**EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES**

**MODEL BUILDING CONFIGURING THE LEARNING PROCESS**

|  |  |
| --- | --- |
| **Date** | 25November 2022 |
| **Team ID** | PNT2022TMID49024 |
| **Project Name** | Emerging Methods for Early Detection of Forest Fires |

***Importing The ImageDataGenerator Library***

import keras

from keras.preprocessing.image import ImageDataGenerator

***Define the parameters/arguments for ImageDataGenerator class***

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)

***Applying ImageDataGenerator functionality to trainset***

x\_train=train\_datagen.flow\_from\_directory(r'/content/drive/MyDrive/Dataset/train\_set', target\_size=(128,128),batch\_size=32, class\_mode='binary')

Found 436 images belonging to 2 classes.

***Applying ImageDataGenerator functionality to testset***

x\_test=test\_datagen.flow\_from\_directory(r'/content/drive/MyDrive/Dataset/test\_set', target\_size=(128,128),batch\_size=32, class\_mode='binary') Found 121 images belonging to 2 classes.

***Import model building libraries***

*#To define Linear initialisation import Sequential*

from keras.models import Sequential *#To add layers import Dense* from keras.layers import Dense

*#To create Convolution 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***

model=Sequential()

***Add CNN Layer***

model.add(Convolution2D(32, (3,3),input\_shape=(128,128,3),activation='relu')) #add maxpooling layer

model.add(MaxPooling2D(pool\_size=(2,2))) #add flatten layer

model.add(Flatten())

***Add Dense Layer***

*#add hidden layer*

model.add(Dense(150,activation='relu'))

*#add output layer*

model.add(Dense(1,activation='sigmoid'))

***Configure the learning process***

model.compile(loss='binary\_crossentropy',optimizer="adam",metrics=["ac curacy"])