University of Derby
Experientia docet

AI model to detect Covid-19 by analysing X-Ray images

Kartheek Raj Mulasa

Supervisor: Mohsen Farid

Submitted to the Department of Electronics, Computing And Mathematics in partial fulfilment of the requirements for the degree of Master of Science in Big Data Analytics

September 11, 2020

Declaration

All the sentences or passages cited from the work of other persons in this text is clearly noted through direct cross-referencing to the source, work and page(s). Any examples, which are not the author's work on this paper, were used with the originator's express permission and are expressly acknowledged. I recognize that failing to do so is plagiarism, which would be viewed as a cause for failing..

Name: Kartheek Raj Mulasa

Date: 09/11/2020

Acknowledgement

I wish to express my sincere gratitude for the mentorship and advice provided by Dr. Mohsen Farid for this research. His advice from the starting concept to the end of this research project has been valuable. My Research project would not have been accomplished without the assistance of my friends Shruthi and Nithip Chuantantigamol. On the other side, I would like to thank the data collection team from the University Of Montreal for providing the data without which this research would not be possible. I wish that my piece of research will help to those who are interested in the study of image detection using chest x-rays. Last but not the least, I will be ever grateful to all the research faculty at the University of Derby for all the assistance through out my research.

Abstract

Chest X-ray is the first imaging technique that plays a vital role in the diagnosis of COVID-19 disease. Due to the high COVID-19 cases, the patient diagnosis is a huge challenge with the existing practice. In this project AI model is developed and evaluated with an accuracy above 90% using the SVM(Support Vector Machines) methodology for image recognition and classification. However, due to the limited availability of COVID -19 annotated medical images, the classification model needs to be trained as the new data is collected and clinical trials need to be done to deploy in medical diagnosis.

Contents

List of Figures

Chapter 1

Literature Review

1.1 Introduction

The Corona Virus pandemic(COVID-19) is the new havoc wrecking globally. This outbreak has caused global social and economic disruptions across all the sections of society. Despite the dreadful impact, this pandemic also paved the way for new ideas, innovations that are helpful in driving the way of life a bit easier. One of the problems that we have evidence at this time is the lack of alternative and effective mechanisms - apart from the current medical tests - for the identification of COVID-19. For this reason, AI can be helpful in providing support for doctors while detecting the virus thus leading to the avoiding of congestion in hospitals, clinics, health centers, etc. Specifically, this dissertation deals with the development of an image classifier using the radiographs of the patients and identifying the one with the COVID-19 indications. So, this development can be regarded as a supplement for the medical professionals in the speedy analysis but not as the replacement of the existed mechanism.(Peters et al., 2020)

1.2 Origin and the Transmission

A bunch of pneumonia cases of unknown cause has been disclosed to the WHO(World Health Organisation) on 31st December 2019, in the Hubei province of Wuhan city in China. A previously unrevealed new virus was identified in January 2020, which is subsequently named as 2019 novel coronavirus. The samples received from different cases and the analysis of the virus' genetics showed that this was the cause of the outbreak. Then WHO named this novel coronavirus as Coronavirus disease 2019, in short as COVID-19 in February 2020. This virus is cited as SARS-CoV-2(Severe Acute Respiratory Syndrome-Corona Virus-2).

There are many speculations on the transmission of coronavirus, but the primary

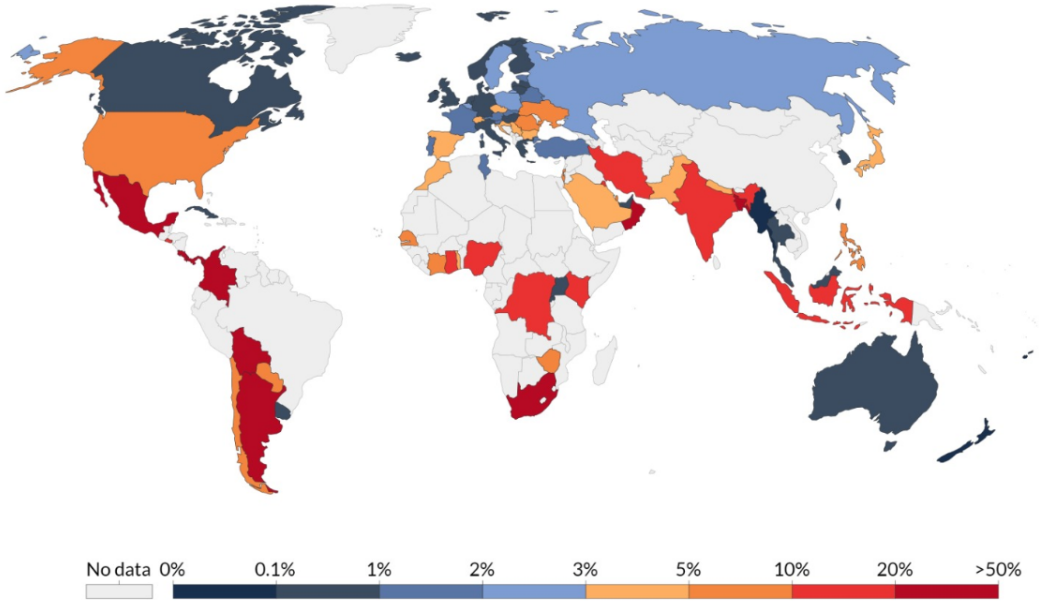


Figure 1.1: *Figure depicting the global distribution of Covid-19 cases.*

transmission is regarded as from human-to-human. This disease spreads through large respiratory droplets and the contact routes either direct or indirect contact with the infected people. There are many guess works around the transmission of this virus-like whether this virus spread is an airborne transmission or a fomite transmission and many more. But, there are no solid reports demonstrating the virus spread through these forms.(Coronavirus Disease (COVID-19), 2020)

The incubation period for this virus presently is between 2 to 14 days. After being in touch with a COVID-19 infected person, if a person remains well and healthy after the duration of 14 days, then they are not infected. (Guo et al., 2020)

1.3 COVID-19 Pandemic

Pandemic - epidemic that crosses the international boundaries and infects a huge amount of people. This describes an infectious disease where there is a notable and ongoing infection from person-to-person simultaneously across the globe.

If a virus is brand new and able to contaminate people quickly and easily in an efficient and sustainable manner across the large parts of geographies, then it usually resembles a pandemic. This coronavirus succeeded in all these characteristics to be considered a pandemic disease.

A pandemic cannot be just regarded as a health crisis but also a revolutionary socio-economic crisis. It has the immense ability to create catastrophic social, economic, and political effects that will leave deep and enduring scars. With no vaccine to avert it yet, curbing its spread is crucial. (Branswell and Joseph, 2020)

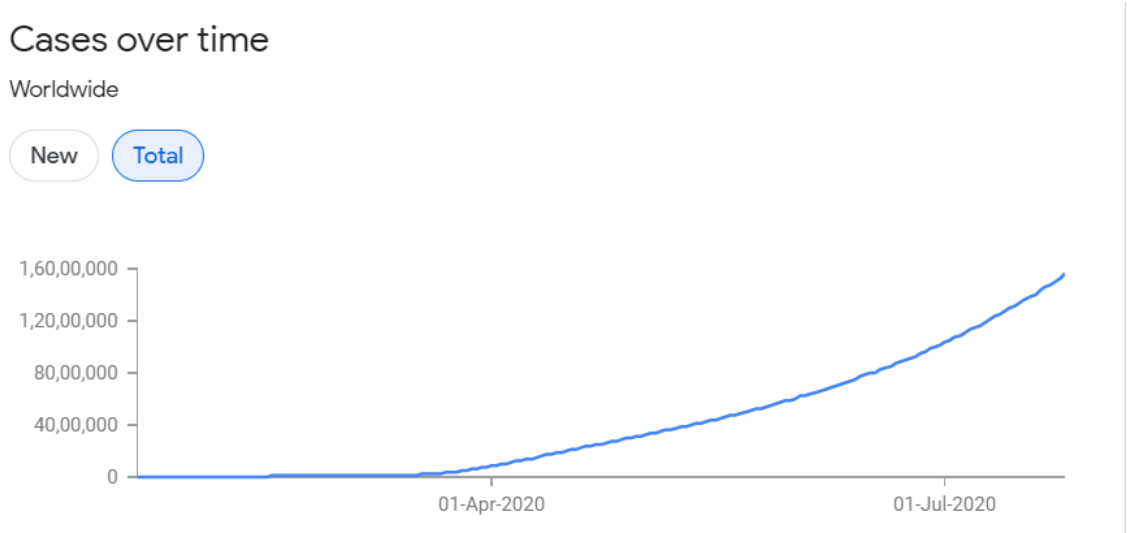


Figure 1.2: *Figure depicting the exponential growth of Covid cases all around the world.*

Whenever a pandemic like situation emerges, then there is a need to gauge the consequences and impacts on the human population. These impacts are assessed in various ways like global mortality rate, morbidity, economic load, and geopolitical implications.

This coronavirus has taken a significant toll on the lives of the people throughout the world. In the pursuit of averting the spread of the virus and for saving the lives of the people, many governments across the world have adhered to implying the stringent rules in their respective countries. Some of such rules include the international and statewide travel ban, imposing the strict lockdown for months. These rules have their own adverse impact on various aspects of a country.

The WHO estimated that an average to acute pandemic would cost nearly 500 billion dollars or 0.6 percent of the global economy. As travel is banned and when a curfew-like situation is prevailing across the globe, the lives of people will come to a standstill. People need to permit themselves to their houses which impacts the revenue generation. The revenue obtained from various sectors like aviation, cargo, retail, industries, banking, e-commerce and many more have come to halt, thus creating a void. This adverse economic burden would take some years to get back to the normalcy.

Because of the pervasive virus, multiple health system considerations including the availability of treatment or vaccination, and the level of resources (human, material and financial) available for distribution, whether or not the population is immunologically naive, resilience, population density has a vital role to play. The healthier the countries perform in the ways alluded to above, the healthier the prevention steps..

As the individual behavior is defined by the cultural beliefs, these impact the movement within a community. Literacy rate of the people, awareness on the general infection prevention measures, the ability of a governing body to respond to the outbreak also play an important role in managing adverse situations. If the health system is fragile or preoccupied with the daily routine cases, then in case of such pandemic emergency there will be a complete destabilization of the health sector.

An outbreak in one country not only affects its own economy but also impacts the world's economy and its contribution to the global Gross Domestic Product(GDP). This also leads to the country's geopolitical relationships with other countries like trade agreements, national travel permits, and many more.

1.4 Role of Chest x-ray in the covid-19 diagnosis

Radiology is the most elementary in this process. The main contribution of a radiologist is to assist and accelerate the exploration as much as possible, support in designing specific circuits and to prepare an accurate and quick report of the radiological findings which stipulate whether or not the findings are consistent with the COVID-19 coronavirus infection.(Cleverley, Piper and Jones, 2020)

1.4.1 Findings in a covid-19 chest x-ray

There are a set of findings or patterns that makes strong speculation about dealing with the coronavirus infection. Such symptoms are called as the ground glass patterned areas, where both lungs are affected, even in the starting stages. This infection occurs particularly in the lower lobes, and especially the posterior segments, with a fundamentally peripheral and subpleural distribution. 50 percent of patients have these symptoms for the first two days. As the days progress, these lesions become more diffused creating the combination of interlobular and intralobular lines presenting a pattern called a crazy-paving pattern. Another typical finding is the widening of the vessels called vascular dilatation. Traction bronchiectasis is one more pattern present in the areas of ground glass, strengthening the presence of coronavirus. (The Radiology Assistant : COVID-19 Imaging findings, 2020)

All the above findings are recognized in the chest x-rays. This helps in expediting the fast treatment of the patients thus contributing to saving the lives of people.

1.5 Infection Prevention and Control Measures

Interpretation of the diverse ways of the transmission of the virus is crucial for developing and implementing the control measures to break chains of virus carriers. As the availability of the scientific education is increased all over the world, the studies that investigate the transmission should be interpreted by bearing in mind the context and settings in which they took place, including the preventive measures of infection, the methods used in the investigation and the limitations and biases of the study designs.

It is quoted that "Prevention is always better than cure". When the society faces this kind of sudden breakdown of a disease with no vaccine, the effective measures to prevent the spread are more vital than curing the disease because till the stage of cure is reached, the damage would have already taken place. In this COVID-19 scenario, the precautions are best achieved by identifying the defendant cases as soon as possible, testing, and isolating infectious cases. In addition, it is imperative to identify all close contacts of infected people so that they can be put down into self-isolation thus breaking the chain of transmission. By quarantining the infected contacts, the potential chances of transmission will already be separated from others before they develop symptoms or before they get infected thus limiting the chance of further onward spread. The time between exposure to the virus and the onset of symptoms is termed as the incubation period of COVID-19. This period is of average 5-6 days but the virus can last as long as 14 days. Thus, quarantine should be in place for 14 days from the last subjection to a confirmed case. If it's impossible for contact to quarantine in a separate area, then 14 days of self-quarantine is essential. The people in self-quarantine may need support for using the physical distancing measures in curbing the expansion of the virus. (Transmission of SARS-CoV-2: implications for infection prevention precautions, 2020)

As there are chances of virus transmission through the contacts who do not show any symptoms, it is prudent to use the face masks mainly in public places where there are many chances of community transmission. The usage of face masks should be followed as part of a comprehensive package of preventive measures including physical distancing, frequent hand hygiene, environmental cleaning and disinfection, respiratory etiquette. Avoiding the gatherings both indoor and outdoor as much as possible ensuring the good environmental ventilation in a closed setting.

The health care facilities are most prone to the danger of this virus when caring for COVID-19 patients. So, WHO recommends taking droplet and contact precautions, airborne precautions while treating the patients. WHO also recommends transmission-

based precautions for other patients through an approach that is guided by risk assessment. All these recommendations are consistent with other national and international guidelines.(Abdi and Sun, 2020)

Moreover, in areas with COVID-19 community transmission, WHO advises that medical masks should be put on by all health care workers working in clinical areas for the entire shift. In the surroundings where aerosol-generating procedures are executed, they should wear an N95, FFP2, or FFP3 respirator. However, they also consider the use of medical masks as a viable option in case of shortages of respirators.

1.6 Covid - testing

COVID-19 testing requires examining samples to assess the current or past presence of SARS-CoV-2. The two main phases of this testing include detecting the presence of the virus or of antibodies produced in response to infection. Tests checking the viral presence are more important in diagnosing the individual cases which in turn allows the public health authorities to trace and restrict the outbreaks. Antibody tests show whether a person once had the presence of the virus. These antibody tests are not of much help in diagnosing current infections because there are fewer chances of developing antibodies in contacts after the infection. But these antibody tests are used to evaluate the disease prevalence leading to the estimation of the fatality rate of the infection.

Various protocols have been adopted by various countries and their respective jurisdictions. These protocols involved whom to test, how often to test, sample collection, and the use of test results, analysis protocols. All these protocols significantly impacted the statistics reported which includes case fatality rates, test numbers, and case demographics.

Medical laboratory scientists conducted the test analysis in automated, high-throughput, medical laboratories. Alternatively, this point of care testing can be done in physician's offices, workplaces, institutional settings, or transit hubs.(Diagnostic Testing for the Novel Coronavirus, 2020)

Chapter 2

Methodology

2.1 Objective

The objective of this project is to train an image classifier to determine if a person has COVID-19 or not, using radiographs. This involves the implementation of the image classification technique to classify the chest x-ray images of infected and normal patients. The methodology chapter will describe the data source, algorithms used to implement the image detection. These sets of algorithms will be reviewed for clarity and thus the algorithm with the highest accuracy is selected to perform this process. Thus, we will use Convolutional Neural Networks as an important technique for analysis to get the most accurate results.

2.2 Research design and approach

To carry out any kind of research, building the way of research analysis plays a crucial role. The research approaches consist of two main ways - Qualitative data research and quantitative data research. Qualitative research is multimethod focused, which incorporates an interpretive and naturalistic approach to its subject matter. This is concerned with understanding the behavior of humans from the perspective of the informant. Whereas, quantitative research is related to identifying facts about social phenomena. The choice of research will be very important as it helps the researcher in finding the correct research strategy and also helps in finding suitable solutions.

Research can be divided into two methods: deductive and induction. In the deductive research methods,research approach is designed by the researchers to test the hypothesis for theoretical proof. For the induction method, researchers gather data first and then develop the theory after data analysis.(Research Methods Knowledge Base, 2020)

Researchers should possess clear theories and the information since the beginning of their studies when planning to design a research approach. The choice of research assists and helps the researcher in finding the correct strategy and also in finding acceptable and satisfactory solutions.

The distinction between deductive and inductive analysis is the framework whose aim is related to qualitative study, starting with observation and developing a theory or hypothesis. This research architecture is often extremely versatile from many laboratory approaches, which include a wide variety of strategies and frameworks for study. However, this analysis does not have a standard framework, so research must be carefully and properly planned using this approach. Researchers must ensure that current analysis is performed without open prejudice, and that unreliable data sources are tracked.

The research analysis of this dissertation can be classified as the qualitative research approach and as a classification problem because identification of COVID-19 symptoms in a chest x-ray ultimately leads to the binary outcome of YES(if the chest x-ray resembles the COVID-19 symptoms) or NO(if the chest x-ray does not resemble that of COVID-19).

2.2.1 Purpose of study

The chest radiography (CXR) is the most frequently utilized method for identification and follow up of lung abnormalities especially during this COVID-19 crisis as this virus mainly affects the respiratory organs. In fact, the American College of Radiology (ACR) states that CT decontamination needed after scanning COVID-19 patients may cause disruptions in the availability of radiological service and suggests that portable chest radiography should be considered to minimize the risk of cross-infection.

The main purpose of this analysis would be to differentiate the radiographs of COVID-19 infected patients from others by applying deep learning methodologies. The measurement processed by Python or other analytical tools, e.g. R, which facilitates the computational aspects of the AI modeling using the chest x-ray dataset. Furthermore, we will analyze and classify the chest radiographs of infected and non-infected people by supervised learning and the deep learning techniques.

2.2.2 Research Method

We will do research using supervised learning-based models. One such methodology is the classification technique. Classification models are based on the texture, description, or resemblance of items or things. There are two kinds of approaches in image

classification - supervised and unsupervised. Pixels are the unit represented in a picture. Image classification groups the pixels in different classes. The categorization of images includes- image acquisition, image pre-processing, image segmentation. Some image classification methods are K- Nearest Neighbors(KNN), Support Vector Machine (SVM), Decision Trees (DT) and the deep learning methodologies like Convolutional Neural Networks (CNN)

2.3 Data Collection

This section describes the source path from where the data required to execute the project is collected. To detect the images exhibiting the symptoms of COVID-19, a model should be built using the required image classification techniques. For a model to classify the images, the first and foremost thing is to supply the required data. The data to be supplied here is a bunch of chest x-ray images of various symptoms. While there are a huge number of public datasets of more typical chest X-rays from the various universities like NIH [Wang 2017], Spain [Bustos 2019], Stanford [Irvin 2019], MIT [Johnson 2019] and Indiana University [Demner-Fushman 2016], there is no collection of COVID-19 chest X-rays that can be used for computational analysis.

The novel coronavirus (COVID-19) exhibits many unique features. These chest x-ray images are of various kinds. They will include x-ray images of coronavirus infected persons, x-ray images of contacts who do not have any virus or the x-ray images having symptoms of flu-like pneumonia. Infected patients with pneumonia cannot be classified as the COVID x-ray.so, this is one of the reasons to include pneumonia based x-ray images into the problem to enhance the capturing of various patterns and thus boosting the performance of the algorithm in classifying class COVID as COVID, class normal as normal and pneumonia as pneumonia. Our model will be trained on all these images and learns the various patterns needed to correctly classify the image.

2.3.1 Data Source

The data which we are referring to is collected from public sources as well as through indirect collection from hospitals and physicians. This data collection is approved by the University of Montreal's ETHIC Committee.(covid-chestxray-dataset, 2020)

2.4 Data analysis

This section explains the various ways of analysing the image data.

2.4.1 Image Data Analysis

An image - group of pixels

Images are stored as a mosaic of tiny squares by computers. These mosaics are sometimes referred to as the resolution of the images. The smoothness of an image depends on the size of these tiny squares. The bigger these squares are, the less is the resolution. If these square tiles are too big, it's then hard to make smooth edges and curves. The word pixel refers to a picture element i.e., these are the smallest unit of information which forms into an image. Each pixel is described using the combination of three colours. They are red, green and blue. These pixels are commonly arranged in a two-dimensional grid. These colour combinations result in different colours and shades of the pixel colour. If all the RBG values are at high intensity means they are 255. If all these RBG colour values are muted then it is shown as white colour. If the RBG colours have 0 as their value, then it shows black colour. Here each number is an 8-bit number, so the values range from 0–255. So, each value will have 256 different intensity making the total 16.8 million shades. There are 4 basic properties of an image. These are the height, width, shape and dimension of an image. The size of an RBG image should be calculated as height*weight*layers(i.e., 3) of an image.(Image Data Analysis, 2020)

Splitting Layers

Each pixel of the image is indicated by three integers. Splitting the layers is the way of extracting separate colour components by pulling out the correct slice of the image array.

GreyScale

Black and white images are stored in 2-Dimensional arrays. There are two kinds of black and white images:

Binary: Pixel will be either one of these, black or white:0 or 255

Greyscale: Ranges of shades of grey:0 255

Now, The process of converting an image from full color to shades of grey is called grey scaling. Many image processing tools use this grey scaling. for example OpenCV. The main use of grey scale is that it simplifies the image by reducing the noise and enhancing the processing time. Many functions use this grey scale for these two reasons. In python, we have few ways of converting an image into grey scale. But, if we want to implement directly then matplotlib provides the way to do this. We need to take

the weighted mean of the RGB value of the original image using this formula.

$$Y' = 0.299R + 0.587G + 0.114B \tag{2.1}$$

Masking

Masking an image is one of the image processing techniques which is used to remove the background having the fuzzy edges, transparent or hair portions.

2.5 Classification Algorithms

Classification algorithms play a serious role in image processing techniques. The features are classified from an image into different classes based on various characteristics. The classification techniques are widely used in multiple applications across different fields. Amongst these classifications algorithms most optimal ones in various domains are also analyzed.

The classification techniques are used for predicting data accuracy at different levels. This is one of the techniques of data mining used to classify particular target groups. The primary aim is to predict the nature of items or dataset according to data levels. This technique mainly depends on the application and nature of the data set. Classification comes under the category of supervised learning where there are a set of input variables and also a discrete target variable. The target variable can be either binary(which contains only 0 and 1 where 0 and 1 are two classes) or multi-class(where target variable can be any number of discrete values).

There are different types of classification algorithms that are categorized based on the learners. There are two types of learners in classification, they are lazy learners and eager learners. (Asiri, 2020)

1. Lazy learners: These learners are termed as lazy because they simply store the training data and will wait for the test data to appear. When the testing data comes into picture then the classification is conducted based on the most related data which is stored in the already trained data. With this feature, these lazy learners have short training periods and long predicting periods.
Ex. k-nearest neighbor(KNN), Case-based reasoning

2. Eager learners: In these learners, the classification model is already built based on the given training data even before the classification data for the model is provided. This feature enables us to generate a single hypothesis covering the entire

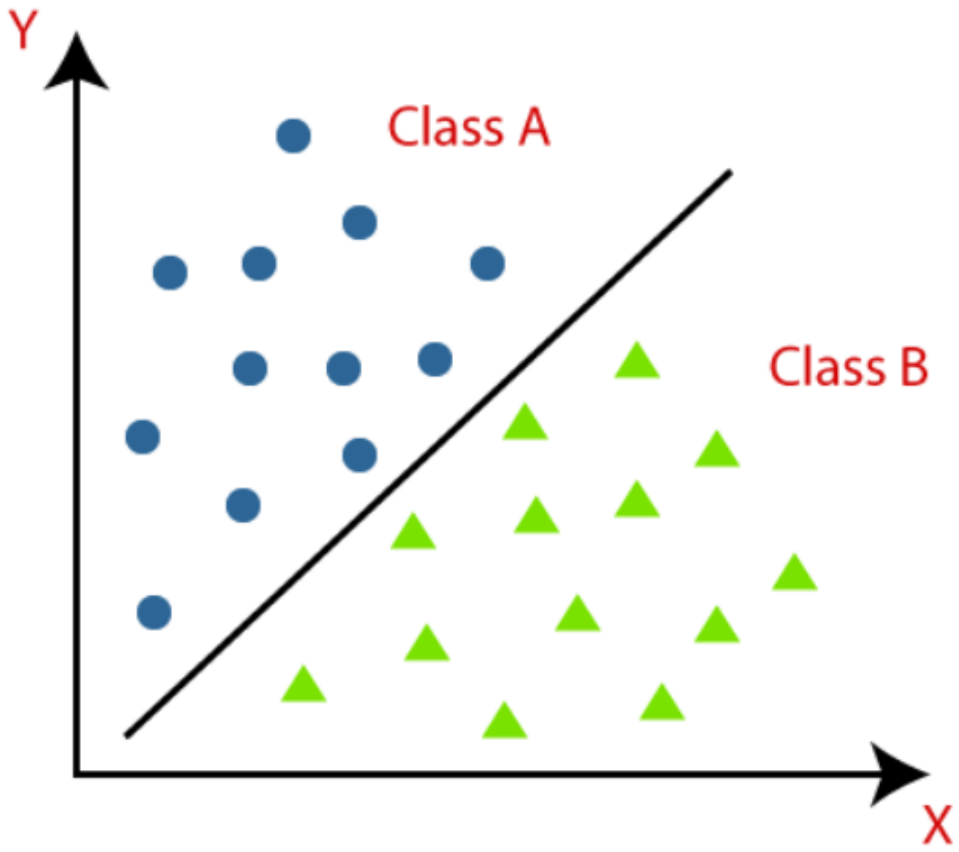


Figure 2.1: *Figure showing the separation of different classes using classification .*

instance space. Because of the model construction, eager learners take significant time for training and very less time for a prediction.
Ex. Decision Tree(DT), Naive Bayes, Artificial Neural Networks(ANN)

The entire classification process is carried out in two phases: training phase, where a classification model is built using the training dataset, and the testing phase, where the model built is assessed by using the test data set. In the training stage, the data is cleansed in all the possible ways to make it ready to the model building. All the required transformations are applied to the data set. Once the data is ready, then a classification algorithm is applied to the data set. In this duration, the model learns all the patterns from the training data set and gets ready to be validated. A new data set is used in the testing process to verify the model, and the values for the sample class are not revealed. In this step only the outcomes will be shown after the forecast, so that the algorithm can reveal the exact category that has just been identified. There are many performance measures like accuracy, precision, recall that can be captured

and assessed in model performance.

2.5.1 Logistic Regression

This concept comes under the category of supervised learning referring to a regression model used for classification. The main theme is to classify the objects into respective classes. Every supervised machine learning models have a set of independent variables or input variables and a target variable or dependant variable. This method is majorly used for binary classification i.e., when the outcome or target is categorized into just two classes either 1 or 0 like pass or fail, win or lose.Here, the dependent variable is categorical:

$$y = 0, 1$$

Logistic regression internally uses the sigmoid function. This function reduces the range of the outcome variable to [0,1] both inclusive. Whatever may be the values, the target variable range will be fitted between 01. This is called the sigmoid probability (σ).

$$If \sigma(\theta Tx) > 0.5,\ set\ y = 1, else\ set\ y = 0\ where\ 0.5\ here\ \ is\ the\ threshold\ value$$

Unlike Linear regression where we try to fit the equation by finding the optimal weights, here we try to find the global minima. This is achieved by implementing a cost function internally called Log Loss function.This cost function is also termed as Binary Cross-Entropy function.This cost function actually measures the classifier's accuracy. It tells the probability of output for each class rather than just giving the class.(Molnar, 2020)

$$H_p(q) = -1/N \sum_{i=1}^{N} y_i.log(p(y_i) + (1 - y_i).log(1 - p(y_i)) \tag{2.2}$$

Logistic regression is internally built to support the prediction of only binary outcomes i.e., 0 and 1. There is an extension to this where the target variable is multi-class or discrete variables like 0,1,2 and so on.

2.5.2 Decision Trees

Decision Trees(DT) are the non-linear algorithms falling under the category of supervised learning. These are heavily algorithmic driven, unlike logistic regression which

is heavily statistical driven. These are tree-like structures to illustrate every possible outcome of a decision. The main aim of the decision tree is to split the data based on the valid points and classify the data until we get pure data points.

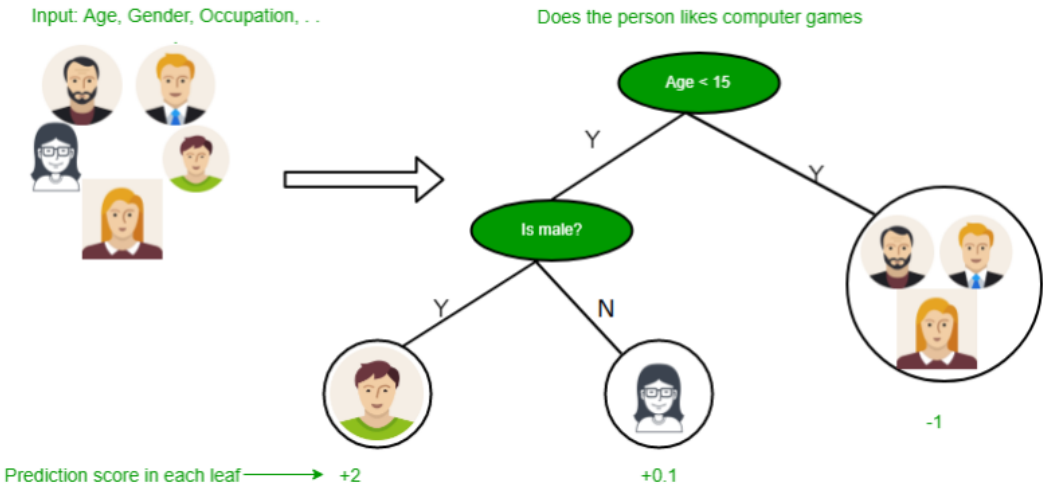


Figure 2.2: *Model of Decision Tree.*

These algorithm uses tree-like representation in which the leaf node represents the label of the class(that is the target label) and the internal nodes represent the attributes(that is the input variables of the data). These are robust in nature. The DT works for multi-class classification problems as well. Each split is an axis parallel split. In order to make a split, all the available features are selected and the calculation entropy, information gain are computed. The feature which results in higher information gain is selected for the split. The node purity is defined by the proper separation or classification of various class labels. There are functions to compute the impurity. One such function is Entropy, which is the amount of disorderliness/disturbances in the system. Entropy is the measure of impurity. The other such functions include the Gini index. The impurity of the nodes are computed before the split. After the split, again the impurity of the leaf node is computed.(Sharma, 2020)

$$Entrophy = -\sum_{i=1}^{n} p_i(log_2 p_i) \tag{2.3}$$

Information gain = entropy before split - entropy after split.

Entropy before split should be greater than entropy after the split. Overall, the information gain must be maximum. The split is done only when the information gain is maximum.

Gini index is also a way to compute the information gain.

$$Gini(node) = 1 - \sum_{k=1}^{K} p_k^2 \tag{2.4}$$

Decision Tree is also called a Greedy algorithm. At each step, it makes the optimal decision without postponing to the next iteration. Thus, it tries to achieve the best split at each level.

2.5.3 Random Forest

Ensemble Method: This is the powerful technique of combining the multiple weak learners to form a single strong learner. All the base models are combined to produce a strong single optimal model. The main goal of these ensemble techniques is to minimize the generalization error of the prediction. If the base models used are distinct and independent then the prediction error is reduced by using this technique. Despite having multiple base models in it, this acts as a single model. The main agenda of these techniques is to enhance the stability and the performance of machine learning models. This ensemble model can outperform all single algorithm models. Random Forest is one such product of ensemble technique.(Breiman and Cutler, 2020)

Random Forest is used for both regression and classification problems. The Random Forest Classifier is a combination of decision trees from a vaguely selected subset of the training set. The final class of the test object is decided by aggregating the votes from different decision trees. The base models can be any of the models. But decision trees are used most of the time.

Random forest prediction can be given as $f(x)$ = Highest vote of all predicted classes over other trees.

There are many advantages of using this model.

1. This is a highly accurate model.
2. This model can efficiently run on the large volumes of data.
3. It can handle thousands of features in the data.
4. This produces the unbiased estimation of the generalization error.
5. This model has the capability to manage huge data noises like having missing values in the data. This model effectively manages to estimate the missing values.

As the random forest has a decision tree as the underlying base model, the split implementation, entropy, and information gain calculation work as per the decision tree. But there are some additional tweaks to the random forest. This algorithm combines the input data with bootstrapping (input with replacement) to produce sev-

eral data sets (weak learners). A decision tree is generated for each slow learner. In this case, if there are M input variables, then random selection of 'm' input variables (where m <<M)is made. These 'm' variables are used to build the optimal split at each node when the decision tree for each subset is being built. The main difference between the decision tree and the random forest is, we can select only a particular set of features for each node split in the random forest whereas in decision tree classifier only single feature resulting in the higher information gain is selected for the node split.

Measurement of error: In Random Forest, the error rate is based on two things.
1. The association between any two trees within the forest. Increasing the connection increases the probability of error.
2. The strength of a single tree within the forest. The strong classifier would be a tree which results in less error rate. Growing a tree's intensity diminishes the forest output.
If there are m sub-features out of M features where (m<<M) then
If m decreases, then both correlation and strength decreases.
If m increases, then both correlation and strength increases.

With these models, the disadvantages are the over fitting and bias towards the categorical data. Over fitting is nothing but the model will be able to learn each and every pattern of the training data but fails to generalize over the test data thus leading to higher accuracy during the training phase and lower accuracy during the testing phase. If the model contains the categorical data with more number of levels, then the model is biased towards the categorical variable with more levels.

2.5.4 Support vector machines

Support Vector Machine (SVM) algorithm falls under the category of supervised learning which is heavily based on logistic regression. This can be used in both classification and regression problems. In the SVM algorithm, each data item is plotted as a point in the n-dimensional space (where n= number of features) where the value of each feature is nothing but the value of a particular coordinate. Then, classification is performed by finding the hyper-plane that can differentiate the classes very well. Many hyperplanes can be chosen to distinguish the classes of data points. But the objective of the SVM is to find a plane that gives the maximum margin of separability between the classes. This maximum margin gives some boosting so that the classification of the future data points can be achieved with more confidence.

Hyperplanes are the decision boundaries that categorize the data points. Data points present on either side of the hyperplane belong to different classes. The number of features decides the dimension of the hyperplane. For example, if there are 2 features then there will be 2-dimensional hyperplane. If the number of input features is n, then the hyperplane is n-dimensional. As the features go on increasing, it becomes difficult

to imagine the dimensions.(Gandhi, 2020)

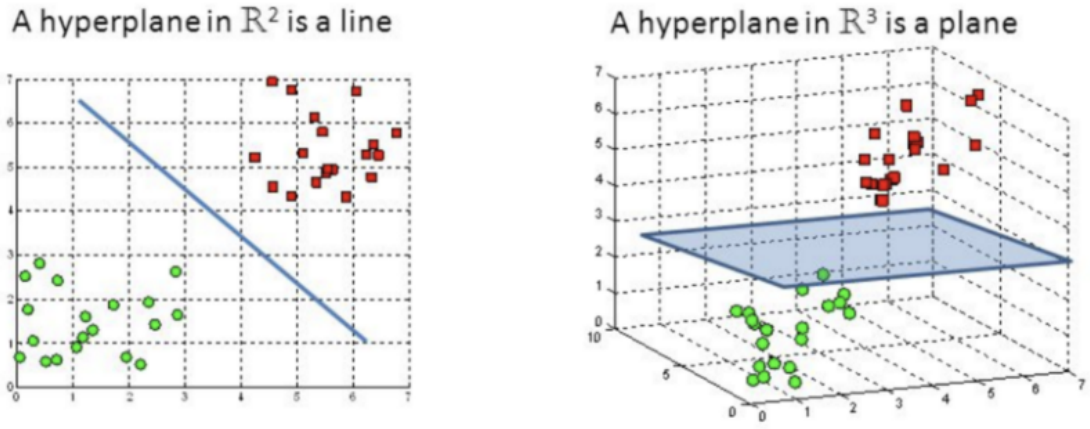


Figure 2.3: *Figure showing Support Vectors and Hyper Planes*

The data points that are closer to the hyperplane are called Support vectors and the position and orientation of the hyperplane are influenced by these support vectors. These support vectors help in maximizing the margin of the classifier. The position and dimension of the hyperplane will be affected by deleting the support vectors. So, SVM is mainly built by these points.

The output of the linear function is squashed to the range of [0,1] in logistic regression using the sigmoid function. If this squeezed value is greater than 0.5 (threshold value) then it is assigned to label 1, else it is assigned to label 0. But in SVM, if the output of the linear function is greater than 1, then it is identified with one class and if the output is -1, it is identified with another class.so, the threshold values are changed to the range of [-1,1] in SVM. This range of values([-1,1]) acts as margin.

In the SVM algorithm, the main agenda is to maximize the margin between the data points and the hyperplane for better classification. Hinge Loss is the loss function which helps SVM in reaching the goal.

$$C\sum_{i=1}^{m}\left[y^{(i)}\,cost_1(\theta^T x^{(i)}) + (1-y^{(i)})\,cost_0(\theta^T x^{(i)})\right] + \frac{1}{2}\sum_{j=1}^{n}\theta_j^2 5 \tag{2.5}$$

If prediction and the actual value are of same class then the cost is 0. The loss value is calculated only when the prediction and the actual values differ. In order to maximise the balance between the margin maximization and loss, a regularization

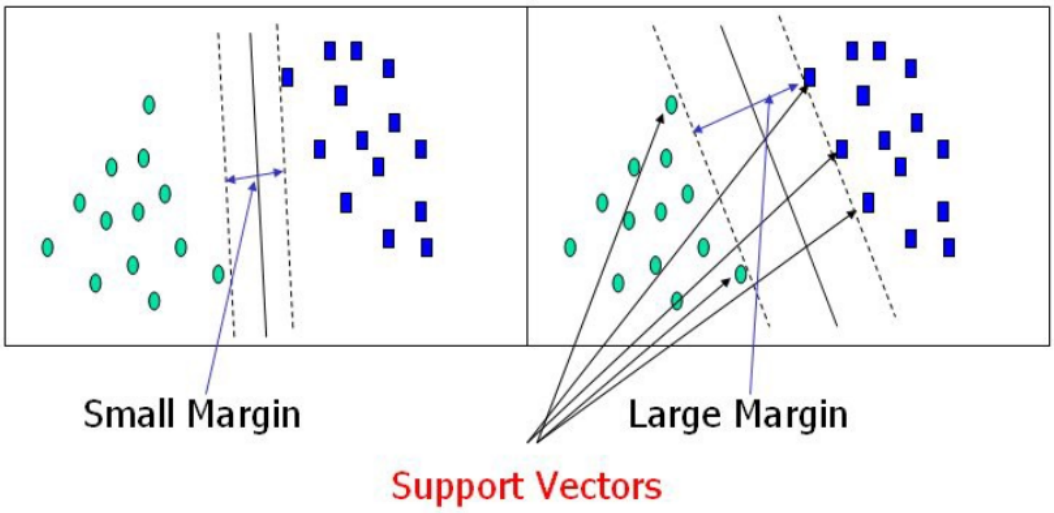


Figure 2.4: *Figure showing Support Vectors and their Margins*

parameter is included.The cost function after adding the regularization parameter is given below.

$$\min_{w} \lambda\|w\|^2 + \sum_{i=1}^{n}(1 - y_i\langle x_i, w\rangle) \tag{2.6}$$

The SVM is the most elegant and powerful algorithm in the machine learning world.

2.5.5 Convolutional Neural Networks

The Convolutional Neural Networks falls under the category of image classification, video analytics in the deep learning world. As the name suggests, Neural networks are a machine learning technique that is carved from the structure of the brain. It consists of a network of learning units called neurons. Whenever the brain comes across any picture or image or scenery, these neurons convert these inputs into the respective output signals. Thus forming the base of automatic recognition.(Convolutional Neural Networks, 2020)

For example, the process of finding whether a picture contains a dog results in the involvement of an activation function.If the neurons have seen the image of a dog before and the input picture also resembles the same, then the respective label of dog will be activated. Hence, the more the neurons are exposed to the labeled images, the better it learns to recognize and distinguish the other unlabelled images. So, these Convolutional Neural Networks are made up of neurons with masterable weights and

biases. Every neuron in the network receives various inputs. Each neuron computes a weighted sum over them, and passes it through an activation function and gives an output.

CNN's comprises of four-layered concepts. They are:
1. Convolution of an image - being translational invariant is one good property of convolution. This is nothing but every filter of convolution represents a feature of interest (e.g pixels in letters) and thus the Convolutional Neural Network algorithm learns which features comprise an alphabet(i.e. resulting reference) There are four steps for convolution:

- Lining up of the image and the feature
- Each image pixel is multiplied by respective feature pixel
- finding the sum by adding the values.
- Now the sum is divided by the total number of pixels in the feature.

2. ReLu(Rectified Linear Unit) - This is one of the available activation functions. The activation function is a mathematical equation that forms the neural network output. Each neuron in the network is attached to this activation function and also determines when it should be activated(or not). This activation or not is based on the relevancy of each neuron's input to the prediction of the model. Activation functions also provide the support in normalizing the output range of each neuron as [0,1] or between [-1,1]. If the input is above a certain quantity, then only this ReLu transform function activates the node. If the input is below zero, then there is no activation and the output is zero, but when the input is above a particular threshold, then it means there is a linear relationship with the dependent variable.(Rao, 2020)

3. Pooling - This layer is used to shrink the image to a smaller size. After passing the input through the activation layer, pooling is applied to the input. This is implemented in four steps:

- Pick a window size (usually 2 or 3)
- Pick a stride (usually 2)
- Walk your window across your filtered images
- Consider the maximum value from each window

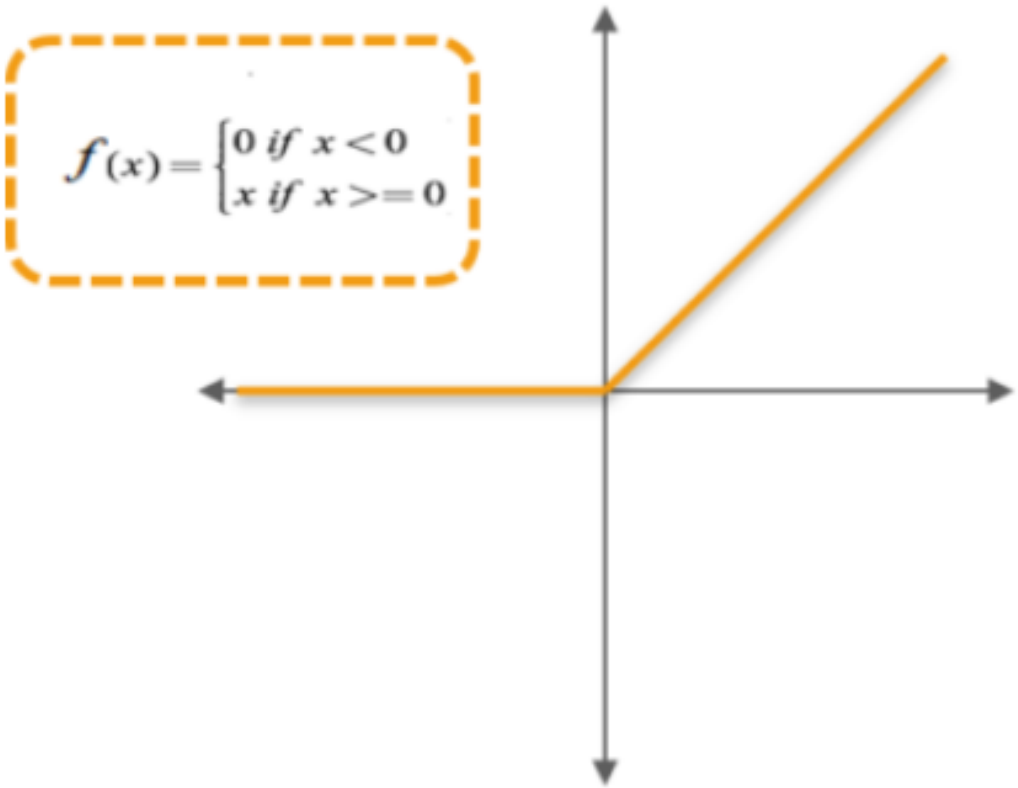


Figure 2.5: *Figure showing the ReLu activation function of Neural Networks*

4. Full Connectedness (Fully Connected Layer) - This layer is mainly involved in prediction and also checking the accuracy of predictions.

.

This CNN concept is the most approved deep learning technique for present visual recognition tasks. Similar to other deep learning techniques, Convolutional Neural Networks depend very much on the quality of training data and the size of the data.

Provided with a fully prepared dataset, these networks are proficient enough of making outstanding visual recognition tasks compared to human beings. Despite these, there is still room for improvement in these networks. These are still not vigorous enough to visual artifacts like noise and glare where human beings can cope easily at this part of the junction.

The Convolutional Neural Networks theory is still under development and researchers are constantly working to improvise the networks with properties like active attention and online memory, thus allowing Convolutional Neural Networks to assess

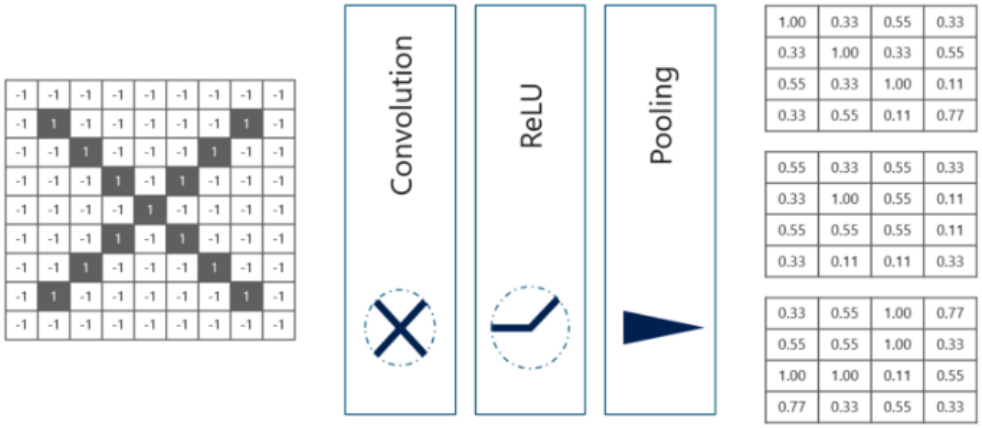


Figure 2.6: *Figure showing the image transformed through the layers*

the new items which are largely different from what they were trained on initially.

2.5.6 Transfer Learning

Transfer Learning is one of the sophisticated machine learning techniques available. This technique is the enhancement of learning a new task by transferring knowledge from a related task that has been already learned or trained. This technique strives on storing knowledge, which is gained by solving one kind of problem and applying this knowledge to a different but related problem. For example, the patterns or information captured while training the cars can be used to apply to recognize the trucks (Transfer Learning, 2020).

Transfer learning has set up the standard of showing reduced training time and also developed mainly for the restricted and less availability of the data for achieving a custom task. It considers CNN which has been pre-trained and then the last fully-connected layer is replaced with the customized fully-connected layer. Thus the original CNN is treated as the feature extractor for the new dataset. After replacing the last fully connected layer, the new dataset is trained using the classifier (Razavian et al,2014).

There are some major steps involved in implementing this technique. Each of these steps is explained below.

Pre-Training

Pre-training involves training the network(Convolutional Neural Network) on a

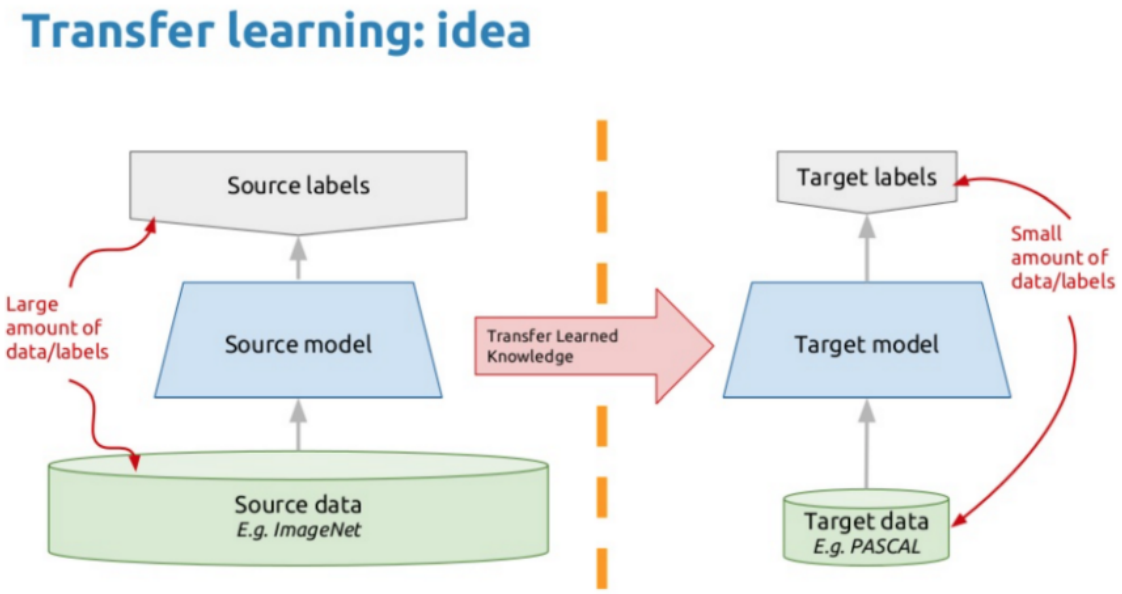


Figure 2.7: *Figure illustrating the Transfer Learning Idea*

huge dataset. For example, usage of imagenet while training the network. All the parameters of the neural network are trained and the model captures the data patterns and thus it learns. This process is very time consuming and ingests more power as this involves the training of large dataset thus needing more computational power.

Fine Tuning

The new data set is passed to the pre-trained CNN for fine-tuning. Let us suppose the new dataset is very much similar to the original dataset which is used for pre-training. Then the same weights of the pre-trained classifier can be used for extracting the features from the new dataset.

There might be chances of having very little new data. In such cases, it is always wise to train only the last layers of the network to circumvent overfitting issues. The final layers of the pre-trained network are removed and the new layers are added. Only these new layers are retrained.

The complete network is retrained using the initial weights of the pre-trained model if the new dataset consists of a large amount of data.

If the original dataset varies with the new dataset then the different fine-tuning methodology should be implemented. In this scenario, the more generic features are present in the initial layers of a convolutional neural network whereas the final layers

contain specific details of the classes present in the original dataset.SO, here the initial layers help us in extracting the features of the new data. It is better to fix the earlier layers and to retrain the remaining layers when there is a small amount of data. The entire network should be retrained with the initial weights of the pre-trained network (Pranoy Radhakrishnan, 2020).

2.5.7 EfficientNet

Numerous convolutional neural networks have been introduced in the field of machine learning with their own pros and con's. Their diversified architectures resulted in immense performances in various applications of computer vision. One of the variants is EfficientNet. The main theme of this technique is that better performance will be achieved by cautiously handling the resolution, width, and depth of the network.(Falah Gatea, 2020)

EfficientNet Architecture

According to this point, a new scaling technique has been introduced which scales all the dimensions of width, depth, and resolution of the network. A new baseline network is built using the quest for neural architecture to obtain a family of profound learning models called EfficientNets. This EfficientNets achieve enhanced accuracy and efficiency compared to the earlier CNN's. Efficient- Net models are implemented as the transfer learning frameworks in the image classification. This technique involves the architecture of Mobile Inverted Bottleneck Net- work(MBConv).(EfficientNet: Model Scaling for Convolutional Neural Networks, 2020)

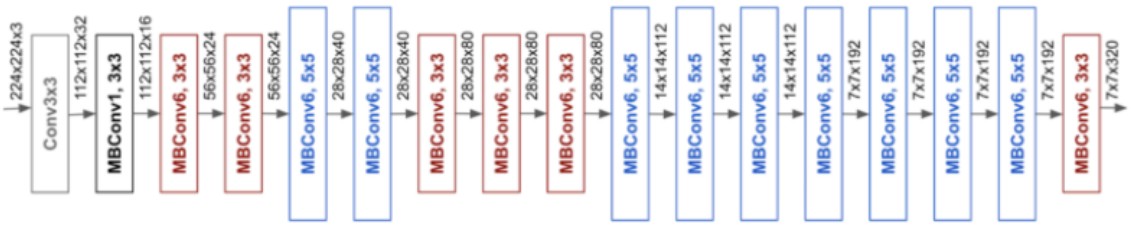


Figure 2.8: *Figure illustrating the architecture of EfficientNets*

There are three main types of scaling techniques in this efficientNets. These are explained in detail below.

Depth Scaling:

Scaling a network by depth is the most routine scaling method. Depth can be scaled up and down by adding and removing the layers respectively. Scaling by depth is chosen because the rich and complex information, features are captured well by the deeper networks. This allows the model to generalize well on new tasks. But in theory, the deeper networks result in higher performance but in practice, this does not work. The main problem that resulted due to this scaling technique is the Vanishing gradients problem.(EfficientNet: Model Scaling for Convolutional Neural Networks, 2020)

Width Scaling:

This scaling by width is mainly used when there is a scope for smaller models. The ideology behind this is that there is a chance of capturing fine-grained features with wider networks. But there is a problem associated with these networks too. Less deep and wider networks represent the shallow models where the networks observe saturation of accuracies quickly. SO, there is a problem with both deeper networks and wider networks.

Resolution:

With high resolution, there is an intuition of having more finely grained features and with this, there is a possibility of an increase in accuracy. But in practice, by increasing the resolution of an image, there is a problem of diminishing accuracy. For example, there will be no significant improvement when the resolution of an image is increased from 512x512 to 560x560.

Combined Scaling:

Combined scaling results in combining the scaling of all the width, depth, and resolution of the networks. There is an intuition that as the resolution increases, the depth and width of the network should be increased as well. Because with an increase in the resolution of an image, the increased depth indicates that large receptive fields tend to capture a larger number of pixels in an image. And with wider networks, fine features are captured.(EfficientNet: Model Scaling for Convolutional Neural Networks, 2020)

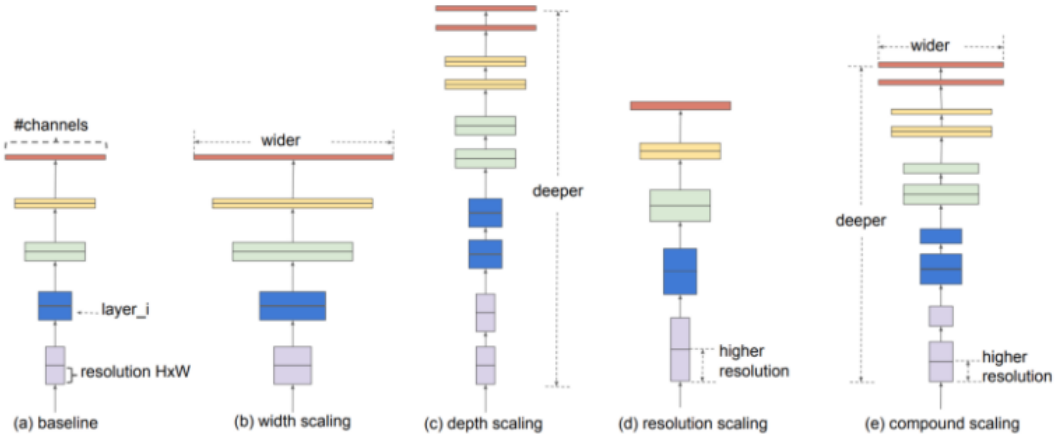


Figure 2.9: *Figure illustrating the different scaling approaches in EfficientNets*

Chapter 3

Experimentation

This chapter deals with the analysis of the data i.e., chest x-ray images of various people which includes both infected people and non-infected people with COVID-19. All the steps involved in building a model are explained here. This study will analyze, train and build an image classifier to detect whether a person has infected with COVID-19 or not using their chest x-rays.

3.1 Data Characteristics

The data is separated into two groups consisting of COVID and no COVID types. The COVID group consists of only COVID infected chest x-ray images while no COVID group consists of images of chest x-rays with or without other viruses as Pneumocystis, etc. The data contains the labels that tell us the disease detected like(COVID-19, Pneumocystis, etc.), the image type (X-ray or CT) and the field of vision that specifies the type of X-ray shot (PA, L, Axial, AP, etc.). The model which is built using the training data set is validated using a test data set which is unseen by the model.

3.2 Image Data Analysis

The data required for building this model is first downloaded into the local data folders. The data here are the images(chest x-ray's). The images are split into an 80:20 ratio where 80 percent of data is used to train the model to capture the required data patterns and the remaining 20 percent images will be used for testing the trained model. The validation data set is kept separately to validate the trained model. The performance measures obtained used this validation set marks the actual performance of our model. Always place the train data set, test data set and the validation data set separately. We should take care of not including the test and validation data set

into the model training.(Hamdaoui, 2020)

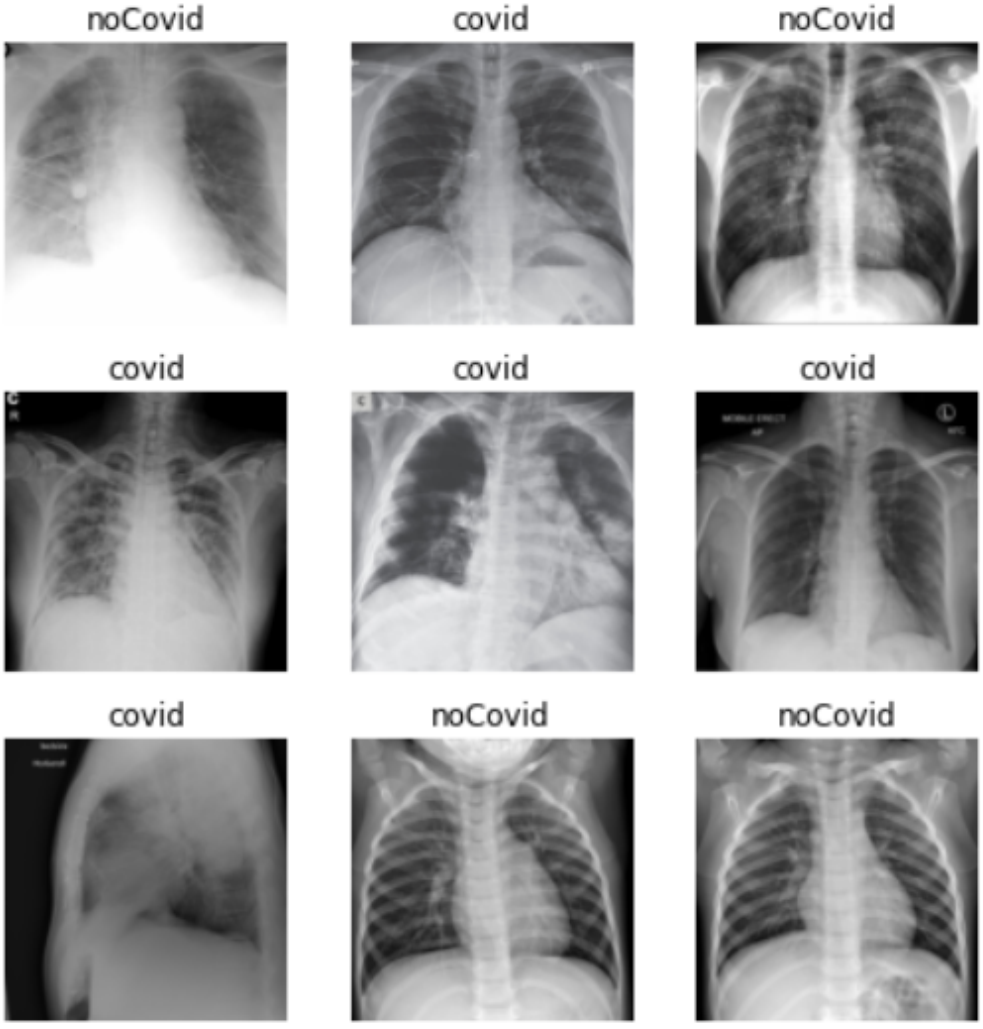


Figure 3.1: *Sample figure showing the covid and no covid chest x-rays used for training the model.*

In this process, firstly, the set of images are loaded keeping the grayscale = True as these images are black and white. Then these sets of images are normalized within the range of [0,1] by dividing the images by 255. Then, these are converted into a NumPy array. This three-step process is repeated for three sets of images. They are COVID, pneumonia and no-COVID images.

```
img_size = (224,224)
img_list2 = glob.glob('C:\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography Database\\whole data\\Tr
ain\\NORMAL\\*')

list_normal = []
for img in img_list2[:]:
    temp_img = load_img(img,grayscale=True,target_size=(img_size))
    temp_img_array = img_to_array(temp_img) /255
    list_normal.append(temp_img_array)
list_normal = np.array(list_normal)
list_normal2 = list_normal.reshape(-1,50176)
df_normal=pd.DataFrame(list_normal2)
df_normal['label'] = np.full(df_normal.shape[0],0)
```

Figure 3.2: *Code base to convert the set of images to numpy arrays.*

3.2.1 Image Data Augmentation

The performance improvement of any data science model(machine learning models, deep learning neural networks) upgrades with the availability of the data. Data augmentation is a principal part of training a machine learning model, mainly when there are limited training images. This is a technique to create new training data from existing data. Domain-specific methods are applied to the existing training data to create new and distinct data. Many augmentation algorithms are defined for the image augmentation process.

Image data augmentation is the most popular data augmentation technique which includes creating the transformed versions of images of the training data where the new versioned images belong to the same class as the original images. Image transformations include various range of operations like field manipulation like flips, shifts, zooms, etc.

The main intention of this augmentation technique is to expand the training data set with the brand new training examples. This allows our model to see, learn the various variations of the data images. For example, a horizontal image of a cat or a dog is meaningful as the picture might have been taken from either the left side or right side. But the vertical flip of the image is pointless because in real-time we do not see an image of an upside-down cat or dog. The drive away from point here is that even though the images are rotated, cropped, and widened, the meaning of the images does not change.

So, the choice of augmentation depends mainly on the domain. These techniques must be chosen wisely and should be in the context of the training data set. These augmentation techniques can be executed in isolation to check whether the performance of the model is enhanced or not.

The deep learning methodologies like Convolutional Neural Networks tend to learn and capture the unique patterns that are invariant to their location in an image. This augmentation technique should be applied only on the training data set but not on

the test or validation data sets. This technique is way different from other image data preparation methods like image resizing and pixel scaling as these are performed across all the data sets which interact with the model.

3.2.2 Principal Component Analysis

Principal Component Analysis (PCA) is the most widely used dimensionality reduction technique in Machine Learning applications. The main goal of PCA is to boil down the information from a huge set of input features into a smaller set of variables by implementing various kinds of transformations onto them. The transformations result in the transformation of linearly correlated variables into uncorrelated variables. Correlation specifies the redundancy among the input variables. If both the variables are highly correlated, then we are not gaining any extra information by keeping these two variables as one variable can be expressed linearly using the other variable. So, there is no use of holding both the variables. Instead, we can compress the information by using PCA, where PCA transforms the variance of the second variable onto the first variable. This transformation is done by rotating, translating the original axes, and projecting the data onto the new axes. This projection is determined by the eigenvalues and eigenvectors. So, the initial transformed features which are known as principal components contain a rich source of information while the last components contain mostly noise with very little information. Thus, the ability to transfer the information into the first few components with minimal loss of information is the specialty of PCA. (Kumar R, 2020)

An image, technically, is the combination of pixels where brightness shows the reflectance of surface features within that pixel. Normally, the reflectance value ranges from 0 to 255 for an 8-bit integer image. So black images are the pixels with zero reflectance, white images are the pixels with value 255 and pixels with value in-between 0 and 255 are in a gray tone.

In this process, PCA is applied for both training and test data sets. The target variables will not be modified i.e., PCA or any other kind of transformations will not be applied to the target features.

X_train_pca = pca.transform(x_train)

X_test_pca = pca.transform(x_test)

In this process, we have considered 40 components and the variance explained by these components can be seen below.

A bar graph is plotted against all these principal components showing the variance

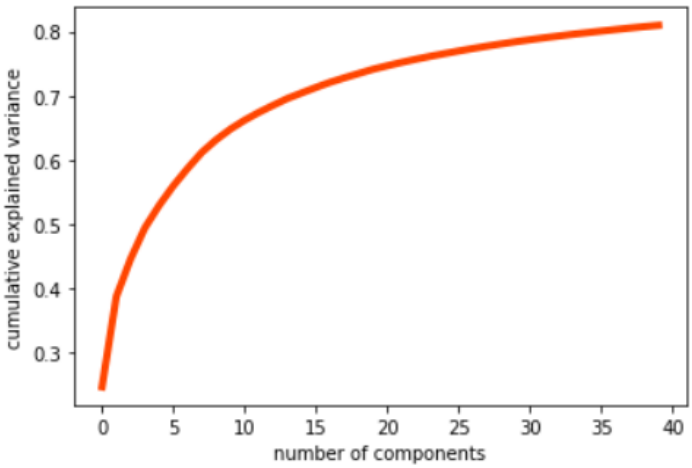


Figure 3.3: *Figure showing the variance graph of the principal components.*

explained by each component.

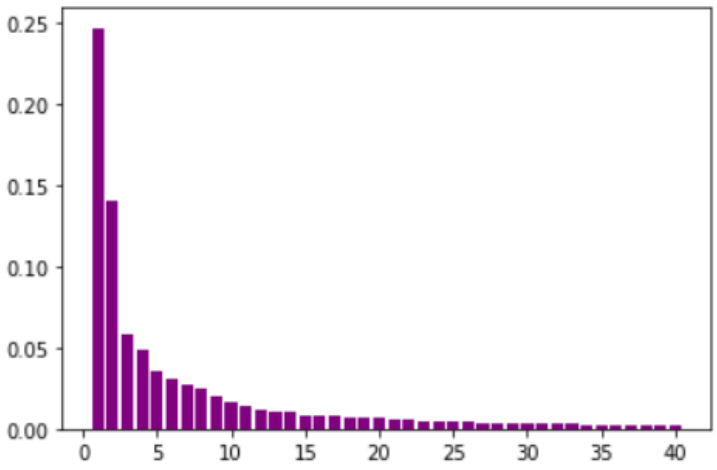


Figure 3.4: *Figure showing the bar graphs explaining the variance by each principal component.*

From the above graph we can see the variance explanation is rich in the first few components. PC1 (component 1) explains the highest variance. The variance explanation decreases as the components increases.

The below graph illustrates the distribution of covid, no covid and pneumonia between the PC1 and PC2 components.

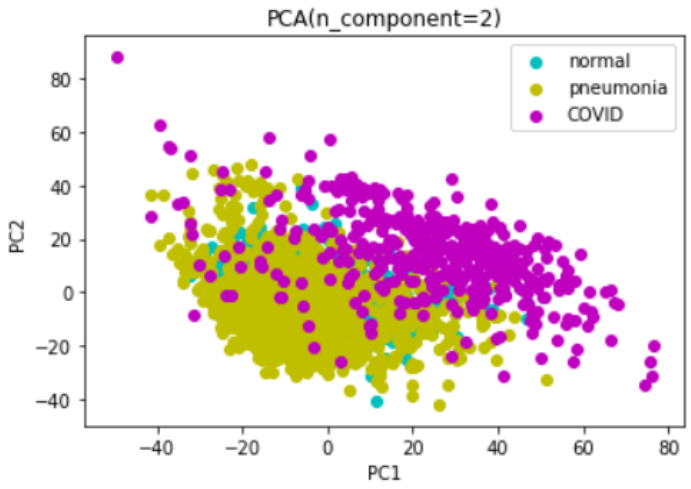


Figure 3.5: *Figure showing the principal components 1 and 2.*

3.2.3 Logistic Regression and Classification Analysis

In this section, as we have the training set of images and the test set of images available, we perform PCA on these two sets. These PCA components are used as the training and testing data. Here the target variables will not be modified both for training and testing data images and the target image categories are category 0 as normal, category 1 as pneumonia and category 2 as COVID.

Logistic regression model is first fitted on the PCA component of a training set of images and the target variable. The FIT method of this model performs the job of training the images i.e., all the essential patterns, pixels, and information of the images are captured in this process. The codebase for training is shown below.

Below is the way to create the logistic regression object.
logreg = LogisticRegression()

Below is the way of training the model.
logreg.fit(X_train_pca, y)

Below is the code for testing the data.
y_pred = logreg.predict(X_test_pca)

This y_pred variable holds the predictions or classifications that is to which category the images of the test data belong to.

3.2.4 Compute Confusion Matrix

Confusion Matrix: Confusion matrix is one form of visualizing the performance of a classification algorithm. This is also known as an error matrix. This matrix is in the form of a table layout where output can be of more than two classes and mainly contains two sections one is actual and the other is predictions. Where each row of the matrix represents the examples in a predicted class and each column represents the examples in an actual class (or vice versa). The name suggests that how the machine learning model is confused between the different target classes during the classification.(Bhandari, 2020) The confusion matrix computed as part of our image classification is

```
              precision    recall  f1-score   support

      normal       0.94      0.92      0.93       258
   pneumonia       0.91      0.92      0.92       235
       covid       0.88      0.89      0.88       116

    accuracy                           0.91       609
   macro avg       0.91      0.91      0.91       609
weighted avg       0.91      0.91      0.91       609

[[237  14   7]
 [ 11 217   7]
 [  5   8 103]]
```

Figure 3.6: *Figure showing the confusion matrix computed and the performance KPI's.*

This table contains the four different combinations of actual and predicted values.

True Positive(TP) means the model predicted positive and in actual it is true.
False Positive(FP) means the model predicted positive and in actual it is false.
False Negative(FN) means the model predicted negative and in actual it is false.
True Negative(TN) means the model predicted negative and in actual it is true.

3.2.5 Evaluation of Performance KPIs

In the logistic regression, the main performance KPI's are precision, recall, accuracy and F1 score.

Precision: Precision tells us that Out of all the positive classes which are predicted correctly, how many classes are actually positive.This value should be a s high

as possible.

$$precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{3.1}$$

Recall: Recall tells us that out of all the positive classes, how many classes have we predicted correctly. This value should be high as possible.

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{3.2}$$

F-score: When there is a need to compare two models with low precision and high recall or vice versa, then it becomes difficult to compare these two models.So in order to make these two models comparable, F-score comes into the picture.This helps in computing precision and recall at the same time.This uses harmonic mean instead of Arithmetic Mean by punishing the extreme values.

$$F - Score = \frac{2 * precision * recall}{precision + recall} \tag{3.3}$$

3.2.6 Receiver Operating Characteristic Curve(ROC)

This curve is termed as AUC-ROC(Area Under Receiver Operating Characteristic Curve). This curve depicts the performance of a classification algorithm at different threshold settings. In this, AUC is called a separability curve that is it shows the degree of separability between the classes. This tells us how the model is able to distinguish between the classes.ROC is termed as a probability curve. The model is said to be performing well, that is the model is able to predict the class 0 as 0 and class 1 as 1 when the AUC value is higher. The ROC curve is plotted with the True Positive Rate(TPR) on the y-axis against the False Positive Rate(FPR) on the x-axis. (Aniruddha, 2020)

$$TPR = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{3.4}$$

$$FPR = \frac{FalsePositives}{FalsePositives + TrueNegatives} \tag{3.5}$$

3.2.7 Performance Evaluation of SVM

In this algorithm, the PCA is applied to the training data set. This PCA applied training data set of images is now sent to the model using the GRID Search CV mechanism.Grid search is the mechanism for implementing hyperparameter tuning which

results in the optimal values for a given model.

```
              precision    recall  f1-score   support

      normal       0.97      0.94      0.95       258
   pneumonia       0.94      0.96      0.95       235
       covid       0.93      0.96      0.94       116

    accuracy                           0.95       609
   macro avg       0.95      0.95      0.95       609
weighted avg       0.95      0.95      0.95       609

[[243  10   5]
 [  7 225   3]
 [  1   4 111]]
```

Figure 3.7: *Figure showing the confusion matrix computed using SVM and the performance KPI's.*

If we see the confusion matrix obtained for SVM, we can see the performance improvement as the F1-score and accuracy both increased to 95. The performance KPI's like precision and recall also increased when compared to the Logistic regression model.

3.2.8 Performance Evaluation of Random Forest

In this algorithm, the training data set of images without applying any principal component analysis are sent to the model building with some set of hyperparameters. Later on, the model is built on the 1000 estimators and the respected performance measures are captured.

```
              precision    recall  f1-score   support

      normal       0.95      0.90      0.93       258
   pneumonia       0.88      0.96      0.92       235
       covid       0.92      0.86      0.89       116

    accuracy                           0.92       609
   macro avg       0.92      0.91      0.91       609
weighted avg       0.92      0.92      0.92       609

[[232  20   6]
 [  6 226   3]
 [  5  11 100]]
```

Figure 3.8: *Figure showing the confusion matrix computed using Random Forest and the performance KPI's.*

From the above confusion matrix, we can see that the precision, recall for each class have been decreased when compared to the metrics of SVM. F-score and accuracy also have been impacted.

3.2.9 CNN building and Training

STEP 1:: Defining the train,test and validation image sets

For Convolutional Neural Networks(CNN), the training, testing, and validation image data sets are defined. The target classes have been assigned for COVID as class 0, normal as class 1 and pneumonia as class 2.

STEP 2:: Initialising the CNN

The CNN is implemented using the Keras framework of python as it is highly flexible for deep learning projects. The CNN is built using the below code.

cnn = tf.keras.models.Sequential()

STEP 3:: Convolution

In this step, the input layer, output layer, and the inner hidden layers and the activation function are defined. The activation function we use here is ReLu.This is defined in the below code.

cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation='relu', input_shape=[64, 64, 1]))

STEP 4: Pooling

Pooling is the concrete block of CNN. Its major role is to gradually reduce the spatial size of representation in order to reduce the computation of the network and the number of parameters. This layer operates independently on each feature. The most common approach used in pooling is max pooling.

cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))

STEP 5: Full Connection

In fully connected layers, the neurons have complete connections with the activations of the previous layer.

cnn.add(tf.keras.layers.Dense(units=128, activation='relu'))

STEP 6: Output Layer

cnn.add(tf.keras.layers.Dense(units=3, activation='softmax'))

STEP 7: Training the CNN

history=cnn.fit(x = training_set, validation_data = test_set, epochs = 10)

3.2.10 Performance KPI's of CNN

Confusion matrix:

	precision	recall	f1-score	support
0	0.33	0.36	0.35	91
1	0.37	0.31	0.33	121
2	0.28	0.32	0.30	88
accuracy			0.33	300
macro avg	0.33	0.33	0.33	300
weighted avg	0.33	0.33	0.33	300

covid19 = 0 , normal = 1, pneumonia = 2

Figure 3.9: *Figure showing the classification report using CNN and the performance KPI's.*

Model Performance:

The model performance can be viewed by comparing the training and validation accuracies.

3.2.11 EfficientNet implementation through Transfer Learning

Implementation of transfer learning is acheived by importing tensorflow, keras and efficientNet.

The model is then trained, tested and validated.

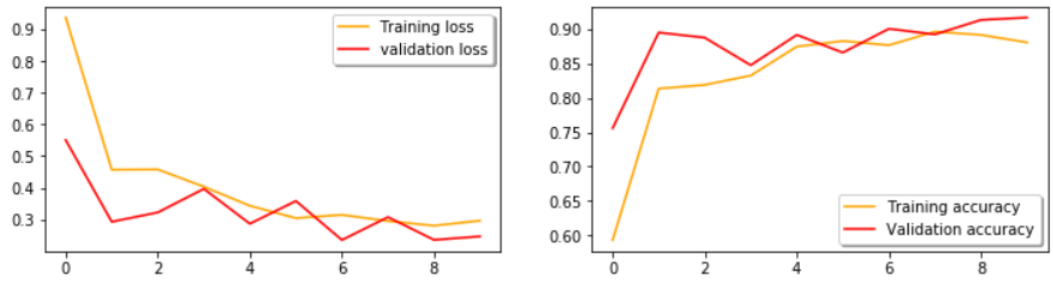


Figure 3.10: *Figure showing the training and validation data accuracy using CNN and the performance KPI's.*

```
img_shape=224
import efficientnet.keras as efn
#using here EfficientNet series B0
baseModel =efn.EfficientNetB0(weights ='noisy-student', include_top=False, input_shape = (img_shape,img_shape,3))
baseModel.summary()
x = baseModel.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.3)(x)
x = Dense(64, activation='relu')(x)
predictions = Dense(3, activation='softmax')(x)
model = Model(inputs=baseModel.input, outputs=predictions)
for layer in baseModel.layers:
    layer.trainable = False
model.compile(optimizer='adam', loss="categorical_crossentropy", metrics=['accuracy'])
```

Figure 3.11: *Figure showing the transfer learning code implementation.*

3.2.12 Performance Evaluation of EfficientNet Results

```
Model: "efficientnet-b0"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, 224, 224, 3)  0
__________________________________________________________________________________________________
stem_conv (Conv2D)              (None, 112, 112, 32) 864         input_1[0][0]
__________________________________________________________________________________________________
stem_bn (BatchNormalization)    (None, 112, 112, 32) 128         stem_conv[0][0]
__________________________________________________________________________________________________
stem_activation (Activation)    (None, 112, 112, 32) 0           stem_bn[0][0]
__________________________________________________________________________________________________
block1a_dwconv (DepthwiseConv2D (None, 112, 112, 32) 288         stem_activation[0][0]
__________________________________________________________________________________________________
block1a_bn (BatchNormalization) (None, 112, 112, 32) 128         block1a_dwconv[0][0]
__________________________________________________________________________________________________
block1a_activation (Activation) (None, 112, 112, 32) 0           block1a_bn[0][0]
__________________________________________________________________________________________________
block1a_se_squeeze (GlobalAvera (None, 32)           0           block1a_activation[0][0]
__________________________________________________________________________________________________
block1a_se_reshape (Reshape)    (None, 1, 1, 32)     0           block1a_se_squeeze[0][0]
__________________________________________________________________________________________________
block1a_se_reduce (Conv2D)      (None, 1, 1, 8)      264         block1a_se_reshape[0][0]
__________________________________________________________________________________________________
block1a_se_expand (Conv2D)      (None, 1, 1, 32)     288         block1a_se_reduce[0][0]
__________________________________________________________________________________________________
block1a_se_excite (Multiply)    (None, 112, 112, 32) 0           block1a_activation[0][0]
                                                                 block1a_se_expand[0][0]
__________________________________________________________________________________________________
block1a_project_conv (Conv2D)   (None, 112, 112, 16) 512         block1a_se_excite[0][0]
__________________________________________________________________________________________________
block1a_project_bn (BatchNormal (None, 112, 112, 16) 64          block1a_project_conv[0][0]
__________________________________________________________________________________________________
block2a_expand_conv (Conv2D)    (None, 112, 112, 96) 1536        block1a_project_bn[0][0]
__________________________________________________________________________________________________
block2a_expand_bn (BatchNormali (None, 112, 112, 96) 384         block2a_expand_conv[0][0]
__________________________________________________________________________________________________
```

Figure 3.12: *Figure showing the transfer learning code implementation.*

	precision	recall	f1-score	support
0	0.26	0.35	0.30	75
1	0.73	0.36	0.48	204
2	0.07	0.33	0.12	21
accuracy			0.35	300
macro avg	0.35	0.35	0.30	300
weighted avg	0.57	0.35	0.41	300

covid19 = 0 , normal = 1, pneumonia = 2

Figure 3.13: *Figure showing the transfer learning confusion matrix results.*

Chapter 4

Results

This chapter will give a brief explanation of the results obtained by various machine learning algorithms implemented in the previous section. In this chapter, we will evaluate the results of (i) Logistic regression (ii) Random Forest (iii) SVM (iv) CNN and (v) EfficientNet. This chapter will be followed by a conclusion.

4.1 Data Observations

This section explains the results of various algorithms starting from the number of images present in the data set to the final algorithm selection and the reasons for choosing the algorithm. Every algorithm results will be observed and analyzed in this section.

The classifier is fed with the sample data images. In this implementation, we have 315 images of chest x-rays. The chest x-rays are divided into three categories. One is a training set, which consists of 153 images. The other is the test set, which consists of 83 chest x-rays and the final set is the validation set which consists of 79 chest x-ray images. The training set contains 77 x-rays images with COVID-19 indications and 76 X-ray images of patients without COVID-19 indications.

The target variable in this project has three categories or belongs to three classes. One is the normal class where the patient is not diagnosed with COVID-19, second class is 'pneumonia' class where the patient is diagnosed with pneumonia disease and third class is 'COVID' where the patient is diagnosed with the COVID-19 symptoms.

In a classification problem, the major concern for a business problem is to decide which performance metric to choose, either precision or recall. There should always be a trade-off between these two metrics. In medical scenarios, the confidence at which a model is giving out the results is very important as the results decide the level of sustainability of living beings. In our project, precision metrics tell us out of positive

predictions(i.e., out of COVID predictions) how many are actually positive(how many are actually COVID). The recall tells us out of total predictions(includes both positive and negative that is missed out ones) how many are actually correct predictions.

In our dissertation, both precision and recall are important. ie., our model needs to maintain the trade-off between the two metrics. These two metrics tell us the miss classifications made by our model. These miss classifications, in turn, tells us the type 1 and type 2 errors of our model.(Schmarzo, 2020)

Type 1 error:
False-positive component captures the Type 1 error. Here, the prediction is positive where in reality, the actual result is false. For example, the model predicted a patient as COVID whereas, in reality, the patient is healthy.(Schmarzo, 2020)

Type 2 error:
False Negative component captures the Type 2 error. Here, the prediction is negative where in reality, the actual result is true. For example, the model predicted a patient as no COVID whereas, in reality, the patient is COVID.(Schmarzo, 2020)

In our dissertation, the detection of COVID x-rays as COVID is vital. Here, the damage caused by predicting COVID patients as healthy is way more than the damage caused by predicting healthy patients as COVID. If a COVID person is a miss classified as healthy, then there will be no treatment for the patient and he is loosely released into society. This causes the virus to spread faster and results in immeasurable havoc. Whereas, if the healthy patient is miss classified as COVID by the model, then there will be the burden of taking the medications and the burden of unnecessary financial expenditure on the person. But this does not result in the damage to the life of human beings. So, here the metric recall(False Negatives) costs us more than the metric precision. Type 2 error is more dangerous than the Type 1 error. So, our model should concentrate on reducing false negatives thereby increasing the recall. The higher the recall metric the lower the false negatives. This should not lead to a decrease in precision. While maintaining the recall ratio, precision should also be balanced.

4.2 Logistic Regression Results

Fig 3.6 depicts the miss classification of classes when logistic regression is implemented. The confusion matrix obtained in the logistic regression tells us the major performance KPI's like precision, recall,f1-score, and accuracy.

4.2.1 Confusion Matrix Results

The precision of capturing the COVID class is 88 while the recall of the COVID class is 89. The F1-score for the COVID class is 88 which is the harmonic mean of precision and recall. There is more possibility of increasing the recall.

4.2.2 ROC Curve

The AUC-ROC curve is the best way to assess our model. It indicates the ability of our model to classify the target variable at various thresholds. The curve is plotted as shown below.

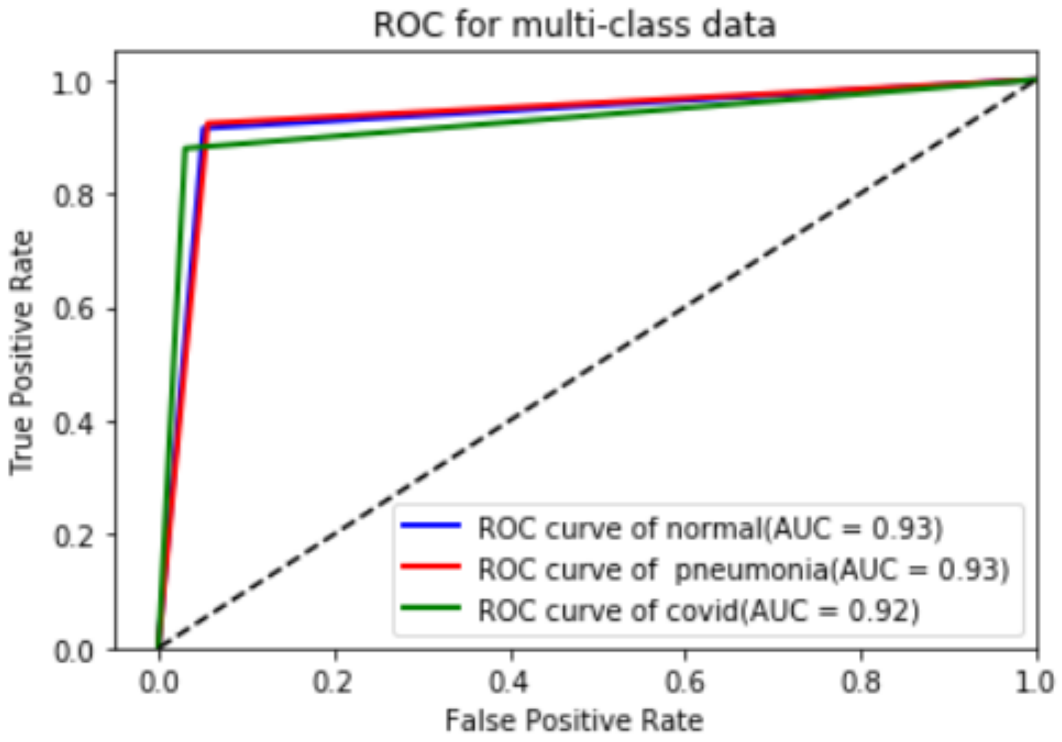


Figure 4.1: *Figure depicting the ROC-AUC for the logistic regression model.*

The highest true positive rate along with the lowest false positive rate gives the best cut-off. Usefulness of a test is usually determined by the area of the curve. The more the area occupied the better the model is. The AUC-ROC of multi class is explained as One vs the rest. The AUC score of 92 and 93 for the three different classes shows the better performance of the model.

4.3 Support Vector Machine Results

Fig 3.7 shows the miss classification of classes when SVM is implemented. The confusion matrix obtained in the support vector machine tells us the major performance KPI's like precision, recall,f1-score, and accuracy. As the hinge loss is the underlying function, the accuracy increases as the classification boundary for each class is increased. SVM has the inbuilt capability of classifying multiple classes in the best possible manner.

The recall and precision metrics for each class have been increased in comparison with logistic regression. The recall for COVID class is 96 which tells us that 96 percent of total predictions are correctly predicted as COVID.

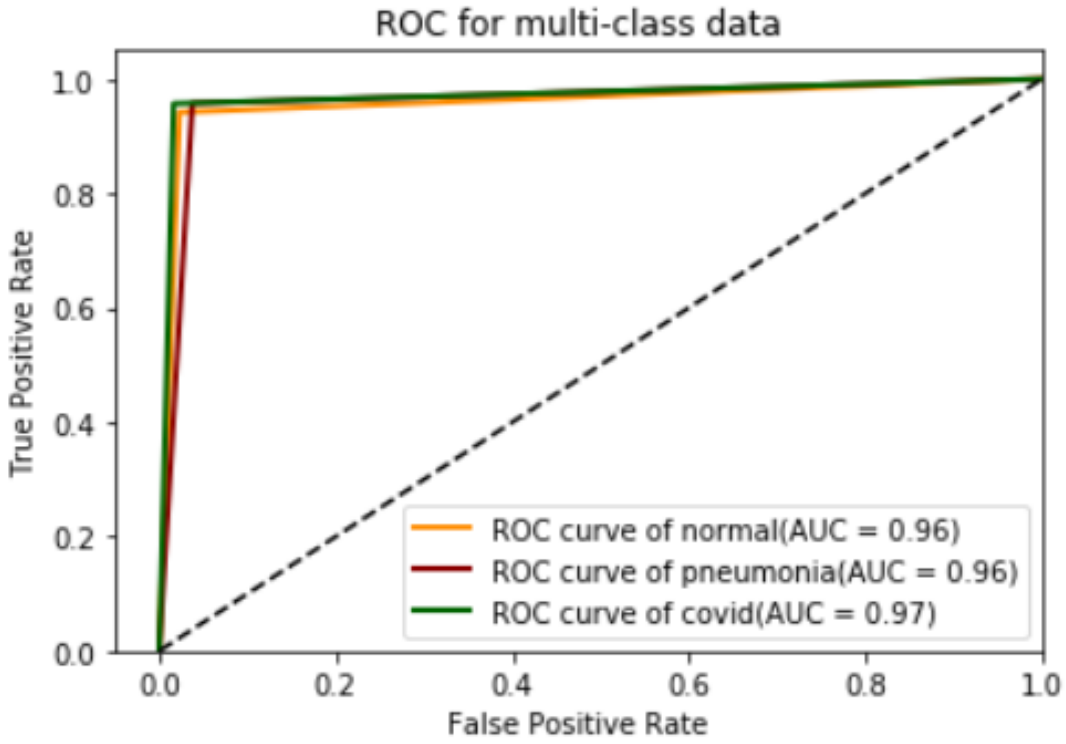


Figure 4.2: *Figure depicting the ROC-AUC for SVM model.*

The good increase in the area under the curve indicates the explainability of the model.The highest true positive rate along with the lowest false positive rate gives the best cut-off. Usefulness of a test is usually determined by the area of the curve. The more the area occupied the better the model is. The AUC-ROC of multi class is explained as One vs the rest. The AUC score of 96 and 97 for the three different classes

shows the best performance of the model. The SVM outperformed the logistic results.

4.4 Random Forest Results

Fig 3.8 shows the miss classification of classes when Random Forest is implemented. The confusion matrix obtained in the random forest tells us the major performance KPI's like precision, recall,f1-score, and accuracy.

As the random forest mainly depends on the underlying decision trees to implement, it considers the combination of multiple outputs. But, in multi-class classification this underperforms the SVM. The results depict the same.

4.4.1 Confusion Matrix Results

The metric which we are concentrating on this project is recall and we can see the recall score is at 86 which is reduced by 10 points when compared to SVM. Even the metric precision is also reduced from 97 to 92 when compared with SVM.

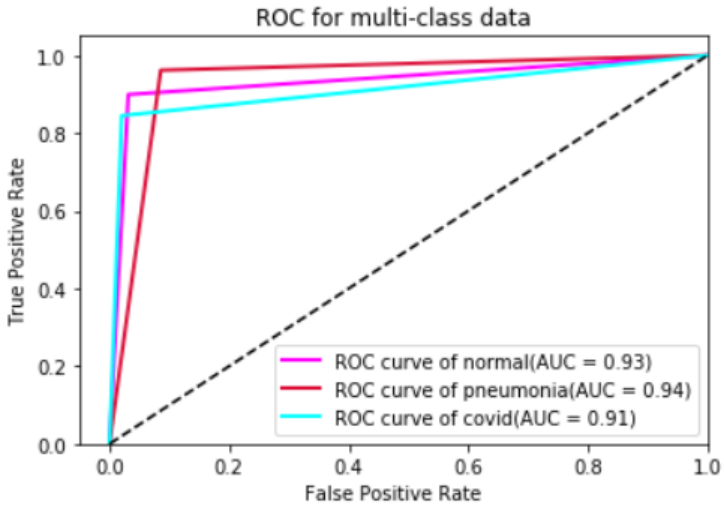


Figure 4.3: *Figure depicting the ROC-AUC for Random Forest model.*

The highest true positive rate along with the lowest false positive rate gives the best cut-off. Usefulness of a test is usually determined by the area of the curve. The more the area occupied the better the model is. The AUC-ROC of multi class is explained as One vs the rest. The AUC score of 93 ,94 and 91 for the three respective

classes shows the good performance of the model.

4.5 CNN Results

Fig 3.9 shows the miss classification of classes when CNN is implemented. The confusion matrix obtained in the implementation of CNN tells us the major performance KPI's like precision, recall,f1-score, and accuracy.

4.5.1 Performance Results Evaluation

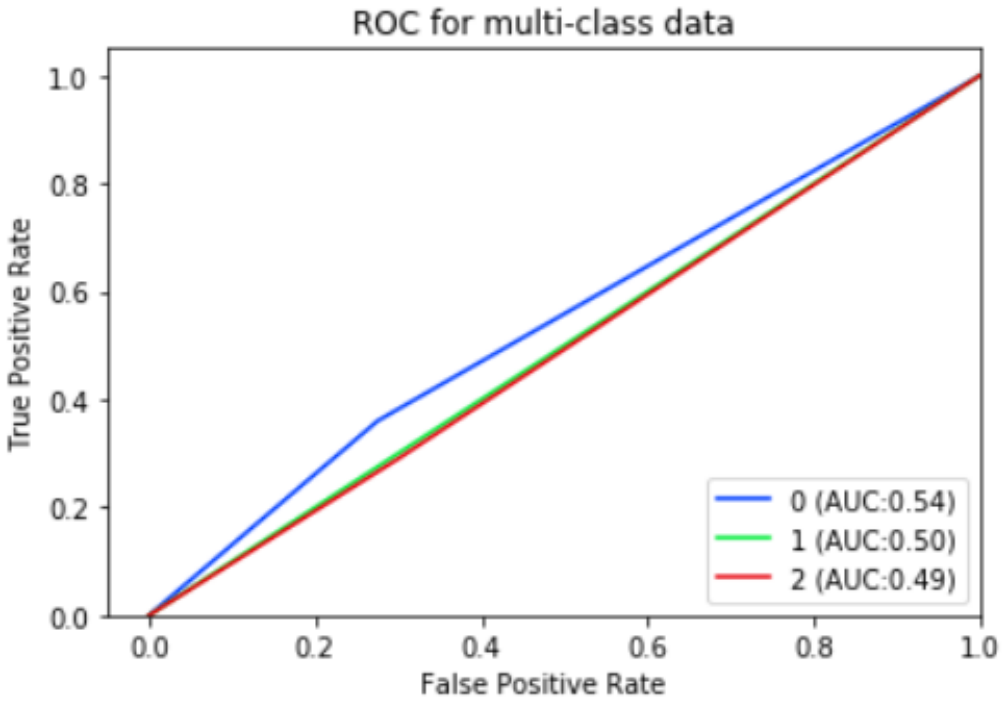


Figure 4.4: *Figure depicting the ROC-AUC for Convolutional Neural Networks.*

class 0 = covid19, class 1 = normal, class 2 = pneumonia
The ROC curve shows the AUC score ranging between 49 and 51 for the target classes present in our problem. This shows our CNN model is not able to recognize or categorise the classes properly. i.e., 0 is not detected as class 0 and so on. This shows the poor performance of the model.

4.6 Transfer Learning with EfficientNet Results

Fig 3.13 shows the miss classification of the classes while implementing the EfficientNet technique. The performance KPI's precision,recall and f1-score have been dropped below 50 percent indicating the poor performance of the model.

4.6.1 Performance Results Evaluation

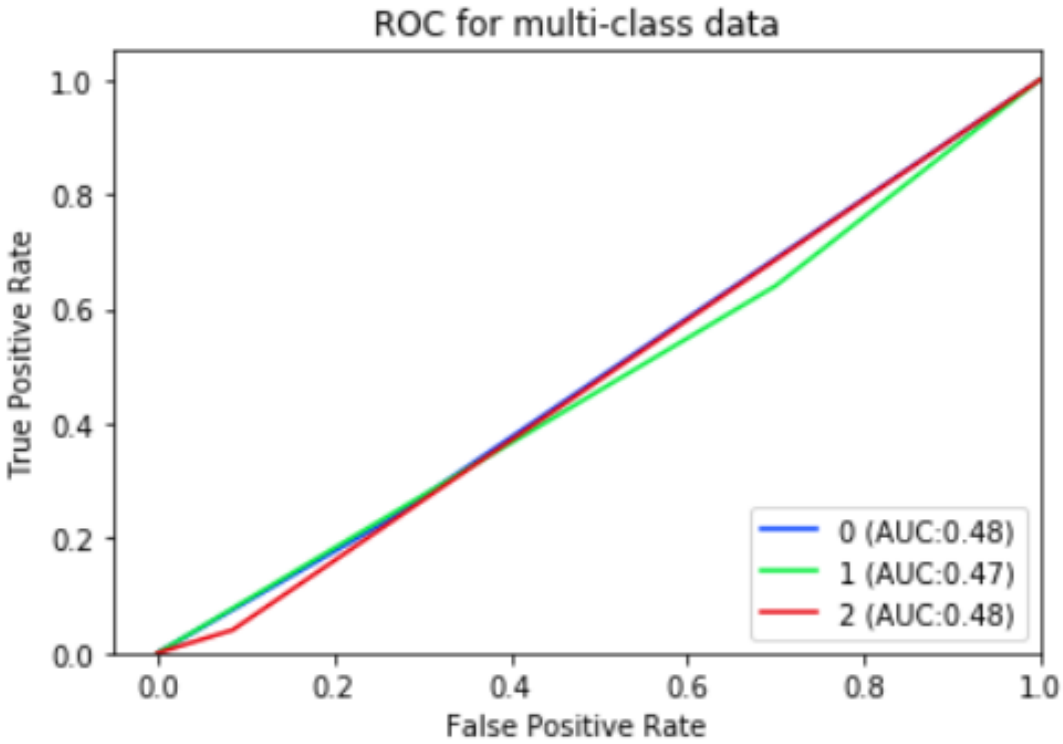


Figure 4.5: *Figure depicting the ROC-AUC for EfficientNet model.*

class 0 = covid19, class 1 = normal, class 2 = pneumonia

The ROC curve shows the AUC score ranging between 48 and 50 for the target classes present in our problem. This shows our EfficientNet model is not able to recognize or categorise the classes properly. i.e., 0 is not detected as class 0 and so on. This shows the poor performance of the model.

4.7 Final Model

After all the model implementations, based on the performance evaluations of all the models, the SVM model is able to categorise the target classes very well. This can be said by analysing the post model synopsis. The precision, recall and f1 score metrics results are far better than the results of other models.So, in this project, SVM classifier is adoped for the image detection model.

Chapter 5

Conclusion

5.1 Introduction

The novel coronavirus(COVID-19) escalated very quickly that the rhythm of the entire globe has been modified by it. In whichever perceptions we may look, all the countries irrespective of their stature, financial background, technological innovations have been put under test. This virus resulted in the most obvious consequences to the entire globe, which yielded in the major setbacks in terms of the losses incurred. The major reverberations are the economic recession, trade exploitation among the countries, poor global governance, impact on the global environment and on the socio-economic culture of the society.

Once the pandemic ends, which will be surely happening, there must be a comprehensive assessment of the ability of the world to maintain solidity in the future when posed with similar kinds of threats or challenges to humankind. With the current scenario, it is evident that the whole world is able to draw the conclusions to some small extent possible.

These kinds of pandemics are not new to the world. However, the COVID-19 pandemic occupies a special position because of its ability to break the interconnectivity and interdependence between the continents, between the countries and between the people. When the roots of dependence are so deep across the globe, such kind of pandemics yields in the unprecedented loss which takes years to recover.(Qian et al., 2020)

In times of such crisis, technological achievements lay down the way for innovation. Artificial Intelligence is one such innovative sector during this COVID-19 global pandemic. This pandemic has come up with a variety of data, which is being used by the data scientists all around the world to analyze the disease, to track and detect the disease, to guide the response and to find the medications. One such way to help the

world is to detect the COVID chest x-ray images resulting in the earlier identification of the virus. The data resulting from the pandemics, i.e., the gap between the occurrence of an outbreak to the visible results of the virus in the community, is hard to analyze for human beings. Artificial Intelligence can be irreplaceable in mitigating these numbers. A various number of projects are already on their way playing the role of battling the pandemic across the complete life cycle of the outbreak: from prediction, detection, response, and the way to recovery. (Pierre-Louis Biaggi, 2020)

5.2 Consequences of the COVID Pandemic

This Coronavirus pandemic is regarded as the major global health catastrophe of the century. This is one of the most substantial challenges faced by the mankind since the world war II. As claimed by the WHO, report on the coronavirus cases states that there are 21294845 cases and 761779 deaths in more than 200 countries across the globe as per August 16, 2020 (WHO-COVID-Report, 2020). The epidemic has spread exponentially across the globe posing serious threats to society as a whole to health, physical, social and environmental. There is no official record of any antiviral drugs or vaccinations clinically approved that are successful against COVID-19. The clinical trials for COVID vaccine are going on relentlessly by most of the countries like Russia, India, the USA, China and many more.

The exponential spread of the virus has become a burden on the functioning of the existing medical system across the world. All the medical staff is routed to work on the patients infected by coronavirus thus resulting in the void of monitoring the patients suffering from other health problems. As the cases are increasing day by day, there is a huge load on the healthcare professionals and doctors. Since the severity of the pandemic is increasing day by day, there is a shortage of N-95 masks and PPE kits for doctors and ventilators and beds to the infected patients in the hospitals. This is resulting in the need for high protection to the doctors who are on the front line of possessing the high risk.

Despite the trials for the vaccine, all the nations simultaneously struggling to slow down the transmission of the disease. This is done by implementing various methods like testing the patients and treating them, imposing a strict quarantine on the suspected people, tracing the people who came in touch with the infected persons through contact tracing method, limiting the huge gatherings, closing down the areas of large gatherings like malls, restaurants, movie theatres, implementing partial or complete lockdown. (Whitelaw and A Mamas, 2020)

The economic recession has started due to this COVID situation. Due to the imposed travel bans, the manufacture of the essentials goods has been slowed down.

The supply chain of the products is disrupted. Businesses both at national and international levels incurred huge shrinkage in the revenue. This also resulted in the poor cash flow in the market. Revenue growth is significantly slowed down.

But on the contrary, there are some sectors of business that utilized the pandemic situation and converted this outbreak of pandemic situation to an opportunity. There are two types of "growth drivers" to businesses that are performing well in these desperate times of the pandemic. The first set includes the companies which are generating benefits on the basis of work from home strategy and the lockdown. The examples include WhiteHatJr - an online platform to teach coding skills to kids. This company has doubled its growth in the last two months. The OTT live streaming platforms like Amazon Prime, Netflix, online education portals, and online gaming sections. The other group of businesses is those who are working closely with the government and some government-funded startups. (Maria, 2020)

In the face of having considerable negative impacts on the geo, socio, and economic sectors, the tech sector is the one providing the various solutions of mitigating the pandemic. Chest x-ray images of the patients play a crucial role in detecting the stage of the virus infection. The use of AI in this area helps to detect the virus-infected patient profiles accurately thus helping the patients to receive the treatment quickly. This simple innovation helps a lot in curbing the spread of the virus.

5.3 Technological Learning's from COVID-19

This pandemic has increased the dependency on technology as social distancing and the work from home choices have become the usual norms. Artificial Intelligence and Machine Learning models are being used to make calculated and informed decisions. There are much such learning's of AI helping in mitigating the virus spread in various stages of the virus commute. (Qian et al., 2020)

There are multiple applications developed to curb the spread at various stages. One such stage of detection is early diagnosis.

Early Diagnosis:

Early stage diagnosis plays a vital role in mitigating the virus. Machine Learning is used ferociously in distinguishing risky patients in the initial stages and therefore assisting in controlling the expansion of the infection in real-time. This identification becomes of utmost importance during this crisis as the real-time observation and monitoring of the spread is the better option because this helps the people to self-isolate and curb the growth of the virus. The covid-19 detection using the chest x-ray images

is one such brainchild of AI used for extraditing the process.

With the help of AI and augmented reality, a startup company of France called "Clevy" is providing assistance in diagnosing the COVID-19 symptoms. This provides remote assistance to the patients thus allowing them to confine to their respective homes. During this critical juncture of isolation rules, this is both cost and time-effective method. It also helps in decreasing the workload on the front line health care workers by avoiding the gathering of a large population.(Hossain, 2020)

Prediction and Tracking:

Artificial Intelligence is also used in bringing better ways of predicting or forecasting the virus spread and also used in tracking the spread of the disease. Various mathematical models in conjunction with machine learning techniques have been implemented. Some of the examples are i) Global Epidemic and Mobility model ii) Susceptible, Infectious and Recovered model iii) Transportation Analysis and Simulation model. The main goal of these developed models is to predict the transmission rate of the virus and also to theoretically calculate the positive cases. Thus the most unsafe regions are identified by the count of the confirmed cases and the required precautionary measures can be implemented.(Hossain, 2020)

Restriction of False Information:

Controlling and mitigating the false information on this coronavirus pandemic is an appreciative step as the misinformation leads to the unnecessary fear in the society. AI is also used in this area as a response to the crisis by a process of screening and removing the false information. With this, false medical information that is getting circulated on social media is removed and the data integrity is restored. This plays a crucial role in this kind of typical health crisis.(Hossain, 2020)

Development of COVID Vaccine:

IBM Watson Health is the product of AI which is being used to determine the correct medication for the patient by scanning the database of symptoms and medicines. This technology can actively determine the probabilistic medication needed for the patient. AI is also used in the process of developing the vaccine as it reduces manual labor and also lowers the risk of selection of doses of different drug combinations. AI can make this process more efficient.(Hossain, 2020)

All the above-mentioned products of AI contributes in some or other way in this critical crisis of health.

5.4 Limitations of the study

The limitations of this research mainly arise from the perception of the data availability and also due to the deployment of the models in real-time scenarios. The limitations are specified as below.

1. The accurate predictions of the image data depends on the resolution of the images. The low resolution of the images may not lead to better results.

2. The chest x-ray images of the patients will not be made public as these radiographs come under the personal information of the patients and there are very few volunteers giving out the data for the research. So, this results in a low volume of data thus affecting the training classifier which may lead to poor generalization of the model on the unseen data.

3. These AI models implemented in response to mitigate the virus should be first passed through the clinical trials before making available to the public usage. This, in general, consumes more time.

4. In the world of Data Science and Artificial Intelligence, it is always quoted that "Garbage In is Garbage Out". So, the better the variants of the data sent to the model, the better and accurate the results are. It is never guaranteed that a particular model like SVM, CNN results in better results all the time. For example: for the initial training of the data and model building, if the SVM classifier works very well then as the data or the images increase, CNN may perform very much better than the SVM. So, it is always the data that plays a major role.

5.5 Future Research

The results of this study can act as the supporting basis to conduct further research.

1. Addition of more labels i.e., the chest x-ray images containing different categories of virus identifications definitely boosts the performance of the model.

2. The more analysis on emphasizing the architecture of the model in the form of an application(ie., as an app that can be installed on a mobile or a PC) is to be researched further. This helps us to bring this as a supportive tool for the doctors and the whole healthcare system in order to expedite the current process.

Bibliography

Bhandari, A. (2020*a*), 'Auc-roc curve in machine learning clearly explained - analytics vidhya'.
URL: *https://www.analyticsvidhya.com/blog/2020/06/auc-roc-curve-machine-learning/*

Bhandari, A. (2020*b*), 'Confusion matrix for machine learning'.
URL: *https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/*

Biaggi, P.-L. (2020), 'Ai and data science tool up to battle covid-19'.
URL: *https://www.orange-business.com/en/blogs/ai-and-data-science-tool-battle-covid-19*

Branswell, H. & Joseph, A. (2020), 'Who declares the coronavirus outbreak a pandemic. stat news', *STAT. Accesat în* **11**.

Breiman, L. (2015), 'Random forests leo breiman and adele cutler', *Random Forests-Classification Description* .

Cleverley, J., Piper, J. & Jones, M. M. (2020), 'The role of chest radiography in confirming covid-19 pneumonia', *bmj* **370**.

Cohen, J. P., Morrison, P., Dao, L., Roth, K., Duong, T. Q. & Ghassemi, M. (2020), 'Covid-19 image data collection: Prospective predictions are the future', *arXiv 2006.11988* .
URL: *https://github.com/ieee8023/covid-chestxray-dataset*

D, M. (2020), '5 businesses that flourish during the covid-19'.
URL: *https://www.cleveroad.com/blog/covid19-business*

Gatea, F. (2020), 'Classification covid 19 x-ray images by using transfer learning with efficientnet and mobilenetv2'.
URL: *https://medium.com/@falahgs07/classification-covid-19-x-ray-images-by-using-transfer-learning-with-efficientnet-and-mobilenetv2-2c461e7402e3*

Guo, Y.-R., Cao, Q.-D., Hong, Z.-S., Tan, Y.-Y., Chen, S.-D., Jin, H.-J., Tan, K.-S., Wang, D.-Y. & Yan, Y. (2020), 'The origin, transmission and clinical therapies on coronavirus disease 2019 (covid-19) outbreak–an update on the status', *Military Medical Research* **7**(1), 1–10.

Hamdaoui, Y. (2020), 'Image data analysis'.
URL: *https://towardsdatascience.com/image-data-analysis-using-python-edddfdf128f4*

Hossain, S. (2020), 'How artificial intelligence is helping us fight against covid-19?'.
URL: *https://towardsdatascience.com/how-artificial-intelligence-is-helping-us-fight-against-covid-19-2f4d885ddbd2*

Islam, M. S., Rahman, K. M., Sun, Y., Qureshi, M. O., Abdi, I., Chughtai, A. A. & Seale, H. (2020), 'Current knowledge of covid-19 and infection prevention and control strategies in healthcare settings: A global analysis', *Infection Control & Hospital Epidemiology* pp. 1–11.

Molnar, C. (2018), 'Interpretable machine learning. christophm. github. io/interpretable-ml-book', *URL: https://christophm. github. io/interpretable-ml-book* .

Nain, A. (2020), 'Efficientnet: Rethinking model scaling for convolutional neural networks'.
URL: *https://medium.com/@nainaakash012/efficientnet-rethinking-model-scaling-for-convolutional-neural-networks-92941c5bfb95*

Ng, M.-Y., Lee, E. Y., Yang, J., Yang, F., Li, X., Wang, H., Lui, M. M.-s., Lo, C. S.-Y., Leung, B., Khong, P.-L. et al. (2020), 'Imaging profile of the covid-19 infection: radiologic findings and literature review', *Radiology: Cardiothoracic Imaging* **2**(1), e200034.

Organization, W. H. et al. (2020), Modes of transmission of virus causing covid-19: implications for ipc precaution recommendations: scientific brief, 27 march 2020, Technical report, World Health Organization.

Ozturk, T., Talo, M., Yildirim, E. A., Baloglu, U. B., Yildirim, O. & Acharya, U. R. (2020), 'Automated detection of covid-19 cases using deep neural networks with x-ray images', *Computers in Biology and Medicine* p. 103792.

Peters, A., Vetter, P., Guitart, C., Lotfinejad, N. & Pittet, D. (2020), 'Understanding the emerging coronavirus: what it means for health security and infection prevention', *Journal of Hospital Infection* **104**(4), 440–448.

Qian, X., Ren, R., Wang, Y., Guo, Y., Fang, J., Wu, Z.-D., Liu, P.-L. & Han, T.-R. (2020), 'Fighting against the common enemy of covid-19: a practice of building a

community with a shared future for mankind', *Infectious Diseases of Poverty* **9**(1), 1–6.

Radhakrishnan, P. (2020), 'What is transfer learning?'.
URL: *https://towardsdatascience.com/what-is-transfer-learning-8b1a0fa42b4*

Rao, A. (2020), 'Convolutional neural network (cnn)'.
URL: *https://www.edureka.co/blog/convolutional-neural-network/*

Rohith, G. (2020), 'Support vector machine introduction to machine learning algorithms'.
URL: *https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47*

Schmarzo, B. (2020), 'Understanding type i and type ii errors'.
URL: *https://www.datasciencecentral.com/profiles/blogs/understanding-type-i-and-type-ii-errors*

Sharfstein, J. M., Becker, S. J. & Mello, M. M. (2020), 'Diagnostic testing for the novel coronavirus', *Jama* **323**(15), 1437–1438.

Sharma, A. (2020), 'Decision tree split methods — decision tree machine learning'.
URL: *https://www.analyticsvidhya.com/blog/2020/06/4-ways-split-decision-tree/*

Sidath, A. (2020), 'Machine learning classifiers'.
URL: *https://towardsdatascience.com/machine-learning-classifiers-a5cc4e1b0623*

Team, S. (2020), 'Superdatascience'.
URL: *https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-3-flattening*

Transfer Learning Explained (2020).
URL: *https://medium.com/the-official-integrate-ai-blog/transfer-learning-explained-7d275c1e34e2*

Trochim, W. M. (2020), 'Deduction induction'.
URL: *https://conjointly.com/kb/deduction-and-induction/*

Whitelaw, S., Mamas, M. A., Topol, E. & Van Spall, H. G. (2020), 'Applications of digital technology in covid-19 pandemic planning and response', *The Lancet Digital Health* .

Appendix A - Code

August 29, 2020

1 COVID-19 Classification

by Kartheek Raj Mulasa

Importing Necessary Python libraries

[1]:
```
import glob
import numpy as np
from keras.preprocessing.image import load_img,img_to_array
import os
import matplotlib.pyplot as plt
import cv2 as cv2
import pandas as pd
import seaborn as sns
import sklearn.metrics as metrics
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import to_categorical
import matplotlib.image as mpimg
from PIL import Image
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, BatchNormalization, Conv2D,␣
 ↪MaxPool2D, Flatten
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score,f1_score
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc
from keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf
from keras.layers import  GlobalAveragePooling2D,BatchNormalization
from skimage.io import imshow
from sklearn.utils.multiclass import unique_labels
```

```
Using TensorFlow backend.
```

[2]:
```
import keras; print(keras.__version__)
```

```
2.3.1
```

Setting the Working Directory

[3]:
```
os.getcwd()
os.chdir('C:\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19␣
 ↪Radiography Database\\whole data\\Train')
```

Loading the Data

1.COVID

[4]:
```
img_size = (224,224)
#dir_name = ''
img_list = glob.glob('C:\\Users\\mkart\\OneDrive\\Documents\\Deep Learning␣
 ↪udemy\\COVID-19 Radiography Database\\whole data\\Train\\COVID-19\\*')

list_covid = []
for img in img_list:
    temp_img = load_img(img,grayscale=True,target_size=(img_size))
    temp_img_array = img_to_array(temp_img) /255
    list_covid.append(temp_img_array)
list_covid = np.array(list_covid)
list_covid2 = list_covid.reshape(-1,50176)
df_covid=pd.DataFrame(list_covid2)
df_covid['label'] = np.full(df_covid.shape[0],2)
```

```
C:\Users\mkart\Anaconda3\lib\site-
packages\keras_preprocessing\image\utils.py:104: UserWarning: grayscale is
deprecated. Please use color_mode = "grayscale"
  warnings.warn('grayscale is deprecated. Please use '
```

[5]:
```
df_covid.shape
```

[5]:
```
(559, 50177)
```

2.NORMAL

[6]:
```
img_size = (224,224)
img_list2 = glob.glob('C:\\Users\\mkart\\OneDrive\\Documents\\Deep Learning␣
 ↪udemy\\COVID-19 Radiography Database\\whole data\\Train\\NORMAL\\*')

list_normal = []
for img in img_list2[:]:
    temp_img = load_img(img,grayscale=True,target_size=(img_size))
    temp_img_array = img_to_array(temp_img) /255
    list_normal.append(temp_img_array)
```

```
list_normal = np.array(list_normal)
list_normal2 = list_normal.reshape(-1,50176)
df_normal=pd.DataFrame(list_normal2)
df_normal['label'] = np.full(df_normal.shape[0],0)
```

[7]:
```
df_normal.shape
```

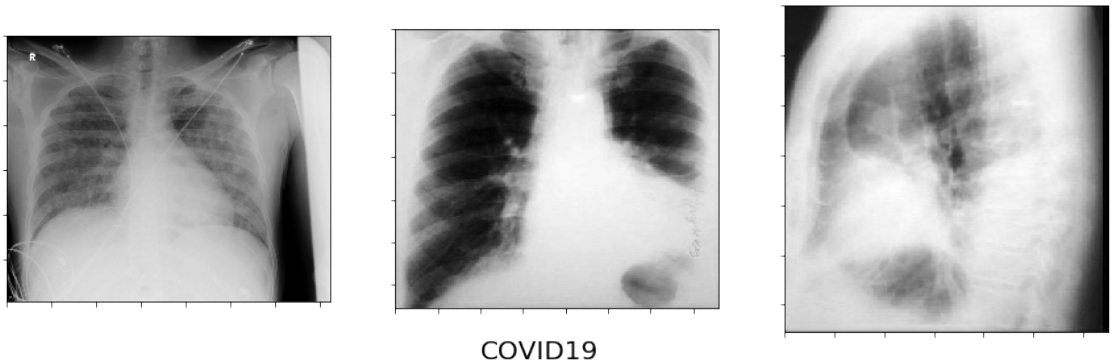
[7]:
```
(1240, 50177)
```

3.PNEUMONIA

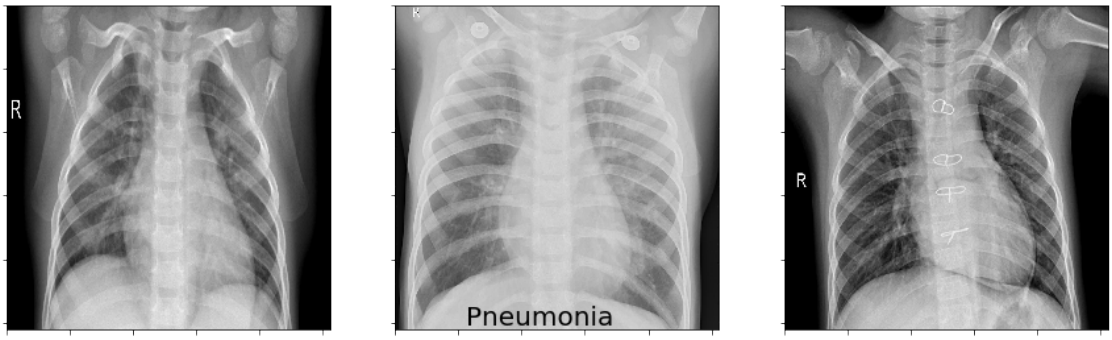
[8]:
```
img_size = (224,224)
img_list3 = glob.glob('C:\\Users\\mkart\\OneDrive\\Documents\\Deep Learning␣
 ↪udemy\\COVID-19 Radiography Database\\whole data\\Train\\Viral Pneumonia\\*')

list_others = []
for img in img_list3[:]:
    temp_img = load_img(img,grayscale=True,target_size=(img_size))
    temp_img_array = img_to_array(temp_img) /255
    list_others.append(temp_img_array)
list_others = np.array(list_others)
list_others2 = list_others.reshape(-1,50176)
df_others=pd.DataFrame(list_others2)
df_others['label'] = np.full(df_others.shape[0],1)
```

Dispalying Imported Images

[9]:
```
f = plt.figure(figsize=(15,7))
f.suptitle('COVID19',fontsize=20)
f.subplots_adjust(top=2.35)
for i in range(3):
    sp = f.add_subplot(1,3,i+1)
    img = cv2.imread(img_list[i])
    img_gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    plt.tick_params(labelbottom=False,
                    labelleft=False,
                    labelright=False,
                    labeltop=False)
    plt.imshow(img_gray)
    plt.gray()
plt.show()
```



[10]:
```
f = plt.figure(figsize=(15,7))
f.suptitle('Pneumonia',fontsize=20)
f.subplots_adjust(top=2.25)
for i in range(3):
    sp = f.add_subplot(1,3,i+1)
    img = cv2.imread(img_list3[i])
    img_gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    plt.tick_params(labelbottom=False,
                    labelleft=False,
                    labelright=False,
                    labeltop=False)
    plt.imshow(img_gray)
plt.gray()
plt.show()
```



[11]:
```
f = plt.figure(figsize=(15,7))
f.suptitle('Normal',fontsize=20)
f.subplots_adjust(top=2.4)
for i in range(3):
    sp = f.add_subplot(1,3,i+1)
    img = cv2.imread(img_list2[i])
```

```
    img_gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    plt.tick_params(labelbottom=False,
                    labelleft=False,
                    labelright=False,
                    labeltop=False)
    plt.imshow(img_gray)
plt.gray()
plt.show()
```

Data Split -Train/Test

[12]:
```
Df = pd.concat([df_covid, df_normal , df_others], ignore_index=True)
```

[13]:
```
print(Df.columns)
```

```
Index([      0,       1,       2,       3,       4,       5,       6,       7,
             8,       9,
       ...
         50167,   50168,   50169,   50170,   50171,   50172,   50173,   50174,
         50175, 'label'],
      dtype='object', length=50177)
```

[14]:
```
x_train, x_test, y_train, y_test = train_test_split(Df.iloc[:,0:-1], Df.iloc[:
 ↪,-1], test_size=0.20, random_state=0)

X_train = x_train.values.reshape(-1,224,224,1)
X_test = x_test.values.reshape(-1,224,224,1)
Y_train = to_categorical(y_train)
Y_test = to_categorical(y_test)
```

1.0.1 Applying PCA

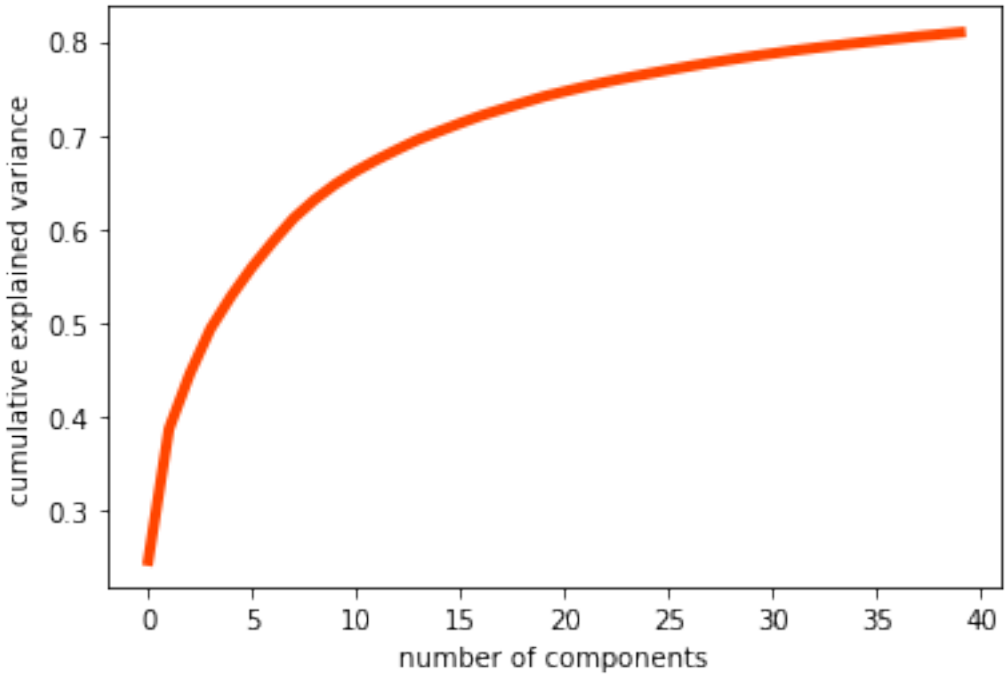
[15]:
```
from sklearn.decomposition import PCA
from time import time

n_components = 40
n_samples=411
h=224
w =224

print("Extracting the top %d eigenfaces from %d cases"
      % (n_components, x_train.shape[0]))
t0 = time()
pca = PCA(n_components=n_components, svd_solver='randomized',
          whiten=True).fit(x_train)
print("done in %0.3fs" % (time() - t0))

eigenfaces = pca.components_.reshape((n_components, h, w))

print("Projecting the input data on the eigenfaces orthonormal basis")
t0 = time()
X_train_pca = pca.transform(x_train)
X_test_pca = pca.transform(x_test)
print("done in %0.3fs" % (time() - t0))
```

```
Extracting the top 40 eigenfaces from 2435 cases
done in 4.892s
Projecting the input data on the eigenfaces orthonormal basis
done in 1.578s
```

PCA Components Expained Variance

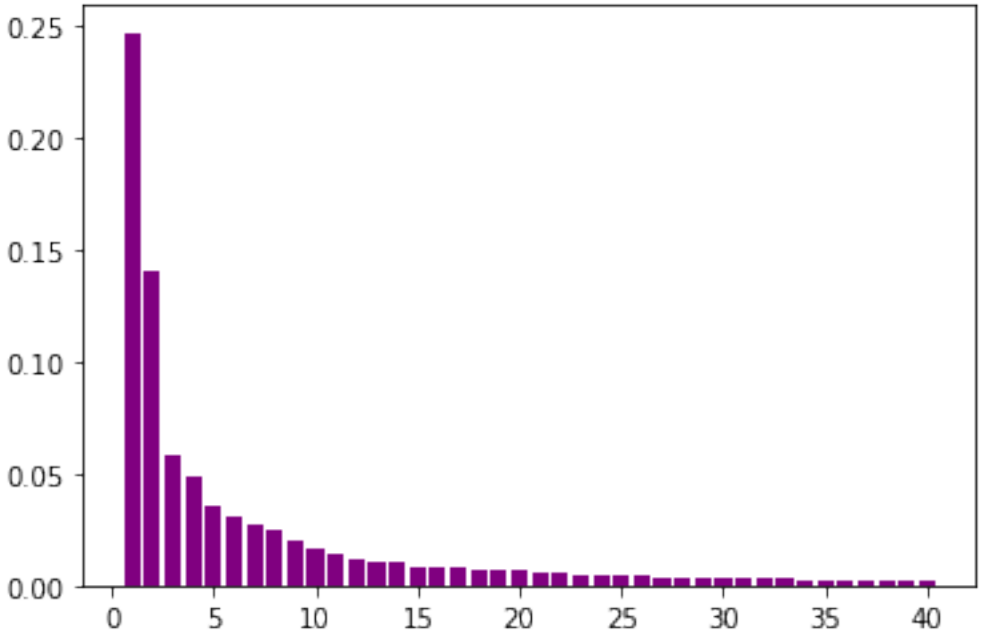
[16]:
```
plt.plot(np.cumsum(pca.explained_variance_ratio_),color='orangered', linewidth=4)
plt.xlabel('number of components')
plt.ylabel('cumulative explained variance')
```

[16]:
```
Text(0, 0.5, 'cumulative explained variance')
```



[17]:
```
plt.bar([n for n in range(1, len(pca.explained_variance_ratio_)+1)], pca.
↪explained_variance_ratio_,color='purple')
```

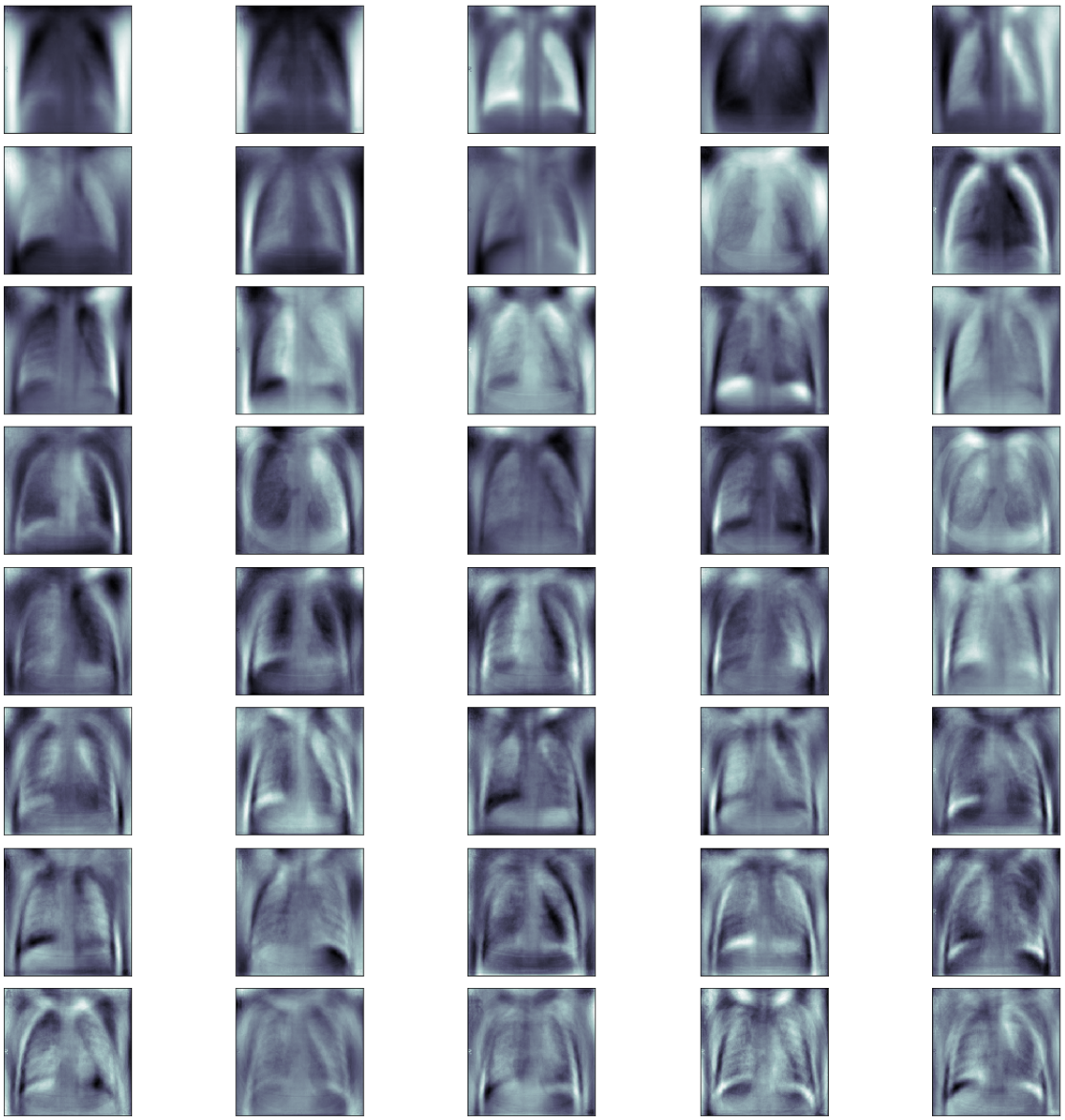
[17]: <BarContainer object of 40 artists>



PCA Visualization

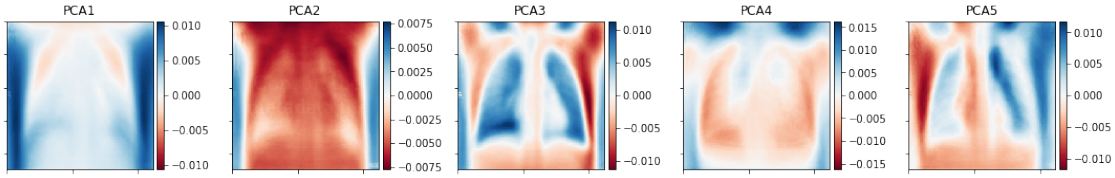
All Components

[18]:
```
fig, axes = plt.subplots(8, 5, figsize=(20, 20),
                         subplot_kw={'xticks':[], 'yticks':[]},
                         gridspec_kw=dict(hspace=0.1, wspace=0.1))
for i, ax in enumerate(axes.flat):
    ax.imshow(pca.components_[i].reshape(224, 224), cmap='bone')
```



First 5 PCA

[19]:
```
from skimage.io import imshow
loadeigen = eigenfaces[0]
f = plt.figure(figsize=(15,4))
f.subplots_adjust(top=1)
for i in range(5):
    sp = f.add_subplot(1,5,i+1)
    plt.tick_params(labelbottom=False,
                    labelleft=False,
                    labelright=False,
                    labeltop=False)
    loadeigen = eigenfaces[i]
    imshow(loadeigen)
    sp.title.set_text('PCA'+str(i+1))
plt.show()
```

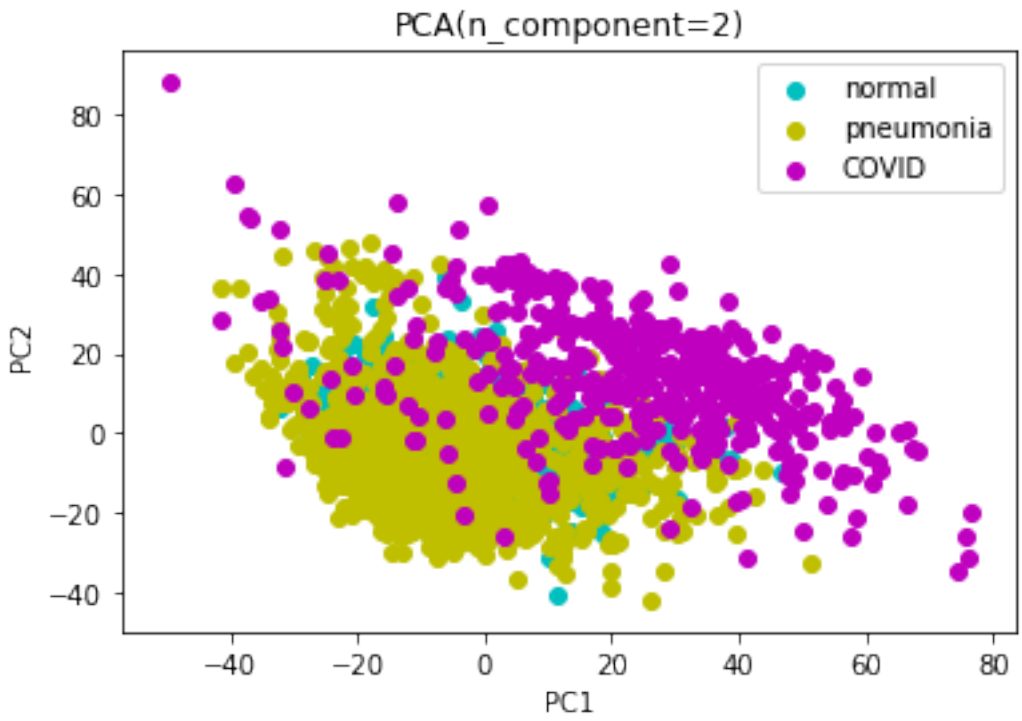


PCA and the classes

[20]:
```
pca_2D = PCA(n_components=2).fit_transform(x_train)
pca_2D
```

[20]:
```
array([[-16.305983 ,  -7.089573 ],
       [-11.453943 ,   2.1625826],
       [ -3.606972 , -10.9638405],
       ...,
       [-10.325927 ,  18.111462 ],
       [ 39.61222  ,  -8.935476 ],
       [ 18.855707 ,  -4.1618166]], dtype=float32)
```

[21]:
```
mycolors=["c","y","m"]
labelTups = ['normal','pneumonia','COVID']
label=y_train
for i,mycolor in enumerate(mycolors):
        plt.scatter(pca_2D[label == i, 0],
                    pca_2D[label == i, 1], color=mycolor)
        plt.xlabel("PC1")
        plt.ylabel("PC2")
        plt.legend(labelTups, loc='upper right')
plt.title('PCA(n_component=2)')
plt.show()
```



1.1 Logi Model

[22]:
```
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(X_train_pca, y_train)
```

[22]:
```
LogisticRegression()
```

[23]:
```
y_pred = logreg.predict(X_test_pca)
print('Accuracy of logistic regression classifier on test set: {:.2f}'.
 ↪format(logreg.score(X_test_pca, y_test)))
```

```
Accuracy of logistic regression classifier on test set: 0.91
```

Confusion Matrix

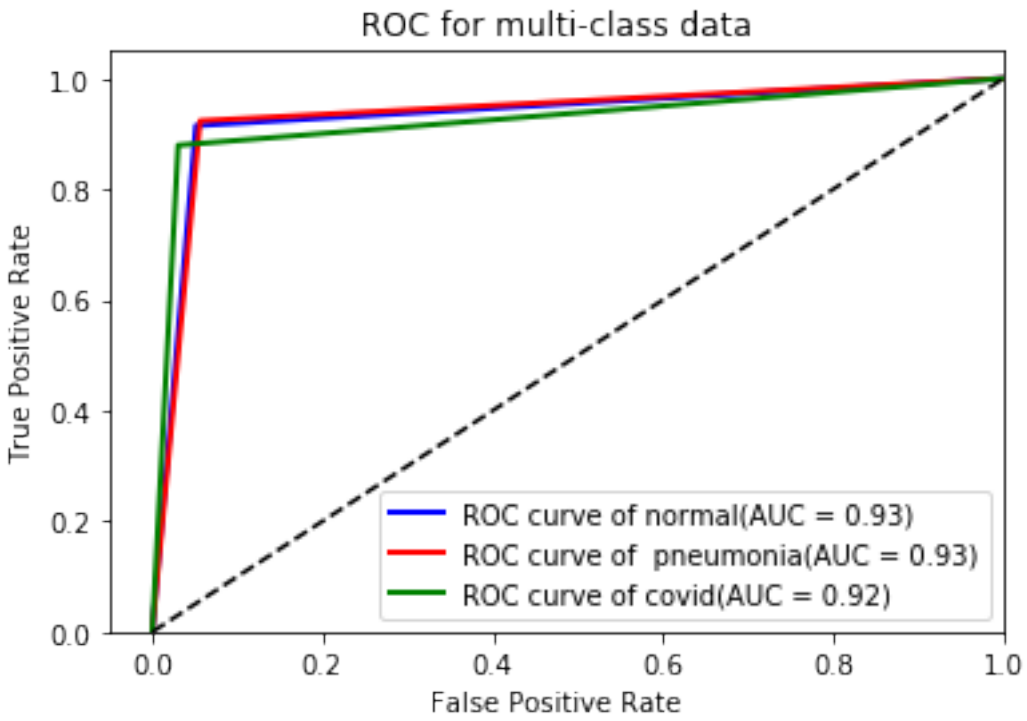
[24]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred)
target_names = ['normal','pneumonia', 'covid']
n_classes=3
print(classification_report(y_test, y_pred, target_names=target_names))
print(confusion_matrix)
```

```
              precision    recall  f1-score   support
```

```
      normal       0.93      0.91      0.92       258
   pneumonia       0.91      0.92      0.92       235
       covid       0.87      0.88      0.88       116

    accuracy                           0.91       609
   macro avg       0.90      0.91      0.90       609
weighted avg       0.91      0.91      0.91       609

[[236  13   9]
 [ 12 217   6]
 [  6   8 102]]
```

ROC Curve

[25]:
```
from tensorflow.keras.utils import to_categorical
from keras.utils.np_utils import to_categorical
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc

PRED = to_categorical(y_pred)
y = Df['label'].values
# Binarize the output
y = label_binarize(y, classes=[0,1,2])
n_classes = y.shape[1]

fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(n_classes):
       fpr[i], tpr[i], _ = roc_curve(Y_test[:,i], PRED[:,i])
       roc_auc[i] = auc(fpr[i], tpr[i])

colors = ['blue', 'red', 'green']
cls = {0:'normal', 1:' pneumonia', 2:'covid'}
for i, color ,c in zip(range(n_classes), colors, cls.values()):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
             label='ROC curve of '+c+ '(AUC = {1:0.2f})'
             ''.format(i, roc_auc[i]))
plt.plot([0, 1], [0, 1], 'k--',linestyle='--')
plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC for multi-class data')
plt.legend(loc="lower right")
plt.show()
```



1.2 SVM

[26]:
```
from sklearn.svm import SVC
from sklearn.model_selection import GridSearchCV

print("Fitting the classifier to the training set")
t0 = time()
param_grid = {'C': [1, 10, 100,1e3, 5e3, 1e4],
              'gamma': [0.001, 0.005, 0.01, 0.1], }
clf = GridSearchCV(
    SVC(kernel='rbf', class_weight='balanced'), param_grid
)
clf = clf.fit(X_train_pca, y_train)
print("done in %0.3fs" % (time() - t0))
print("Best estimator found by grid search:")
print(clf.best_estimator_)
```

```
Fitting the classifier to the training set
done in 26.947s
Best estimator found by grid search:
SVC(C=10, class_weight='balanced', gamma=0.01)
```

Confusion Matrix

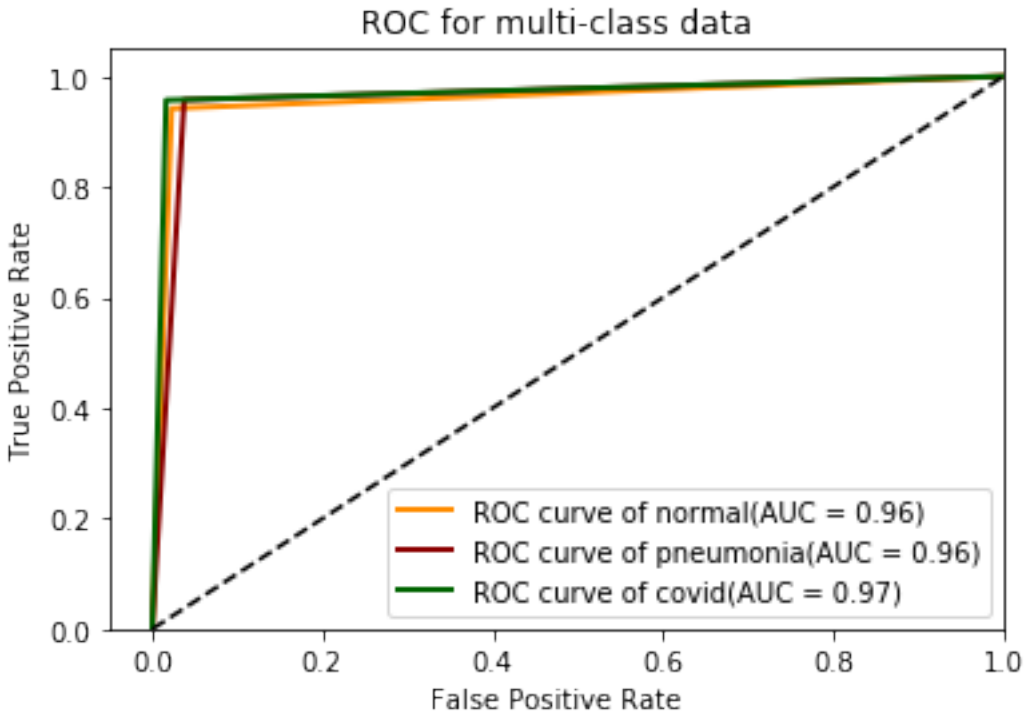
[27]:
```
from sklearn.metrics import classification_report, confusion_matrix,␣
 ↪accuracy_score
t0 = time()
y_pred = clf.predict(X_test_pca)
print("done in %0.3fs" % (time() - t0))

target_names = ['normal','pneumonia', 'covid']
n_classes=3
print(classification_report(y_test, y_pred, target_names=target_names))
print(confusion_matrix(y_test, y_pred, labels=range(n_classes)))
```

```
done in 0.031s
              precision    recall  f1-score   support

      normal       0.97      0.94      0.95       258
   pneumonia       0.94      0.96      0.95       235
       covid       0.93      0.96      0.94       116

    accuracy                           0.95       609
   macro avg       0.95      0.95      0.95       609
weighted avg       0.95      0.95      0.95       609

[[243  10   5]
 [  7 225   3]
 [  1   4 111]]
```

ROC Curve

[28]:
```
from tensorflow.keras.utils import to_categorical
from keras.utils.np_utils import to_categorical
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc

PRED = to_categorical(y_pred)
y = Df['label'].values
# Binarize the output
y = label_binarize(y, classes=[0,1,2])
n_classes = y.shape[1]

fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(n_classes):
       fpr[i], tpr[i], _ = roc_curve(Y_test[:,i], PRED[:,i])
       roc_auc[i] = auc(fpr[i], tpr[i])

colors = ['darkorange', 'darkred', 'darkgreen']
cls = {0:'normal', 1:'pneumonia', 2:'covid'}
```

```
for i, color ,c in zip(range(n_classes), colors, cls.values()):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
             label='ROC curve of '+c+ '(AUC = {1:0.2f})'
             ''.format(i, roc_auc[i]))
plt.plot([0, 1], [0, 1], 'k--',linestyle='--')
plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC for multi-class data')
plt.legend(loc="lower right")
plt.show()
```



1.3 Random Forest

[29]:
```
from sklearn.ensemble import RandomForestClassifier
estimator = RandomForestClassifier(n_estimators=1000)
estimator.fit(x_train, y_train)
```

[29]:
```
RandomForestClassifier(n_estimators=1000)
```

[30]:
```
importances = estimator.feature_importances_
importances = importances.reshape(224,224)
```

Confusion Matrix

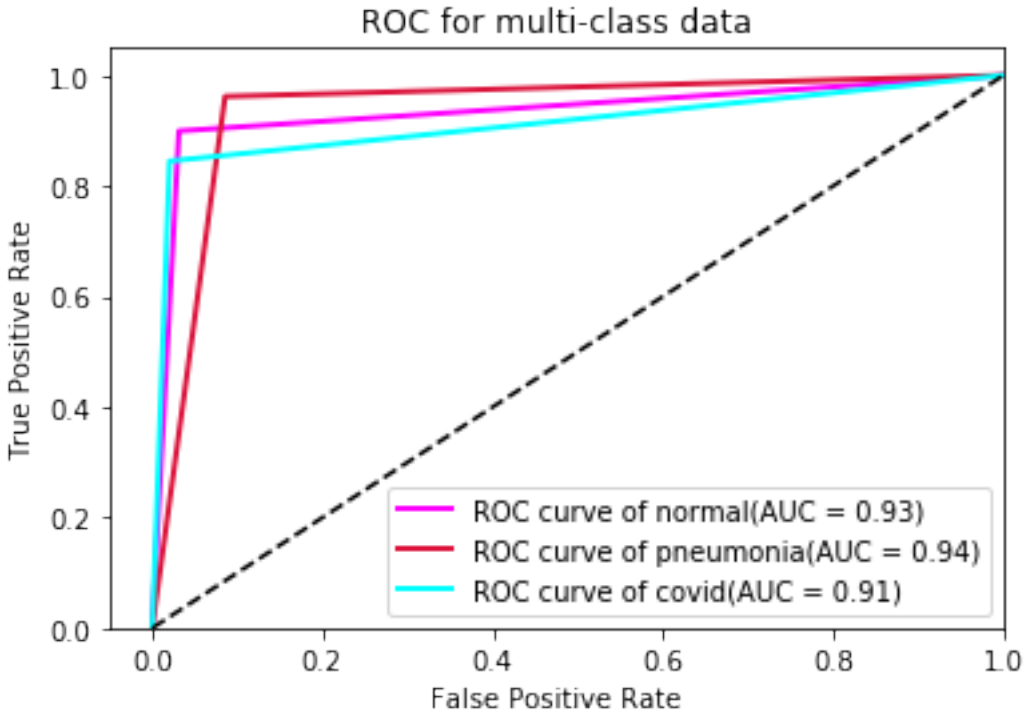
[31]:
```
y_pred = estimator.predict(x_test)
print("done in %0.3fs" % (time() - t0))

target_names = ['normal','pneumonia', 'covid']
n_classes=3
print(classification_report(y_test, y_pred, target_names=target_names))
print(confusion_matrix(y_test, y_pred, labels=range(n_classes)))
```

```
done in 322.565s
              precision    recall  f1-score   support

      normal       0.95      0.90      0.93       258
   pneumonia       0.88      0.96      0.92       235
       covid       0.91      0.84      0.88       116

    accuracy                           0.91       609
   macro avg       0.91      0.90      0.91       609
weighted avg       0.92      0.91      0.91       609

[[232  20   6]
 [  5 226   4]
 [  6  12  98]]
```

ROC Curve

[32]:
```
from tensorflow.keras.utils import to_categorical
from keras.utils.np_utils import to_categorical
from sklearn.preprocessing import label_binarize
from sklearn.metrics import roc_curve, auc

PRED = to_categorical(y_pred)
y = Df['label'].values
# Binarize the output
y = label_binarize(y, classes=[0,1,2])
n_classes = y.shape[1]

fpr = dict()
tpr = dict()
roc_auc = dict()
for i in range(n_classes):
       fpr[i], tpr[i], _ = roc_curve(Y_test[:,i], PRED[:,i])
       roc_auc[i] = auc(fpr[i], tpr[i])

colors = ['magenta', 'crimson', 'cyan']
cls = {0:'normal', 1:'pneumonia', 2:'covid'}
for i, color ,c in zip(range(n_classes), colors, cls.values()):
    plt.plot(fpr[i], tpr[i], color=color, lw=2,
```

```
            label='ROC curve of '+c+ '(AUC = {1:0.2f})'
            ''.format(i, roc_auc[i]))
plt.plot([0, 1], [0, 1], 'k--',linestyle='--')
plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC for multi-class data')
plt.legend(loc="lower right")
plt.show()
```



1.4 Convolutional Neural Network

1.4.1 Data augmentation

Training Dataset

[33]:
```
train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   )
training_set = train_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\Dessertation\\Train',
```

```
                                                target_size = (64, 64),
                                                color_mode="grayscale",
                                                batch_size = 32,
                                                class_mode = 'categorical')
```

```
Found 1469 images belonging to 3 classes.
```

[34]:
```
print(training_set.class_indices)
```

```
{'COVID-19': 0, 'NORMAL': 1, 'Viral Pneumonia': 2}
```

Validation Dataset

[35]:
```
Validation_datagen =ImageDataGenerator(rescale = 1./255,
                                       shear_range = 0.2,
                                       zoom_range = 0.2,
                                       )
Validation_set = Validation_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\Dessertation\\Train1',
                                                target_size = (64, 64),
                                                batch_size = 32,
                                                color_mode="grayscale",
                                                class_mode = 'categorical')
```

```
Found 1470 images belonging to 3 classes.
```

Test Dataset

[36]:
```
test_datagen = ImageDataGenerator(rescale = 1./255)
test_set = test_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\whole data\\Test',
                                                target_size = (64, 64),
                                                color_mode="grayscale" )
```

```
Found 300 images belonging to 3 classes.
```

[37]:
```
print(Validation_set.class_indices)
```

```
{'COVID-19': 0, 'NORMAL': 1, 'Viral Pneumonia': 2}
```

1.4.2 Building the CNN

Initialising the CNN

[38]:
```
cnn = tf.keras.models.Sequential()
```

Step 1 - Convolution

[39]:
```
cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation='relu',␣
↪input_shape=[64, 64, 1]))
```

```
WARNING:tensorflow:From C:\Users\mkart\Anaconda3\lib\site-
packages\tensorflow\python\ops\resource_variable_ops.py:435: colocate_with (from
tensorflow.python.framework.ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Colocations handled automatically by placer.
```

Step 2 - Pooling

[40]:
```
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
```

Adding a second convolutional layer

[41]:
```
cnn.add(tf.keras.layers.Conv2D(filters=32, kernel_size=3, activation='relu'))
cnn.add(tf.keras.layers.MaxPool2D(pool_size=2, strides=2))
```

Step 3 - Flattening

[42]:
```
cnn.add(tf.keras.layers.Flatten())
```

Step 4 - Full Connection

[43]:
```
cnn.add(tf.keras.layers.Dense(units=128, activation='relu'))
```

Step 5 - Output Layer

[44]:
```
cnn.add(tf.keras.layers.Dense(units=3, activation='softmax'))
```

Model summnary

[45]:
```
cnn.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 32)        320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 31, 31, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 29, 29, 32)        9248
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 14, 14, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
dense (Dense)                (None, 128)               802944
```

```
_________________________________________________________________
dense_1 (Dense)              (None, 3)                 387
=================================================================
Total params: 812,899
Trainable params: 812,899
Non-trainable params: 0
_________________________________________________________________
```

1.4.3 Training the CNN

Compiling the CNN

[46]:
```
cnn.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =␣
↪['accuracy'])
```

Training the CNN on the Training set and evaluating it on the Test set

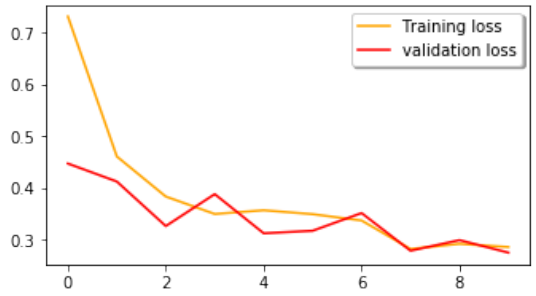
[47]:
```
history=cnn.fit(x = training_set, validation_data = Validation_set , epochs = 10)
```

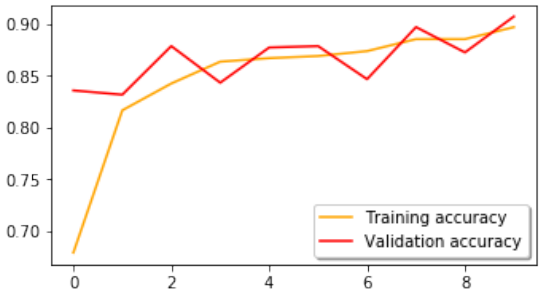
```
WARNING:tensorflow:From C:\Users\mkart\Anaconda3\lib\site-
packages\tensorflow\python\ops\math_ops.py:3066: to_int32 (from
tensorflow.python.ops.math_ops) is deprecated and will be removed in a future
version.
Instructions for updating:
Use tf.cast instead.
Epoch 1/10
46/46 [==============================] - 34s 735ms/step - loss: 0.4471 - acc:
0.8354
46/46 [==============================] - 74s 2s/step - loss: 0.7317 - acc:
0.6787 - val_loss: 0.4471 - val_acc: 0.8354
Epoch 2/10
46/46 [==============================] - 41s 889ms/step - loss: 0.4123 - acc:
0.8313
46/46 [==============================] - 72s 2s/step - loss: 0.4609 - acc:
0.8162 - val_loss: 0.4123 - val_acc: 0.8313
Epoch 3/10
46/46 [==============================] - 34s 734ms/step - loss: 0.3262 - acc:
0.8782
46/46 [==============================] - 66s 1s/step - loss: 0.3830 - acc:
0.8421 - val_loss: 0.3262 - val_acc: 0.8782
Epoch 4/10
46/46 [==============================] - 32s 691ms/step - loss: 0.3880 - acc:
0.8429
46/46 [==============================] - 68s 1s/step - loss: 0.3493 - acc:
0.8632 - val_loss: 0.3880 - val_acc: 0.8429
Epoch 5/10
46/46 [==============================] - 31s 674ms/step - loss: 0.3123 - acc:
0.8769
```

```
46/46 [==============================] - 62s 1s/step - loss: 0.3569 - acc:
0.8666 - val_loss: 0.3123 - val_acc: 0.8769
Epoch 6/10
46/46 [==============================] - 31s 665ms/step - loss: 0.3171 - acc:
0.8782
46/46 [==============================] - 64s 1s/step - loss: 0.3490 - acc:
0.8686 - val_loss: 0.3171 - val_acc: 0.8782
Epoch 7/10
46/46 [==============================] - 35s 772ms/step - loss: 0.3514 - acc:
0.8463
46/46 [==============================] - 72s 2s/step - loss: 0.3371 - acc:
0.8734 - val_loss: 0.3514 - val_acc: 0.8463
Epoch 8/10
46/46 [==============================] - 31s 681ms/step - loss: 0.2786 - acc:
0.8966
46/46 [==============================] - 61s 1s/step - loss: 0.2822 - acc:
0.8850 - val_loss: 0.2786 - val_acc: 0.8966
Epoch 9/10
46/46 [==============================] - 36s 780ms/step - loss: 0.2991 - acc:
0.8721
46/46 [==============================] - 71s 2s/step - loss: 0.2917 - acc:
0.8850 - val_loss: 0.2991 - val_acc: 0.8721
Epoch 10/10
46/46 [==============================] - 39s 847ms/step - loss: 0.2749 - acc:
0.9068
46/46 [==============================] - 76s 2s/step - loss: 0.2859 - acc:
0.8965 - val_loss: 0.2749 - val_acc: 0.9068
```

1.4.4 Model Performance

[48]:
```
plt.style.use('seaborn-bright')
fig, ax = plt.subplots(1,2, figsize=(12, 3))
ax[0].plot(history.history['loss'], color='orange', label="Training loss")
ax[0].plot(history.history['val_loss'], color='r', label="validation loss",axes␣
 ↪=ax[0])
legend = ax[0].legend(loc='best', shadow=True)
ax[1].plot(history.history['acc'], color='orange', label="Training accuracy")
ax[1].plot(history.history['val_acc'], color='r',label="Validation accuracy")
legend = ax[1].legend(loc='best', shadow=True)
```





Confusion Matrix

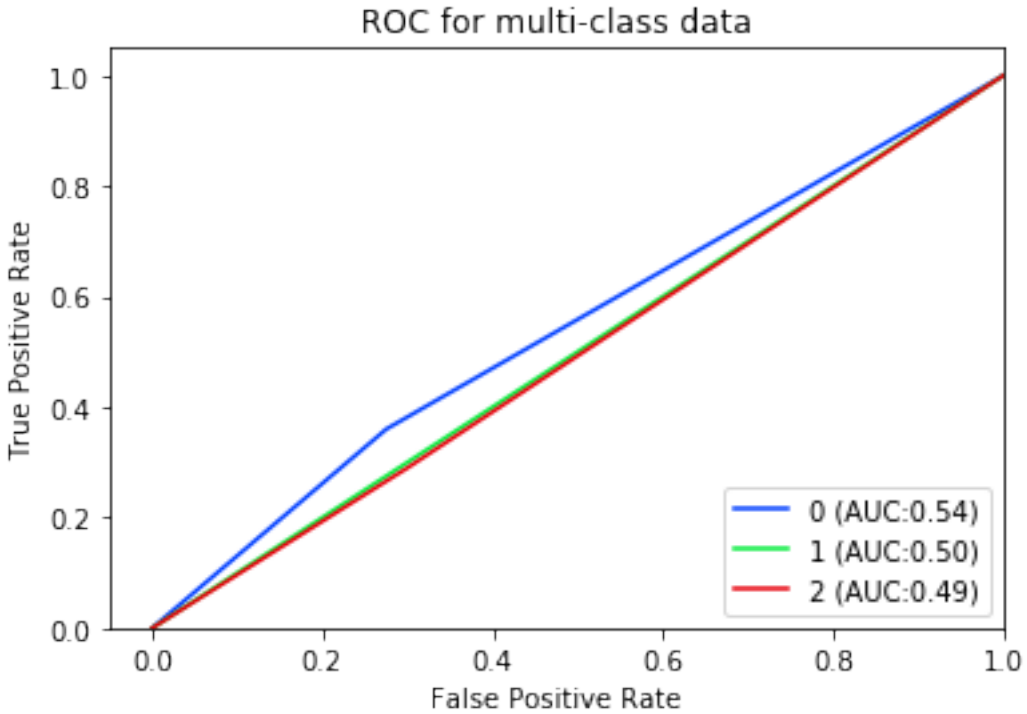
[49]:
```
pred = cnn.predict_classes(test_set)
cnnp=cnn.predict_proba(test_set)
cm = metrics.confusion_matrix(test_set.classes,pred)
from sklearn.metrics import classification_report
print(classification_report(pred,test_set.classes))
print(' covid19 = 0 , normal = 1, pneumonia = 2')
print(cm)
```

```
              precision    recall  f1-score   support

           0       0.36      0.40      0.38        91
           1       0.40      0.33      0.36       120
           2       0.29      0.33      0.31        89

    accuracy                           0.35       300
   macro avg       0.35      0.35      0.35       300
weighted avg       0.36      0.35      0.35       300

 covid19 = 0 , normal = 1, pneumonia = 2
[[36 36 28]
 [28 40 32]
 [27 44 29]]
```

ROC Curve

[50]:
```
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import roc_auc_score
# set plot figure size
fig, c_ax = plt.subplots()

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
    lb = LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
```

```
    #all_labels= {0:'normal', 1:'pneumonia', 2:'covid'}
    all_labels= {'normal', 'pneumonia', 'covid'}

    for (idx, c_label) in enumerate(all_labels): # all_labels: no of the labels,␣
↪for ex. ['cat', 'dog', 'rat']
        fpr, tpr, thresholds = roc_curve(y_test[:,idx].astype(int), y_pred[:
↪,idx])
        c_ax.plot(fpr, tpr, label = '%s (AUC:%0.2f)'  % (c_label, auc(fpr, tpr)))
        colors = ['magenta', 'crimson', 'cyan']

    return roc_auc_score(y_test, y_pred, average=average)



multiclass_roc_auc_score(test_set.classes, pred)

plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC for multi-class data')
plt.legend(loc="lower right")
plt.show()
```



1.5 Transfer Learning with Imagenet

[51]:
```
train_datagen = ImageDataGenerator(
horizontal_flip=True,
vertical_flip=True,
rotation_range=90,
width_shift_range=0.1,
height_shift_range=0.1,
zoom_range=.1,
rescale=1/255,
fill_mode='nearest',
shear_range=0.1,
brightness_range=[0.5, 1.5],
validation_split=0.3)

training_set_tl = train_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\Dessertation\\Train',
                                                    target_size = (224, 224),
                                                    batch_size = 48,
                                                    class_mode = 'categorical')
```

```
Found 1469 images belonging to 3 classes.
```

[52]:
```
Validation_datagen =ImageDataGenerator(
horizontal_flip=True,
vertical_flip=True,
rotation_range=90,
width_shift_range=0.1,
height_shift_range=0.1,
zoom_range=.1,
rescale=1/255,
fill_mode='nearest',
shear_range=0.1,
brightness_range=[0.5, 1.5],
validation_split=0.3)

Validation_set_tl = Validation_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\Dessertation\\Train1',
                                                    target_size = (224, 224),
                                                    batch_size = 48,
                                                    class_mode = 'categorical')
```

```
Found 1470 images belonging to 3 classes.
```

[53]:
```
test_datagen = ImageDataGenerator(rescale = 1./255)
test_set_tl = test_datagen.flow_from_directory('C:
 ↪\\Users\\mkart\\OneDrive\\Documents\\Deep Learning udemy\\COVID-19 Radiography␣
 ↪Database\\whole data\\Test',
                                               target_size = (224, 224)
)
```

```
Found 300 images belonging to 3 classes.
```

[54]:
```
#Imported libraries and modules
import efficientnet.keras as efn
from sklearn.metrics import␣
 ↪classification_report,accuracy_score,f1_score,confusion_matrix
import numpy as np
from keras.preprocessing.image import load_img, img_to_array
import matplotlib.pyplot as plt
import os
from keras.models import Model
from keras.utils import np_utils
import tensorflow as tf
from tensorflow.keras.models import load_model
from keras.utils.np_utils import to_categorical
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Activation,Dense, Dropout, Flatten, Conv2D
from keras.layers import  GlobalAveragePooling2D,BatchNormalization
```

[55]:
```
import efficientnet.keras as efn
from keras.models import Model
```

[56]:
```
img_shape=224
import efficientnet.keras as efn
#using here EfficientNet series B0
baseModel =efn.EfficientNetB0(weights ='noisy-student', include_top=False,␣
 ↪input_shape = (img_shape,img_shape,3))
baseModel.summary()
x = baseModel.output
x = GlobalAveragePooling2D()(x)
x = Dropout(0.3)(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.3)(x)
x = Dense(64, activation='relu')(x)
predictions = Dense(3, activation='softmax')(x)
model = Model(inputs=baseModel.input, outputs=predictions)
for layer in baseModel.layers:
    layer.trainable = False
model.compile(optimizer='adam', loss="categorical_crossentropy",␣
 ↪metrics=['accuracy'])
```

```
Model: "efficientnet-b0"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, 224, 224, 3)  0
__________________________________________________________________________________________________
stem_conv (Conv2D)              (None, 112, 112, 32) 864         input_1[0][0]
__________________________________________________________________________________________________
stem_bn (BatchNormalization)    (None, 112, 112, 32) 128         stem_conv[0][0]
__________________________________________________________________________________________________
stem_activation (Activation)    (None, 112, 112, 32) 0           stem_bn[0][0]
__________________________________________________________________________________________________
block1a_dwconv (DepthwiseConv2D (None, 112, 112, 32) 288
stem_activation[0][0]
__________________________________________________________________________________________________
block1a_bn (BatchNormalization) (None, 112, 112, 32) 128
block1a_dwconv[0][0]
__________________________________________________________________________________________________
block1a_activation (Activation) (None, 112, 112, 32) 0
block1a_bn[0][0]
__________________________________________________________________________________________________
block1a_se_squeeze (GlobalAvera (None, 32)           0
block1a_activation[0][0]
__________________________________________________________________________________________________
block1a_se_reshape (Reshape)    (None, 1, 1, 32)     0
block1a_se_squeeze[0][0]
__________________________________________________________________________________________________
block1a_se_reduce (Conv2D)      (None, 1, 1, 8)      264
block1a_se_reshape[0][0]
__________________________________________________________________________________________________
block1a_se_expand (Conv2D)      (None, 1, 1, 32)     288
block1a_se_reduce[0][0]
__________________________________________________________________________________________________
block1a_se_excite (Multiply)    (None, 112, 112, 32) 0
block1a_activation[0][0]
```

```
block1a_se_expand[0][0]
__________________________________________________________________________________
__________________
block1a_project_conv (Conv2D)   (None, 112, 112, 16) 512
block1a_se_excite[0][0]
__________________________________________________________________________________
__________________
block1a_project_bn (BatchNormal (None, 112, 112, 16) 64
block1a_project_conv[0][0]
__________________________________________________________________________________
__________________
block2a_expand_conv (Conv2D)    (None, 112, 112, 96) 1536
block1a_project_bn[0][0]
__________________________________________________________________________________
__________________
block2a_expand_bn (BatchNormali (None, 112, 112, 96) 384
block2a_expand_conv[0][0]
__________________________________________________________________________________
__________________
block2a_expand_activation (Acti (None, 112, 112, 96) 0
block2a_expand_bn[0][0]
__________________________________________________________________________________
__________________
block2a_dwconv (DepthwiseConv2D (None, 56, 56, 96)   864
block2a_expand_activation[0][0]
__________________________________________________________________________________
__________________
block2a_bn (BatchNormalization) (None, 56, 56, 96)   384
block2a_dwconv[0][0]
__________________________________________________________________________________
__________________
block2a_activation (Activation) (None, 56, 56, 96)   0
block2a_bn[0][0]
__________________________________________________________________________________
__________________
block2a_se_squeeze (GlobalAvera (None, 96)           0
block2a_activation[0][0]
__________________________________________________________________________________
__________________
block2a_se_reshape (Reshape)    (None, 1, 1, 96)     0
block2a_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block2a_se_reduce (Conv2D)      (None, 1, 1, 4)      388
block2a_se_reshape[0][0]
__________________________________________________________________________________
__________________
block2a_se_expand (Conv2D)      (None, 1, 1, 96)     480
```

```
block2a_se_reduce[0][0]
__________________________________________________________________________________
__________________
block2a_se_excite (Multiply)    (None, 56, 56, 96)   0
block2a_activation[0][0]
block2a_se_expand[0][0]
__________________________________________________________________________________
__________________
block2a_project_conv (Conv2D)   (None, 56, 56, 24)   2304
block2a_se_excite[0][0]
__________________________________________________________________________________
__________________
block2a_project_bn (BatchNormal (None, 56, 56, 24)   96
block2a_project_conv[0][0]
__________________________________________________________________________________
__________________
block2b_expand_conv (Conv2D)    (None, 56, 56, 144)  3456
block2a_project_bn[0][0]
__________________________________________________________________________________
__________________
block2b_expand_bn (BatchNormali (None, 56, 56, 144)  576
block2b_expand_conv[0][0]
__________________________________________________________________________________
__________________
block2b_expand_activation (Acti (None, 56, 56, 144)  0
block2b_expand_bn[0][0]
__________________________________________________________________________________
__________________
block2b_dwconv (DepthwiseConv2D (None, 56, 56, 144)  1296
block2b_expand_activation[0][0]
__________________________________________________________________________________
__________________
block2b_bn (BatchNormalization) (None, 56, 56, 144)  576
block2b_dwconv[0][0]
__________________________________________________________________________________
__________________
block2b_activation (Activation) (None, 56, 56, 144)  0
block2b_bn[0][0]
__________________________________________________________________________________
__________________
block2b_se_squeeze (GlobalAvera (None, 144)          0
block2b_activation[0][0]
__________________________________________________________________________________
__________________
block2b_se_reshape (Reshape)    (None, 1, 1, 144)    0
block2b_se_squeeze[0][0]
__________________________________________________________________________________
__________________
```

```
block2b_se_reduce (Conv2D)      (None, 1, 1, 6)      870
block2b_se_reshape[0][0]
-------------------------------------------------------------------------------
------------------
block2b_se_expand (Conv2D)      (None, 1, 1, 144)    1008
block2b_se_reduce[0][0]
-------------------------------------------------------------------------------
------------------
block2b_se_excite (Multiply)    (None, 56, 56, 144)  0
block2b_activation[0][0]
block2b_se_expand[0][0]
-------------------------------------------------------------------------------
------------------
block2b_project_conv (Conv2D)   (None, 56, 56, 24)   3456
block2b_se_excite[0][0]
-------------------------------------------------------------------------------
------------------
block2b_project_bn (BatchNormal (None, 56, 56, 24)   96
block2b_project_conv[0][0]
-------------------------------------------------------------------------------
------------------
block2b_drop (FixedDropout)     (None, 56, 56, 24)   0
block2b_project_bn[0][0]
-------------------------------------------------------------------------------
------------------
block2b_add (Add)               (None, 56, 56, 24)   0
block2b_drop[0][0]
block2a_project_bn[0][0]
-------------------------------------------------------------------------------
------------------
block3a_expand_conv (Conv2D)    (None, 56, 56, 144)  3456
block2b_add[0][0]
-------------------------------------------------------------------------------
------------------
block3a_expand_bn (BatchNormali (None, 56, 56, 144)  576
block3a_expand_conv[0][0]
-------------------------------------------------------------------------------
------------------
block3a_expand_activation (Acti (None, 56, 56, 144)  0
block3a_expand_bn[0][0]
-------------------------------------------------------------------------------
------------------
block3a_dwconv (DepthwiseConv2D (None, 28, 28, 144)  3600
block3a_expand_activation[0][0]
-------------------------------------------------------------------------------
------------------
block3a_bn (BatchNormalization) (None, 28, 28, 144)  576
block3a_dwconv[0][0]
```

```
------------------------------------------------------------------------------
------------------
block3a_activation (Activation) (None, 28, 28, 144)  0
block3a_bn[0][0]
------------------------------------------------------------------------------
------------------
block3a_se_squeeze (GlobalAvera (None, 144)          0
block3a_activation[0][0]
------------------------------------------------------------------------------
------------------
block3a_se_reshape (Reshape)    (None, 1, 1, 144)    0
block3a_se_squeeze[0][0]
------------------------------------------------------------------------------
------------------
block3a_se_reduce (Conv2D)      (None, 1, 1, 6)      870
block3a_se_reshape[0][0]
------------------------------------------------------------------------------
------------------
block3a_se_expand (Conv2D)      (None, 1, 1, 144)    1008
block3a_se_reduce[0][0]
------------------------------------------------------------------------------
------------------
block3a_se_excite (Multiply)    (None, 28, 28, 144)  0
block3a_activation[0][0]
block3a_se_expand[0][0]
------------------------------------------------------------------------------
------------------
block3a_project_conv (Conv2D)   (None, 28, 28, 40)   5760
block3a_se_excite[0][0]
------------------------------------------------------------------------------
------------------
block3a_project_bn (BatchNormal (None, 28, 28, 40)   160
block3a_project_conv[0][0]
------------------------------------------------------------------------------
------------------
block3b_expand_conv (Conv2D)    (None, 28, 28, 240)  9600
block3a_project_bn[0][0]
------------------------------------------------------------------------------
------------------
block3b_expand_bn (BatchNormali (None, 28, 28, 240)  960
block3b_expand_conv[0][0]
------------------------------------------------------------------------------
------------------
block3b_expand_activation (Acti (None, 28, 28, 240)  0
block3b_expand_bn[0][0]
------------------------------------------------------------------------------
------------------
block3b_dwconv (DepthwiseConv2D (None, 28, 28, 240)  6000
```

```
block3b_expand_activation[0][0]
------------------------------------------------------------------------------------
------------------
block3b_bn (BatchNormalization) (None, 28, 28, 240)  960
block3b_dwconv[0][0]
------------------------------------------------------------------------------------
------------------
block3b_activation (Activation) (None, 28, 28, 240)  0
block3b_bn[0][0]
------------------------------------------------------------------------------------
------------------
block3b_se_squeeze (GlobalAvera (None, 240)          0
block3b_activation[0][0]
------------------------------------------------------------------------------------
------------------
block3b_se_reshape (Reshape)    (None, 1, 1, 240)    0
block3b_se_squeeze[0][0]
------------------------------------------------------------------------------------
------------------
block3b_se_reduce (Conv2D)      (None, 1, 1, 10)     2410
block3b_se_reshape[0][0]
------------------------------------------------------------------------------------
------------------
block3b_se_expand (Conv2D)      (None, 1, 1, 240)    2640
block3b_se_reduce[0][0]
------------------------------------------------------------------------------------
------------------
block3b_se_excite (Multiply)    (None, 28, 28, 240)  0
block3b_activation[0][0]
block3b_se_expand[0][0]
------------------------------------------------------------------------------------
------------------
block3b_project_conv (Conv2D)   (None, 28, 28, 40)   9600
block3b_se_excite[0][0]
------------------------------------------------------------------------------------
------------------
block3b_project_bn (BatchNormal (None, 28, 28, 40)   160
block3b_project_conv[0][0]
------------------------------------------------------------------------------------
------------------
block3b_drop (FixedDropout)     (None, 28, 28, 40)   0
block3b_project_bn[0][0]
------------------------------------------------------------------------------------
------------------
block3b_add (Add)               (None, 28, 28, 40)   0
block3b_drop[0][0]
block3a_project_bn[0][0]
------------------------------------------------------------------------------------
```

```
------------------
block4a_expand_conv (Conv2D)    (None, 28, 28, 240)  9600
block3b_add[0][0]
------------------------------------------------------------------------------------
------------------
block4a_expand_bn (BatchNormali (None, 28, 28, 240)  960
block4a_expand_conv[0][0]
------------------------------------------------------------------------------------
------------------
block4a_expand_activation (Acti (None, 28, 28, 240)  0
block4a_expand_bn[0][0]
------------------------------------------------------------------------------------
------------------
block4a_dwconv (DepthwiseConv2D (None, 14, 14, 240)  2160
block4a_expand_activation[0][0]
------------------------------------------------------------------------------------
------------------
block4a_bn (BatchNormalization) (None, 14, 14, 240)  960
block4a_dwconv[0][0]
------------------------------------------------------------------------------------
------------------
block4a_activation (Activation) (None, 14, 14, 240)  0
block4a_bn[0][0]
------------------------------------------------------------------------------------
------------------
block4a_se_squeeze (GlobalAvera (None, 240)          0
block4a_activation[0][0]
------------------------------------------------------------------------------------
------------------
block4a_se_reshape (Reshape)    (None, 1, 1, 240)    0
block4a_se_squeeze[0][0]
------------------------------------------------------------------------------------
------------------
block4a_se_reduce (Conv2D)      (None, 1, 1, 10)     2410
block4a_se_reshape[0][0]
------------------------------------------------------------------------------------
------------------
block4a_se_expand (Conv2D)      (None, 1, 1, 240)    2640
block4a_se_reduce[0][0]
------------------------------------------------------------------------------------
------------------
block4a_se_excite (Multiply)    (None, 14, 14, 240)  0
block4a_activation[0][0]
block4a_se_expand[0][0]
------------------------------------------------------------------------------------
------------------
block4a_project_conv (Conv2D)   (None, 14, 14, 80)   19200
block4a_se_excite[0][0]
```

```
------------------------------------------------------------------------------
------------------
block4a_project_bn (BatchNormal (None, 14, 14, 80)   320
block4a_project_conv[0][0]
------------------------------------------------------------------------------
------------------
block4b_expand_conv (Conv2D)    (None, 14, 14, 480)  38400
block4a_project_bn[0][0]
------------------------------------------------------------------------------
------------------
block4b_expand_bn (BatchNormali (None, 14, 14, 480)  1920
block4b_expand_conv[0][0]
------------------------------------------------------------------------------
------------------
block4b_expand_activation (Acti (None, 14, 14, 480)  0
block4b_expand_bn[0][0]
------------------------------------------------------------------------------
------------------
block4b_dwconv (DepthwiseConv2D (None, 14, 14, 480)  4320
block4b_expand_activation[0][0]
------------------------------------------------------------------------------
------------------
block4b_bn (BatchNormalization) (None, 14, 14, 480)  1920
block4b_dwconv[0][0]
------------------------------------------------------------------------------
------------------
block4b_activation (Activation) (None, 14, 14, 480)  0
block4b_bn[0][0]
------------------------------------------------------------------------------
------------------
block4b_se_squeeze (GlobalAvera (None, 480)          0
block4b_activation[0][0]
------------------------------------------------------------------------------
------------------
block4b_se_reshape (Reshape)    (None, 1, 1, 480)    0
block4b_se_squeeze[0][0]
------------------------------------------------------------------------------
------------------
block4b_se_reduce (Conv2D)      (None, 1, 1, 20)     9620
block4b_se_reshape[0][0]
------------------------------------------------------------------------------
------------------
block4b_se_expand (Conv2D)      (None, 1, 1, 480)    10080
block4b_se_reduce[0][0]
------------------------------------------------------------------------------
------------------
block4b_se_excite (Multiply)    (None, 14, 14, 480)  0
block4b_activation[0][0]
```

```
block4b_se_expand[0][0]
__________________________________________________________________________________
__________________
block4b_project_conv (Conv2D)   (None, 14, 14, 80)   38400
block4b_se_excite[0][0]
__________________________________________________________________________________
__________________
block4b_project_bn (BatchNormal (None, 14, 14, 80)   320
block4b_project_conv[0][0]
__________________________________________________________________________________
__________________
block4b_drop (FixedDropout)     (None, 14, 14, 80)   0
block4b_project_bn[0][0]
__________________________________________________________________________________
__________________
block4b_add (Add)               (None, 14, 14, 80)   0
block4b_drop[0][0]
block4a_project_bn[0][0]
__________________________________________________________________________________
__________________
block4c_expand_conv (Conv2D)    (None, 14, 14, 480)  38400
block4b_add[0][0]
__________________________________________________________________________________
__________________
block4c_expand_bn (BatchNormali (None, 14, 14, 480)  1920
block4c_expand_conv[0][0]
__________________________________________________________________________________
__________________
block4c_expand_activation (Acti (None, 14, 14, 480)  0
block4c_expand_bn[0][0]
__________________________________________________________________________________
__________________
block4c_dwconv (DepthwiseConv2D (None, 14, 14, 480)  4320
block4c_expand_activation[0][0]
__________________________________________________________________________________
__________________
block4c_bn (BatchNormalization) (None, 14, 14, 480)  1920
block4c_dwconv[0][0]
__________________________________________________________________________________
__________________
block4c_activation (Activation) (None, 14, 14, 480)  0
block4c_bn[0][0]
__________________________________________________________________________________
__________________
block4c_se_squeeze (GlobalAvera (None, 480)          0
block4c_activation[0][0]
__________________________________________________________________________________
__________________
```

```
block4c_se_reshape (Reshape)    (None, 1, 1, 480)    0
block4c_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block4c_se_reduce (Conv2D)      (None, 1, 1, 20)     9620
block4c_se_reshape[0][0]
__________________________________________________________________________________
__________________
block4c_se_expand (Conv2D)      (None, 1, 1, 480)    10080
block4c_se_reduce[0][0]
__________________________________________________________________________________
__________________
block4c_se_excite (Multiply)    (None, 14, 14, 480)  0
block4c_activation[0][0]
block4c_se_expand[0][0]
__________________________________________________________________________________
__________________
block4c_project_conv (Conv2D)   (None, 14, 14, 80)   38400
block4c_se_excite[0][0]
__________________________________________________________________________________
__________________
block4c_project_bn (BatchNormal (None, 14, 14, 80)   320
block4c_project_conv[0][0]
__________________________________________________________________________________
__________________
block4c_drop (FixedDropout)     (None, 14, 14, 80)   0
block4c_project_bn[0][0]
__________________________________________________________________________________
__________________
block4c_add (Add)               (None, 14, 14, 80)   0
block4c_drop[0][0]
block4b_add[0][0]
__________________________________________________________________________________
__________________
block5a_expand_conv (Conv2D)    (None, 14, 14, 480)  38400
block4c_add[0][0]
__________________________________________________________________________________
__________________
block5a_expand_bn (BatchNormali (None, 14, 14, 480)  1920
block5a_expand_conv[0][0]
__________________________________________________________________________________
__________________
block5a_expand_activation (Acti (None, 14, 14, 480)  0
block5a_expand_bn[0][0]
__________________________________________________________________________________
__________________
block5a_dwconv (DepthwiseConv2D (None, 14, 14, 480)  12000
block5a_expand_activation[0][0]
```

```
------------------------------------------------------------------------------
------------------
block5a_bn (BatchNormalization) (None, 14, 14, 480)  1920
block5a_dwconv[0][0]
------------------------------------------------------------------------------
------------------
block5a_activation (Activation) (None, 14, 14, 480)  0
block5a_bn[0][0]
------------------------------------------------------------------------------
------------------
block5a_se_squeeze (GlobalAvera (None, 480)          0
block5a_activation[0][0]
------------------------------------------------------------------------------
------------------
block5a_se_reshape (Reshape)    (None, 1, 1, 480)    0
block5a_se_squeeze[0][0]
------------------------------------------------------------------------------
------------------
block5a_se_reduce (Conv2D)      (None, 1, 1, 20)     9620
block5a_se_reshape[0][0]
------------------------------------------------------------------------------
------------------
block5a_se_expand (Conv2D)      (None, 1, 1, 480)    10080
block5a_se_reduce[0][0]
------------------------------------------------------------------------------
------------------
block5a_se_excite (Multiply)    (None, 14, 14, 480)  0
block5a_activation[0][0]
block5a_se_expand[0][0]
------------------------------------------------------------------------------
------------------
block5a_project_conv (Conv2D)   (None, 14, 14, 112)  53760
block5a_se_excite[0][0]
------------------------------------------------------------------------------
------------------
block5a_project_bn (BatchNormal (None, 14, 14, 112)  448
block5a_project_conv[0][0]
------------------------------------------------------------------------------
------------------
block5b_expand_conv (Conv2D)    (None, 14, 14, 672)  75264
block5a_project_bn[0][0]
------------------------------------------------------------------------------
------------------
block5b_expand_bn (BatchNormali (None, 14, 14, 672)  2688
block5b_expand_conv[0][0]
------------------------------------------------------------------------------
------------------
block5b_expand_activation (Acti (None, 14, 14, 672)  0
```

```
block5b_expand_bn[0][0]
__________________________________________________________________________________
__________________
block5b_dwconv (DepthwiseConv2D (None, 14, 14, 672)  16800
block5b_expand_activation[0][0]
__________________________________________________________________________________
__________________
block5b_bn (BatchNormalization) (None, 14, 14, 672)  2688
block5b_dwconv[0][0]
__________________________________________________________________________________
__________________
block5b_activation (Activation) (None, 14, 14, 672)  0
block5b_bn[0][0]
__________________________________________________________________________________
__________________
block5b_se_squeeze (GlobalAvera (None, 672)          0
block5b_activation[0][0]
__________________________________________________________________________________
__________________
block5b_se_reshape (Reshape)    (None, 1, 1, 672)    0
block5b_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block5b_se_reduce (Conv2D)      (None, 1, 1, 28)     18844
block5b_se_reshape[0][0]
__________________________________________________________________________________
__________________
block5b_se_expand (Conv2D)      (None, 1, 1, 672)    19488
block5b_se_reduce[0][0]
__________________________________________________________________________________
__________________
block5b_se_excite (Multiply)    (None, 14, 14, 672)  0
block5b_activation[0][0]
block5b_se_expand[0][0]
__________________________________________________________________________________
__________________
block5b_project_conv (Conv2D)   (None, 14, 14, 112)  75264
block5b_se_excite[0][0]
__________________________________________________________________________________
__________________
block5b_project_bn (BatchNormal (None, 14, 14, 112)  448
block5b_project_conv[0][0]
__________________________________________________________________________________
__________________
block5b_drop (FixedDropout)     (None, 14, 14, 112)  0
block5b_project_bn[0][0]
__________________________________________________________________________________
__________________
```

```
block5b_add (Add)               (None, 14, 14, 112)  0
block5b_drop[0][0]
block5a_project_bn[0][0]
__________________________________________________________________________________
__________________
block5c_expand_conv (Conv2D)    (None, 14, 14, 672)  75264
block5b_add[0][0]
__________________________________________________________________________________
__________________
block5c_expand_bn (BatchNormali (None, 14, 14, 672)  2688
block5c_expand_conv[0][0]
__________________________________________________________________________________
__________________
block5c_expand_activation (Acti (None, 14, 14, 672)  0
block5c_expand_bn[0][0]
__________________________________________________________________________________
__________________
block5c_dwconv (DepthwiseConv2D (None, 14, 14, 672)  16800
block5c_expand_activation[0][0]
__________________________________________________________________________________
__________________
block5c_bn (BatchNormalization) (None, 14, 14, 672)  2688
block5c_dwconv[0][0]
__________________________________________________________________________________
__________________
block5c_activation (Activation) (None, 14, 14, 672)  0
block5c_bn[0][0]
__________________________________________________________________________________
__________________
block5c_se_squeeze (GlobalAvera (None, 672)          0
block5c_activation[0][0]
__________________________________________________________________________________
__________________
block5c_se_reshape (Reshape)    (None, 1, 1, 672)    0
block5c_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block5c_se_reduce (Conv2D)      (None, 1, 1, 28)     18844
block5c_se_reshape[0][0]
__________________________________________________________________________________
__________________
block5c_se_expand (Conv2D)      (None, 1, 1, 672)    19488
block5c_se_reduce[0][0]
__________________________________________________________________________________
__________________
block5c_se_excite (Multiply)    (None, 14, 14, 672)  0
block5c_activation[0][0]
block5c_se_expand[0][0]
```

```
----------------------------------------------------------------------------------------------------
block5c_project_conv (Conv2D)   (None, 14, 14, 112)  75264
block5c_se_excite[0][0]
----------------------------------------------------------------------------------------------------
block5c_project_bn (BatchNormal (None, 14, 14, 112)  448
block5c_project_conv[0][0]
----------------------------------------------------------------------------------------------------
block5c_drop (FixedDropout)     (None, 14, 14, 112)  0
block5c_project_bn[0][0]
----------------------------------------------------------------------------------------------------
block5c_add (Add)               (None, 14, 14, 112)  0
block5c_drop[0][0]
block5b_add[0][0]
----------------------------------------------------------------------------------------------------
block6a_expand_conv (Conv2D)    (None, 14, 14, 672)  75264
block5c_add[0][0]
----------------------------------------------------------------------------------------------------
block6a_expand_bn (BatchNormali (None, 14, 14, 672)  2688
block6a_expand_conv[0][0]
----------------------------------------------------------------------------------------------------
block6a_expand_activation (Acti (None, 14, 14, 672)  0
block6a_expand_bn[0][0]
----------------------------------------------------------------------------------------------------
block6a_dwconv (DepthwiseConv2D (None, 7, 7, 672)    16800
block6a_expand_activation[0][0]
----------------------------------------------------------------------------------------------------
block6a_bn (BatchNormalization) (None, 7, 7, 672)    2688
block6a_dwconv[0][0]
----------------------------------------------------------------------------------------------------
block6a_activation (Activation) (None, 7, 7, 672)    0
block6a_bn[0][0]
----------------------------------------------------------------------------------------------------
block6a_se_squeeze (GlobalAvera (None, 672)          0
block6a_activation[0][0]
----------------------------------------------------------------------------------------------------
block6a_se_reshape (Reshape)    (None, 1, 1, 672)    0
```

```
block6a_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block6a_se_reduce (Conv2D)      (None, 1, 1, 28)     18844
block6a_se_reshape[0][0]
__________________________________________________________________________________
__________________
block6a_se_expand (Conv2D)      (None, 1, 1, 672)    19488
block6a_se_reduce[0][0]
__________________________________________________________________________________
__________________
block6a_se_excite (Multiply)    (None, 7, 7, 672)    0
block6a_activation[0][0]
block6a_se_expand[0][0]
__________________________________________________________________________________
__________________
block6a_project_conv (Conv2D)   (None, 7, 7, 192)    129024
block6a_se_excite[0][0]
__________________________________________________________________________________
__________________
block6a_project_bn (BatchNormal (None, 7, 7, 192)    768
block6a_project_conv[0][0]
__________________________________________________________________________________
__________________
block6b_expand_conv (Conv2D)    (None, 7, 7, 1152)   221184
block6a_project_bn[0][0]
__________________________________________________________________________________
__________________
block6b_expand_bn (BatchNormali (None, 7, 7, 1152)   4608
block6b_expand_conv[0][0]
__________________________________________________________________________________
__________________
block6b_expand_activation (Acti (None, 7, 7, 1152)   0
block6b_expand_bn[0][0]
__________________________________________________________________________________
__________________
block6b_dwconv (DepthwiseConv2D (None, 7, 7, 1152)   28800
block6b_expand_activation[0][0]
__________________________________________________________________________________
__________________
block6b_bn (BatchNormalization) (None, 7, 7, 1152)   4608
block6b_dwconv[0][0]
__________________________________________________________________________________
__________________
block6b_activation (Activation) (None, 7, 7, 1152)   0
block6b_bn[0][0]
__________________________________________________________________________________
__________________
```

```
block6b_se_squeeze (GlobalAvera (None, 1152)         0
block6b_activation[0][0]
__________________________________________________________________________________
__________________
block6b_se_reshape (Reshape)    (None, 1, 1, 1152)   0
block6b_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block6b_se_reduce (Conv2D)      (None, 1, 1, 48)     55344
block6b_se_reshape[0][0]
__________________________________________________________________________________
__________________
block6b_se_expand (Conv2D)      (None, 1, 1, 1152)   56448
block6b_se_reduce[0][0]
__________________________________________________________________________________
__________________
block6b_se_excite (Multiply)    (None, 7, 7, 1152)   0
block6b_activation[0][0]
block6b_se_expand[0][0]
__________________________________________________________________________________
__________________
block6b_project_conv (Conv2D)   (None, 7, 7, 192)    221184
block6b_se_excite[0][0]
__________________________________________________________________________________
__________________
block6b_project_bn (BatchNormal (None, 7, 7, 192)    768
block6b_project_conv[0][0]
__________________________________________________________________________________
__________________
block6b_drop (FixedDropout)     (None, 7, 7, 192)    0
block6b_project_bn[0][0]
__________________________________________________________________________________
__________________
block6b_add (Add)               (None, 7, 7, 192)    0
block6b_drop[0][0]
block6a_project_bn[0][0]
__________________________________________________________________________________
__________________
block6c_expand_conv (Conv2D)    (None, 7, 7, 1152)   221184
block6b_add[0][0]
__________________________________________________________________________________
__________________
block6c_expand_bn (BatchNormali (None, 7, 7, 1152)   4608
block6c_expand_conv[0][0]
__________________________________________________________________________________
__________________
block6c_expand_activation (Acti (None, 7, 7, 1152)   0
block6c_expand_bn[0][0]
```

```
------------------------------------------------------------------------------
------------------
block6c_dwconv (DepthwiseConv2D (None, 7, 7, 1152)   28800
block6c_expand_activation[0][0]
------------------------------------------------------------------------------
------------------
block6c_bn (BatchNormalization) (None, 7, 7, 1152)   4608
block6c_dwconv[0][0]
------------------------------------------------------------------------------
------------------
block6c_activation (Activation) (None, 7, 7, 1152)   0
block6c_bn[0][0]
------------------------------------------------------------------------------
------------------
block6c_se_squeeze (GlobalAvera (None, 1152)         0
block6c_activation[0][0]
------------------------------------------------------------------------------
------------------
block6c_se_reshape (Reshape)    (None, 1, 1, 1152)   0
block6c_se_squeeze[0][0]
------------------------------------------------------------------------------
------------------
block6c_se_reduce (Conv2D)      (None, 1, 1, 48)     55344
block6c_se_reshape[0][0]
------------------------------------------------------------------------------
------------------
block6c_se_expand (Conv2D)      (None, 1, 1, 1152)   56448
block6c_se_reduce[0][0]
------------------------------------------------------------------------------
------------------
block6c_se_excite (Multiply)    (None, 7, 7, 1152)   0
block6c_activation[0][0]
block6c_se_expand[0][0]
------------------------------------------------------------------------------
------------------
block6c_project_conv (Conv2D)   (None, 7, 7, 192)    221184
block6c_se_excite[0][0]
------------------------------------------------------------------------------
------------------
block6c_project_bn (BatchNormal (None, 7, 7, 192)    768
block6c_project_conv[0][0]
------------------------------------------------------------------------------
------------------
block6c_drop (FixedDropout)     (None, 7, 7, 192)    0
block6c_project_bn[0][0]
------------------------------------------------------------------------------
------------------
block6c_add (Add)               (None, 7, 7, 192)    0
```

```
block6c_drop[0][0]
block6b_add[0][0]
__________________________________________________________________________________
__________________
block6d_expand_conv (Conv2D)    (None, 7, 7, 1152)   221184
block6c_add[0][0]
__________________________________________________________________________________
__________________
block6d_expand_bn (BatchNormali (None, 7, 7, 1152)   4608
block6d_expand_conv[0][0]
__________________________________________________________________________________
__________________
block6d_expand_activation (Acti (None, 7, 7, 1152)   0
block6d_expand_bn[0][0]
__________________________________________________________________________________
__________________
block6d_dwconv (DepthwiseConv2D (None, 7, 7, 1152)   28800
block6d_expand_activation[0][0]
__________________________________________________________________________________
__________________
block6d_bn (BatchNormalization) (None, 7, 7, 1152)   4608
block6d_dwconv[0][0]
__________________________________________________________________________________
__________________
block6d_activation (Activation) (None, 7, 7, 1152)   0
block6d_bn[0][0]
__________________________________________________________________________________
__________________
block6d_se_squeeze (GlobalAvera (None, 1152)         0
block6d_activation[0][0]
__________________________________________________________________________________
__________________
block6d_se_reshape (Reshape)    (None, 1, 1, 1152)   0
block6d_se_squeeze[0][0]
__________________________________________________________________________________
__________________
block6d_se_reduce (Conv2D)      (None, 1, 1, 48)     55344
block6d_se_reshape[0][0]
__________________________________________________________________________________
__________________
block6d_se_expand (Conv2D)      (None, 1, 1, 1152)   56448
block6d_se_reduce[0][0]
__________________________________________________________________________________
__________________
block6d_se_excite (Multiply)    (None, 7, 7, 1152)   0
block6d_activation[0][0]
block6d_se_expand[0][0]
__________________________________________________________________________________
```

```
------------------
block6d_project_conv (Conv2D)   (None, 7, 7, 192)    221184
block6d_se_excite[0][0]
--------------------------------------------------------------------------------
------------------
block6d_project_bn (BatchNormal (None, 7, 7, 192)    768
block6d_project_conv[0][0]
--------------------------------------------------------------------------------
------------------
block6d_drop (FixedDropout)     (None, 7, 7, 192)    0
block6d_project_bn[0][0]
--------------------------------------------------------------------------------
------------------
block6d_add (Add)               (None, 7, 7, 192)    0
block6d_drop[0][0]
block6c_add[0][0]
--------------------------------------------------------------------------------
------------------
block7a_expand_conv (Conv2D)    (None, 7, 7, 1152)   221184
block6d_add[0][0]
--------------------------------------------------------------------------------
------------------
block7a_expand_bn (BatchNormali (None, 7, 7, 1152)   4608
block7a_expand_conv[0][0]
--------------------------------------------------------------------------------
------------------
block7a_expand_activation (Acti (None, 7, 7, 1152)   0
block7a_expand_bn[0][0]
--------------------------------------------------------------------------------
------------------
block7a_dwconv (DepthwiseConv2D (None, 7, 7, 1152)   10368
block7a_expand_activation[0][0]
--------------------------------------------------------------------------------
------------------
block7a_bn (BatchNormalization) (None, 7, 7, 1152)   4608
block7a_dwconv[0][0]
--------------------------------------------------------------------------------
------------------
block7a_activation (Activation) (None, 7, 7, 1152)   0
block7a_bn[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_squeeze (GlobalAvera (None, 1152)         0
block7a_activation[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_reshape (Reshape)    (None, 1, 1, 1152)   0
block7a_se_squeeze[0][0]
```

```
__________________________________________________________________________________
__________________
block7a_se_reduce (Conv2D)      (None, 1, 1, 48)     55344
block7a_se_reshape[0][0]
__________________________________________________________________________________
__________________
block7a_se_expand (Conv2D)      (None, 1, 1, 1152)   56448
block7a_se_reduce[0][0]
__________________________________________________________________________________
__________________
block7a_se_excite (Multiply)    (None, 7, 7, 1152)   0
block7a_activation[0][0]
block7a_se_expand[0][0]
__________________________________________________________________________________
__________________
block7a_project_conv (Conv2D)   (None, 7, 7, 320)    368640
block7a_se_excite[0][0]
__________________________________________________________________________________
__________________
block7a_project_bn (BatchNormal (None, 7, 7, 320)    1280
block7a_project_conv[0][0]
__________________________________________________________________________________
__________________
top_conv (Conv2D)               (None, 7, 7, 1280)   409600
block7a_project_bn[0][0]
__________________________________________________________________________________
__________________
top_bn (BatchNormalization)     (None, 7, 7, 1280)   5120        top_conv[0][0]
__________________________________________________________________________________
__________________
top_activation (Activation)     (None, 7, 7, 1280)   0           top_bn[0][0]
==================================================================================
==================
Total params: 4,049,564
Trainable params: 4,007,548
Non-trainable params: 42,016
__________________________________________________________________________________
__________________
```

[57]:
```
results = model.
 ↪fit(training_set_tl,epochs=10,steps_per_epoch=5,validation_data=Validation_set_tl,validation_
```

```
Epoch 1/10
5/5 [==============================] - 340s 68s/step - loss: 1.1115 - accuracy:
0.3801 - val_loss: 1.1055 - val_accuracy: 0.3347
Epoch 2/10
5/5 [==============================] - 294s 59s/step - loss: 0.9697 - accuracy:
0.5375 - val_loss: 1.1288 - val_accuracy: 0.3028
```

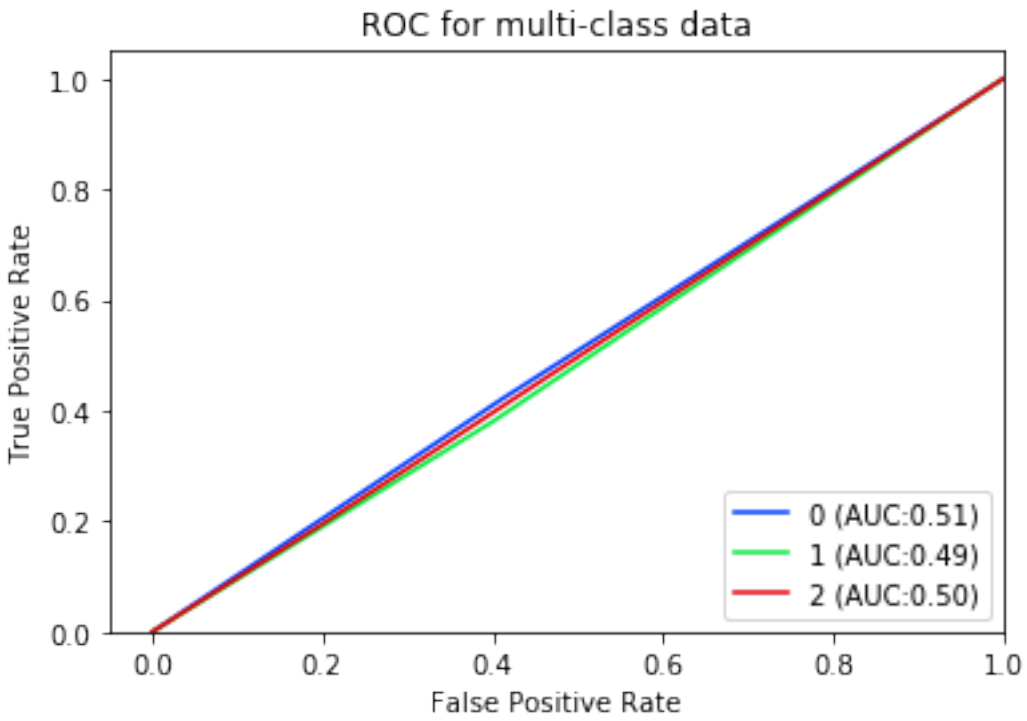
```
Epoch 3/10
5/5 [==============================] - 304s 61s/step - loss: 0.9270 - accuracy:
0.5375 - val_loss: 1.0700 - val_accuracy: 0.4188
Epoch 4/10
5/5 [==============================] - 315s 63s/step - loss: 0.8821 - accuracy:
0.6125 - val_loss: 1.0326 - val_accuracy: 0.4667
Epoch 5/10
5/5 [==============================] - 297s 59s/step - loss: 0.7768 - accuracy:
0.7042 - val_loss: 1.0606 - val_accuracy: 0.4601
Epoch 6/10
5/5 [==============================] - 286s 57s/step - loss: 0.7477 - accuracy:
0.7000 - val_loss: 0.9793 - val_accuracy: 0.5306
Epoch 7/10
5/5 [==============================] - 292s 58s/step - loss: 0.6435 - accuracy:
0.7421 - val_loss: 0.8627 - val_accuracy: 0.5627
Epoch 8/10
5/5 [==============================] - 280s 56s/step - loss: 0.6235 - accuracy:
0.7625 - val_loss: 0.9500 - val_accuracy: 0.5583
Epoch 9/10
5/5 [==============================] - 280s 56s/step - loss: 0.5295 - accuracy:
0.8000 - val_loss: 0.8870 - val_accuracy: 0.6011
Epoch 10/10
5/5 [==============================] - 260s 52s/step - loss: 0.5827 - accuracy:
0.7833 - val_loss: 0.8654 - val_accuracy: 0.5861
```

[59]:
```
from sklearn.metrics import classification_report
print(classification_report(pred_tl,test_set_tl.classes))
print(' covid19 = 0 , normal = 1, pneumonia = 2')
```

```
              precision    recall  f1-score   support

           0       0.42      0.34      0.38       124
           1       0.38      0.32      0.35       118
           2       0.19      0.33      0.24        58

    accuracy                           0.33       300
   macro avg       0.33      0.33      0.32       300
weighted avg       0.36      0.33      0.34       300

 covid19 = 0 , normal = 1, pneumonia = 2
```

[60]:
```
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import roc_auc_score
# set plot figure size
fig, c_ax = plt.subplots()

def multiclass_roc_auc_score(y_test, y_pred, average="macro"):
```

```
    lb = LabelBinarizer()
    lb.fit(y_test)
    y_test = lb.transform(y_test)
    y_pred = lb.transform(y_pred)
    all_labels= {0:'normal', 1:'pneumonia', 2:'covid'}

    for (idx, c_label) in enumerate(all_labels): # all_labels: no of the labels,␣
↪for ex. ['cat', 'dog', 'rat']
        fpr, tpr, thresholds = roc_curve(y_test[:,idx].astype(int), y_pred[:
↪,idx])
        c_ax.plot(fpr, tpr, label = '%s (AUC:%0.2f)'  % (c_label, auc(fpr, tpr)))
        colors = ['magenta', 'crimson', 'cyan']

    return roc_auc_score(y_test, y_pred, average=average)



multiclass_roc_auc_score(test_set_tl.classes, pred_tl)

plt.xlim([-0.05, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC for multi-class data')
plt.legend(loc="lower right")
plt.show()
```


ROC for multi-class data
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
0 (AUC:0.51)
1 (AUC:0.49)
2 (AUC:0.50)