Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/

Collecting twilio

Downloading twilio-7.15.2-py2.py3-none-any.whl (1.4 MB)

|████████████████████████████████| 1.4 MB 14.6 MB/s

Collecting PyJWT<3.0.0,>=2.0.0

Downloading PyJWT-2.6.0-py3-none-any.whl (20 kB)

Requirement already satisfied: requests>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from twilio) (2.23.0)

Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from twilio) (2022.6)

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (2022.9.24)

Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (1.24.3)

Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (3.0.4)

Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests>=2.0.0->twilio) (2.10)

Installing collected packages: PyJWT, twilio

Successfully installed PyJWT-2.6.0 twilio-7.15.2

**Importing Image Data Generator from Keras**

In [3]:

**from** matplotlib **import** pyplot **as** plt

**from** keras.preprocessing.image **import** ImageDataGenerator

**Defining the Parameters**

In [4]:

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,shear\_range**=**0.2,rotation\_range**=**180,zoom\_range**=**0.2,horizontal\_flip**=True**)

**Applying Image Data Generator functionality to train dataset**

In [5]:

**from** google.colab **import** drive

drive**.**mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force\_remount=True).

In [6]:

x\_train**=**train\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/train\_set',target\_size**=**(64,64),batch\_size**=**32,class\_mode**=**'binary')

Found 436 images belonging to 2 classes.

**Applying Image Data Generator functionality to test dataset**

In [7]:

x\_test**=**test\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/test\_set',target\_size**=**(64,64),batch\_size**=**32,class\_mode**=**'binary')

Found 121 images belonging to 2 classes.

**Importing Model Building Libraries**

In [8]:

*#to define the linear Initialisation import sequential*

**from** keras.models **import** Sequential

*#to add layers import Dense*

**from** keras.layers **import** Dense

*#to create Convolutional 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**

In [9]:

model **=** Sequential()

**Adding CNN Layers**

In [10]:

model**.**add(Convolution2D(32,(3,3),input\_shape**=**(64,64,3),activation**=**'relu'))

*#add maxpooling layers*

model**.**add(MaxPooling2D(pool\_size**=**(2,2)))

*#add faltten layer*

model**.**add(Flatten())

**Add Dense layers**

In [11]:

*#add hidden layers*

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

*#add output layer*

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

**configuring the learning process**

In [12]:

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

**Training the model**

In [13]:

model**.**fit\_generator(x\_train,steps\_per\_epoch**=**14,epochs**=**10,validation\_data**=**x\_test,validation\_steps**=**4)

Epoch 1/10

14/14 [==============================] - 137s 10s/step - loss: 0.9627 - accuracy: 0.7294 - val\_loss: 0.3491 - val\_accuracy: 0.8595

Epoch 2/10

14/14 [==============================] - 22s 2s/step - loss: 0.3139 - accuracy: 0.8532 - val\_loss: 0.1041 - val\_accuracy: 0.9752

Epoch 3/10

14/14 [==============================] - 20s 1s/step - loss: 0.2354 - accuracy: 0.9037 - val\_loss: 0.1252 - val\_accuracy: 0.9421

Epoch 4/10

14/14 [==============================] - 22s 2s/step - loss: 0.2281 - accuracy: 0.9106 - val\_loss: 0.0831 - val\_accuracy: 0.9752

Epoch 5/10

14/14 [==============================] - 20s 1s/step - loss: 0.1916 - accuracy: 0.9289 - val\_loss: 0.0753 - val\_accuracy: 0.9752

Epoch 6/10

14/14 [==============================] - 22s 2s/step - loss: 0.1879 - accuracy: 0.9128 - val\_loss: 0.0738 - val\_accuracy: 0.9669

Epoch 7/10

14/14 [==============================] - 20s 2s/step - loss: 0.1720 - accuracy: 0.9312 - val\_loss: 0.0745 - val\_accuracy: 0.9669

Epoch 8/10

14/14 [==============================] - 20s 1s/step - loss: 0.1707 - accuracy: 0.9266 - val\_loss: 0.1375 - val\_accuracy: 0.9339

Epoch 9/10

14/14 [==============================] - 22s 2s/step - loss: 0.1539 - accuracy: 0.9358 - val\_loss: 0.0504 - val\_accuracy: 0.9917

Epoch 10/10

14/14 [==============================] - 20s 1s/step - loss: 0.1467 - accuracy: 0.9450 - val\_loss: 0.0694 - val\_accuracy: 0.9752

Out[13]:

**Save the model**

In [14]:

model**.**save("forest.h5")

**Predictions**

In [31]:

*#import load model from keras.model*

**from** keras.models **import** load\_model

*#import image from keras*

**from** tensorflow.keras.preprocessing **import** image

**import** numpy **as** np

*#import cv2*

**import** cv2

*#load the saved model*

model**=**load\_model("/content/drive/MyDrive/IBM PROJECT/dataset/forest.h5")

img**=**image**.**load\_img('/content/drive/MyDrive/IBM PROJECT/dataset/DATA SET/archive/Dataset/Dataset/test\_set/with fire/Bandipur\_fires\_2019.jpg')

plt**.**imshow(img)

plt**.**show()

x**=**image**.**img\_to\_array(img)

res**=**cv2**.**resize(x,dsize**=**(64,64),interpolation**=**cv2**.**INTER\_CUBIC)

*#expand the image shape*

x**=**np**.**expand\_dims(res,axis**=**0)



In [32]:

pred**=**model**.**predict(x)

pred **=** int(pred[0][0])

pred

int(pred)

1/1 [==============================] - 0s 56ms/step

Out[32]:

0

In [33]:

**if** pred**==**0:

print('Forest fire')

**from** twilio.rest **import** Client

print('Forest fire')

account\_sid**=**'AC34c4bee5e03df7bc7dba1eef29761275'

auth\_token**=**'1fc522239435d0c251c1fd870d715295'

client**=**Client(account\_sid,auth\_token)

message**=**client**.**messages \

**.**create(

body**=**'forest fire is detected,stay alert',

*#use twilio free number*

from\_**=**'+19803934024',

*#to number*

to**=**'+919962082226')

print(message**.**sid)

print("Fire detected")

print("SMS Sent!")

**elif** pred**==**1:

print('No Fire')

Forest fire

Forest fire

SM5b6e9adf70d50c9b7c39e5a82faae39d

Fire detected

SMS Sent!