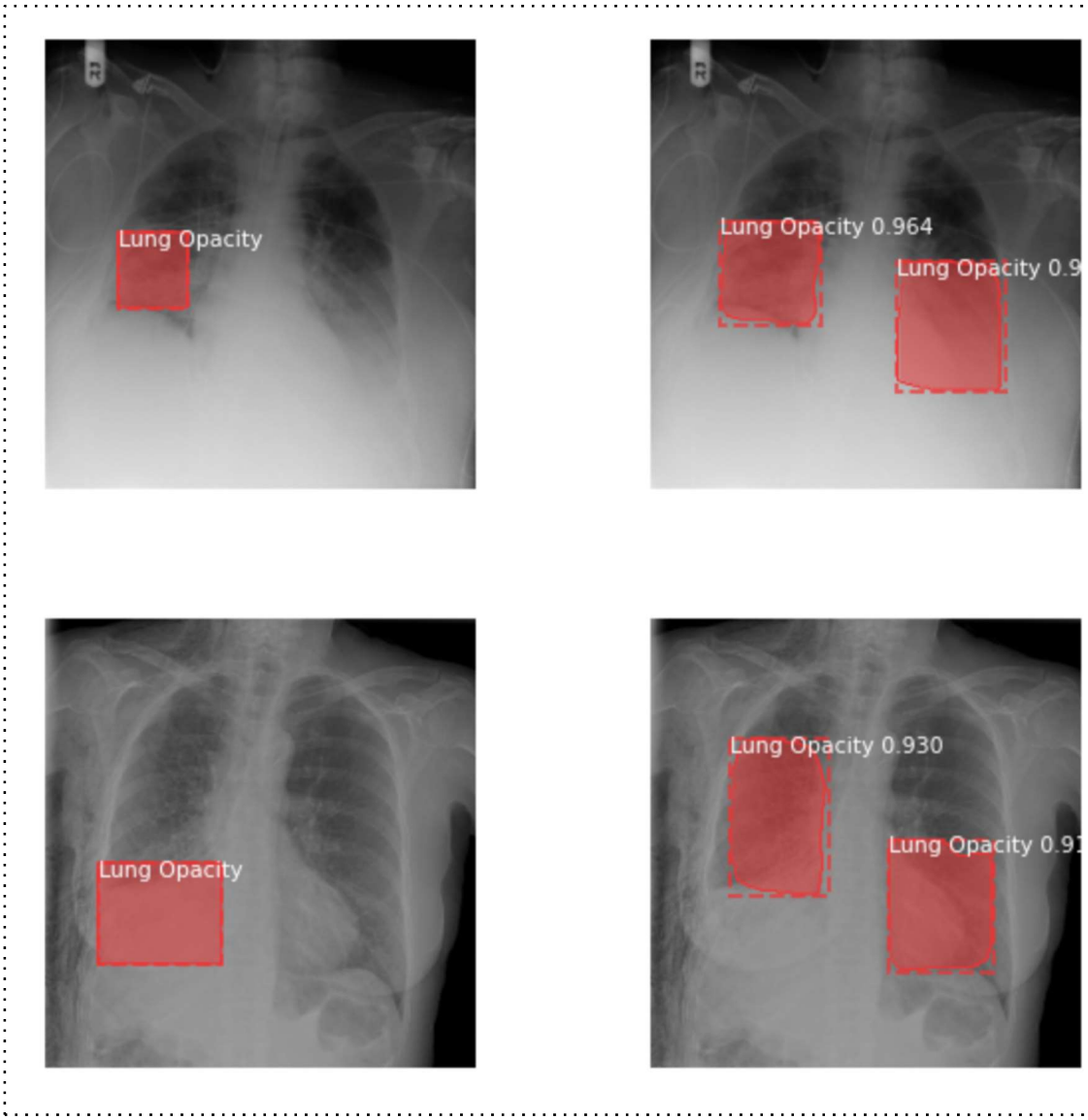
```
Epoch 7/16
200/200 [==============================] - 1192s 6s/step - loss: 1.3769 - val_loss: 1.4024
Epoch 8/16
200/200 [==============================] - 1167s 6s/step - loss: 1.3186 - val_loss: 1.2631
Epoch 9/16
200/200 [==============================] - 1138s 6s/step - loss: 1.2730 - val_loss: 1.0868
Epoch 10/16
200/200 [==============================] - 1074s 5s/step - loss: 1.2535 - val_loss: 1.4548
Epoch 11/16
200/200 [==============================] - 1444s 7s/step - loss: 1.2909 - val_loss: 1.2340
Epoch 12/16
200/200 [==============================] - 1796s 9s/step - loss: 1.3148 - val_loss: 1.0807
Epoch 13/16
200/200 [==============================] - 1811s 9s/step - loss: 1.3206 - val_loss: 1.3553
Epoch 14/16
200/200 [==============================] - 1106s 6s/step - loss: 1.2859 - val_loss: 1.0069
Epoch 15/16
200/200 [==============================] - 1068s 5s/step - loss: 1.2643 - val_loss: 1.4702
Epoch 16/16
200/200 [==============================] - 1056s 5s/step - loss: 1.2652 - val_loss: 1.3062
```

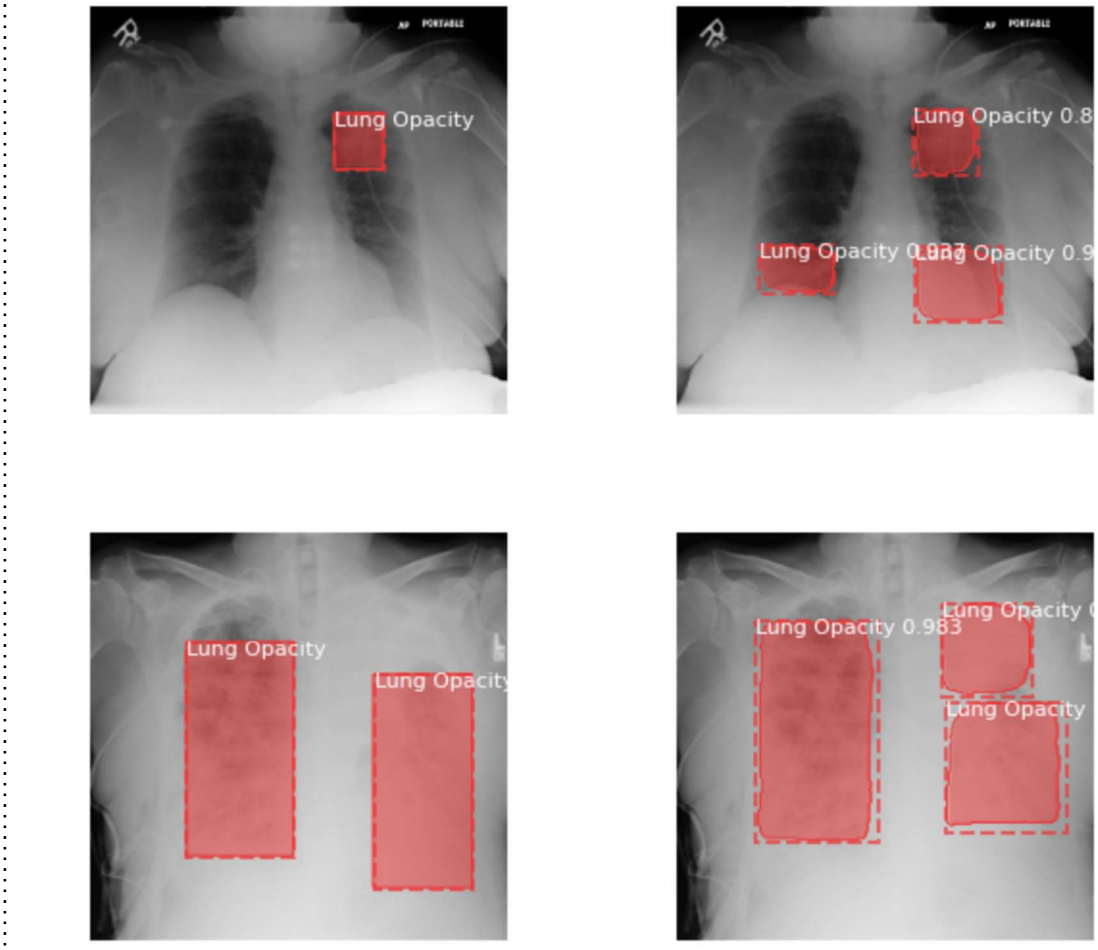
| | val_loss | loss |
|---|---|---|
| 1 | 3.043316 | 1.838753 |
| 2 | 1.765986 | 2.006115 |
| 3 | 1.759799 | 1.821804 |
| 4 | 1.860938 | 1.738347 |
| 5 | 1.473331 | 1.592146 |
| 6 | 1.388740 | 1.604584 |
| 7 | 1.402411 | 1.376887 |
| 8 | 1.263067 | 1.318632 |
| 9 | 1.086779 | 1.272991 |
| 10 | 1.454799 | 1.253452 |
| 11 | 1.234037 | 1.290862 |
| 12 | 1.080733 | 1.314803 |
| 13 | 1.355265 | 1.320634 |
| 14 | 1.006877 | 1.285908 |
| 15 | 1.470187 | 1.264263 |
| 16 | 1.306225 | 1.265177 |



**Prediction on training images**:

**On left: Ground truth** **on right: Prediction**

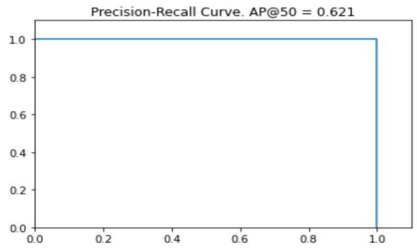





## Model Evaluation:

```
[ ] APs, precisions, recalls = compute_batch_ap(detailed_image_ids)
    print("mAP: ", np.mean(APs))

    visualize.plot_precision_recall(np.mean(APs), precisions, recalls)
```

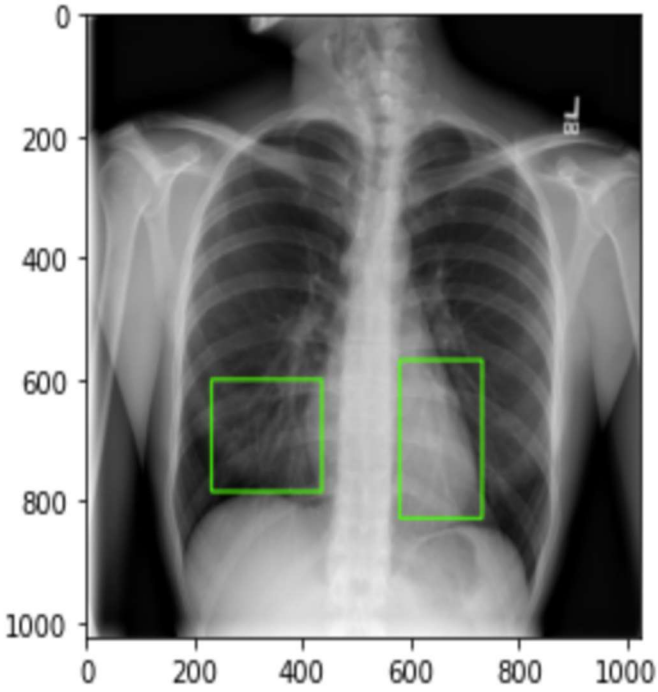
```
mAP:  0.6207951073222447
```

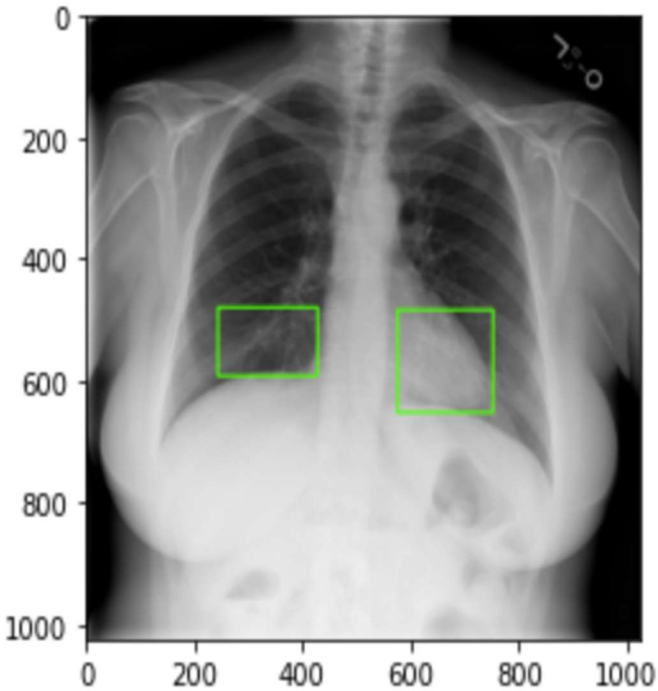



**The mAP across the validation dataset is : 0.62**

## Avg IOU Score ~ 70%

**Prediction on test images**:



# Summary:

- EDA has given some meaningful insights about the data.

**RetinaNet:**

- Retina Net Model gives the good results.
- In the Iteration# 3, With Learning rate= 1e-4 & 50 Epochs.
  - Classification Value reaches to the 0.335 and Given the MAP Value is **0.4814**
  - This Iteration gave the Best IOU Values between **0.5 to 0.7.**

**Mask RCCN:**

- Mask RCCN also gives good results on Predicting the Bounding Boxes.
- It gives the Val_loss 1.3 at the 16$^{th}$ Epoch which is saturated.
- The MAP Value of the model is 0.62.
- Avg IOU Score is around 0.7.

**References:**


- RetinaNet Paper
- Feature Pyramid Network Paper