*TEAM ID -PNT2022TMID04308 * TEAM MEMBERS:- DHANA SREE R, HARISH K P, DHARANIDHARAN R, DHIVAKAR P

```
!unzip "/content/drive/MyDrive/dhana1810 AI-Based-Natural-Disaster-Intensity-Analysis·main·dataset.zip"
```

```
  extracting: dataset/train_set/Cyclone/18.jpg
  extracting: dataset/train_set/Cyclone/184.jpg
  extracting: dataset/train_set/Cyclone/17.jpg
  extracting: dataset/train_set/Cyclone/185.jpg
  extracting: dataset/train_set/Cyclone/186.jpg
  extracting: dataset/train_set/Cyclone/188.jpg
  extracting: dataset/train_set/Cyclone/165.jpg
  extracting: dataset/train_set/Cyclone/19.jpg
  extracting: dataset/train_set/Cyclone/187.jpg
  extracting: dataset/train_set/Cyclone/189.jpg
  extracting: dataset/train_set/Cyclone/192.jpg
  extracting: dataset/train_set/Cyclone/194.jpg
  extracting: dataset/train_set/Cyclone/190.jpg
  extracting: dataset/train_set/Cyclone/193.jpg
  extracting: dataset/train_set/Cyclone/195.jpg
  extracting: dataset/train_set/Cyclone/196.jpg
  extracting: dataset/train_set/Cyclone/197.jpg
  extracting: dataset/train_set/Cyclone/199.jpg
  extracting: dataset/train_set/Cyclone/2.jpg
  extracting: dataset/train_set/Cyclone/202.jpg
  extracting: dataset/train_set/Cyclone/191.jpg
  extracting: dataset/train_set/Cyclone/20.jpg
  extracting: dataset/train_set/Cyclone/201.jpg
  extracting: dataset/train_set/Cyclone/200.jpg
  extracting: dataset/train_set/Cyclone/203.jpg
  extracting: dataset/train_set/Cyclone/206.jpg
  extracting: dataset/train_set/Cyclone/204.jpg
  extracting: dataset/train_set/Cyclone/205.jpg
  extracting: dataset/train_set/Cyclone/207.jpg
  extracting: dataset/train_set/Cyclone/21.jpg
  extracting: dataset/train_set/Cyclone/208.jpg
  extracting: dataset/train_set/Cyclone/209.jpg
  extracting: dataset/train_set/Cyclone/210.jpg
  extracting: dataset/train_set/Cyclone/212.jpg
  extracting: dataset/train_set/Cyclone/211.jpg
  extracting: dataset/train_set/Cyclone/214.jpg
  extracting: dataset/train_set/Cyclone/215.jpg
  extracting: dataset/train_set/Cyclone/213.jpg
  extracting: dataset/train_set/Cyclone/216.jpg
  extracting: dataset/train_set/Cyclone/218.jpg
  extracting: dataset/train_set/Cyclone/217.jpg
  extracting: dataset/train_set/Cyclone/219.jpg
  extracting: dataset/train_set/Cyclone/22.jpg
  extracting: dataset/train_set/Cyclone/221.jpg
  extracting: dataset/train_set/Cyclone/220.jpg
  extracting: dataset/train_set/Cyclone/222.jpg
  extracting: dataset/train_set/Cyclone/223.jpg
  extracting: dataset/train_set/Cyclone/224.jpg
  extracting: dataset/train_set/Cyclone/226.jpg
  extracting: dataset/train_set/Cyclone/225.jpg
  extracting: dataset/train_set/Cyclone/227.jpg
  extracting: dataset/train_set/Cyclone/228.jpg
  extracting: dataset/train_set/Cyclone/229.jpg
  extracting: dataset/train_set/Cyclone/230.jpg
  extracting: dataset/train_set/Cyclone/23.jpg
  extracting: dataset/train_set/Cyclone/232.jpg
  extracting: dataset/train_set/Cyclone/25.jpg
  extracting: dataset/train_set/Cyclone/231.jpg
  extracting: dataset/train_set/Cyclone/26.jpg
```

```
#data agumentation
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_gen=ImageDataGenerator(rescale=1./255,zoom_range=0.2,horizontal_flip=True)
test_gen=ImageDataGenerator(rescale=1./255)
```

```
#passing the data
xtrain=train_gen.flow_from_directory("/content/dataset/train_set",
                                     target_size=(64,64),
                                     class_mode="categorical",
                                     batch_size=50,)
```

```
Found 742 images belonging to 4 classes.
```

```
xtest=test_gen.flow_from_directory("/content/dataset/test_set",
                                   target_size=(64,64),
                                   class_mode="categorical",
                                   batch_size=50)
```

```
Found 198 images belonging to 4 classes.
```

```
#creating cnn model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPool2D,Flatten,Dense

CNN_model=Sequential()
CNN_model.add(Convolution2D(32,(3,3),activation="relu",input_shape=(64,64,3)))
CNN_model.add(MaxPool2D(pool_size=(2,2)))
CNN_model.add(Flatten())
#fully connected layers
CNN_model.add(Dense(300,activation="relu"))
CNN_model.add(Dense(200,activation="relu"))
CNN_model.add(Dense(150,activation="relu"))
CNN_model.add(Dense(120,activation="relu"))
CNN_model.add(Dense(500,activation="relu"))
CNN_model.add(Dense(650,activation="relu"))
CNN_model.add(Dense(750,activation="relu"))
CNN_model.add(Dense(50,activation="relu"))
CNN_model.add(Dense(750,activation="relu"))
CNN_model.add(Dense(350,activation="relu"))
CNN_model.add(Dense(150,activation="relu"))
CNN_model.add(Dense(450,activation="relu"))
CNN_model.add(Dense(950,activation="relu"))
CNN_model.add(Dense(100,activation="relu"))
CNN_model.add(Dense(105,activation="relu"))
CNN_model.add(Dense(190,activation="relu"))
CNN_model.add(Dense(130,activation="relu"))
CNN_model.add(Dense(4,activation="softmax"))
```

```
CNN_model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 62, 62, 32)        896

 max_pooling2d_2 (MaxPooling  (None, 31, 31, 32)       0
 2D)

 flatten_2 (Flatten)         (None, 30752)             0

 dense_8 (Dense)             (None, 300)               9225900

 dense_9 (Dense)             (None, 200)               60200

 dense_10 (Dense)            (None, 150)               30150

 dense_11 (Dense)            (None, 120)               18120

 dense_12 (Dense)            (None, 500)               60500

 dense_13 (Dense)            (None, 650)               325650

 dense_14 (Dense)            (None, 750)               488250

 dense_15 (Dense)            (None, 50)                37550

 dense_16 (Dense)            (None, 750)               38250

 dense_17 (Dense)            (None, 350)               262850

 dense_18 (Dense)            (None, 150)               52650

 dense_19 (Dense)            (None, 450)               67950

 dense_20 (Dense)            (None, 950)               428450

 dense_21 (Dense)            (None, 100)               95100

 dense_22 (Dense)            (None, 105)               10605

 dense_23 (Dense)            (None, 190)               20140

 dense_24 (Dense)            (None, 130)               24830

 dense_25 (Dense)            (None, 4)                 524

=================================================================
Total params: 11,248,565
Trainable params: 11,248,565
Non-trainable params: 0
_________________________________________________________________
```

```
CNN_model.compile(optimizer="adam",loss="categorical_crossentropy",metrics=["accuracy"])
```

```
CNN_model.save("Disasters.h5")
```

Testing

```
#tuning
from keras.callbacks import EarlyStopping,ReduceLROnPlateau

earlystopping=EarlyStopping(monitor="val_accuracy",patience=5)
reduce_lr=ReduceLROnPlateau(monitor="val_accuracy",patience=5,factor=0.5,min_lr=0.00001)
callback=[reduce_lr,earlystopping]
```

```
CNN_model.fit_generator(xtrain,
                        steps_per_epoch=len(xtrain),
                        epochs=100,
                        callbacks=callback,
                        validation_data=xtest,
                        validation_steps=len(xtest))
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:6: UserWarning: `Model.fit_generator` is deprecated and will be remove

Epoch 1/100
15/15 [==============================] - 30s 2s/step - loss: 0.2799 - accuracy: 0.9057 - val_loss: 1.5248 - val_accuracy: 0.7020 -
Epoch 2/100
15/15 [==============================] - 27s 2s/step - loss: 0.2491 - accuracy: 0.9313 - val_loss: 1.2206 - val_accuracy: 0.7424 -
Epoch 3/100
15/15 [==============================] - 27s 2s/step - loss: 0.2302 - accuracy: 0.9245 - val_loss: 1.3768 - val_accuracy: 0.7475 -
Epoch 4/100
15/15 [==============================] - 26s 2s/step - loss: 0.2183 - accuracy: 0.9340 - val_loss: 1.3843 - val_accuracy: 0.7475 -
Epoch 5/100
15/15 [==============================] - 27s 2s/step - loss: 0.2313 - accuracy: 0.9367 - val_loss: 1.2302 - val_accuracy: 0.7525 -
Epoch 6/100
15/15 [==============================] - 27s 2s/step - loss: 0.2140 - accuracy: 0.9340 - val_loss: 1.3193 - val_accuracy: 0.7323 -
Epoch 7/100
15/15 [==============================] - 27s 2s/step - loss: 0.1746 - accuracy: 0.9528 - val_loss: 1.3630 - val_accuracy: 0.7323 -
Epoch 8/100
15/15 [==============================] - 27s 2s/step - loss: 0.2306 - accuracy: 0.9326 - val_loss: 1.4956 - val_accuracy: 0.7374 -
Epoch 9/100
15/15 [==============================] - 27s 2s/step - loss: 0.1954 - accuracy: 0.9299 - val_loss: 1.5619 - val_accuracy: 0.7374 -
Epoch 10/100
15/15 [==============================] - 28s 2s/step - loss: 0.1896 - accuracy: 0.9394 - val_loss: 1.5368 - val_accuracy: 0.7273 -
<keras.callbacks.History at 0x7f9d1403d5d0>
```

```
import numpy as np
from tensorflow.keras.preprocessing import image
img=image.load_img("/content/dataset/test_set/Flood/993.jpg",target_size=(64,64))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
op=['Cyclone', 'Earthquake', 'Flood', 'Wildfire']
pred=np.argmax(CNN_model.predict(x))
op[pred]
```

```
1/1 [==============================] - 0s 123ms/step
'Flood'
```

```
#saving in tar
!tar -zvcf natural-disaster.tgz Disasters.h5
```

```
Disasters.h5
```

**IBM DEPLOYMENT**

```
!pip install watson-machine-learning-client
```