Topic	IMAGE DATA PREPROCESSING & AUGMENTATION	
Class Description	**The student will learn to modify the image dataset before feeding the data for training the model.**	
Class	**PRO C111**	
Class time	**50 mins**	
Goal	• Learn to modify the images.	

Resources Required	• Teacher Resources:
	  ○ Laptop with internet connectivity
	  ○ Webcam
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Smartphone
	• Student Resources:
	  ○ Laptop with internet connectivity
	  ○ Webcam
	  ○ Earphones with mic
	  ○ Notebook and pen	

Class structure	**Warm-Up** **Teacher-led Activity 1** **Student-led Activity 1** **Wrap-Up**	**10 mins** **15 mins** **20 mins** **05 mins**
Credit	Teachable Machines by Google. Tensorflow by Google Brain team. Keras by Google Engineer Francois Chollet. Kaggle: Pneumothorax Dataset	
WARM-UP SESSION - 10 mins		

Teacher Starts Slideshow  **Slide 1 to 3** Refer to speaker notes and follow the instructions on each slide.	
Teacher Action	**Student Action**
Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** • Greet the student. • Revision of previous class activities. • Quizzes.	**ESR**: Hi, thanks! Yes I am excited about it! Click on the slide show tab and present the slides
WARM-UP QUIZ Click on In-Class Quiz	
Continue WARM-UP Session  **Slide 4 to 15**	
Following are the session deliverables: • Appreciate the student. • Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.	
Teacher Action	**Student Action**
In the previous class, we trained a machine model using Teachable Machine to classify images into different classes(categories). Now, we will begin to go a bit deeper into each stage of training the machine model for image classification. We will learn to:	

<table>
<tr><td>

- Process our own Image Dataset before feeding it into the machine for model training.
- Define our own model for training.
- Evaluate and Test if the model is able to classify the new image.

In today's class, we start with image dataset processing.

Are you excited?

Let's get started.

</td><td>

ESR: Yes.

</td></tr>
<tr><td colspan="2">

Teacher Ends Slideshow 

</td></tr>
<tr><td colspan="2">

TEACHER-LED ACTIVITY - 15 mins

</td></tr>
<tr><td colspan="2">

Teacher Initiates Screen Share

</td></tr>
<tr><td colspan="2">

ACTIVITY

- **Understanding Pneumothorax Disease.**
- **Image data preprocessing and augmentation using Keras.**

</td></tr>
<tr><td>

Teacher Action

</td><td>

Student Action

</td></tr>
<tr><td>

Need of Machine Learning in Healthcare:

Technology is advancing day by day and has drastically improved the way we live today.

Even though we have advanced so much in technology, we still have one major problem to solve - how to increase the usage of technology in the healthcare sector.

Before we can actually cure a disease, its proper diagnosis(finding out the disease) is very important. When

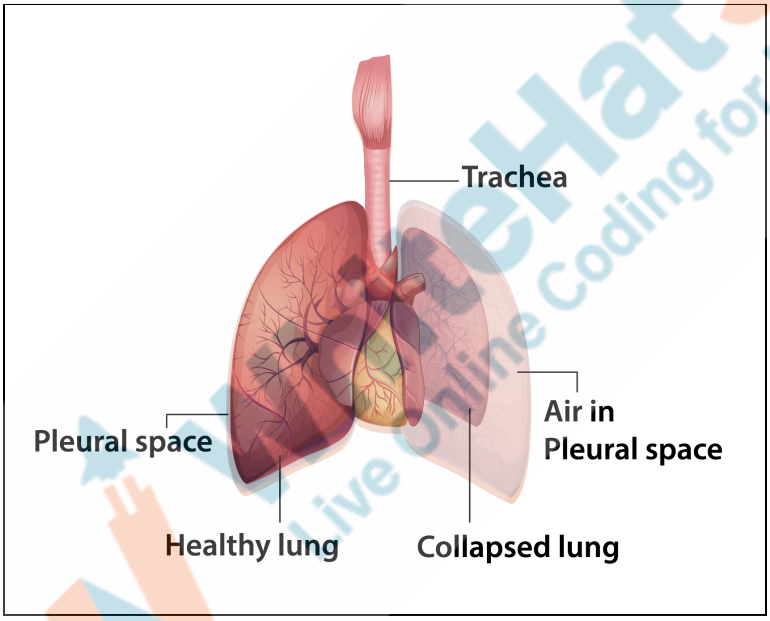
</td><td></td></tr>
</table>

doctors make wrong decisions about a disease, it might cause life-altering complications or prolong recovery of a patient. Misdiagnosis is a major issue, how can doctors improve the accuracy of detecting a disease? Making use of machine learning can help us achieve more accurate diagnosis of a disease. Can you tell me how can we do that? Well, one possible way is to collect patient's medical image data(images captured through X-Rays, CT Scans, Microscopic Cells images) who have been already diagnosed with a disease and patients with no disease, and then train a machine using those images. In the future if a doctor is not able to diagnose the disease by looking at the patient's test results, then probably a machine can predict a result, and this might help a patient recover a lot earlier! Healthcare sector is already embracing the idea of using machine learning to get better solutions to cure diseases. **Pneumothorax:** In today's class, we will start working with the Pneumothorax(pronunciation: nyu-moo-thor-aks) Image Dataset for detecting the X-Rays images with infection. Pneumothorax is a respiratory disease. This occurs when air gets filled up between the lungs and chest walls. The air pushes outside the lungs and makes it collapse.	**ESR:** Varied. **ESR**: Varied.

Open Teacher Activity 1 to show Pneumothorax animation.

Roughly 1% of the patients having COVID-19 develop Pneumothorax.

***Note**: At the time of writing this document, the world is suffering from coronavirus disease(COVID-19) pandemic. The virus that causes COVID-19 is mainly transmitted through droplets generated when an infected person coughs, sneezes, or exhales.*

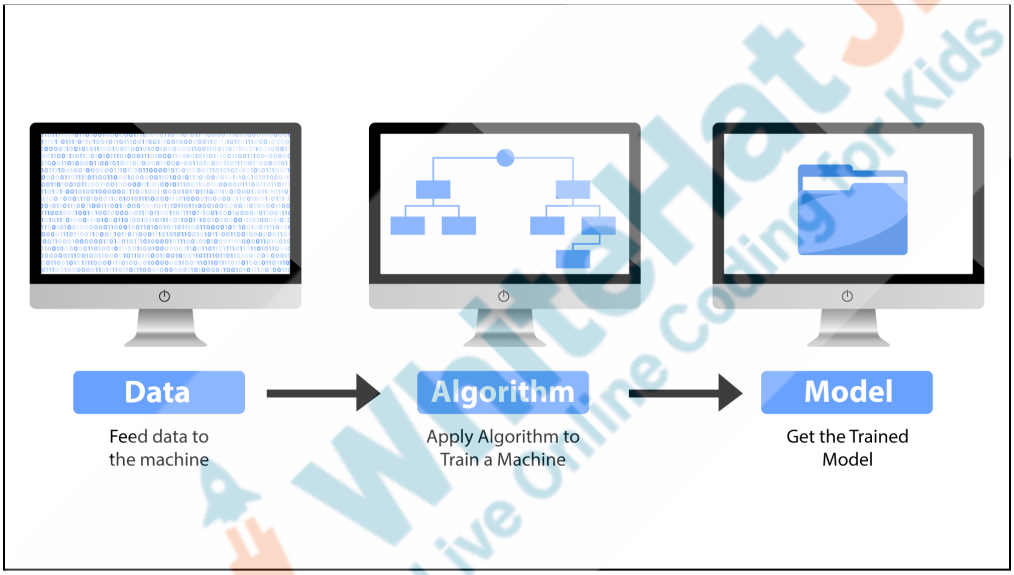


Now that we know what Pneumothorax is, let's take a look at one of the chest X-Ray images.

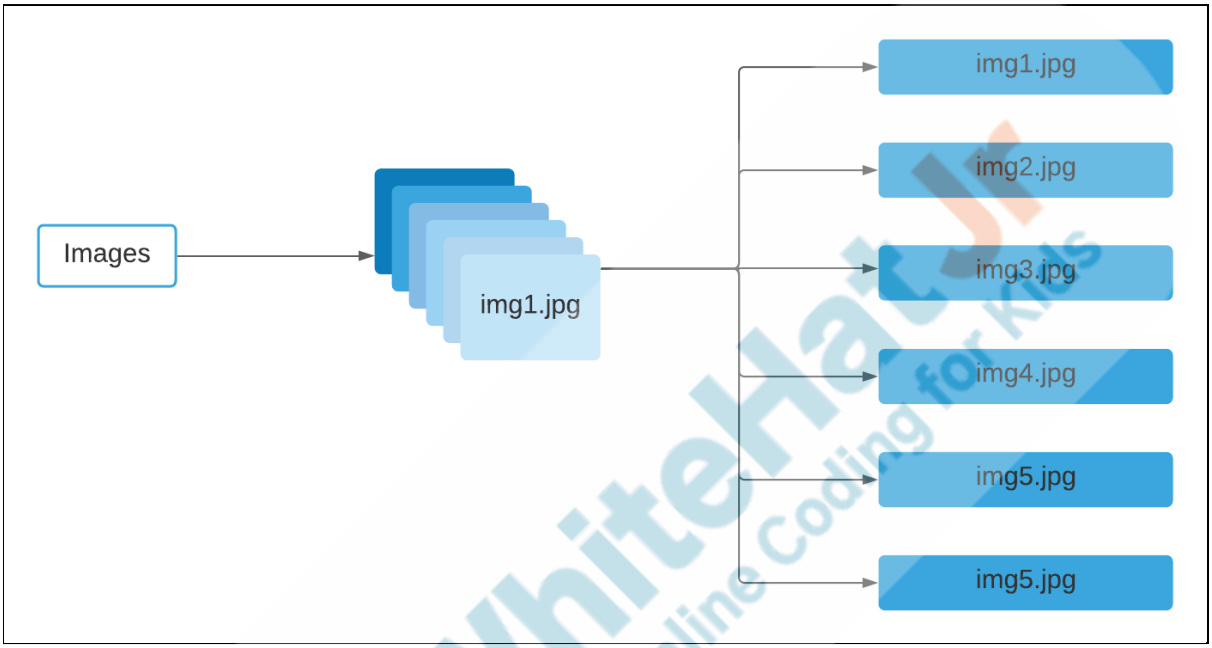
Open Teacher Activity 2 to show the chest X-Ray image.

I understand that it is difficult to read X-Ray images for people who are not qualified in the medical field, but can

you try to tell me if this image has Pneumothorax(collapsed lung(s)) or not? ***Note**: Ask to look at the image carefully and try to understand. Wait for the student to give responses.* The image has Pneumothorax(collapsed lung(s)). Although these X-Rays can be read by the doctors/qualified medical expert easily, sometimes it is difficult to understand whether the patient has Pneumothorax(collapsed lung(s)) or not. To tackle this problem, we will be using an image classification process in the machine to identify if the X-Ray image has Pneumothorax(collapsed lung(s)) infection or not.	**ESR:** Yes/No.
	
Remember in the previous class we talked about the process of machine learning to classify images, can you tell the steps involved in that process?	**ESR:** • Image input data with different categories. • Train the machine

Superb!	using the input data. • Use the trained information(i.e. model) to identify if the new image belongs to any trained category.
	
Since we want to detect if the patient's chest X-Ray images have Pneumothorax(collapsed lung(s)) or not, the first step is to have the patient's X-Ray images, with Pneumothorax infection and without Pneumothorax infection. Can you tell me how many classes(or categories) of image data the computer needs to be trained on?	**ESR**: We need 2 classes: • Images with Pneumothorax disease infection • Images without

Amazing! Using the Teachable Machine we could capture images directly to generate images of different classes, right? But here we need the X-Rays images, which need to be accessed from the websites which have datasets available for learning purposes. People can take the dataset and learn to build machine learning models which can be useful in solving health problems. Before we can train the model with the images dataset, remember the image dataset needs to be in proper format! This is known as **Image Data Preprocessing**. **Image Data Preprocessing includes:** 1. Mapping Each Image With A Label. 2. Converting All Image to Same Sized Array. 3. Augmenting Image Data. Do you remember what are the requirements that we need to keep in mind? **Labeling Images:** First, we need to make sure each image has their classes(or categories) defined. Let's understand this by an example: *Open Teacher Activity 3 to show the image.* Suppose we have all the X-Ray images to be used for training all together in one directory(folder).	Pneumothorax infection **ESR**: Yes. **ESR**: Varied.

Would you be able to tell if the X-Ray image has an infection or not if all the images are together?	**ESR**: No.
	
So what should we do? Great! We need to define class names for our image classification dataset and assign labels to each class. **Classes**: The classes are names of all categories for the model to be trained to identify the image. The class names are human-readable names which can be anything or in any language, like “infected” and “uninfected" to identify the disease in the image dataset.	**ESR**: We should tell the computer which image belongs to which class.

Labels: The labels are numbers assigned to each class, generally starting from 0(unless specified by us manually) till 1 less than the number of classes.

But why do you think we should assign label numbers to each class?

This is important because let's say the class name is "infected" which is in English, it can be "infiziert" in German, or "infectado" in Spanish.

If the model is trained using the English class names, people having the same image dataset with class names in their native language, the model will fail to understand other class names.

Training models in every language would waste a lot of time and will be very costly. Instead, adding labels of each class will help us to keep it universal in different parts of the world.

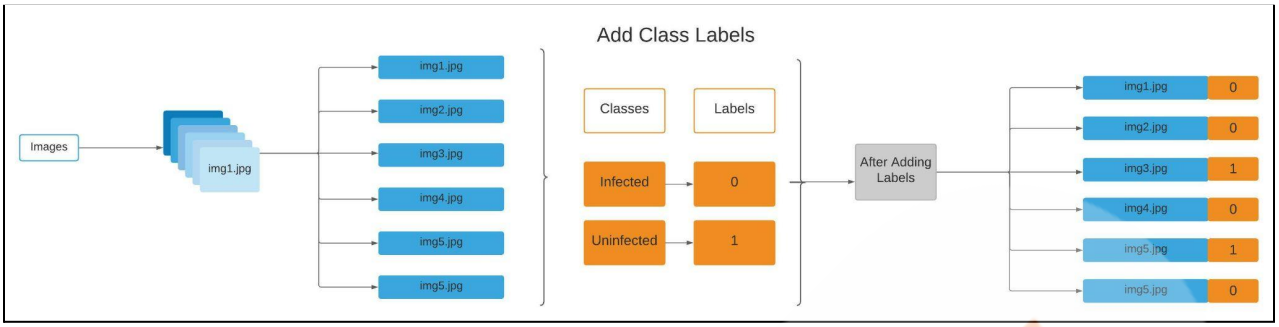
But adding labels every time would result in the same output, no matter what we choose as class name:

{"infected": 0, "uninfected" : 1}	→English
{"infiziert": 0, "nicht infiziert" : 1}	→German
{"infectado": 0, "no infectado" : 1}	→Spanish

After adding labels, each image can be classified as one part of one of the classes.

Open Teacher Activity 4 to show the image.

ESR: Varied.



We can use TensorFlow(& Keras) to add labels to each image easily.

Before we can start working with image datasets to add labels, let's quickly take a look at various Python machine learning libraries which are designed for using machine learning functionalities/algorithms.

Machine Learning Libraries:

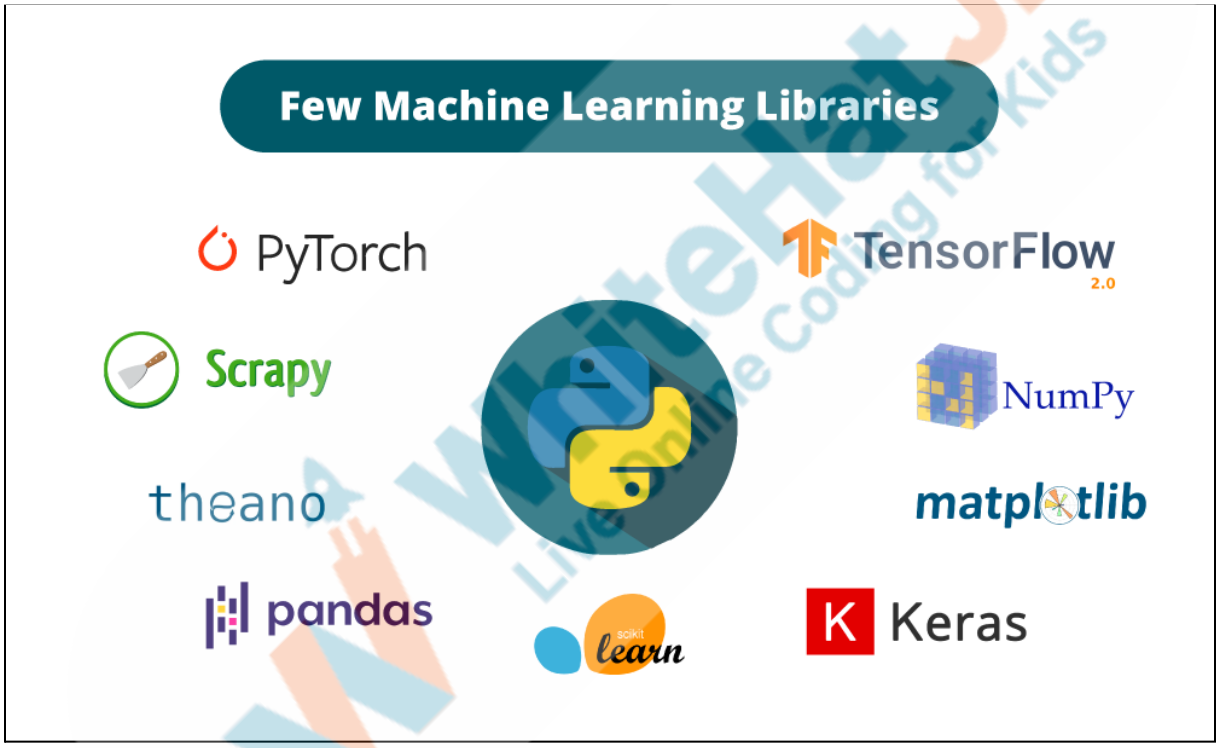
A few of the machines learning libraries available which can be used for various machines learning task are:

Open the links of 2-3 libraries to show it to the students.

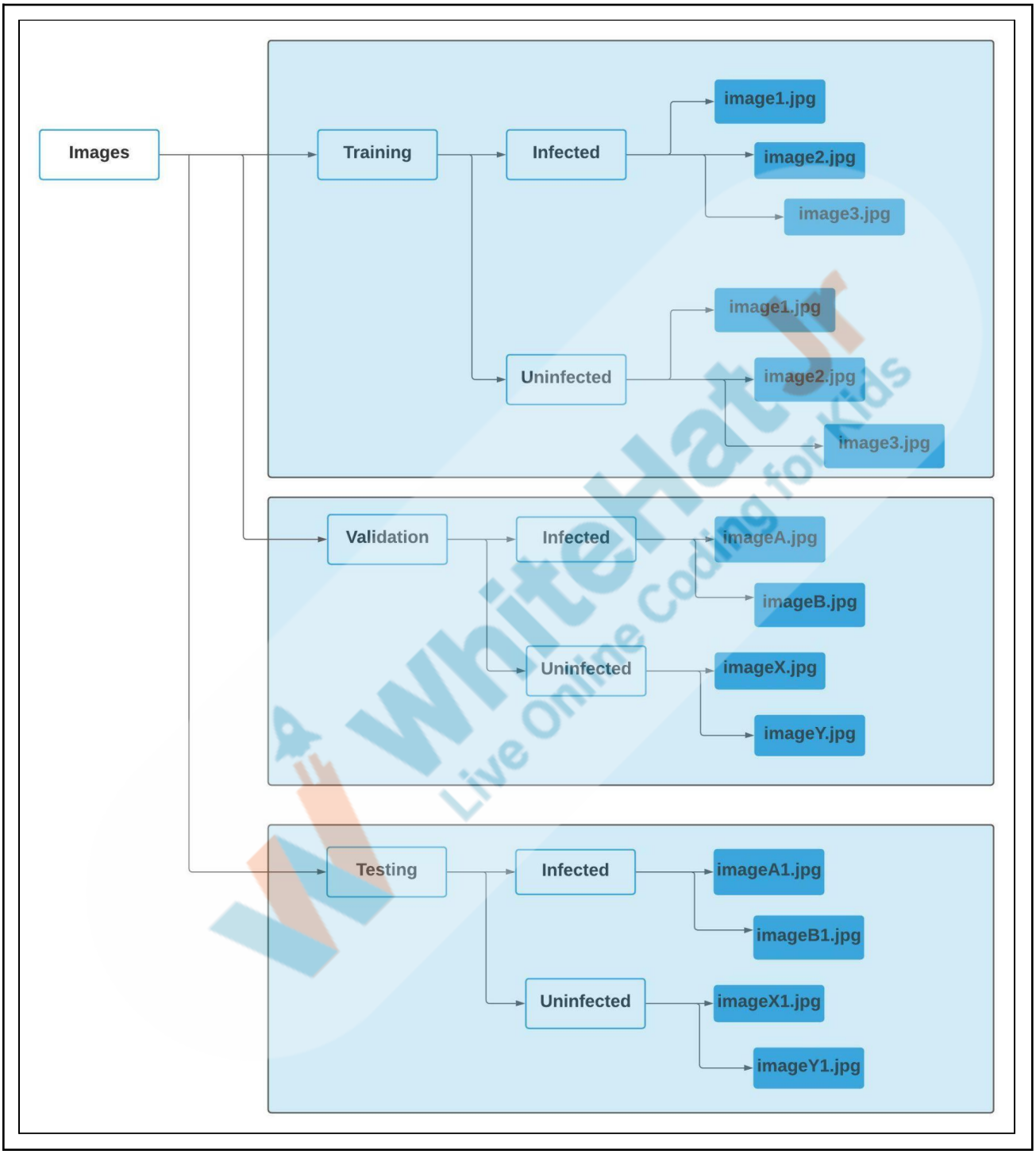
- PyTorch
- Scrapy
- Theano
- pandas
- TensorFlow
- scikit-learn
- Numpy
- Keras

We can choose the machine learning libraries based on our project requirement. We will not discuss every library in detail.

For the image classification process to detect the X-Ray image with Pneumothorax disease infection, we will be using TensorFlow and Keras(which is now a part of Tensorflow after the release of Tensorflow version 2.5) library.



Now to use TensorFlow to add labels on each X-Ray image in the image dataset we need to **store data into a sequence of subdirectories**. **Image Dataset Directory Structure:** Open Teacher Activity 5 to show the subdirectory structure to the student. Suppose if we have a master directory(folder) of the **Images** then we can subdivide it into "**Training**", "**Validation**" & "**Testing**" images sub-directories(sub-folder). And then the "**Training**" directories contain sub-directories(sub-folders) called "**Infected**" and "**Uninfected**" which contain appropriate images in the respective sub-directories. Similarly, the "**Validation"** & "**Testing**" directory also contains sub-directories(sub-folders) called "**Infected**" and "**Uninfected**" which contain appropriate images in the respective sub-directories. ***Note**: The directory and sub-directory names shown here are only for explanation purposes, which might differ from the code.* **Training:** Images in this directory will be used for the training of the data. **Validation:** Images in this directory will be used to validate the model training. The validation dataset allows us to see how well the data generalizes the classification. **Testing:** Images in this directory will be used to test how well the model is trained.	



Now let's quickly load the images. The image data is stored in the above.

Load Image Data:

1. Open Google Colaboratory [Teacher Activity 6](#).
2. Open [Teacher Activity 7](#) for Pneumothorax Image Dataset.
3. Clone the data into the Google Colab Note using the following command:

   !git clone <GitHub Repository URL>

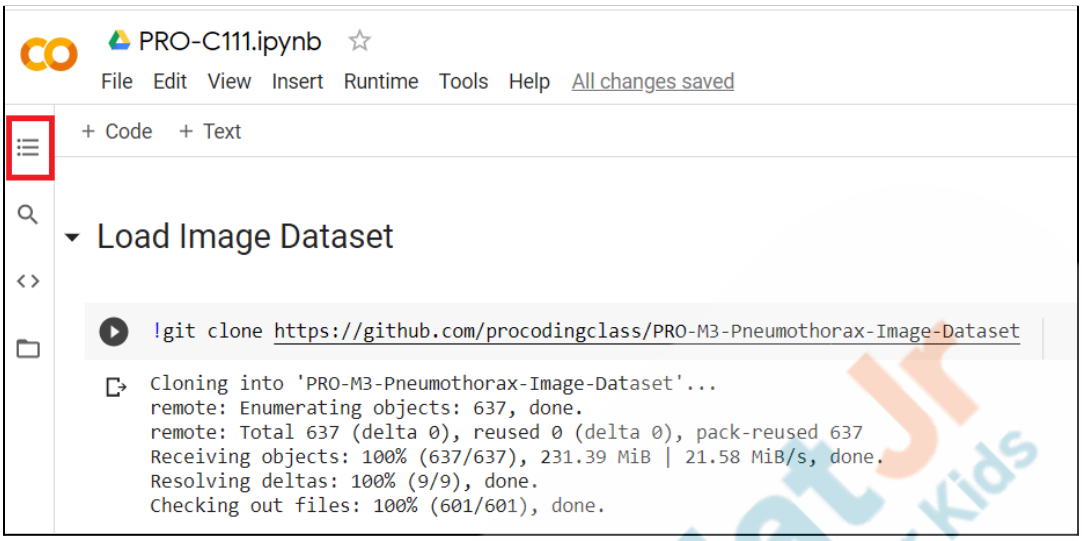
   The exclamatory sign is used to tell the computer to run the command as a Shell command instead of an iPython Notebook command.

   ***Note**: The data will be cloned into the current Google Colab Notebook while you are working.*

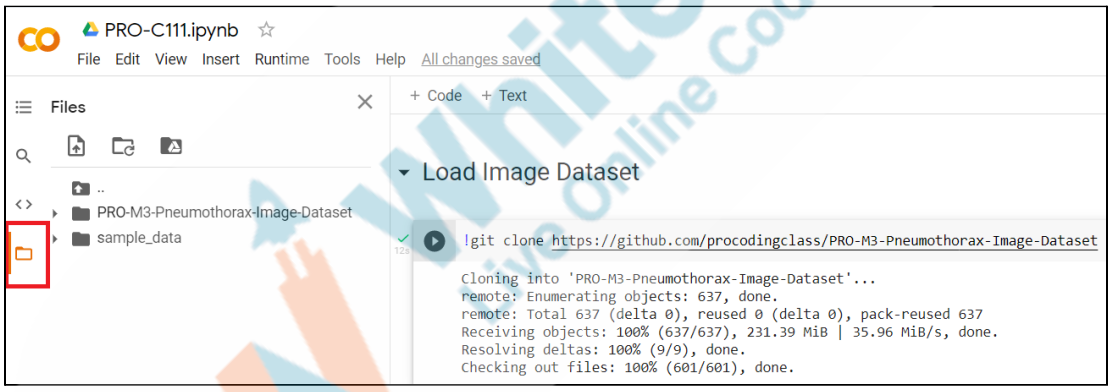
```
!git clone https://github.com/procodingclass/PRO-M3-Pneumothorax-Image-Dataset

Cloning into 'PRO-M3-Pneumothorax-Image-Dataset'...
remote: Enumerating objects: 426, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 426 (delta 0), reused 5 (delta 0), pack-reused 419
Receiving objects: 100% (426/426), 153.87 MiB | 34.65 MiB/s, done.
Resolving deltas: 100% (2/2), done.
Checking out files: 100% (401/401), done.
```

Once the data is cloned:

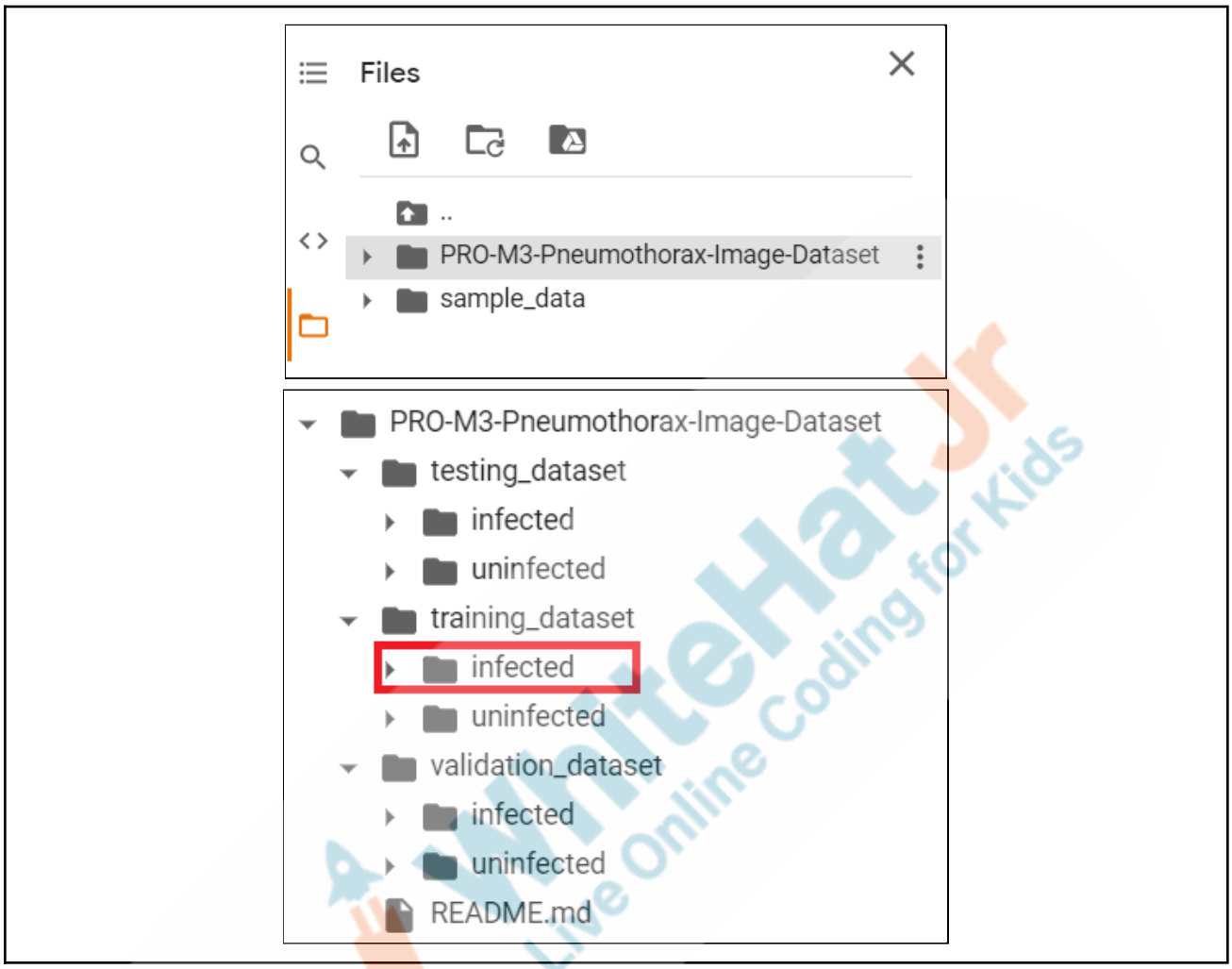
1. Open the "**Table of contents**" icon on the left side:

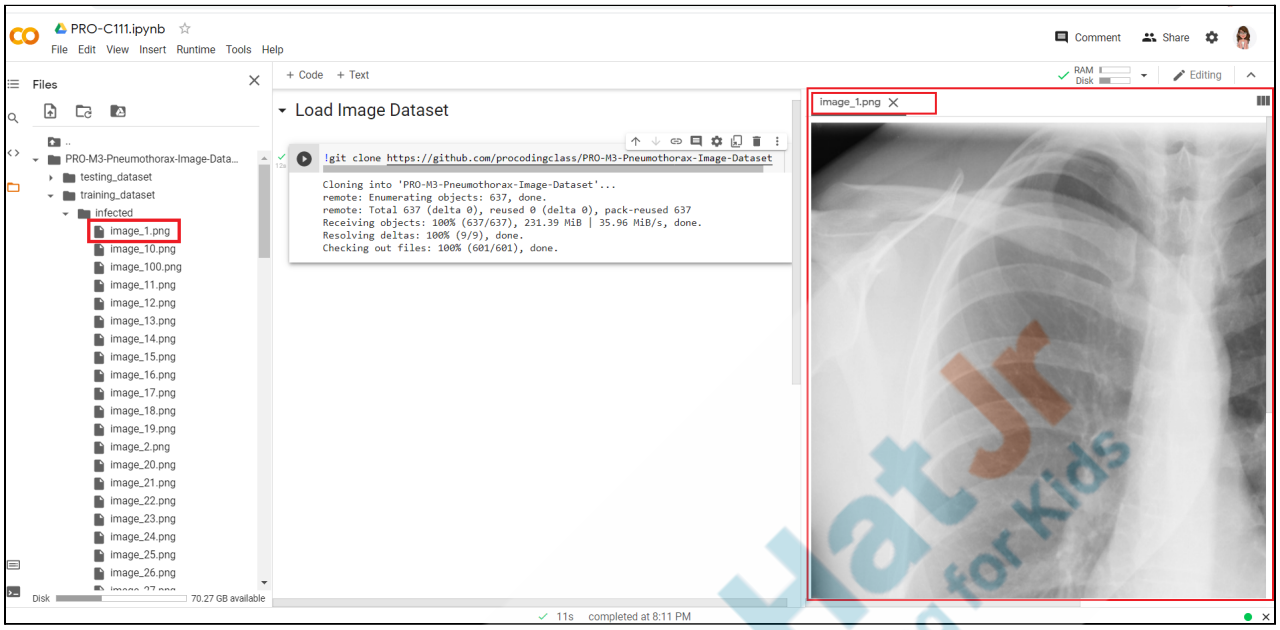


2. Open the "**File**" icon on the left side, to see the Pneumotorax Image Dataset.



3. Check the Dataset Directory Structure:
   - Double-click on the image inside *PRO-M3-Pneumothorax-Image-Dataset/training_dataset/infected/image_1.png* to see it on the right panel.





Visualize the Data:

Instead of checking the image on the right panel, we can show the image as an output of the code.

We are going to use a library called the matplotlib library in Python. This library can be used to plot numerical data as graphs.

Remember, images are also represented as pixel numbers?

We can also use the matplotlib library to plot images using pixel array values and show it as the output.

Let's try to plot the first image in the "**infected**" **trainng_dataset**:

1. Import **matplotlib** as **pyplot**
   The **pyplot** module is used to plot the graphs.

   Reference: Teacher Activity 8.

ESR: Yes.

2. Import **matplotlib.image** as **imread**
   The **image** module in **matplotlib** is used to work with images in python. We use **imread()** in the **image** method to read the images as an array.

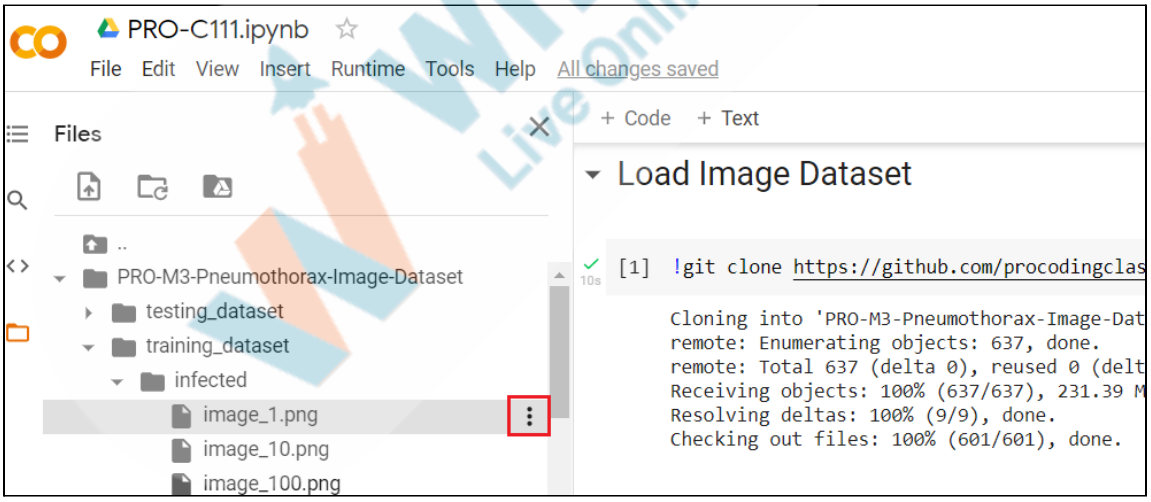
3. Take a variable **training_infected_image_dir**.

Visualise the Data

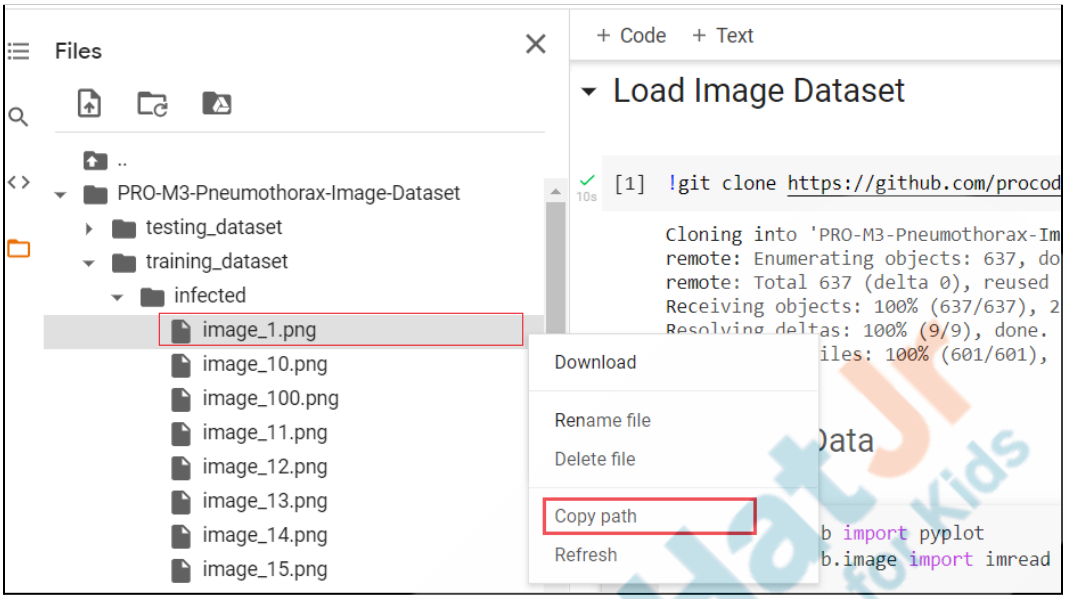
```
from matplotlib import pyplot
from matplotlib.image import imread

training_infected_image = ""
```

4. Copy the path of the first image in the training dataset infected directory.

Click on the 3 dots against the image name in the directory.



Click on "Copy path" to copy the path of the image name in the directory.



5. Assign the path as string to **traininng_infected_image_dir**.

```
from matplotlib import pyplot
from matplotlib.image import imread

training_infected_image = "/content/PRO-M3-Pneumothorax-Image-Dataset/training_dataset/infected/image_1.png"
```

6. Read the image as an array using the **imread()** method in the **image** module.

7. Add a title for the plot using the **title()** method in the **pyplot** module.

8. Display the image using **imshow()** in the **pyplot** module. The **imshow()** method is responsible for displaying it in the correct format.
   Reference: Teacher Activity 9.

9. Use **show()** to create the graph figure outline. The **show()** plots the function created by the **imshow()** method.

10. Run the code cell to see the output.

```
# load image pixels
image = imread(training_infected_image)

pyplot.title("Infected: Image 1")

# plot raw pixel data
pyplot.imshow(image)

# show the figure
pyplot.show()
```

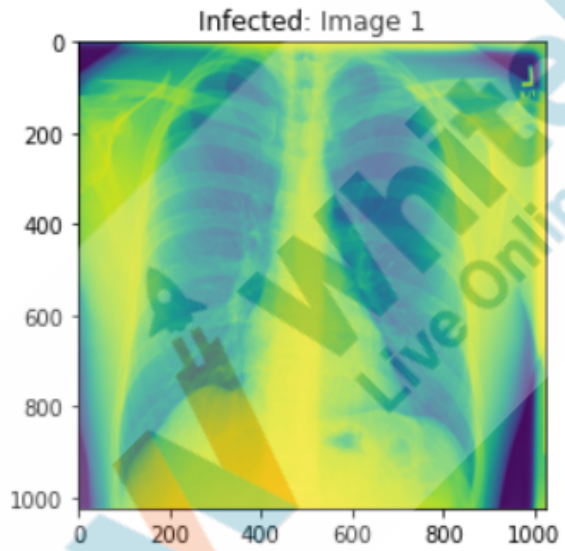
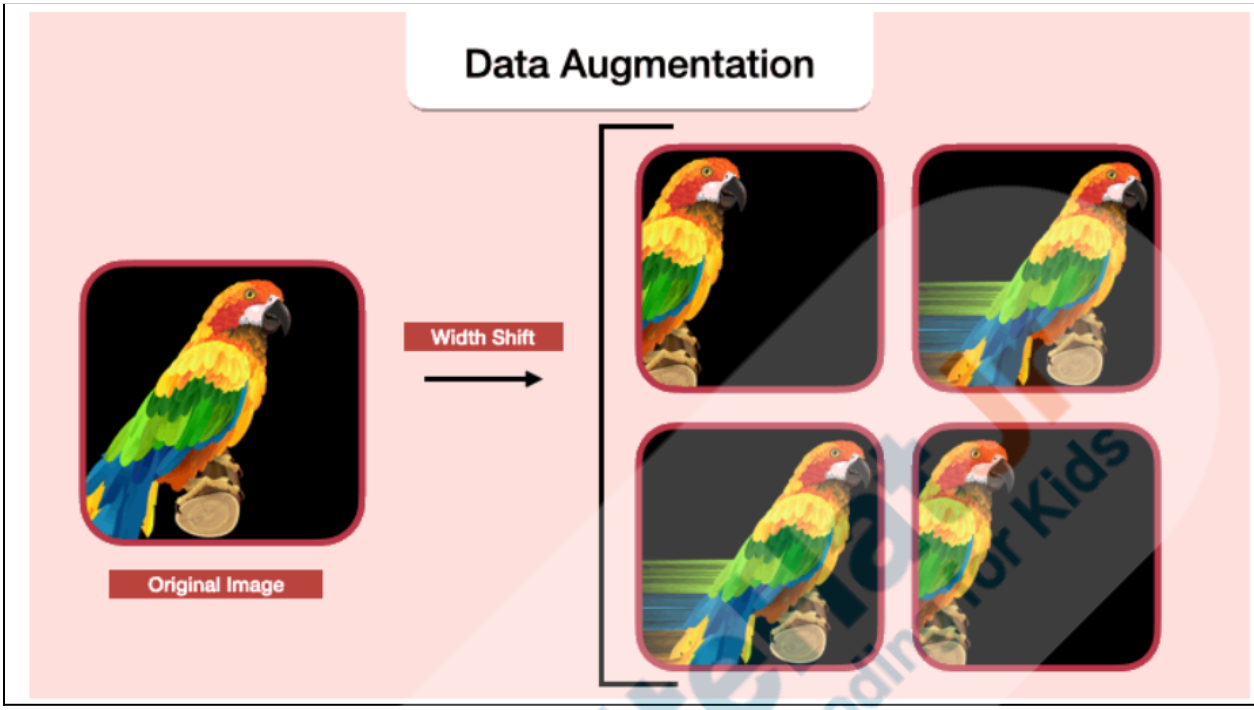
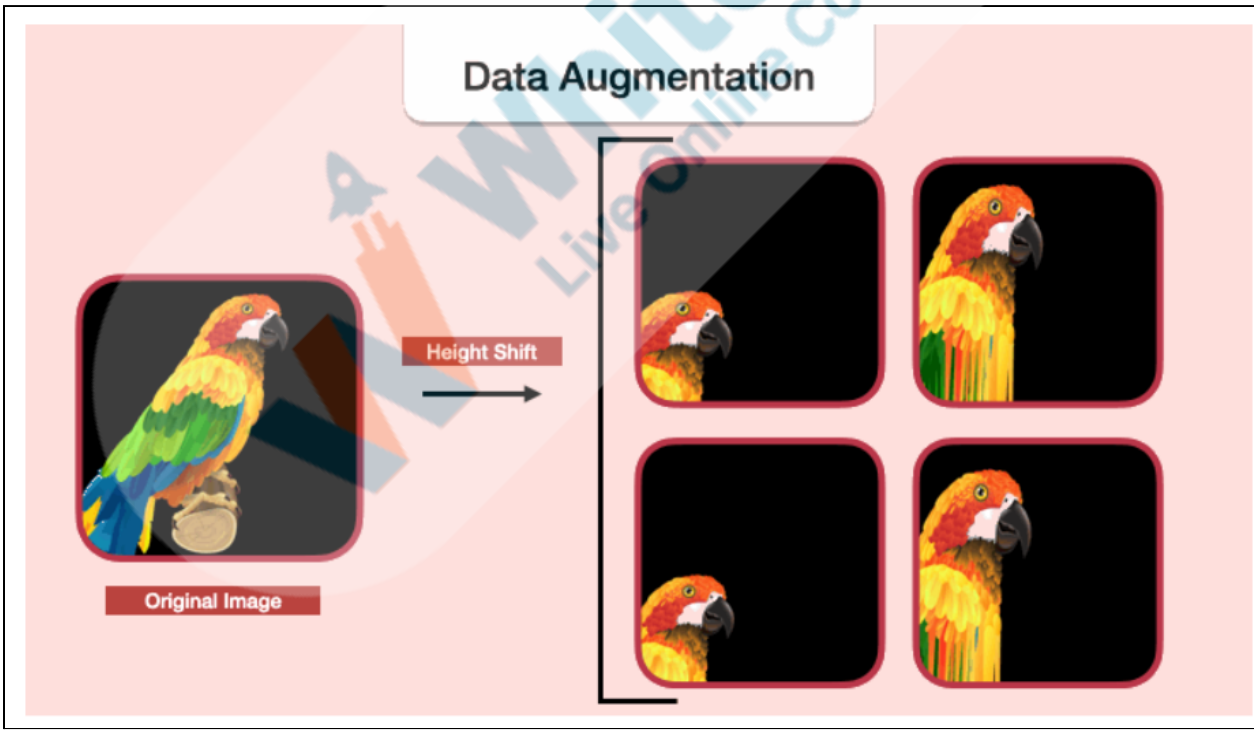


Image Data Augmentation:

Now that we have seen how our data looks, we will use data augmentation techniques such as resizing an image, rotating an image, horizontally flipping an image, vertically flipping an image etc.

Data augmentation techniques help us to virtually create images with more variety by generating the different versions of each image in the directory. We will be using a few of the image data augmentation techniques: *Open Teacher Activity 10 to show Data augmentation techniques to the student.* • **Rescale**: Increase or decrease the pixel values, keeping the size of the image fixed. • **Rotation**: Rotate the image in degrees. • **Width Shift**: Shift the image along its width(left to right or vice-versa). • **Height Shift**: Shift the image along its height(upside down or vice-versa) • **Horizontal Flip**: Flipping the image horizontally. • **Vertical Flip**: Flipping the image vertically. • **Zoom**: Magnifying in or out the image.	





Training a model on a wider variety of images helps the model increase the accuracy to identify the complex and different images more accurately.

We are going to use the Keras **ImageDataGenerator** Class to apply data augmentation techniques(like rotation, flipping, rescaling etc.).

Reference: Teacher Activity 11.

Rotation Augmentation:

Let's try to apply rotation to an image:

1. Import **tensorflow** library.
2. Import **ImageDataGenerator** Class form **keras** image preprocessing library

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

3. Initialize **ImageDataGenerator** class by creating an object variable named **data_generator** and passing **rotation_range** parameter value as **90°** (possible values can be 0 to 90 degrees) and **fill_mode** parameter value with "**nearest**", that fills all the pixels of the image with its nearby pixels.

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Random Rotation using Image Data Generator
training_data_generator = ImageDataGenerator(rotation_range=90, fill_mode='nearest')
```

4. Take a variable, **training_image_directory**, and assign the path value of the "**taining_dataset**" directory. (This directory should have the images divided into subdirectories "**infected**" and "**uninfected**"). Note that, this time, we are not providing the path to the exact image.

5. Call the **flow_from_directory()** method of the **ImageDataGenerator** class using the **data_generator** object variable of **ImageDataGenerator** class and assign the value to a variable called **training_augmented_images**. We need to pass the **training_image_directory** and the **target_size** of the image, which will be a new size of augmented images.

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Random Rotation using Image Data Generator
training_data_generator = ImageDataGenerator(rotation_range=90, fill_mode='nearest')

# Image Directory
training_image_directory = "/content/PRO-M3-Pneumothorax-Image-Dataset/training_dataset"

# Generate Randomly Rotated Image files
training_augmented_images = training_data_generator.flow_from_directory(training_image_directory,target_size=(180,180))
```

```
Found 200 images belonging to 2 classes.
```

This will randomly(without any order) generate a large amount of the augmented images(as a 4D array)for all the images present in the "**training_dataset**" directory.

Visualize the Image After Rotation Augmentation:

To see how the images will look after applying "Rotation Augmentation", we can again use matplotlib library and plot 3-4 images to understand the effect of "Rotation Augmentation" on the images:

1. Import the **pyplot** module in the **matplotlib** library.
2. Import the **imread** method from the **image** module in the **matplotlib** library.
3. Define a **for-in** loop for a range of **4** numbers. This will help us to plot 4 plots inside one main plot.

```
from matplotlib import pyplot
from matplotlib.image import imread

for i in range(4):
```

4. Loop through the range of 4 numbers:
    a. Define sub figures/plots using the **subplot()** method.

We can use the **subplot()** method to plot multiple plots/figures inside one main plot/figure.

Syntax:

```
pyplot.subplot(nrows, ncols, index)
```

nrows : It is the number of rows needed in the main plot.

ncols : It is the number of columns needed in the main plot.

index : It the index of each subplot starting from 1

Reference: *Teacher Activity 12*

```
from matplotlib import pyplot
from matplotlib.image import imread

for i in range(4):

  # Define a subplot
  pyplot.subplot(2, 2, i+1)
```

b. Take a variable called, **batch**, use the **next()** method to get the next image from the **training_augmented_images** variable and assign it to **batch**.

The **next()** method in Python is used to get the next value in lists or any other data types with multiple records in Python.

```
from matplotlib import pyplot
from matplotlib.image import imread

for i in range(4):

  # Define a subplot
  pyplot.subplot(2, 2, i+1)

  # Generate batch of images
  batch = training_augmented_images.next()
```

c. Access the Image Pixel Array and Convert the type of the image to an unsigned integer:

Augmented images are as 4D arrays. We can access the image pixel arrays using **batch[0][0]**

After augmentation, the image pixel array values get converted to with different pixel values which have some negative values.

The **imshow()** method in the **pyplot** module can accept arrays with only positive values.

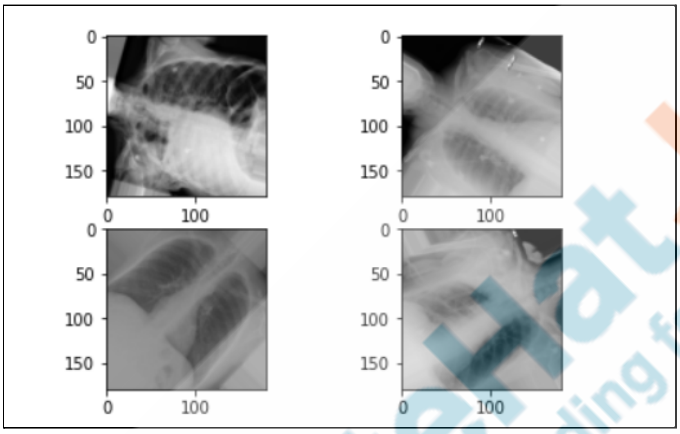
We can use the **astype()** method with value **'uint8'** in Python to convert the type of the array to unsigned integer.

```
for i in range(4):

  # Define a subplot
  pyplot.subplot(2, 2, i+1)

  # Generate batch of images
  batch = training_augmented_images.next()

  # Convert to unsigned integers for viewing
  image = batch[0][0].astype('uint8')
```

d. Display the image using **imshow()** in the **pyplot** module. The **imshow()** method is responsible for processing images and displaying it in the correct format.

5. Plot the whole figure(with the 4 sub figures/plots). Note that, it should be printed outside the loop, as this the whole plot with multiple plots inside it.

```
from matplotlib import pyplot
from matplotlib.image import imread

for i in range(4):

  # Define a subplot
  pyplot.subplot(2, 2, i+1)

  # Generate batch of images
  batch = training_augmented_images.next()

  # Convert to unsigned integers for viewing
  image = batch[0][0].astype('uint8')

  # Plot raw pixel data
  pyplot.imshow(image)

# Plot the main figure(with 4 sub figures)
pyplot.show()
```

Run the code cell to test the output. We can see the images are rotated by 90°. ***Note**: The images are randomly picked up from the augmented images. The set of images in the output may vary.*	
	
That was interesting! We could see how the images have augmented that helps to train the model with more accuracy. Now you will try to augment images using both rotation & width shift augmentation together and see the images. Are you excited?	**ESR:** Yes.
Teacher Stops Screen Share	
Teacher Starts Slideshow **Slide 16 to 17** Refer to speaker notes and follow the instructions on each slide.	
We have one more class challenge for you. Can you solve it?	

Let's try. I will guide you through it.	
Teacher Ends Slideshow 	
STUDENT-LED ACTIVITY - 20 mins	
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start Screen Share.** • **The teacher gets into Fullscreen.**	
ACTIVITY • **Image data preprocessing and augmentation using Keras.**	
Teacher Action	**Student Action**
Guide the student to open the colab notebook and make a copy of the boilerplate code using Student Activity 1. Run all the cells to load the dataset and update the code cells output.	
Rotation & Width Shift Augmentation: *Guide the student to:* 1. *Create an object of the ImageDataGenerator Class using **rotation_range**, **width_shift_range** and fill_mode parameter.* a. *The value of **width_shift_range** is given in a decimal number which represents the percentage shift in the image along the width of the image.* 2. *Call the flow_from_directory method using the object.* *For example: width_shift_range= 0.3 will shift the image*	

along the width by 30%(0.3X100)

***Note**: Follow the teacher's activity to explain the rest of the code.*

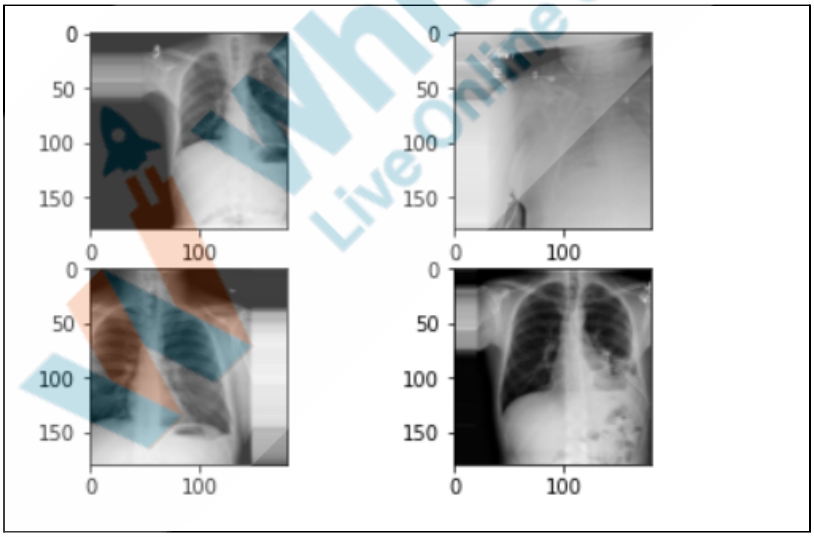
Random Rotations & Width Shift

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Random Roation & Width Shift using Image Data Generator
training_data_generator = ImageDataGenerator(
    rotation_range=90,
    width_shift_range=0.3,
    fill_mode='nearest')


# Image Directory
training_image_directory = "/content/PRO-M3-Pneumothorax-Image-Dataset/training_dataset"

# Generate Randomly Rotated Image files
training_augmented_images = training_data_generator.flow_from_directory(
    training_image_directory,
    target_size=(180,180))
```

Guide the student to visualize the augmented images.

***Note**: Follow the teacher's activity to explain the code.*

```
from matplotlib import pyplot
from matplotlib.image import imread

for i in range(4):

  # define subplot
  pyplot.subplot(2, 2, i+1)

  # generate batch of images
  batch = training_augmented_images.next()

  # convert to unsigned integers for viewing
  image = batch[0][0].astype('uint8')

  # plot raw pixel data
  pyplot.imshow(image)


pyplot.show()
```

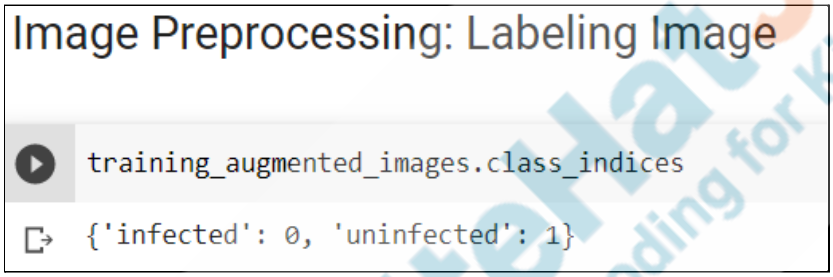


Remember, we also talked about image labeling? The **ImageDataGenerator** class makes it very easy to label each image stored in their respective directory. It	**ESR:** Yes.

automatically generates the labels for the class names(considering the names in alphabetical order) starting from 0.

We can check the labels to our classes “infected” and “uninfected” using **class_indices** property on augmented images stored in the variable called **training_augmented_images**.

You can see that “infected” class images are labeled as 0 and “uninfected” class images are labeled as 1.



You did amazing work today!

We preprocessed the images dataset using **ImageDataGenerator** Class which converts the images into an array of pixel values, label each image based on the number of classes in the subdirectory structure and apply augmentation which helps to make better machine learning models!

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 05 mins

Teacher Starts Slideshow
Slide 19 to 23

Activity Details:

Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz

Continue WRAP-UP Session
Slide 24 to 29

Activity Details:

Following are the session deliverables:

- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK

- **Appreciate the student for his/her efforts in the class.**
- **Ask the student to make notes for the reflection journal along with the code they wrote in today's class.**

Teacher Action	Student Action
You get Hats off for your excellent work!	*Make sure you have given at least 2 Hats Off during the class for:*  

PROJECT OVERVIEW DISCUSSION

Refer the document below in Activity Links Sections

Teacher Clicks ✖ End Class

ADDITIONAL ACTIVITIES

Encourage the student to apply multiple data augmentation using ImageDataGenerator Class:

- **Rescale**: Convert all values from [0, 255] to [0, 1] by dividing it with 255.
- **Rotation**: 40 degrees
- **Width Shift**: 0.3 (30%)
- **Height Shift**: 0.3 (30%)
- **Horizontal Flip**: True
- **Vertical Flip**: True
- **Zoom**: 0.3 (30%)

```
import tensorflow
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Random Data Augmentation(Rescale, Rotation, Flips, Zoom, Shifts) using ImageDataGenerator
training_data_generator = ImageDataGenerator(
    rescale = 1.0/255,
    rotation_range=40,
    width_shift_range=0.3,
    height_shift_range=0.3,
    zoom_range=0.3,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest')
```

*Guide the student to create the augmented images using **flow_from_directory()** method*

***Note**: Follow the teacher's activity to explain the rest of the code.*

```
# Image Directory
training_image_directory = "/content/PRO-M3-Pneumothorax-Image-Dataset/training_dataset"

# Generate Augmented Image files
training_augmented_images = training_data_generator.flow_from_directory(
    training_image_directory,
    target_size=(180,180))
```

```
Found 200 images belonging to 2 classes.
```

ACTIVITY LINKS		
Activity Name	**Description**	**Link**
Teacher Activity 1	Understanding Pneumothorax Disease	https://s3-whjr-curriculum-uploads.whjr.online/ace33d04-a02e-4700-84f0-89885ab5f8c0.gif
Teacher Activity 2	Sample X-Ray Image	https://s3-whjr-curriculum-uploads.whjr.online/0a1b4cd4-563b-4666-912e-f6bbb3dab246.jpg
Teacher Activity 3	Raw Image Dataset Structure	https://s3-whjr-curriculum-uploads.whjr.online/f907f235-4c7e-47e5-bf68-e3f53cb8b9b4.png
Teacher Activity 4	Image Labeling	https://s3-whjr-curriculum-uploads.whjr.online/1bdade80-2a32-4fc2-8902-f18067803dba.jpeg
Teacher Activity 5	Image Directory Structure of Keras Image Preprocessing	https://s3-whjr-curriculum-uploads.whjr.online/2467514a-e93f-4a0f-8e20-b3893dfa9144.jpeg
Teacher Activity 6	Google Colab Notebook	Google Colab Notebook
Teacher Activity 7	Pneumothorax Image	https://github.com/procodingclass/PRO-

	Dataset	M3-Pneumothorax-Image-Dataset
Teacher Activity 8	Matplotlib Pyplot	Pyplot tutorial — Matplotlib 3.4.3 documentation
Teacher Activity 9	Matplotlib Pyplot imshow	matplotlib.pyplot.imshow — Matplotlib 3.4.3 documentation
Teacher Activity 10	Image Data Augmentation	https://s3-whjr-curriculum-uploads.whjr.online/5403e9b1-a339-405d-98b3-36826ec3f04a.gif
Teacher Activity 11	Keras ImageDataGenerator Class	Image data preprocessing keras.io
Teacher Activity 12	Matplotlib Pyplot subplot	matplotlib.pyplot.subplot — Matplotlib 3.4.3 documentation
Teacher Activity 13	Teacher Activity Reference	https://colab.research.google.com/drive/1PokFogv8GkdWJ5vMh6AIM4jxnMt_sg02?usp=sharing
Teacher Activity 14	Reference Code	https://colab.research.google.com/drive/1dHESnqJlgVAMWnJ9tQDC_sSDQu8ydKM9?usp=sharing
Student Activity 1	Student Boilerplate Code	https://colab.research.google.com/drive/1XWZuXDgR2HE5Oms_mfDZKV-SIxSQ9aR3?usp=sharing
Teacher Reference 1	Project Document	https://s3-whjr-curriculum-uploads.whjr.online/4a49aad3-1738-4a36-abfa-7b9ce7e07179.pdf
Teacher Reference 2	Project Solution	https://colab.research.google.com/drive/1-HJ3Eg4tNDhy378OPmEQFe_r8nJKi0kE?usp=sharing
Teacher Reference 3	Visual Aid Link	https://s3-whjr-curriculum-uploads.whjr.online/10e94352-4d93-4a22-9d7b-b3b8c

		296dbd3.html
Teacher Reference 4	In Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/50103666-dca9-4271-b2a5-fabc8694704c.pdf