Sprint - III

Model building

Date	19 November 2022
Team ID	PNT2022TMID10415
Project Name	Natural Disasters Intensity Analysis and Classification using Artificial Intelligence

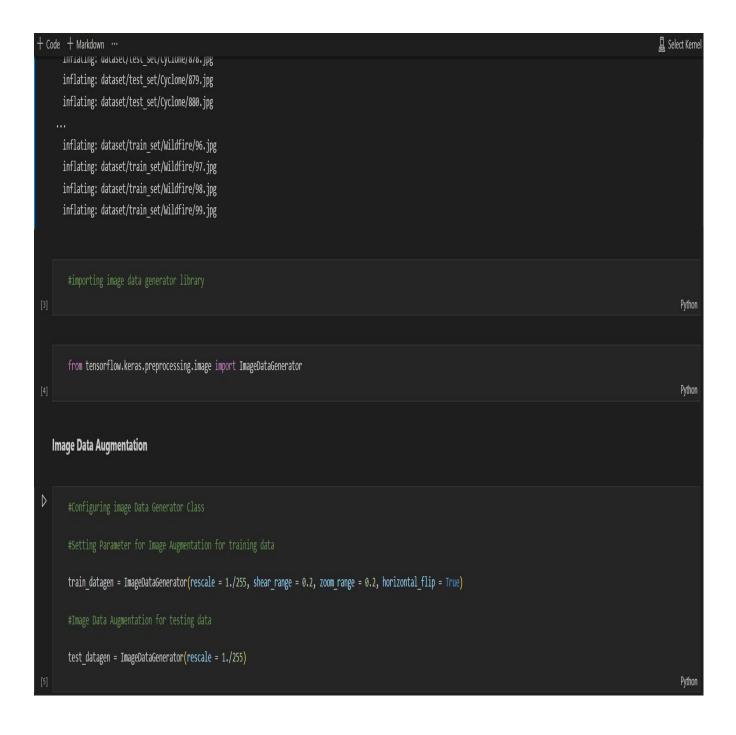
Extract zip file

ZIP is an archive file format that supports lossless data compression. By lossless compression, we mean that the compression algorithm allows the original data to be perfectly reconstructed from the compressed data.

```
D₁ D₄ 日 ··· 📵
    !unzip '/content/drive/MyDrive/IBM/dataset.zip'
                                                                                                                                                                                  Python
Output exceeds the size limit. Open the full output data in a text editor
Archive: /content/drive/MyDrive/IBM/dataset.zip
replace dataset/readme.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: yes
 inflating: dataset/readme.txt
replace dataset/test_set/Cyclone/867.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: yes
 inflating: dataset/test_set/Cyclone/867.jpg
replace dataset/test_set/Cyclone/868.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: yes
 inflating: dataset/test_set/Cyclone/868.jpg
replace dataset/test_set/Cyclone/869.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: yes
 inflating: dataset/test set/Cyclone/869.jpg
replace dataset/test_set/Cyclone/870.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
 inflating: dataset/test_set/Cyclone/870.jpg
replace\ dataset/test\_set/Cyclone/871.jpg?\ [y]es,\ [n]o,\ [A]ll,\ [N]one,\ [r]ename:\ yes
 inflating: dataset/test_set/Cyclone/871.jpg
replace dataset/test_set/Cyclone/872.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
 inflating: dataset/test set/Cyclone/872.jpg
replace dataset/test_set/Cyclone/873.jpg? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
  inflating: dataset/test_set/Cyclone/873.jpg
replace\ dataset/test\_set/Cyclone/874.jpg?\ [y]es,\ [n]o,\ [A]ll,\ [N]one,\ [r]ename:\ ALL\ yes
 inflating: dataset/test_set/Cyclone/874.jpg
  inflating: dataset/test set/Cyclone/875.jpg
  inflating: dataset/test_set/Cyclone/876.jpg
  inflating: dataset/test_set/Cyclone/877.jpg
  inflating: dataset/test_set/Cyclone/878.jpg
  inflating: dataset/test_set/Cyclone/879.jpg
```

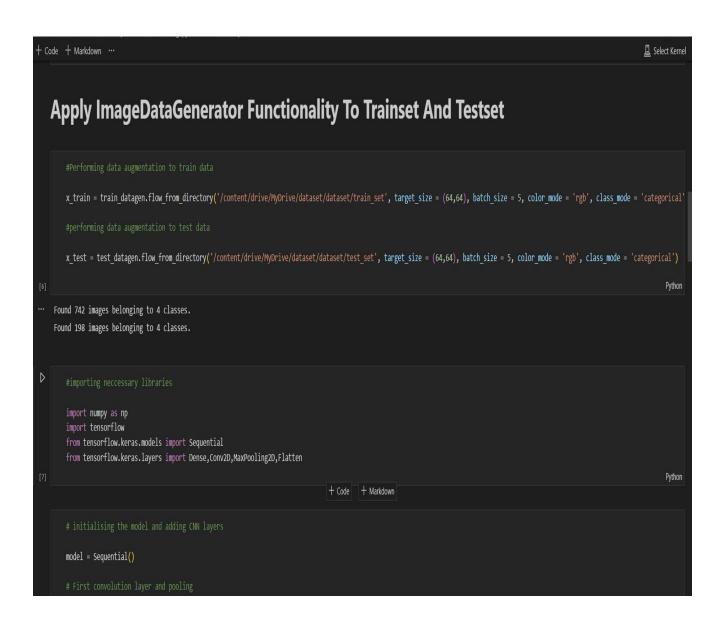
Importing image data generator library/Image data Augmentation

Keras Image Data Generator is used for getting the input of the original data and further, it makes the transformation of this data on a random basis and gives the output resultant containing only the data that is newly transformed.



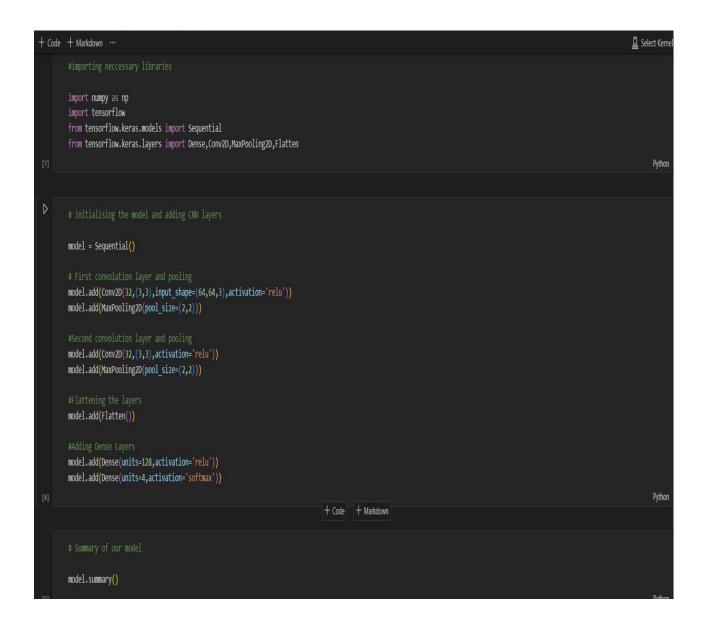
Apply Image Data Generator Functionality to train set and test set

You probably encountered a situation where you try to load a dataset but there is not enough memory in your machine. As the field of machine learning progresses, this problem becomes more and more common. Today this is already one of the challenges in the field of vision where large datasets of images and video files are processed



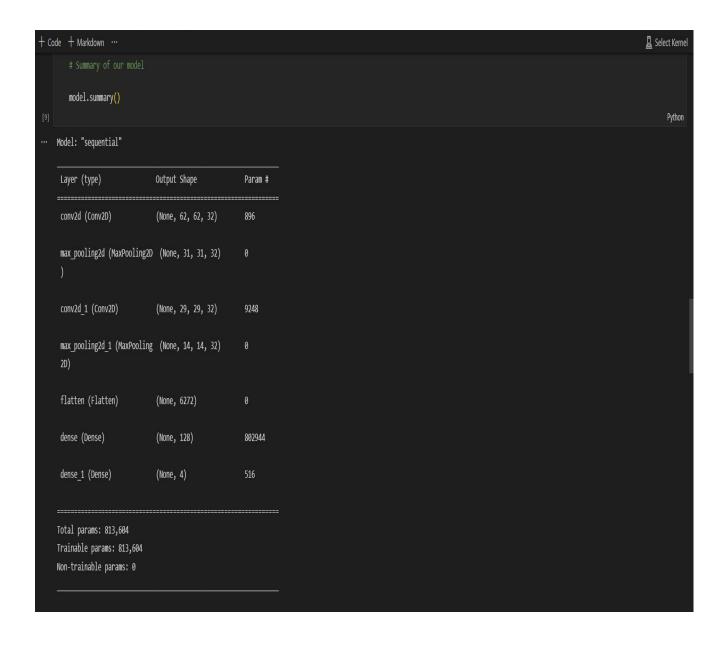
Importing necessary libraries/Initializing the model and adding CNN layers

TensorFlow is a popular deep learning framework. In this tutorial, you will learnthe basics of this Python library and understand how to implement these deep, feed-forward artificial neural networks with it.



Summary of our model

The model summary gives us a fine visualization of our model and the aim is to provide complete information that is not provided by the print statement.



Fitting the model

We'll define the Keras sequential model and add a one-dimensional convolutional layer. Input shape becomes as it is confirmed above We'll add Dense, MaxPooling1D, and Flatten layers into the model. The output layer contains the number of output classes and 'SoftMax' activation.

```
+ Code + Markdown ···
                                                                                                                                                      A Select Kernel
      model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
                                                                                                                                                           Python
      model.fit generator(generator=x train, steps per_epoch=len(x train), epochs=20, validation_data=x test, validation_steps=len(x test))
... /usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which
    supports generators.
     """Entry point for launching an IPython kernel.
    Output exceeds the size limit. Open the full output data in a text editor
    Epoch 1/20
    149/149 [======] - 246s 2s/step - loss: 0.4583 - accuracy: 0.5283 - val loss: 0.3589 - val accuracy: 0.7273
    Epoch 2/20
    149/149 [=======] - 40s 268ms/step - loss: 0.3377 - accuracy: 0.6995 - val loss: 0.4073 - val accuracy: 0.6263
    Epoch 3/20
    149/149 [======] - 40s 266ms/step - loss: 0.2974 - accuracy: 0.7345 - val_loss: 0.3060 - val_accuracy: 0.7626
    Epoch 4/20
                    149/149 [==
    149/149 [======] - 40s 262ms/step - loss: 0.2445 - accuracy: 0.7803 - val loss: 0.4709 - val accuracy: 0.6465
    Epoch 6/20
    149/149 [======] - 40s 270ms/step - loss: 0.2430 - accuracy: 0.7925 - val_loss: 0.3750 - val_accuracy: 0.6970
    Epoch 7/20
    149/149 [======] - 40s 268ms/step - loss: 0.2047 - accuracy: 0.8423 - val_loss: 0.2808 - val_accuracy: 0.7727
    Epoch 8/20
    149/149 [======= - - 40s 267ms/step - loss; 0.1946 - accuracy; 0.8396 - val loss; 0.2907 - val accuracy; 0.8030
    Epoch 9/20
    149/149 [==================] - 40s 266ms/step - loss: 0.1900 - accuracy: 0.8410 - val_loss: 0.2787 - val_accuracy: 0.7778
```

Save the model/Load the saved model/Taking image as input

The Saved Model format is another way to serialize models. Models saved in this format can be restored using and are compatible with TensorFlow Serving. The Saved Model goes into detail about how to serve/inspect the Saved Model. The section below illustrates the steps to save and restore the model.

```
+ Code + Markdown ···
                                                                                                                                                                                 Select Kernel
       model.save('disaster.h5')
       model_json = model.to_json()
        with open("/content/drive/MyDrive/IBM/model-bw.json", "w") as json_file:
          json file.write(model json)
        from tensorflow.keras.models import load model
        from tensorflow.keras.preprocessing import image
       model = load model('disaster.h5')
                                                                                                                                                                                       Pythor
        x_train.class_indices
                                                                                                                                                                                       Python
   {'Cyclone': 0, 'Earthquake': 1, 'Flood': 2, 'Wildfire': 3}
        img = image.load_img('/content/drive/MyDrive/dataset/dataset/test_set/Wildfire/1040.jpg', target_size=(64,64))
        x=image.img to array(img)
        x=np.expand_dims(x,axis=0)
        index=['Cyclone','Earthquake','Flood','Wildfire']
        y=np.argmax(model.predict(x),axis=1)
        print(index[int(y)])
```