Loading and pre-processing the data:

Data is gold as far as deep learning models are concerned. Your image classification model has a far better chance of performing well if you have a good amount of images in the training set. Also, the shape of the data varies according to the architecture/framework that we use.

Hence, the critical data pre-processing step (the eternally important step in any project). I highly recommend going through the "basics of image processing using Python

we use Keras' ImageDataGenerator class to perform data augmentation. i.e, we are using some kind of parameters to process our collected data. The word "augment" means to make something "greater" or "increase" something (in this case, data), the Keras ImageDataGenerator class actually works by:

- Accepting a batch of images used for training.
- Taking this batch and applying a series of random transformations to each image in the batch (including random rotation, resizing, shearing, etc.).
- Replacing the original batch with the new, randomly transformed batch.
- Training the CNN on this randomly transformed batch (i.e., the original data itself is not used for training).

Note: The ImageDataGenerator accepts the original data, randomly transforms it, and returns only the new, transformed data.

Import the library:

```
#its used for different augmentation of the image
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

Define the parameters /arguments for ImageDataGenerator class:

```
#setting parameter for Image Data agumentation to the traing data
train_datagen = ImageDataGenerator(rescale=1./255,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   horizontal_flip=True)
mage Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)
```

Applying ImageDataGenerator functionality to trainset and testset:

```
#performing data agumentation to train data
train = train_datagen.flow_from_directory(r'D:\Python\gesture based control of raidiology images\Dataset\train',
                                          target_size=(64, 64),
                                          batch_size=3,
                                          color_mode='grayscale',
                                          class_mode='categorical')
#performing data agumentation to test data
x_test = test_datagen.flow_from_directory(r'D:\Python\gesture based control of raidiology images\Dataset\test',
                                          target_size=(64, 64),
                                          batch_size=3,
                                          color_mode='grayscale',
                                          class_mode='categorical')
```