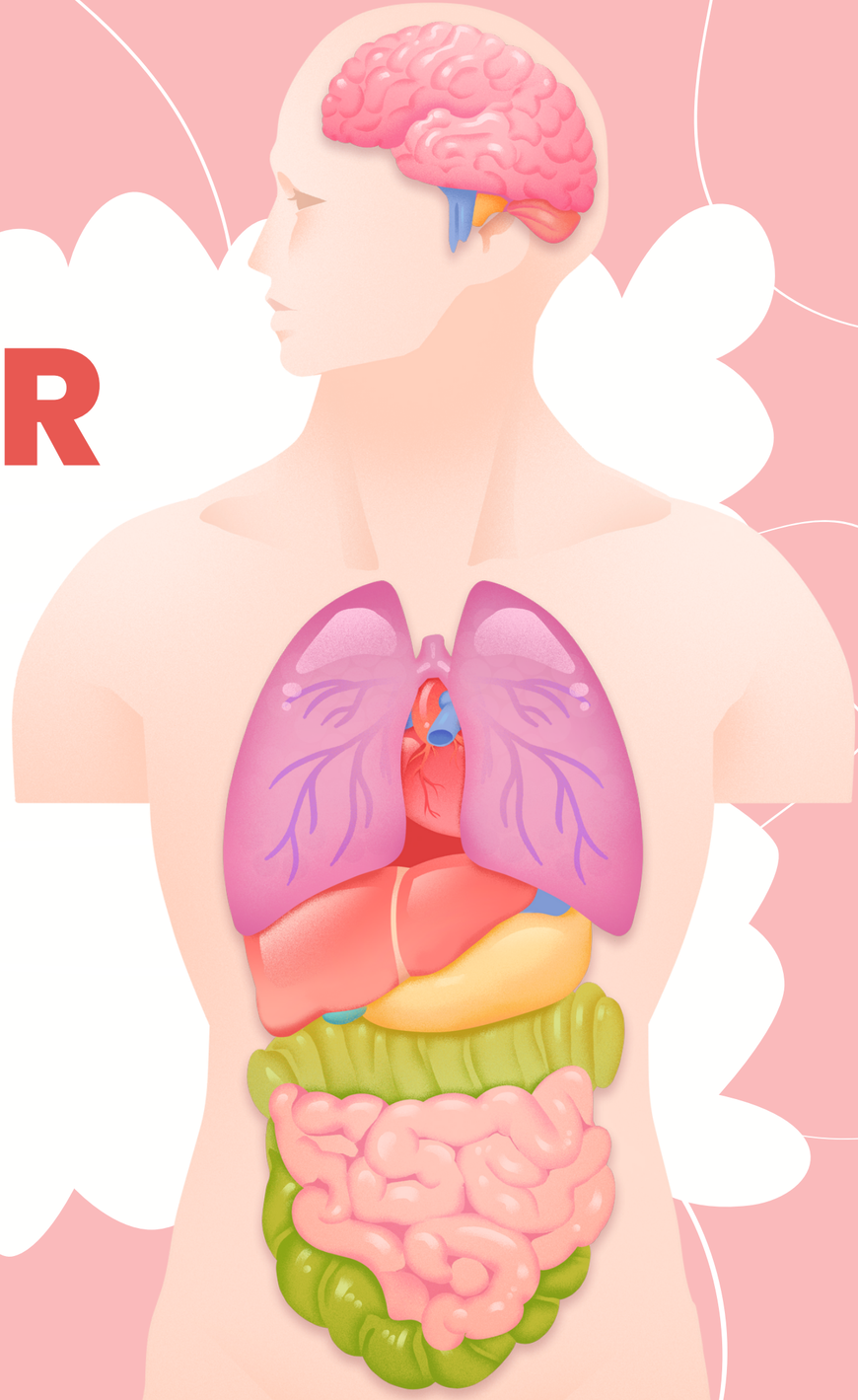
LUNGS CANCER

RECURRENCE PREDICTOR



Masters Thesis by

Tathagat Saha

Mat: 902046

University of Milan - Bicocca

Agenda

01 Introduction

02 Problem Statement

03 Dataset

04 Literature Referred

05 Process Flow

06 Data Preprocessing

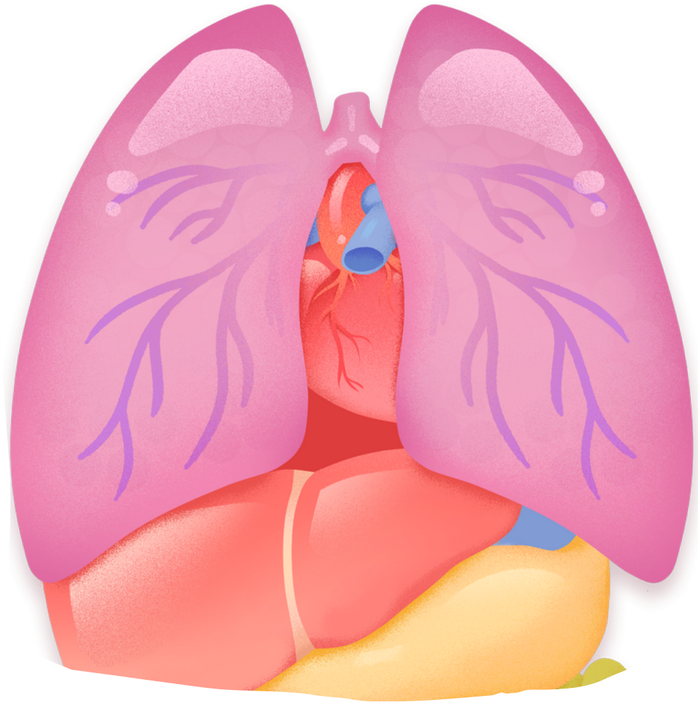
07 Data Augmentation

08 Dataset Splitting

09 Model Training

10 Performance Testing

11 Future Steps



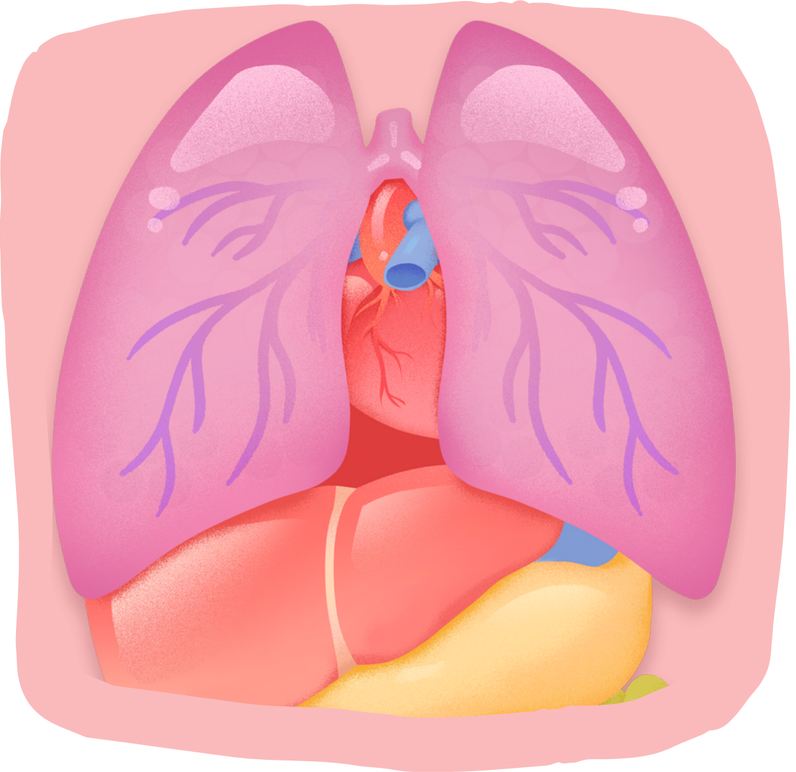
Introduction

General concepts

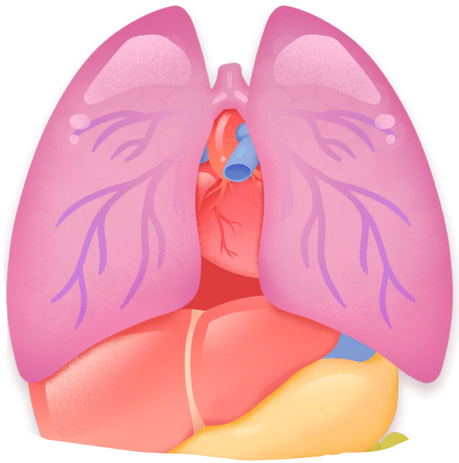
A CT scan, or computed tomography scan, is a medical imaging procedure that uses X-rays to create detailed cross-sectional images of the body. It provides more detailed information than conventional X-rays and is commonly used in various medical fields for diagnostic and treatment planning purposes

Output

It provides us a 3D view of the affected area in question, by taking multiples 2D slices of the entire area. A combination of all the 2D slices, gives us a holistic 3D view



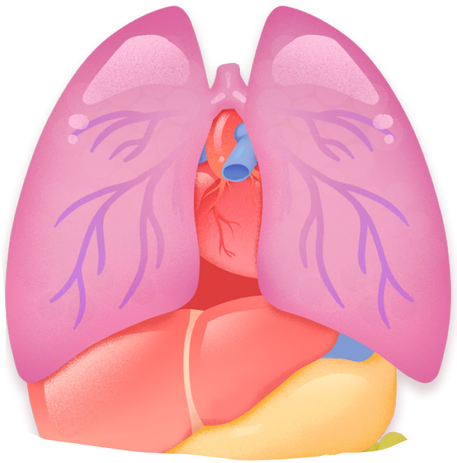
Problem Statement



Prediction of recurrency of lungs cancer before the actual surgery of the lungs tumour takes place.

Thus, helping surgeons to take precautionary measures before the actual surgery.

Dataset



01

The data we are using has been obtained from Cancer Imaging Archive. The dataset contains lungs CT scans of 211 patients

02

There also exists a csv file which contains the labels. The dataset has all patients with lungs cancer, but the labels indicate whether they had a recurrence or not

LITERATURE REFERRED

Uniformizing Techniques to Process CT scans with 3D CNNs for Tuberculosis Prediction by Hasib Zunair, Aimon Rahman, Nabeel Mohammed, Joseph Paul Cohen.

https://github.com/hepxpansion/Tatz_Thesis/blob/main/Research%20Papers/2007.13224.pdf

Automated detection and segmentation of non-small cell lung cancer computed tomography images

https://github.com/hepxpansion/Tatz_Thesis/blob/main/Research%20Papers/s41467-022-30841-3.pdf

Process Flow Diagram

Data Loading

Loading the data into numpy arrays from the available nii.gz fikes using the nibabel library

Data Pre-Processing

Data preproprocessing steps have been implemented on the whole dataset.

Labels Mapping

Labels have been acquired from the labels.csv file and mapped

Data Augmentation

The existing dataset was not balanced, hence Data Augmentation(scaling) was used to balance it.

Database Splitting

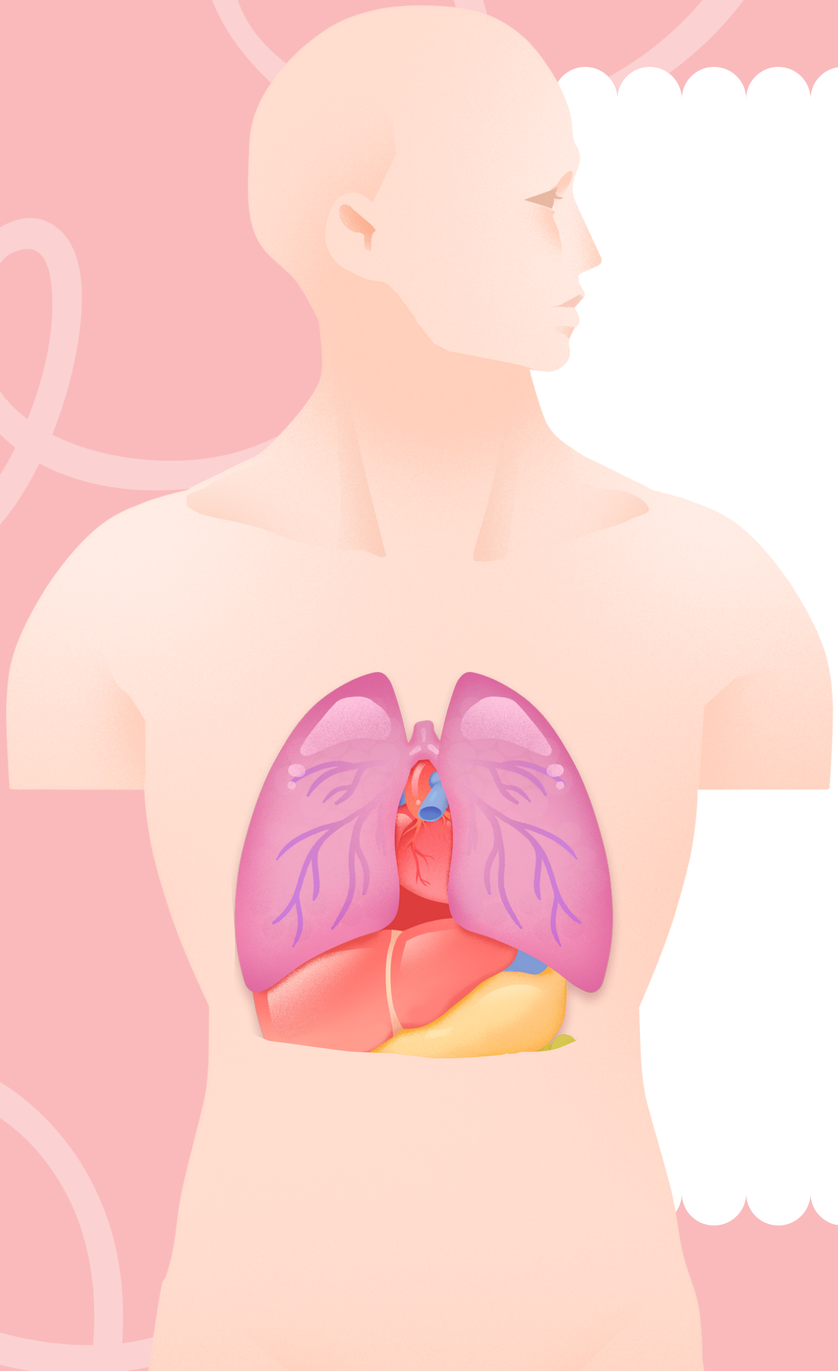
The given dataset was split in to 210(train), 64(validate), 28 (test)

Model Training

3D CNN has been used as a model and the model was trained with batches of 5

Performace Testing

After the mdoel is trained, the unused test data has been used for performance testing evaluation.



Data Pre-Processing

50%

Step 01

CT scans store raw voxel intensity in Hounsfield units (HU). Above 400 are bones with different radiointensity, so this is used as a higher bound. A threshold between -1000 and 400 is commonly used to normalize CT scans.

75%

Step 02

#We first rotate the volumes by 90 degrees, so the orientation is fixed
#We scale the HU values to be between 0 and 1

100%

Step 03

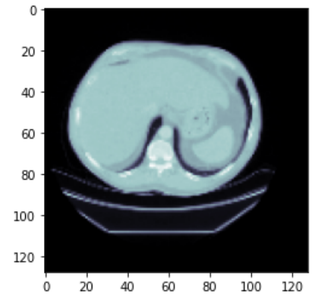
Downsample the scans to have shape of 128x128x64

BEFORE

CT scans with non-recurrent lung tissue: 156
CT scans with recurrent lung tissue: 54

CONSEQUENCE

The dataset is not balanced. Training with this dataset will inject a definite bias towards the non-recurrent class as the model will learn more from the more represented class.



Data Augmentation

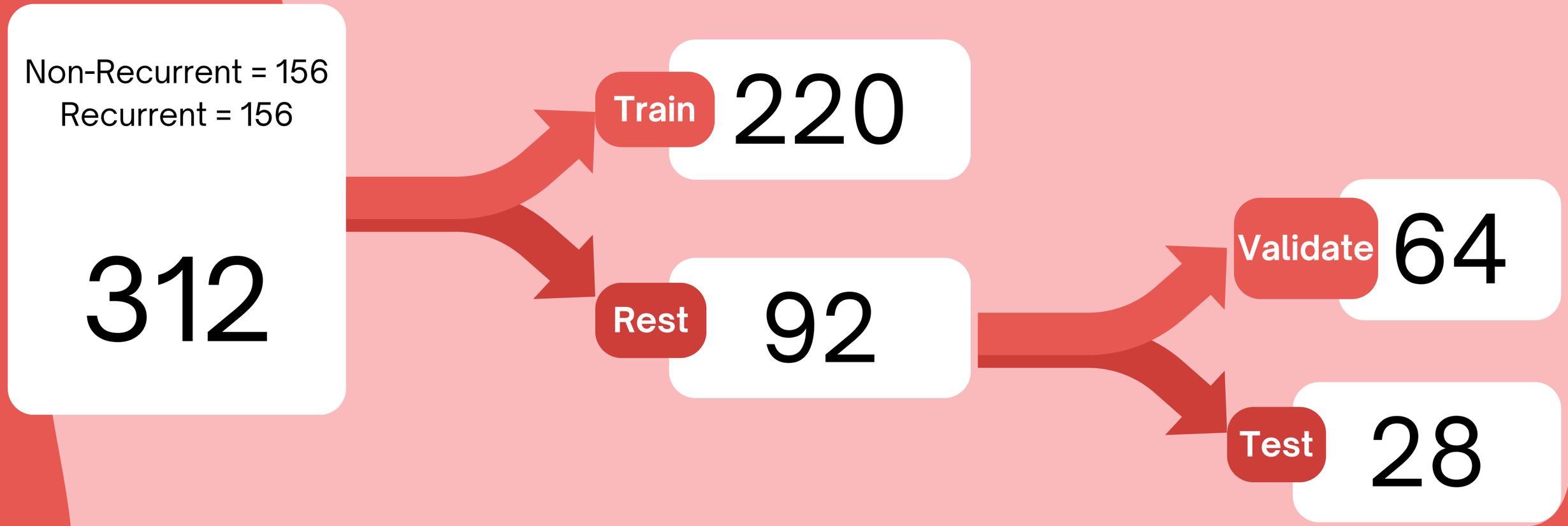
AFTER

CT scans with non-recurrent lung tissue: 156
CT scans with recurrent lung tissue: 156

STEPS

1. Use Affine transform
2. Apply scaling
3. Resample every image using these transform
4. Convert back to numpy array.
5. Apply the above steps only on the less represented class.

Dataset Splitting

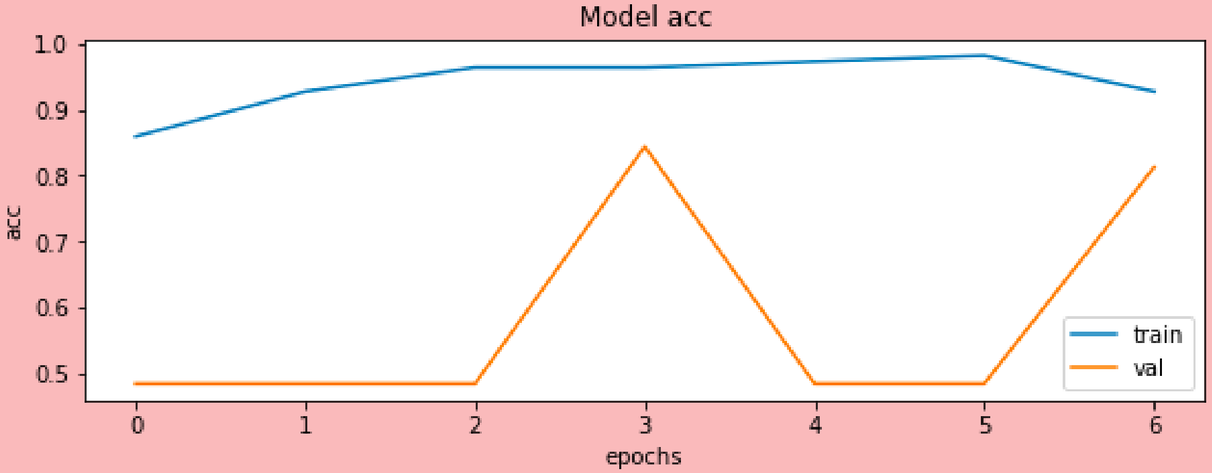


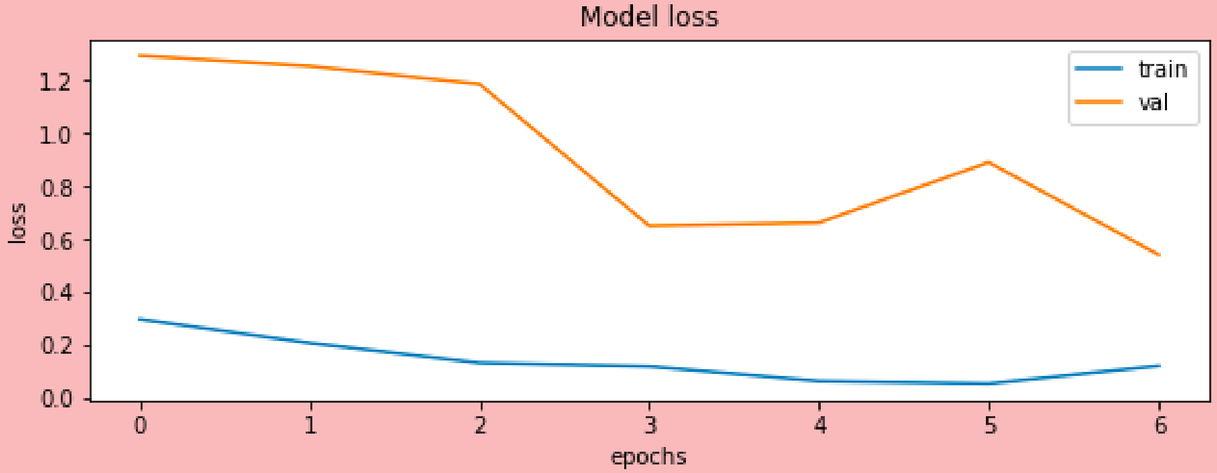
Model Training

EXPERIMENT 1

Data Augmentaion is done only in low represented class and the sugmentated data is present in Train, validation as well as test dataset.

Round 1: Training for 7 epochs



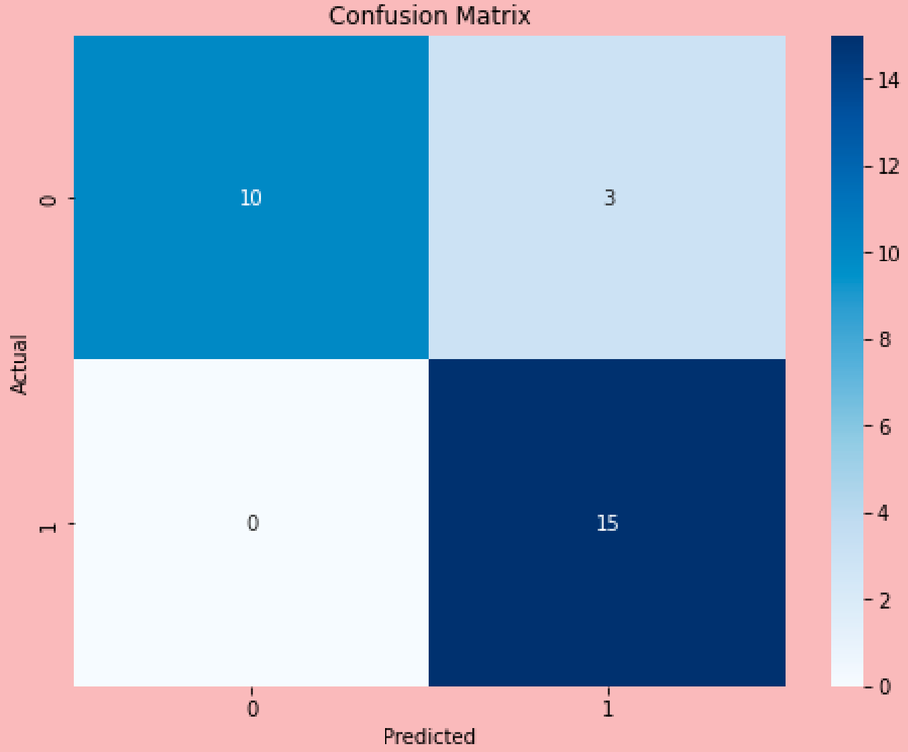


Train data points: 220

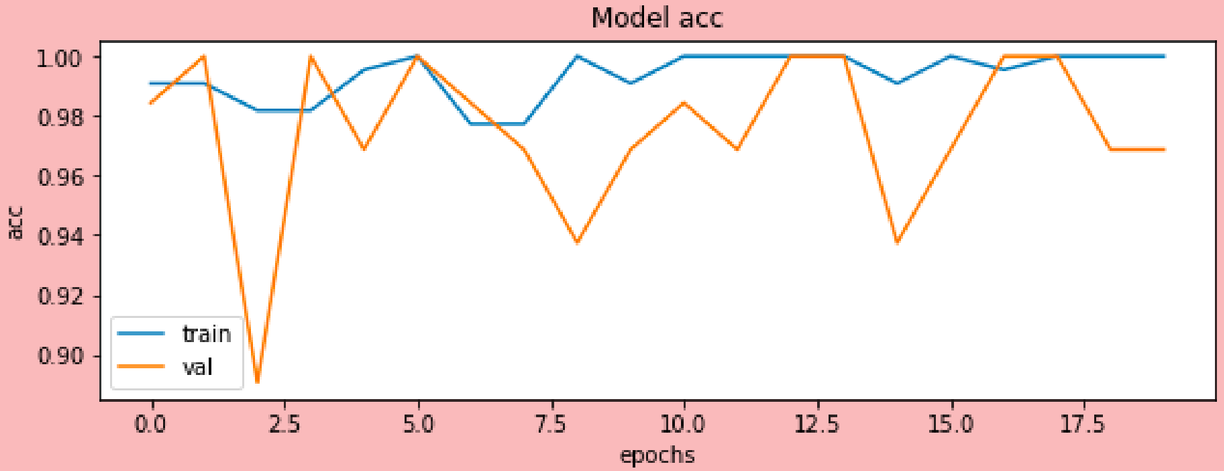
Validation data : 64

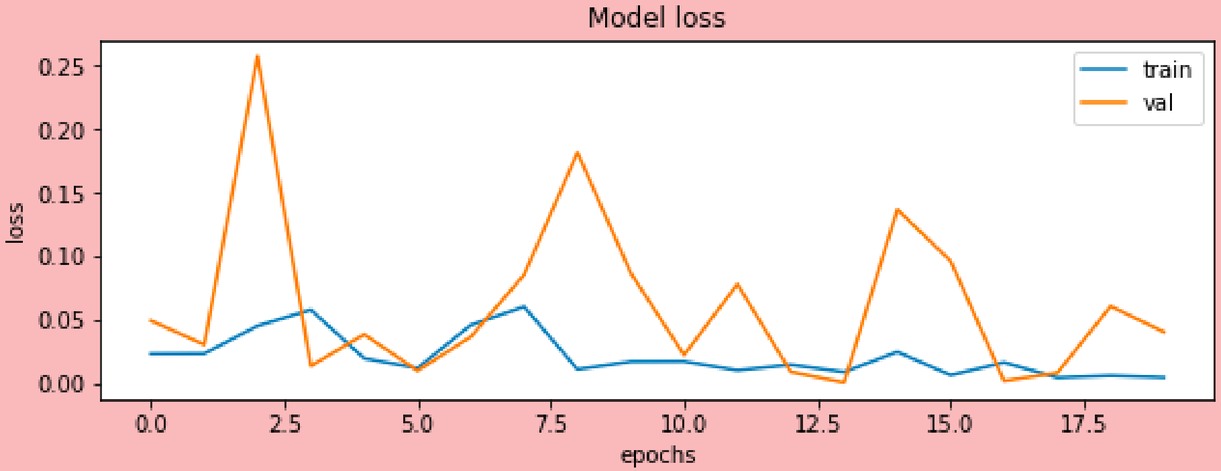
Test data: 28

Batch Size: 5



Round 2: After training for 20 epochs



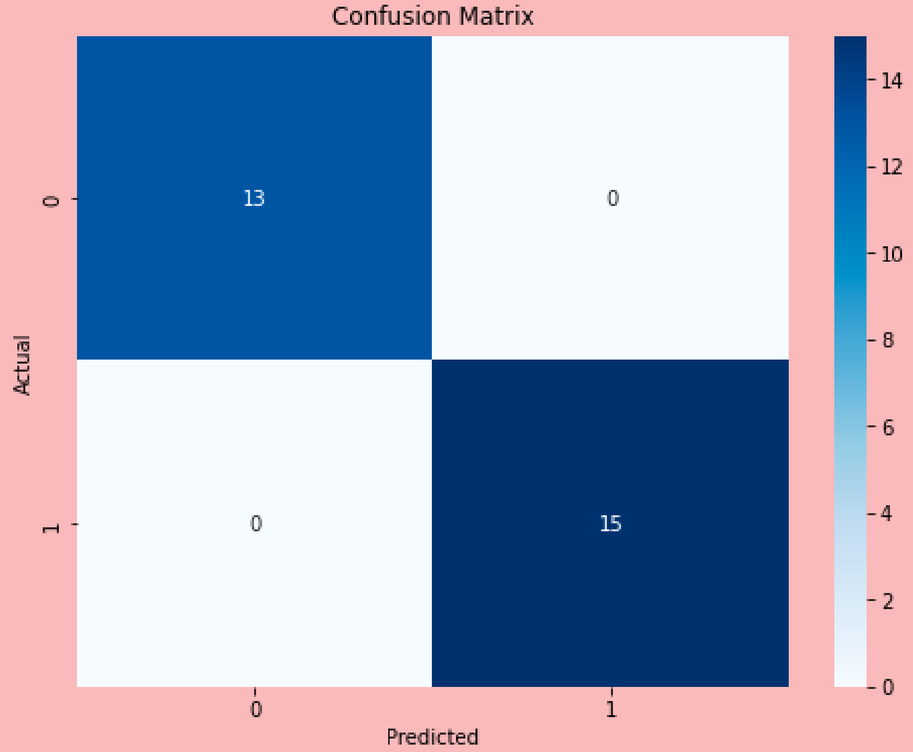


Train data points: 220

Validation data : 64

Test data: 28

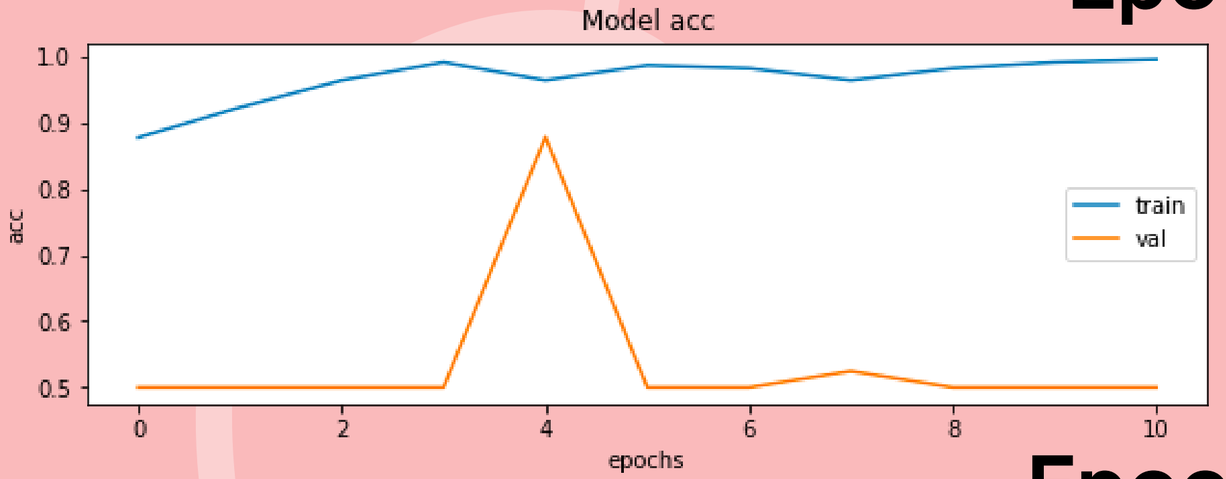
Batch Size: 5



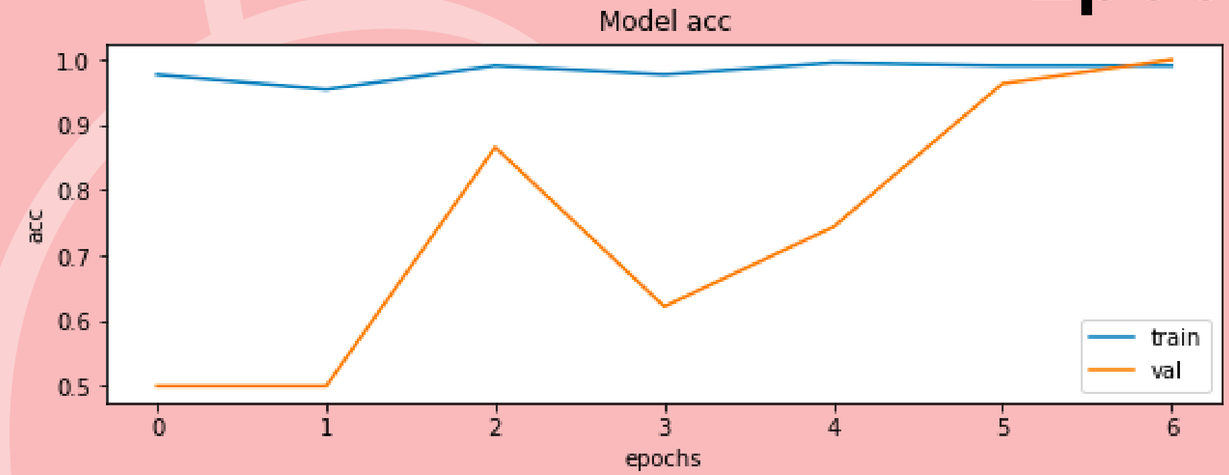
EXPERIMENT 2

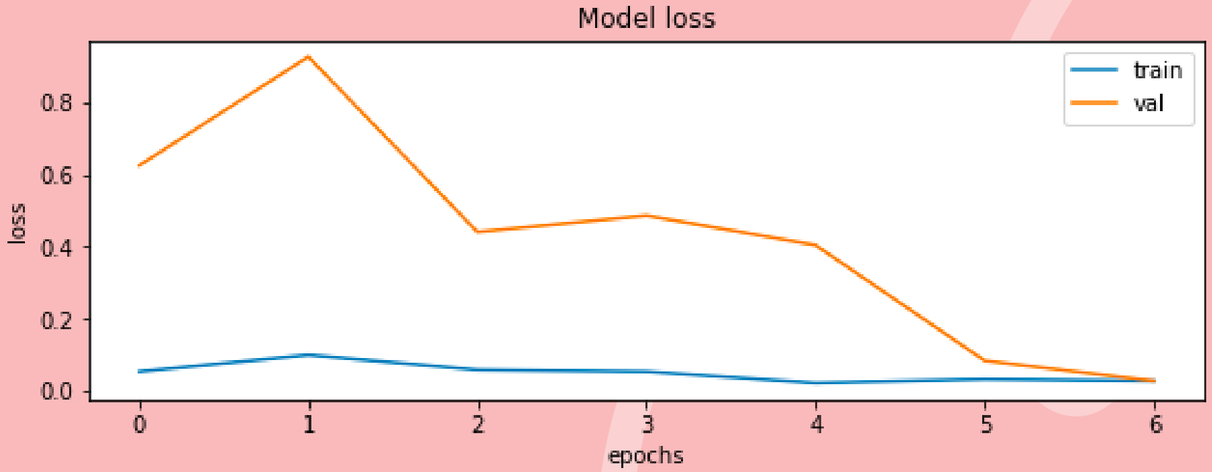
REMOVE AUGMENTED IMAGES FROM TEST

Epoch 1-10

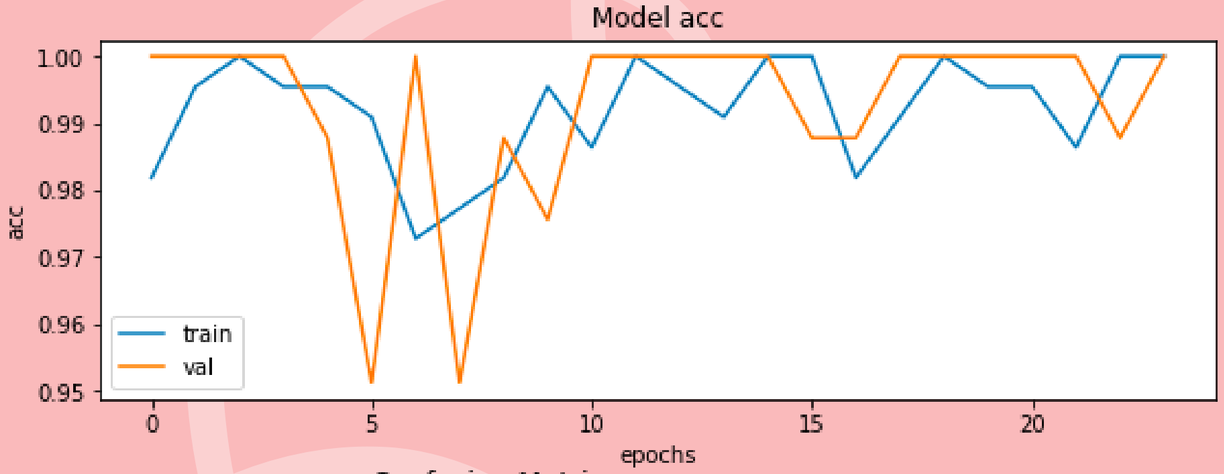


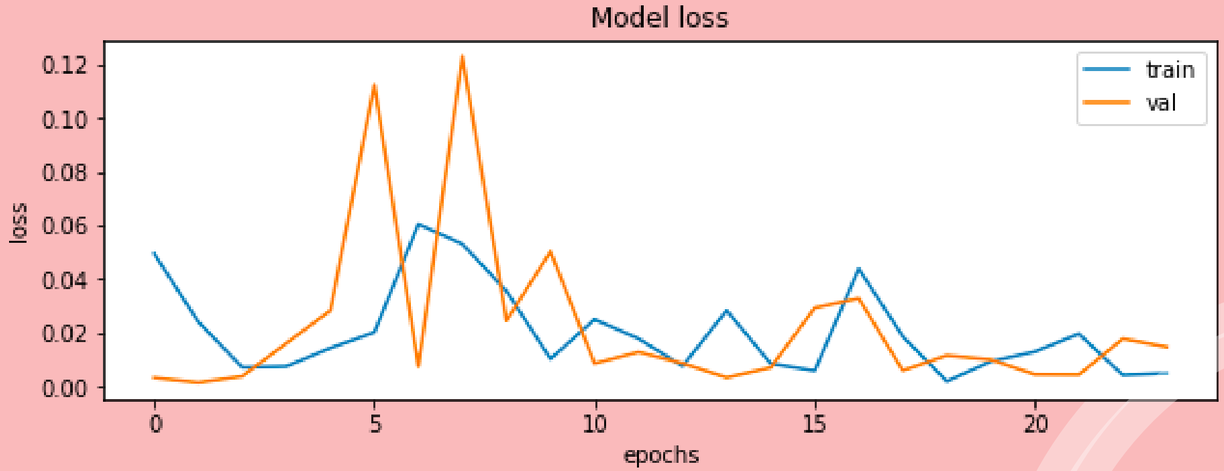
Epoch 10-17

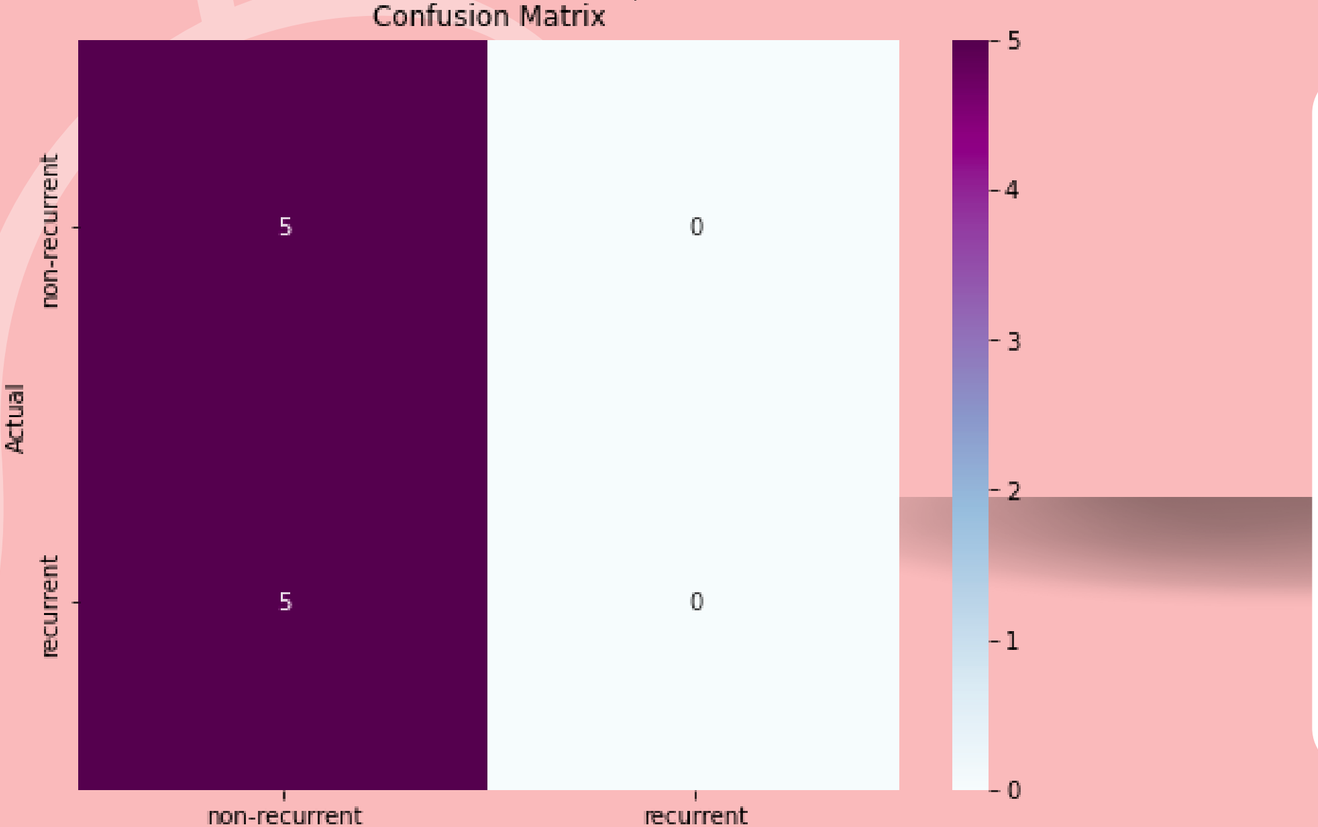




Epoch 17- 41







Accuracy of the model: 0.5714285714285714

Sensitivity: 0.0

Specificity:1.0

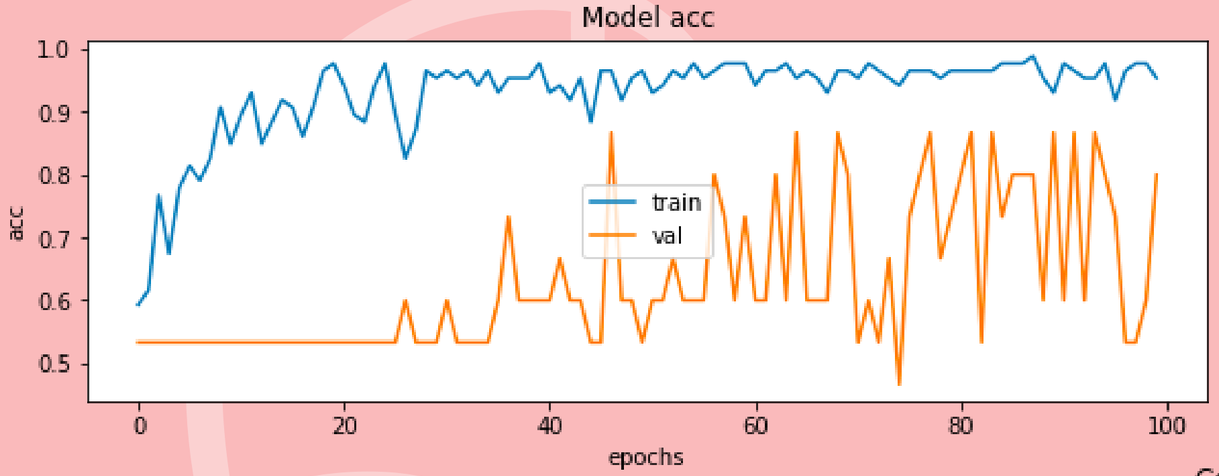
ROC :0.5

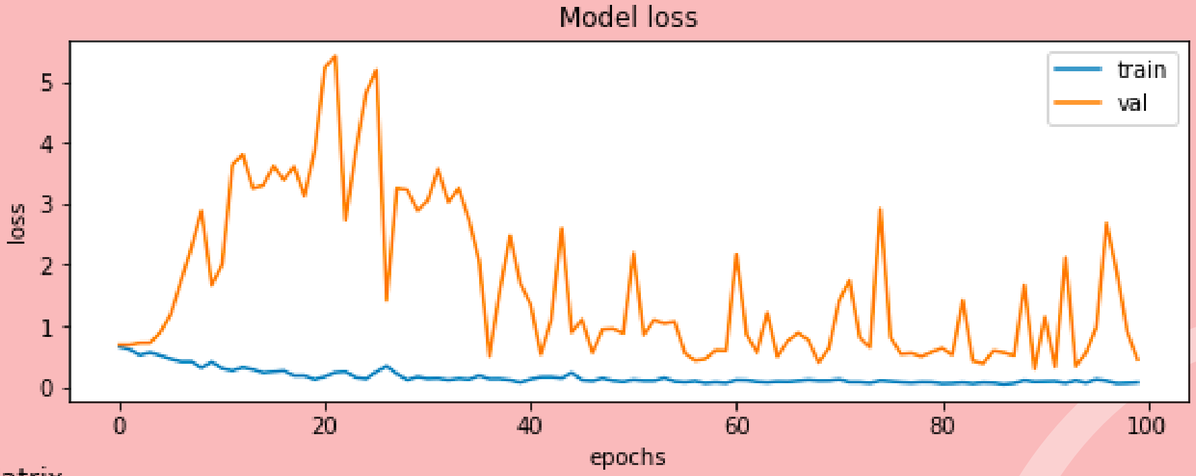
Balanced Accuracy:0.5

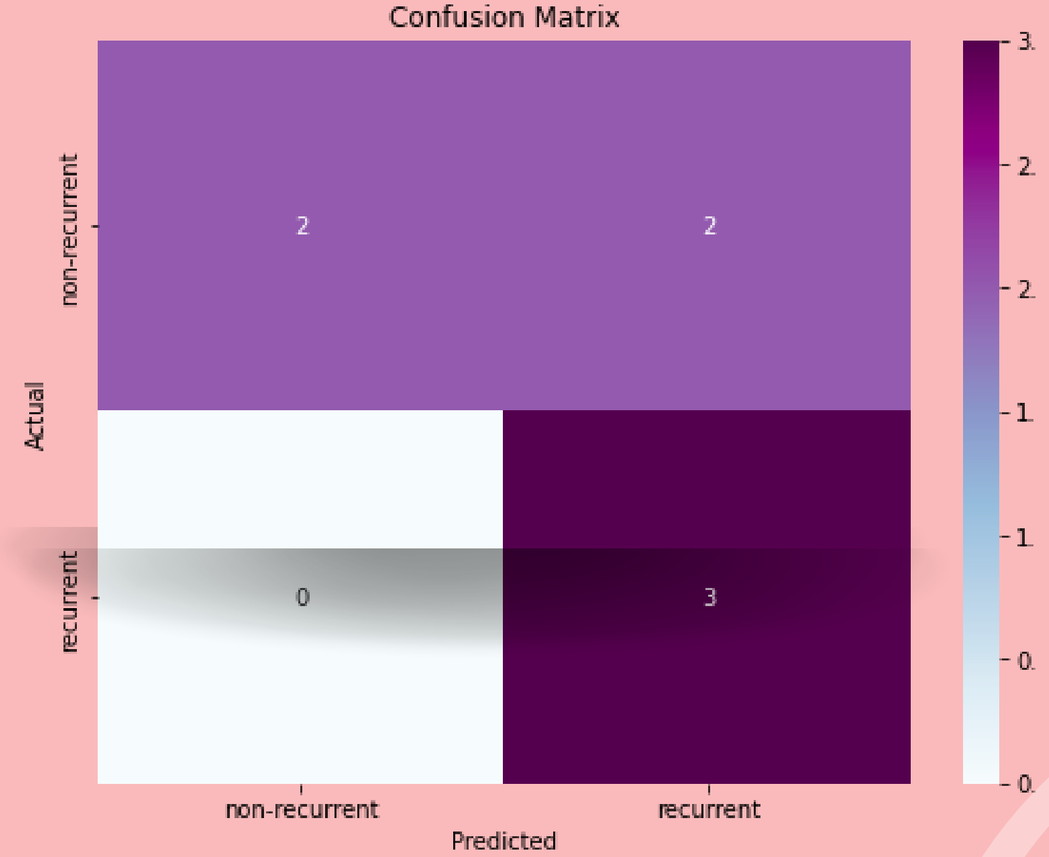
EXPERIMENT 3

REDUCE DATASET FROM HIGH REPRESENTED CLASS TO MATCH LOW REPRESENTED CLASS. NO DATA AUGMENTATION DONE

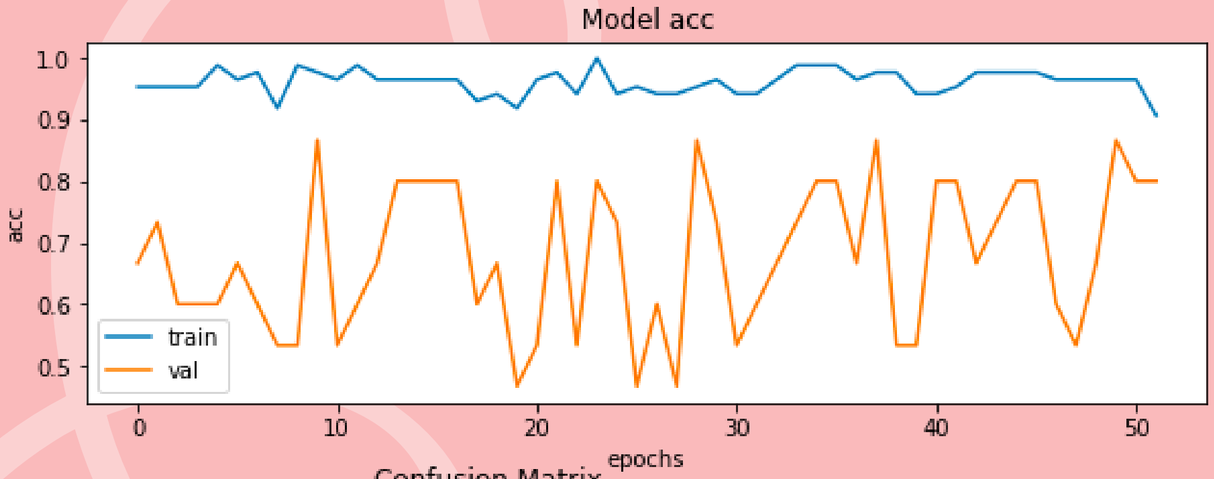
Epoch 1-100

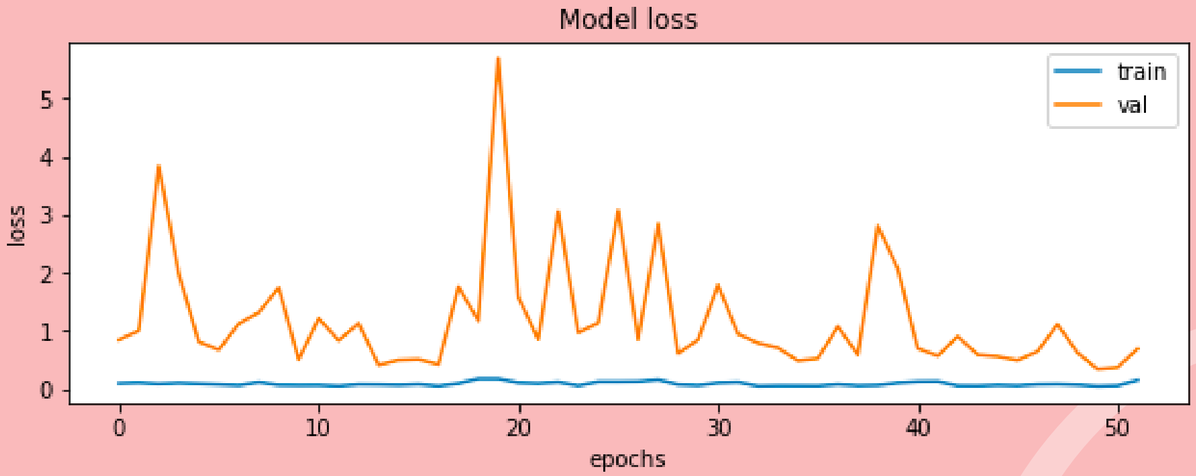

Model acc
acc
epochs
train
val
1.0
0.9
0.8
0.7
0.6
0.5
0
20
40
60
80
100

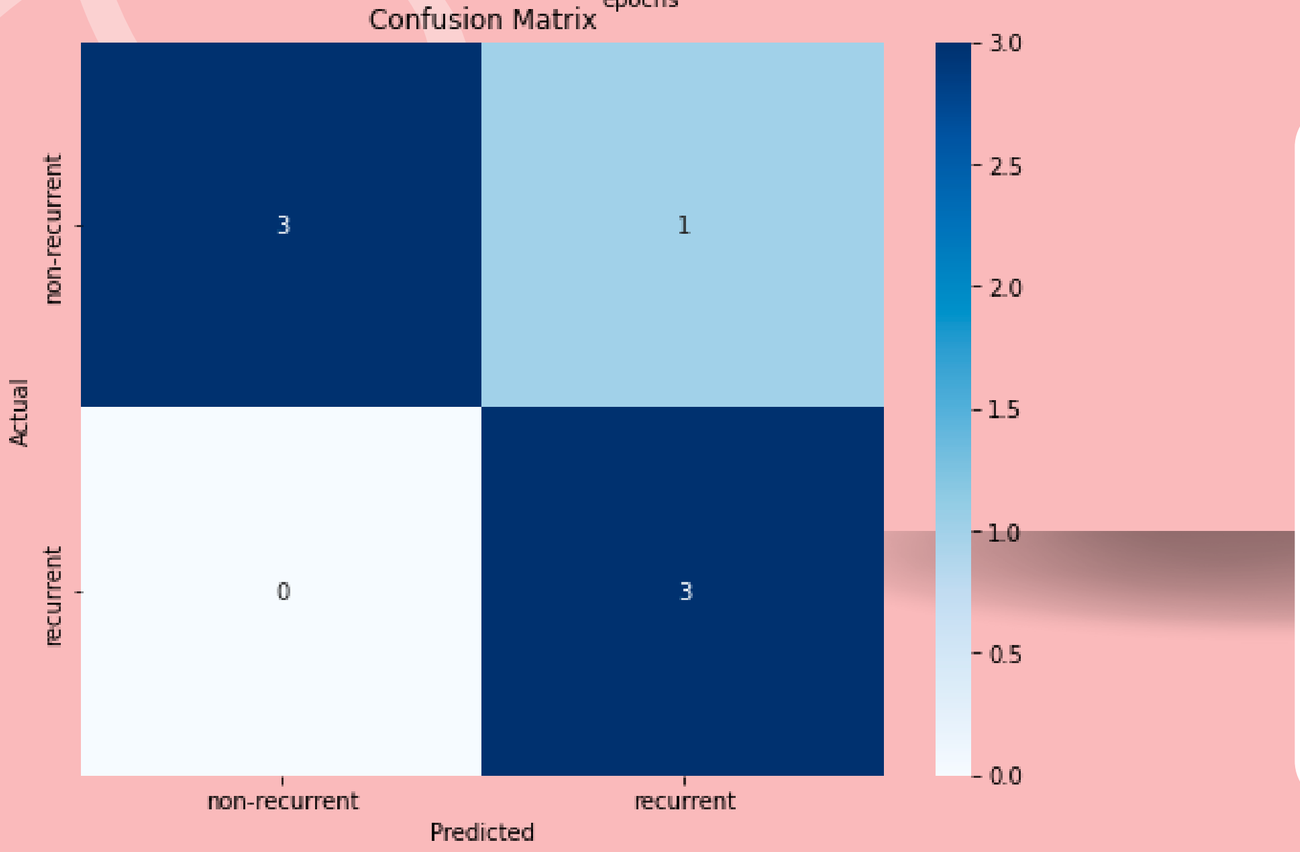

Model loss
loss
epochs
train
val
5
4
3
2
1
0
0
20
40
60
80
100


Confusion Matrix
Actual
non-recurrent
recurrent
2
2
0
3
non-recurrent
recurrent
Predicted
3.
2.
2.
1.
1.
0.
0.

Epoch 100-147







Accuracy of the model: 0.8571428571428571

Sensitivity: 1.0

Specificity:0.75

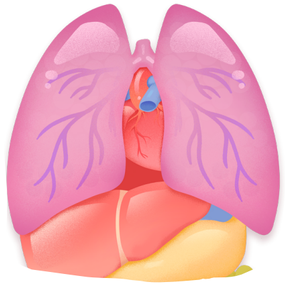
ROC :0.875

Balanced Accuracy:0.875

EXPERIMENT 4

No Data Augmentation. Train on real data. Add weights to the low represented class to address data imbalance.

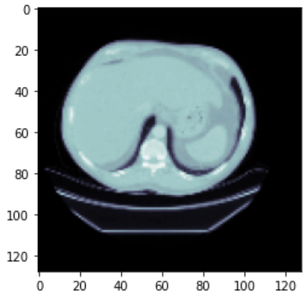
Class Weights



NON RECURRENT CLASS

This class has 156 samples. 6 samples has been kept as test data points.

So we end up having 150 data samples

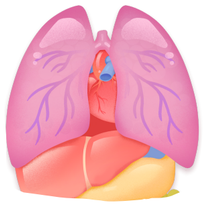


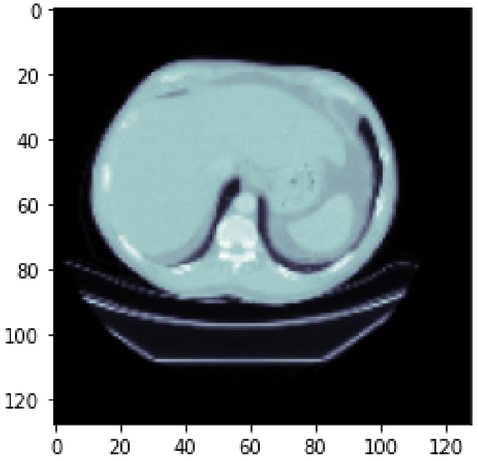
RECURRENT CLASS

This class has 54 samples. 5 samples has been kept as test data points.

So we end up having 49 data samples

WEIGHTS

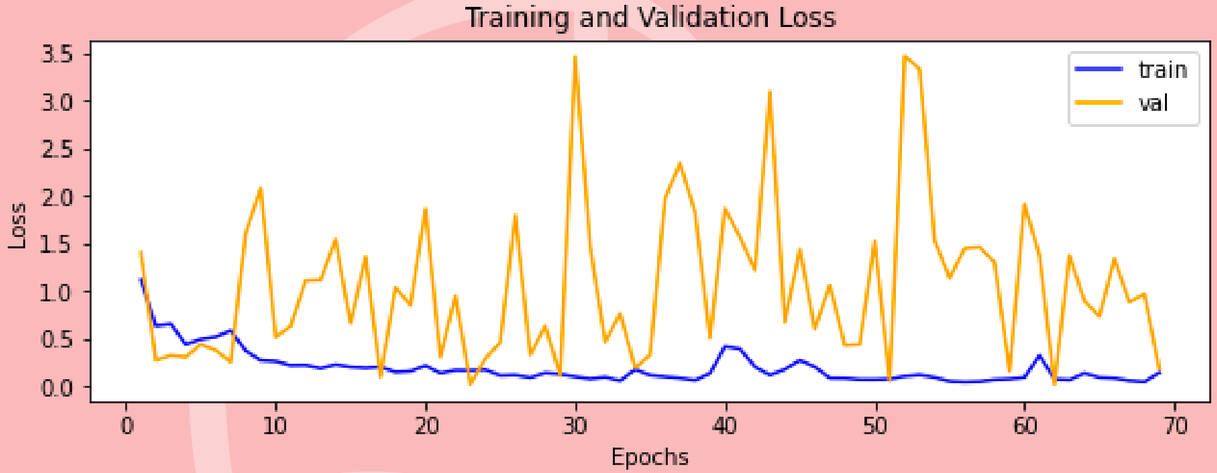


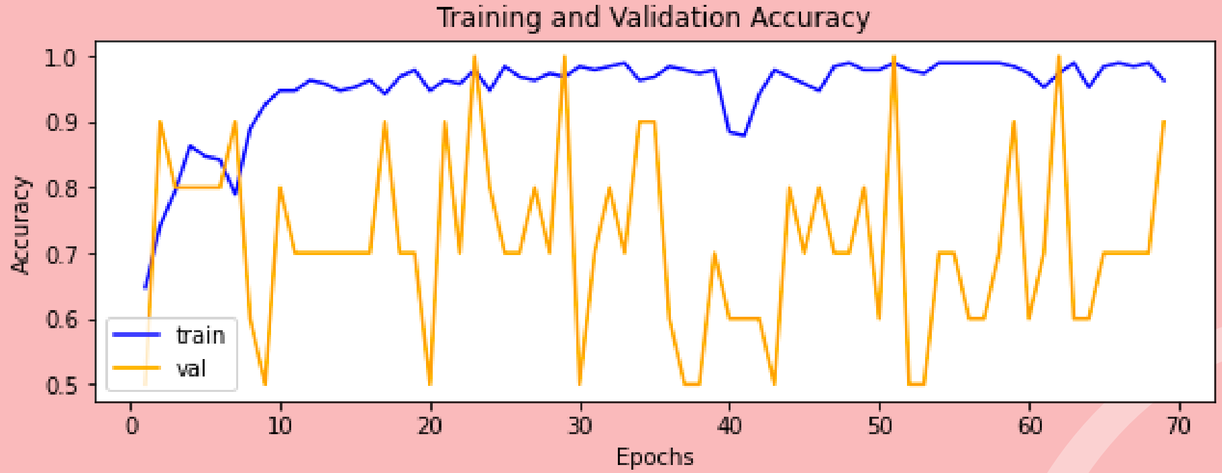


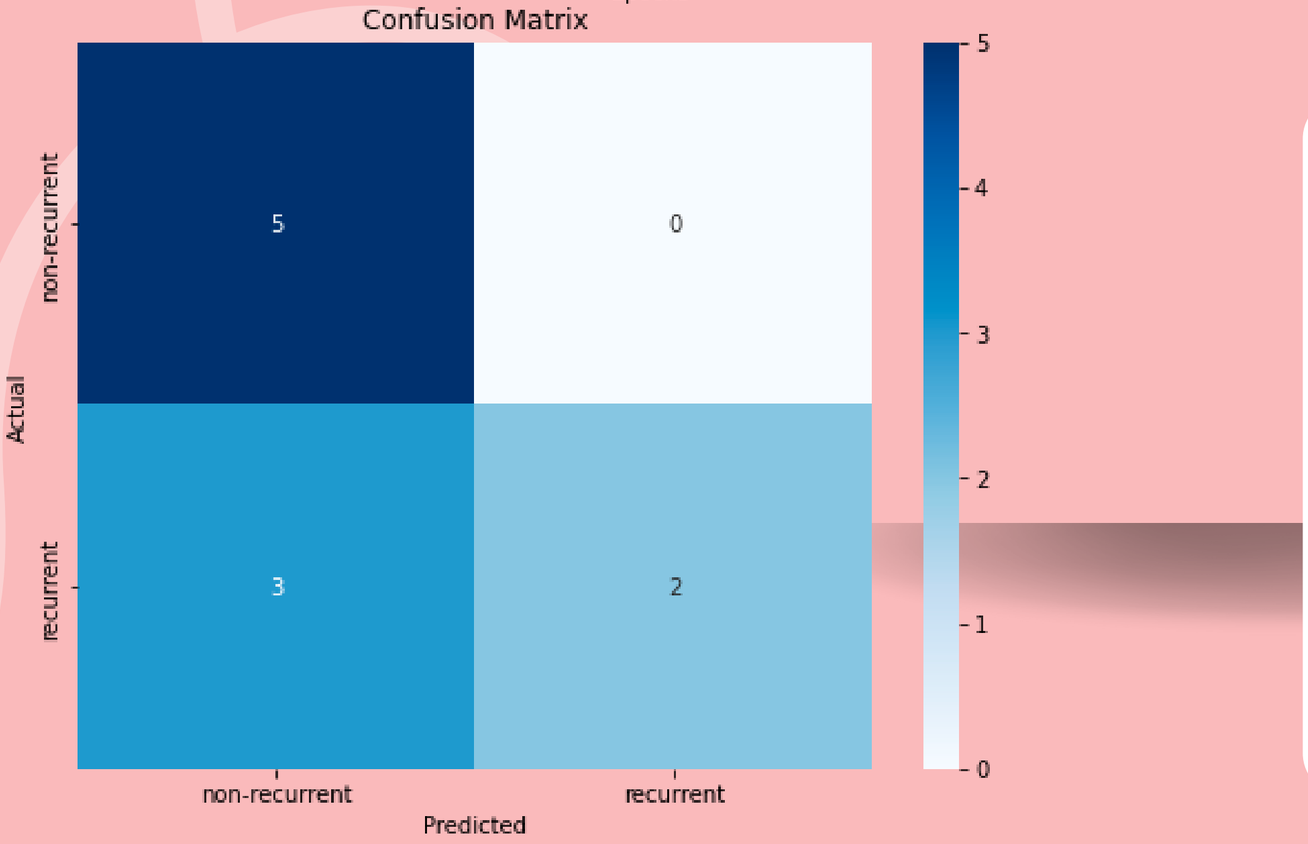
{0: 1, 1: 3.061224489795918}

So, since the Recurrent class has 3.06 times the lesser data, we increase the weights of Recurrent Class by 3.06 during training to address this data imbalance.

Epoch 1-69







Accuracy of the model: 0.727

Sensitivity: 0.4

Specificity: 1.0

ROC : 0.73

Balanced Accuracy: 0.73