PROJECT DEVELOPMENT PHASE

SPRINT-II

Date	14 November 2022
Team ID	PNT2022TMID37631
Project Name	Natural Disaster Intensity Analysis and Classification using Artificial Intelligence

INSERTING NECESSARY LIBRARIES:

Numpy: It is an open source numerical python library.

Scikit-learn: It is a machine learning library for python.

OpenCV: OpenCV is a library of programming functions mainly aimed at real-time computer vision.

Flask: Web framework used for building web application.



LOADING DATA AND PERFORMING DATA AUGUMENTATION:

Loading the data into the Jupyter notebook by using RR dataset path.

CREATING THE MODEL:

Creating the Model a Classifier Sequential. Classifier is a machine learning algorithm that determines the class of the input element based on the set of the feature. In this model using convolution2D function. Convolution2D parameter is an number of filters that convolution layer will be learn from. Then we will be using MaxPooling2D function. Then, using a Flatten() function that flatten the multidimensional input denser into the denser.

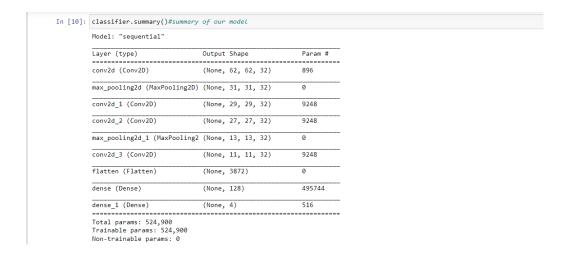
```
In [9]: # Initializing the CNN
    classifier = Sequential()

# 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=128, activation='relu'))
    classifier.add(Dense(units=4, activation='relu'))
```

Using classifier.summary() function summary of our model



COMPILING THE MODEL:

The model is compiled using the following code.

```
In [11]: # Compiling the CNN
# categorical_crossentropy for more than 2
classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

FITTING THE MODEL:

Fitting the Model with 70 epoch.



SAVING THE MODEL:

Saving the Model as disaster.h5. disaster.h5 file is used to find the image classification files. Model.json represents that Jason stands for JavaScript object rotation, Jason is a lite weight data format used for data inserting between multiple different language.

PREDICTING RESULTS:

Loading model from the tensorflow keras models and loading the image then converting image into array. Then predicting our model.