

# EMERGING METHODS FOR EARLY DETECTION OF FOREST FIRES

## MODEL BUILDING

### CONFIGURING THE LEARNING PROCESS

<b>Date</b>	06 November 2022
<b>Team ID</b>	PNT2022TMID12327
<b>Project Name</b>	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=["accuracy"])
```

