Milestone- Model Building

Date	16 November 2022
Team ID	PNT2022TMID18498
Project Name	Project – Natural Disasters Intensity Analysis and Classification Using Artificial Intelligence

```
# Initializing the CNN
classifier = Sequential()

# First convolution layer and poolingo
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
# Second convolution layer and pooling
classifier.add(Conv2D(32, (3, 3), activation='relu'))
# input_shape is going to be the pooled feature maps from the previous convolution layer
classifier.add(MaxPooling2D(pool_size=(2, 2)))
classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))

# Flattening the layers
classifier.add(Flatten())

# Adding a fully connected layer
classifier.add(Dense(units=128, activation='relu'))
classifier.add(Dense(units=4, activation='softmax')) # softmax for more than 2
```

classifier.summary()#summary of our model

```
+ Code   + Text

 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248

 conv2d_2 (Conv2D)           (None, 27, 27, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 13, 13, 32)       0
 2D)

 conv2d_3 (Conv2D)           (None, 11, 11, 32)        9248

 flatten (Flatten)           (None, 3872)              0

 dense (Dense)               (None, 128)               495744

 dense_1 (Dense)             (None, 4)                 516

=================================================================
Total params: 524,900
Trainable params: 524,900
Non-trainable params: 0
_________________________________________________________________
```

```
# Compiling the CNN
# categorical_crossentropy for more than 2
classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

classifier.fit_generator(
        generator=x_train,steps_per_epoch = len(x_train),
        epochs=20, validation_data=x_test,validation_steps = len(x_test))# No of images in test set
```

```
+ Code   + Text                                                    Connect   Editing

[ ]  Epoch 1/20
     C:\Users\hp\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please
       This is separate from the ipykernel package so we can avoid doing imports until
     40/40 [==============================] - 17s 440ms/step - loss: 1.3599 - accuracy: 0.3434 - val_loss: 1.3235 - val_accuracy: 0.3081
     Epoch 2/20
     40/40 [==============================] - 17s 421ms/step - loss: 1.2764 - accuracy: 0.3485 - val_loss: 1.3702 - val_accuracy: 0.3384
     Epoch 3/20
     40/40 [==============================] - 18s 448ms/step - loss: 1.0652 - accuracy: 0.5808 - val_loss: 0.7824 - val_accuracy: 0.6970
     Epoch 4/20
     40/40 [==============================] - 20s 514ms/step - loss: 0.9344 - accuracy: 0.6566 - val_loss: 0.7065 - val_accuracy: 0.7576
     Epoch 5/20
     40/40 [==============================] - 19s 468ms/step - loss: 0.7658 - accuracy: 0.6768 - val_loss: 1.0319 - val_accuracy: 0.6515
     Epoch 6/20
     40/40 [==============================] - 18s 447ms/step - loss: 0.7263 - accuracy: 0.7222 - val_loss: 0.6478 - val_accuracy: 0.7727
     Epoch 7/20
     40/40 [==============================] - 16s 397ms/step - loss: 0.6744 - accuracy: 0.7525 - val_loss: 0.6544 - val_accuracy: 0.7677
     Epoch 8/20
     40/40 [==============================] - 21s 524ms/step - loss: 0.5660 - accuracy: 0.7677 - val_loss: 1.1223 - val_accuracy: 0.6263
     Epoch 9/20
     40/40 [==============================] - 24s 601ms/step - loss: 0.8633 - accuracy: 0.6667 - val_loss: 0.8999 - val_accuracy: 0.6010
     Epoch 10/20
     40/40 [==============================] - 18s 469ms/step - loss: 0.6305 - accuracy: 0.7929 - val_loss: 0.5919 - val_accuracy: 0.8030
     Epoch 11/20
     40/40 [==============================] - 22s 548ms/step - loss: 0.6084 - accuracy: 0.7677 - val_loss: 0.4047 - val_accuracy: 0.8434
     Epoch 12/20
     40/40 [==============================] - 19s 474ms/step - loss: 0.5069 - accuracy: 0.7980 - val_loss: 1.0144 - val_accuracy: 0.7424
     Epoch 13/20
     40/40 [==============================] - 18s 456ms/step - loss: 0.4486 - accuracy: 0.8333 - val_loss: 0.4645 - val_accuracy: 0.8384
     Epoch 14/20
     40/40 [==============================] - 18s 461ms/step - loss: 0.4557 - accuracy: 0.8434 - val_loss: 0.5087 - val_accuracy: 0.7879
```



```
# Save the model
classifier.save('disaster.h5')
```

```
model_json = classifier.to_json()
with open("model-bw.json", "w") as json_file:
    json_file.write(model_json)

from tensorflow.keras.models import load_model
from keras.preprocessing import image
#model = load_model("disaster.h5") #loading the model for testing

img = image.load_img(r"C:\Users\hp\Downloads\e2.jpg",grayscale=False,
                     target_size= (64,64))#loading of the image
x = image.img_to_array(img)#image to array
x = np.expand_dims(x,axis = 0)#changing the shape
#pred = classifier.predict_classes(x)#predicting the classes
predict=classifier.predict(x)
classes_x=np.argmax(predict,axis=1)
classes_x
```

```
index=['Cyclone','Earthquake','Flood','Wildfire']
result=str(index[classes_x[0]])
result
```

```
'Earthquake'
```

Submitted by:

S. Jeyashree (9517201906019)
K. Kaviya Varshini(9517201906021)
J. Kavipriya(9517201906020)
G. Lakshmi Priya(9517201906023)