Emerging methods for early detection of forest fire

Model building

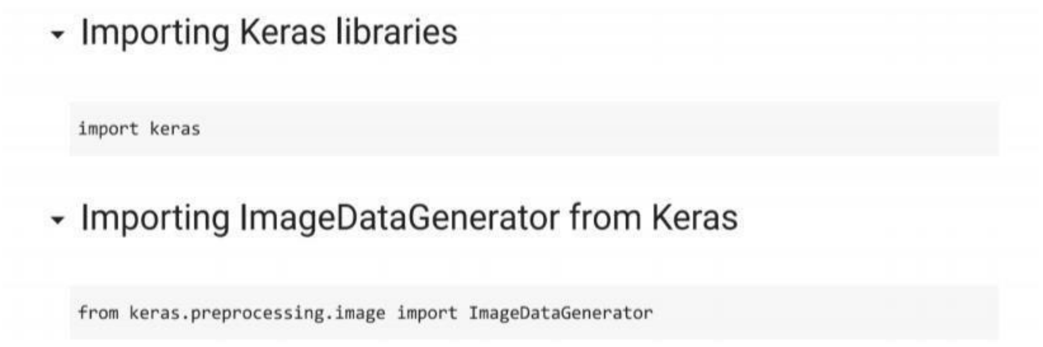
Configuring the learning process

Team ID	PNT2022TMID16960
Project Name	Emerging methods for early detection of forest fire

Configuring the learning process

With both the training data defined and model defined, it's time to configure the learning process. This is accomplished with a call to the compile () method of the Sequential model class. Compilation requires 3 arguments: an optimizer, a loss function, and a list of metrics.

IMPORT LIBRARIES:



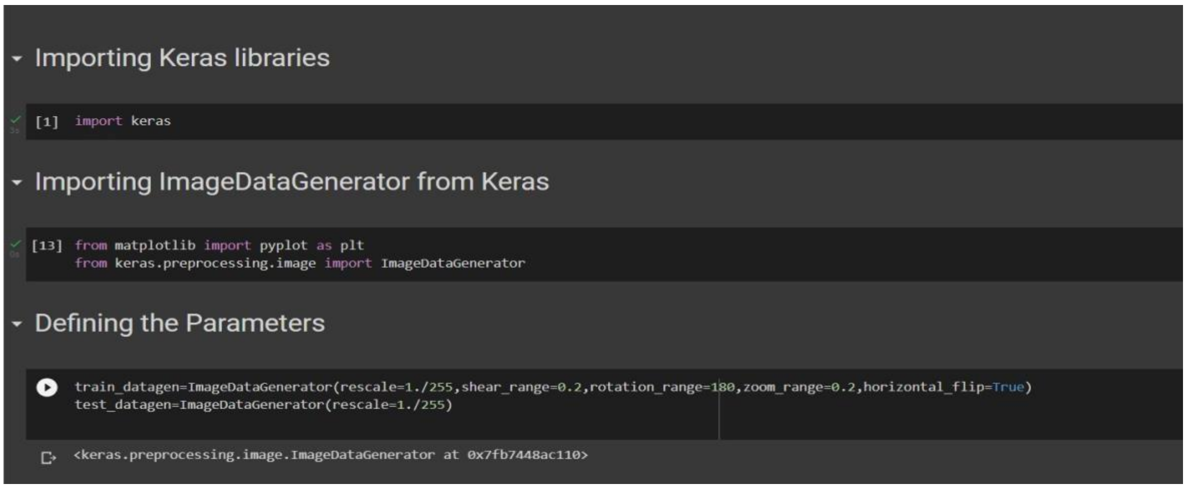
Importing Keras libraries

```
import keras
```

Importing ImageDataGenerator from Keras

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

IMPORT ImageDataGenerator FROM KERAS:



Importing Keras libraries

```
[1]  import keras
```

Importing ImageDataGenerator from Keras

```
[13] from matplotlib import pyplot as plt
     from keras.preprocessing.image import ImageDataGenerator
```

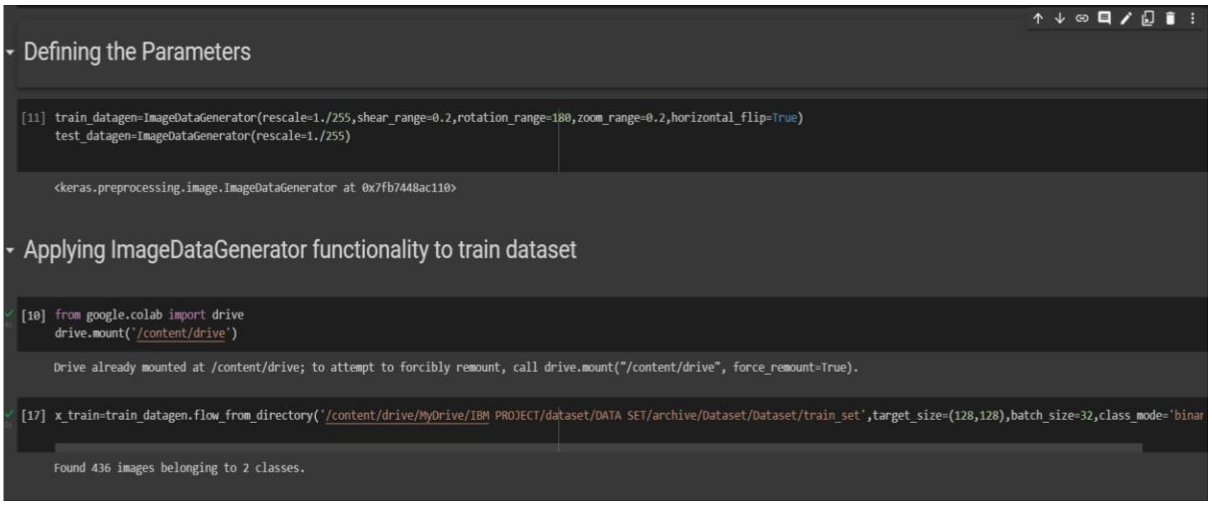
Defining the Parameters

```
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotation_range=180,zoom_range=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
```

```
<keras.preprocessing.image.ImageDataGenerator at 0x7fb7448ac110>
```

APPLYING ImageDataGenerator to train dataset:

ply**flow_from_directory ( )**methodfor Train folder.



Defining the Parameters

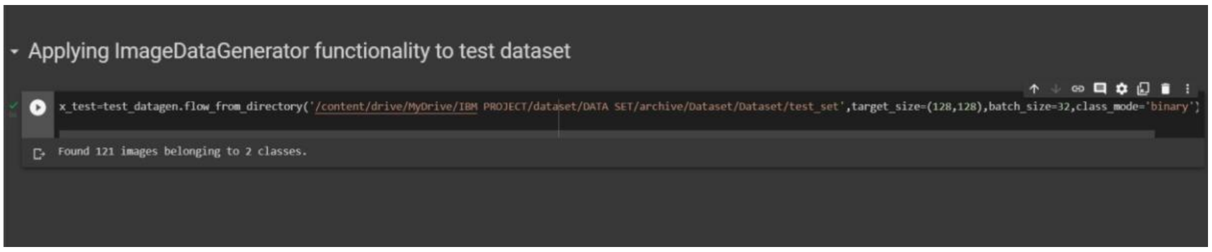
```
[11] train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,rotation_range=180,zoom_range=0.2,horizontal_flip=True)
     test_datagen=ImageDataGenerator(rescale=1./255)

     <keras.preprocessing.image.ImageDataGenerator at 0x7fb7448ac110>
```

Applying ImageDataGenerator functionality to train dataset

```
[10] from google.colab import drive
     drive.mount('/content/drive')

     Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[17] x_train=train_datagen.flow_from_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/train_set',target_size=(128,128),batch_size=32,class_mode='binar

     Found 436 images belonging to 2 classes.
```

APPLYING ImageDataGenerator to test dataset:

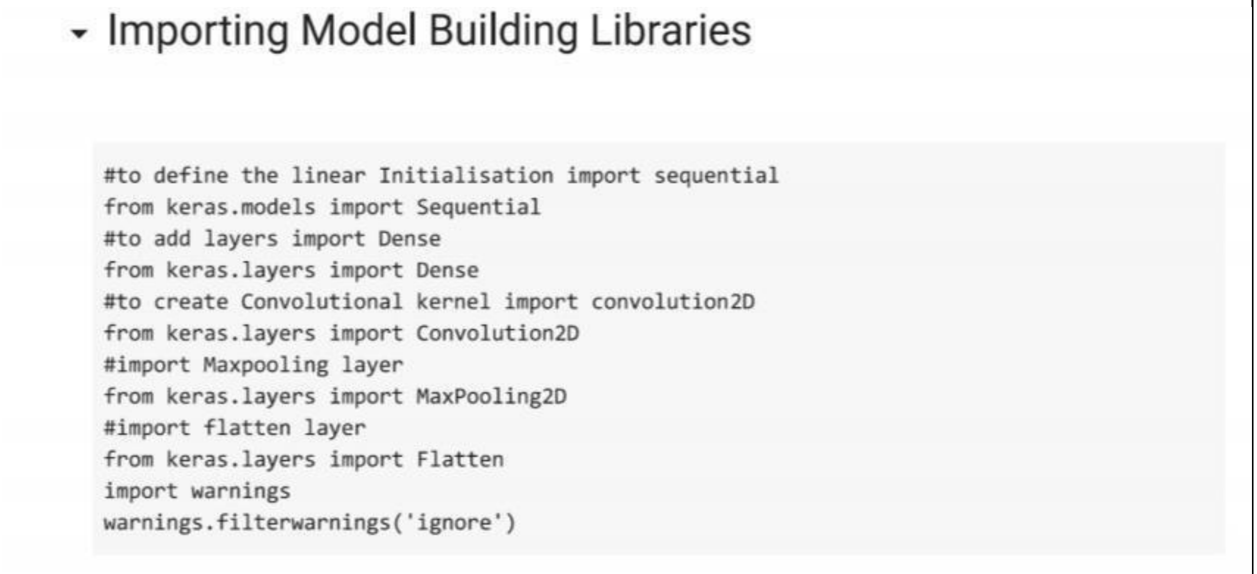
Applying the **flow_from_directory ( )** methodfortest folder.



Applying ImageDataGenerator functionality to test dataset

```
x_test=test_datagen.flow_from_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/test_set',target_size=(128,128),batch_size=32,class_mode='binary')

Found 121 images belonging to 2 classes.
```

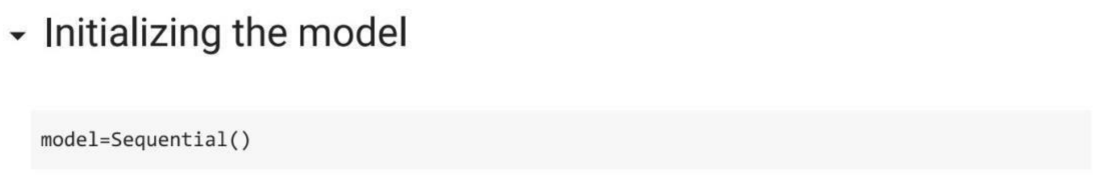
IMPORTING MODEL BUILDING LIBRARIES:



Importing Model Building Libraries

```
#to define the linear Initialisation import sequential
from keras.models import Sequential
#to add layers import Dense
from keras.layers import Dense
#to create Convolutional kernel import convolution2D
from keras.layers import Convolution2D
#import Maxpooling layer
from keras.layers import MaxPooling2D
#import flatten layer
from keras.layers import Flatten
import warnings
warnings.filterwarnings('ignore')
```

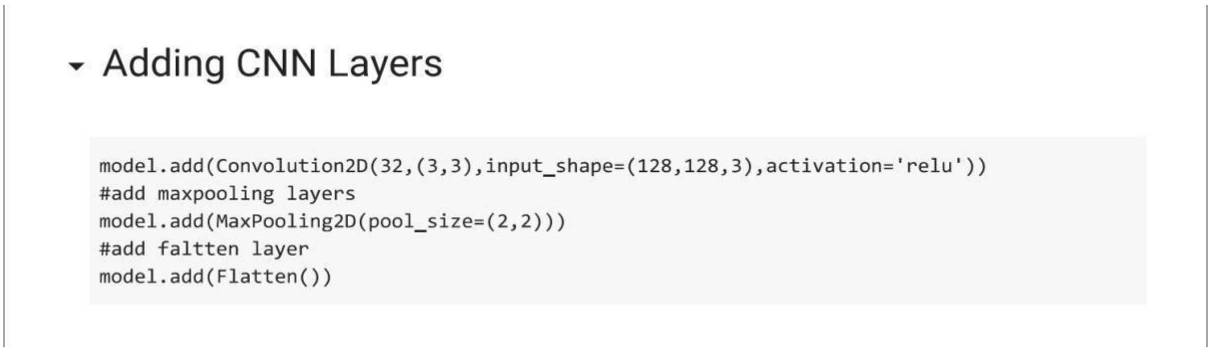
INITIALIZING THE MODEL:



Initializing the model

```
model=Sequential()
```

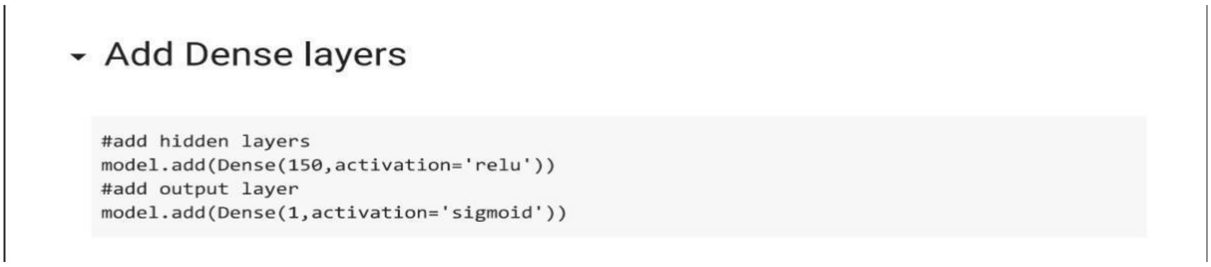
ADDING CNN LAYERS:



Adding CNN Layers

```
model.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='relu'))
#add maxpooling layers
model.add(MaxPooling2D(pool_size=(2,2)))
#add faltten layer
model.add(Flatten())
```

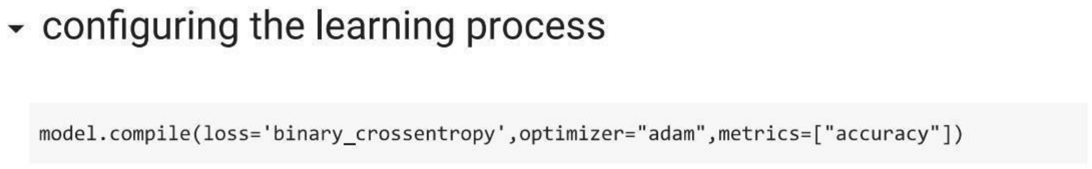
ADDING DENSE LAYERS:



Add Dense layers

```
#add hidden layers
model.add(Dense(150,activation='relu'))
#add output layer
model.add(Dense(1,activation='sigmoid'))
```

CONFIGURING THE LEARNING PROCESS:



configuring the learning process

```
model.compile(loss='binary_crossentropy',optimizer="adam",metrics=["accuracy"])
```