**Project title:**  Emerging Methods for Early Detection of Forest Fires

**TEAM ID**: PNT2022TMID13265

**TEAM MEMBERS**: HABIBA BANU S

ELLAMMAL V

AARTHI R

RAJASHELSHITA

**Introduction:**

**1.1 Project overview:**

Fire can make major hazards in this hectic world. All buildings and vehicles used in public transportation have fire prevention and fire protection systems due to the accelerated number in the fire incidents. Also, many of the firms conduct a mock fire drill in every occurrence of months to protect their employees from the fire. This would help them to understand what to do or what not to do when a fire situation happens. Forests are one of the main factors in balancing the ecology. It is very harmful when a fire occurs in a forest. But most of the time, the detection of forest fire happens when it spread over a wide region. Sometimes, it could not be possible to stop the fire. As a result, the damage of the environment is higher than predictable. The emission of large amount of carbon dioxide (CO2) from the forest fire damages the environment. As well as it would lead to complete disappearance of rare species in the world (Alkhatib, 2014). Also, it can make an impact on the weather, and this make major issues like earthquakes, heavy rains, floods and so on.

A research study shows an automatic fire detection can be divided into three groups: aerial, ground and borne detection. The ground-based systems use several staring black and white video cameras are used in fire detection which detect the smoke and compares it with the natural smoke. The main benefit of using this system is high temporal resolution and spatial resolution. So that, the detection is easier (Eric den breejen, 1998). But these mechanisms still have some drawbacks in detecting the early stage of the fire. So that, it is highly important to introduce a system to detect the fire early as possible.

*1.2 Purpose:*

**we propose a novel system for detecting fire using Convolutional Neural Networks (CNN). Detection of fire can be extremely difficult using existing methods of smoke sensors installed in the buildings. They are slow and cost inefficient due to their primitive design and technology. This paper critically analyzes the scope of Artificial intelligence for detection and sending alerts with video from CCTV footages. This project uses self-built dataset containing video frames with fire. The data is then preprocessed and use the CNN to build a machine learning model. The test set of the dataset is given as input for validating the algorithm and experiments are noted. The project focus on building cost efficient and highly accurate machine that can be used in almost any use case of fire detection.**

***Keywords:Fire detection, Convolutional neural networks, Machine learning, CCTV, Object detection.***

2. LITERATURE SURVEY:

2.1 Existing Problem:

The existing system for detecting fire are smoke alarms and heat alarms. The main disadvantage of the smoke sensor alarm and heat sensor alarms are that just one module is not enough to monitor all the potential fire prone places. The only way to prevent a fire is to be cautious al the time. Even if they are installed in every nook and corner, it just is not sufficient for an efficient output consistently. As the number of smoke sensor requirement increase the cost will also increase to its multiple. The proposed system can produce consistent and highly accurate alerts within seconds of accident of the fire. It reduces cost because only one software is enough to power the entire network of surveillance. Research is active on this field by data scientists and machine learning researchers. The real challenge is to minimize the error in detection of fire and sending alerts at the right time.

2.2 References

[1] Saponara, S., Elhanashi, A. & Gagliardi, A. Real-time video fire/smoke detection based on CNN in antifire surveillance systems. J Real-Time Image Proc 18, 889–900 (2021).

[2]Qingjie Zhang, Jiaolong Xu, Liang Xu and Haifeng Guo, Deep Convolutional Neural Networks for Forest Fire Detection. IFMEITA 2016.

[3] Pragati, S. S. (2019-2020). International Journal Of Advance Scientific Research. Forest Fire Detection Using Machine Learning,

[

4] A Arul, R. S. (2021, May). Fire Detection System Using Machine Learning. Retrieved from ResearchGate: <https://www.researchgate.net/publication/351926970_Fire_Detection_System_Using_Machine_Learning>.

[5] Yuanbin Wang, L. D. (2019). Journal of algorithms and Computational technology. Forest fire image recognition based on convolutional neural network, 1.

[6] Kim, Byoungjun, and Joonwhoan Lee. 2019. "A Video-Based Fire Detection Using Deep Learning Models" Applied Sciences 9, no. 14: 2862

[7] Xu, R.; Lin, H.; Lu, K.; Cao, L.; Liu, Y. A Forest Fire Detection System Based on Ensemble Learning. Forests **2021**, 12, 217. https://doi.org/10.3390/f12020217

[8] BoWFire Dataset. Available online: https://bitbucket.org/gbdi/bowfire-dataset/downloads/ (accessed on 1 January 2021). 29.

[9] FD-Dataset. Available online: http://www.nnmtl.cn/EFDNet/ (accessed on 1 January 2021).

[10] ForestryImages. Available online: https://www.forestryimages.org/browse/subthumb.cfm?sub=740 (accessed on 1 January 2021). 31. [10].

[11] VisiFire. Available online: http://signal.ee.bilkent.edu.tr/VisiFire/ (accessed on 1 January 2021).

Problem Statement:

• Wildfires are blazes that are uncontrolled and fueled by different types of weather, dry underbrush, and wind. They cause significant destruction in a very short amount of time and are difficult to extinguish.

• Over 9 million acres of land have been destroyed due to treacherous wildfires. Wildfires also cause 339,000 deaths a year. In contrast, prescribed fires (fires ignited for ecosystem restoration and reduction of risk of wildfires) are typically managed to minimize downwind impacts on populated areas, however those in close proximity may be exposed to smoke 4.

• Multi-temporal threshold algorithm in forest fire detection using MSG satellite was the solution built in the Zimbabwe 1.

• A solution is required to monitor and display key atmospheric data in wildfire-prone areas so that potential wildfires can be identified before they grow significantly large. The benefits or gains:

• Decrease in animal, plant and human deaths due to wildfires.

• The effects of smoke and pollution can be reduced. Primary beneficiary:

• People living in wildfire prone areas.

• Firms and industries dependent on forest produce.

Secondary beneficiary:

• Government. The minimum set of features/functionality:

• Solution should be capable of on-board data storage as well as wireless transmission of data to the central receiving unit at the 5-minute measurement time resolution.

• The data transmission range from should support a minimum of 15 km distance.

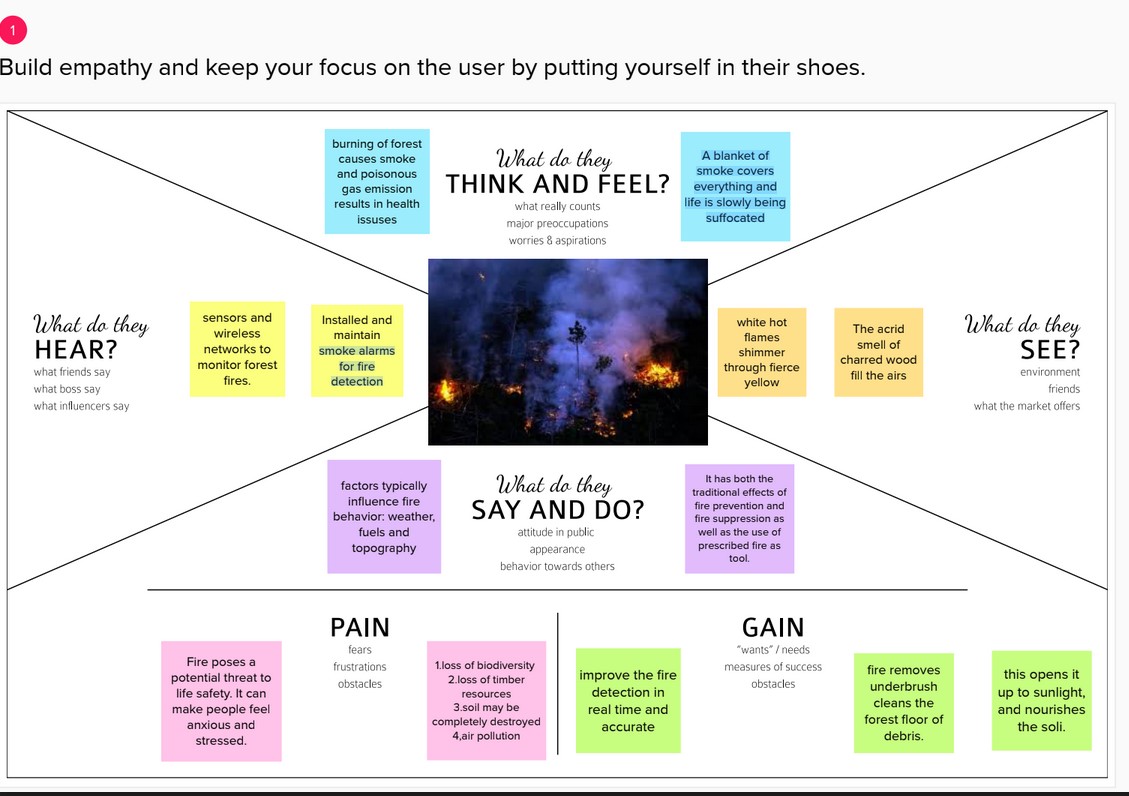
• The system should require minimal infrastructure and easy set-up.

• The system should be capable of operation for a minimum of 15 days without operator maintenance. The market and business potential:

• Solution units could be licensed or sold to people and industries living in wildland fire prone areas. Revenue generated from maintenance and sale.

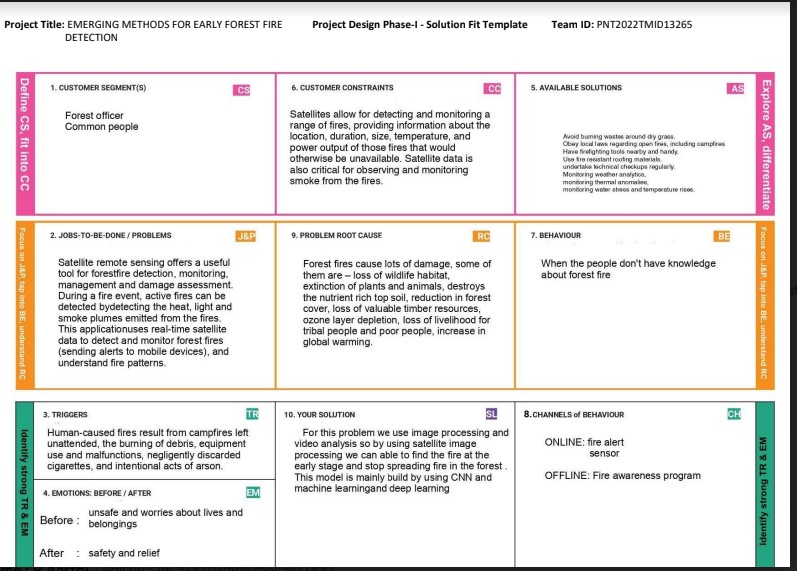
**Ideation & Proposed Solution:**

**Empathy Map Canvas:**



**Ideation & Brainstorming:**

**Proposed solution:**



**Proposed solution fit:**

|  |  |  |
| --- | --- | --- |
| **S/no** | **Parameter** | **Description** |
| • | ProblemStatement(Problemtobesolved) | Aforestfireriskpredictionalgorithm,basedonsupport vector machines, is presented. Thealgorithm depends on previous weatherconditionsinordertopredictthefirehazard  levelofaday. |
| • | Idea/Solutiondescription | Use computer vision methods for recognitionand detection of smoke or fire, based on thestillimagesorthevideoinputfromthedrone  cameras. |
| • | Novelty/Uniqueness | Realtimecomputerprogramdetectforestfireinearliestbeforeitspreadtolargerarea. |
| • | Impactonsociety | Blockedroadsandrailwaylines,electricity,mobile and land telephone lines cut,destructionofhomesandindustries. |
| • | BusinessModel(RevenueModel) | The proposed method was implemented usingthe Python programming language on a Core i3orgreater(CPU and4GBRAM.) |
| • | ScalabilityoftheSolution | Computer vision models enable land coverclassificationandsmokedetectionfrom  satelliteandgroundcameras |

**4. REQUIREMENT ANALYSIS :**

**4.1FUNCTIONAL REQUIREMENT:**

SYSTEM : 13

HARD DISK :500

RAM :4gb DDR2

BOARD :LG 104 KEYS KEYBOARD

MOUSE ;LOGITRCH

MONITOR ; 15INCH TFT COLOR MONITOR

**4.2 NON-FUNCTIONAL REQUIREMENT :**

Software Specifications Google Colaboratory — Colaboratory is a free Jupyter notebook environment provided by Google where one can use free GPUs and TPUs which requires no setup and runs entirely in the cloud. The Jupyter Notebook is an open-source web application which allows to create and share documents that contain live code, equations, visualizations and narrative text[11]. A notebook is a list of cells. Cells contain either explanatory text or executable code and its output. With Colaboratory one can write and execute code, save and share their analyses, and access powerful computing resources.

**5.PROJECT DESIGN:**

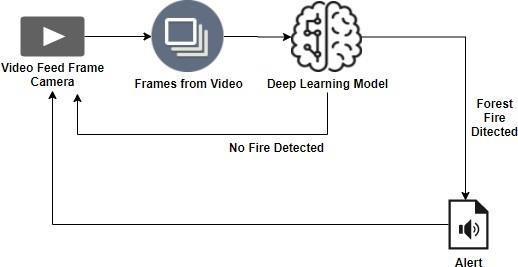
**5.1 DATA FLOW DIAGRAM**

**Data Flow Diagrams:**

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

**Example:**

FLOW □ It is difficult to predict and detect Forest Fire in a sparsely populated forest area.

□ it is more difficult if the prediction is done using ground-based methods like Camera or Video-Based approach.

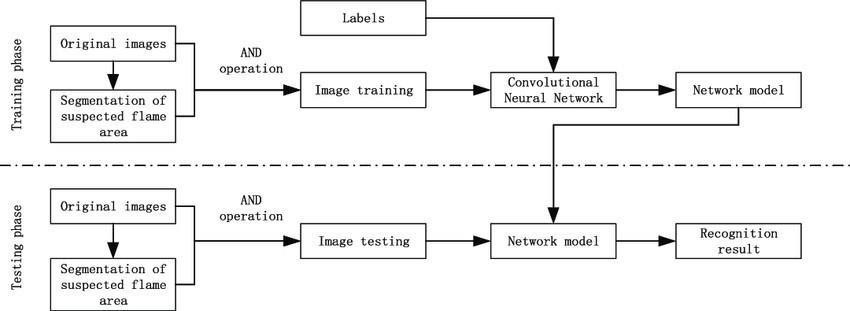
□ Satellites can be an important source of data prior to and also during the Fire due to its reliability and efficiency.

□ The various real-time forest fire detection and prediction approaches, with the goal of

informing the local fire authorities.

□ If the fire is not detected ,it will send the result to the frame camera.if the forest fire will detected the alert will go to the video feed frame camera.

**DFD:**

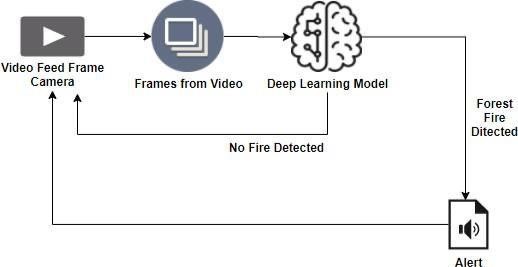


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **User Type** | **Functional**  **Requirement**  **(Epic)** | **User**  **Story**  **Number** | **User Story** I **Task** | **Acceptance criteria** | **Priority** | **Release** |
| Environmenta  list | Collect the data | USN-1 | As an Environmentalist.it isnecessary to collect the data of the | It is necessary tocollect the right | High | Sprint-1 |
|  |  |  | forest which includes | data else the |  |  |
|  |  |  | temperature,humidity,wind and rain | prediction may |  |  |
|  |  |  | of the forest | become wrona |  |  |
|  |  | USN-2 | Identify algorithms that can be used for prediction | To collect the algorithm toidentify the accuracylevel of each algorithms | Medium | Sprint-2 |
|  | Implement | USN-3 | Identify the accuracy of each | Accuracy of each | High | Sprint-2 |
|  | Algorithm |  | algorithms | algorithm-calculated so that it is easy toobtain the  most accurate output |  |  |
|  |  | USN-4 | Evaluate the Dataset | Data is evaluated before processing | Medium | Sprint-1 |
|  | Evaluate | USN-5 | Identify accuracy,precision,recallof | These values are | High | Sprint-3 |
|  | Accuracy of  Algorithm |  | each algorithms | important for obtaining the riahtoutput |  |  |

**SOLUTION AND TECHNICAL ARCHITECTURE:**

**Technical Architecture:**

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **Component** | **Description** |  | **Technology** |
| 1. | User Interface | How user interacts with application e.g. Mobile App, database system |  | HTML, CSS, JavaScript / Angular Js / React Js etc. |
| 2. | Application Logic-1 | Logic for a process in the application |  | Java / Python |

**Table-1 : Components & Technologies:**

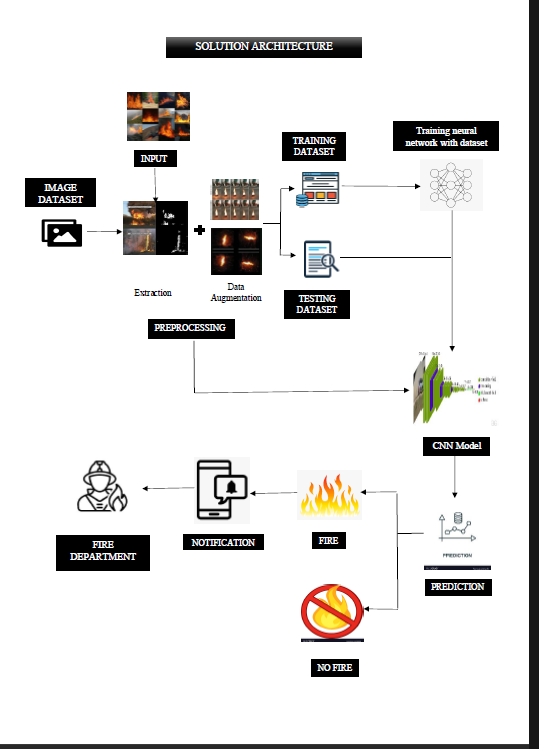
|  |  |  |  |
| --- | --- | --- | --- |
| 3. | Camera | Logic for a process in the application | FPV Camera technology |
| 4. | Smoke sensor | Logic for a process in the application | MQZ, etct |
| 5. | Database | Data Type, Configurations etc. | MySQL, NoSQL, etc. |
| 6. | Cloud Database | Database Service on Cloud | IBM DB2, IBM Cloudant etc. |
| 7. | database system | File storage requirements | Other Storage Service or Local Filesystem |
| 8. | Rotary--wing UAV | Purpose of firefighting used in the application | IBM Weather API, etc. |
| 9. | EFixed--wing UAV | Purpose of weather monitoring.used in the application | Aadhar API, etc. |
| 10. | Machine Learning Model | Purpose of Machine Learning Model | Object Recognition Model, etc. |
| 11. | Infrastructure (Server / Cloud) | Application Deployment on Local System / Cloud Local Server Configuration: | Local, Cloud Foundry, Kubernetes, etc. |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Cloud Server Configuration : |  |

**Table-2: Application Characteristics:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Characteristics** | **Description** | **Technology** |
| 1. | Open-Source Frameworks | List the open-source frameworks used | Technology of Opensource framework |
| 2. | Security Implementations | List all the security / access controls implemented, use of firewalls etc. | e.g. SHA-256, Encryptions, IAM Controls, OWASP etc. |
| 3. | Scalable Architecture | Justify the scalability of architecture (3 – tier, Micro- services) | Technology used |
| 4. | Availability | Justify the availability of application (e.g. use of load balancers, distributed servers etc.) | Technology used |
| 5. | Performance | Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN’s) etc. | Technology used |

**SOLUTION ARCHITECTURE**



USER STORIES:

**6.PROJECT PLANNING &ESTIMATION**

**6.1 SPRINT PLANNING & ESTIMATION &**

**6.2 SPRINT DELIVERY SCHEDULE**

**Product Backlog, Sprint Schedule, and Estimation (4 Marks)**

Use the below template to create product backlog and sprint schedule

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | **Team Members** |
| Sprint-1 | Registration | USN-1 | As a user, I can register for the application by entering my email, password, and confirming my password. | 20 | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |
| Sprint-1 |  | USN-2 | As a user, I will receive confirmation email once I have registered for the application usage. | 20 | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |
| Sprint-2 | Input | USN-3 | Whenever the fire is detected, the information is given to the database. | 20 | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |
| Sprint-2 |  | USN-4 | When it is the wildfire then the alarming system is activated. | 20 | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Total Story Points** | **Duration** | **Sprint Start Date** | **Sprint End Date (Planned)** | **Story Points**  **Completed (as on**  **Planned End Date)** | **Sprint Release Date (Actual)** |
| Sprint-1 | 20 | 6 Days | 24 Oct 2022 | 29 Oct 2022 | 20 | 29 Oct 2022 |
| Sprint-2 | 20 | 6 Days | 31 Oct 2022 | 05 Nov 2022 | 20 | 05 Nov 2022 |
| Sprint-3 | 20 | 6 Days | 07 Nov 2022 | 12 Nov 2022 | 20 | 12 Nov 2022 |
| Sprint-4 | 20 | 6 Days | 14 Nov 2022 | 19 Nov 2022 | 20 | 19 Nov 2022 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sprint** | **Functional**  **Requirement (Epic)** | **User Story Number** | **User Story / Task** | **Story Points** | **Priority** | | **Team Members** |
| Sprint-3 | Output | USN-5 | And the alarm also sent to the corresponding departments and made them know that the wildfire is erupted. | 20 | | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |
| Sprint-4 | Action | USN-6 | Required actions will be taken in order to controlled erupted wildfire by reaching as early as possible to the destination with the help of detecting systems. | 20 | | High | Habiba Banu S  Ellammal V  Aarthi  Rajashelshita |

**Project Tracker, Velocity & Burndown Chart: (4 Marks) Velocity:**

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let’s calculate the team’s average velocity (AV) per iteration unit (story points per

day)

7.**CODING & SOLUTIONING**

**7.1 FEATURE1**

Program

!pip install tensorflow

!pip install opencv-python

!pip install opencv-contrib-python

5s

pip install opencv-python



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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

[3]

import tensorflow as tf  
import numpy as np  
from tensorflow import keras  
import os  
from tensorflow.keras.preprocessing.image import ImageDataGenerator  
from tensorflow.keras.preprocessing import image

[4]

train=ImageDataGenerator(rescale=1./255,  
                                 shear\_range=0.2,  
                                 rotation\_range=180,  
                                 zoom\_range=0.2,  
                                 horizontal\_flip=True)  
train = ImageDataGenerator(rescale=1/255)  
test = ImageDataGenerator(rescale=1/255)

[5]

0s

train\_dataset = train.flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing",  
                                          target\_size=(128,128),  
                                          batch\_size = 32,  
                                          class\_mode = 'binary' )

Found 380 images belonging to 2 classes.

[6]

2s

test\_dataset = test.flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/train\_set",  
  
                                          target\_size=(128,128),  
                                          batch\_size = 32,  
                                          class\_mode = 'binary' )

Found 1520 images belonging to 2 classes.

[7]

0s

test\_dataset.class\_indices

{'fire': 0, 'nofire': 1}

[8]

#to define linear initialisation import sequential  
from keras.models import Sequential  
#to add layer 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')

[9]

model = keras.Sequential()  
model.add(Convolution2D(32,(3,3),input\_shape=(128,128,3),activation='relu'))  
model.add(MaxPooling2D(pool\_size=(2,2)))  
model.add(Convolution2D(32,(3,3),activation='relu'))  
model.add(MaxPooling2D(pool\_size=(2,2)))  
model.add(Convolution2D(32,(3,3),activation='relu'))  
model.add(MaxPooling2D(pool\_size=(2,2)))  
model.add(Convolution2D(32,(3,3),activation='relu'))  
model.add(MaxPooling2D(pool\_size=(2,2)))  
model.add(Flatten())

[10]

model.add(Dense(150,activation='relu'))  
  
model.add(Dense(1,activation='sigmoid'))

[11]

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

[12]

13m

r = model.fit(train\_dataset, epochs = 5, validation\_data = test\_dataset)

Epoch 1/5

12/12 [==============================] - 724s 66s/step - loss: 0.5401 - accuracy: 0.7868 - val\_loss: 0.3640 - val\_accuracy: 0.8329

Epoch 2/5

12/12 [==============================] - 19s 2s/step - loss: 0.3495 - accuracy: 0.8684 - val\_loss: 0.2594 - val\_accuracy: 0.9171

Epoch 3/5

12/12 [==============================] - 20s 2s/step - loss: 0.2189 - accuracy: 0.9237 - val\_loss: 0.1627 - val\_accuracy: 0.9454

Epoch 4/5

12/12 [==============================] - 19s 2s/step - loss: 0.1603 - accuracy: 0.9526 - val\_loss: 0.1404 - val\_accuracy: 0.9520

Epoch 5/5

12/12 [==============================] - 21s 2s/step - loss: 0.1412 - accuracy: 0.9421 - val\_loss: 0.1792 - val\_accuracy: 0.9257

[13]

11s

predictions = model.predict(test\_dataset)  
predictions = np.round(predictions)

48/48 [==============================] - 12s 239ms/step

[14]

0s

print(len(predictions))

1520

[15]

model.save("/content/forest1.h5")

[16]

#import load\_model from keras.model  
from keras.models import load\_model  
#import image class from keras  
import tensorflow as tf  
from tensorflow.keras.preprocessing import image  
#import numpy  
import numpy as np  
#import cv2  
import cv2

[17]

model = load\_model("/content/forest1.h5")

[18]

0s

import matplotlib.pyplot as plt  
plt.plot(r.history['loss'],label='loss')  
plt.plot(r.history['val\_loss'],label='val\_loss')  
plt.legend()

[19]

1s

plt.plot(r.history['accuracy'],label='acc')  
plt.plot(r.history['val\_accuracy'],label='val\_acc')  
plt.legend()

[22]

def predictImage(filename):  
  img1=image.load\_img(filename,target\_size=(128,128))  
  plt.imshow(img1)  
  y=image.img\_to\_array(img1)  
  x=np.expand\_dims(y,axis=0)  
  val=model.predict(x)  
  print(val)  
  if val==0:  
    plt.xlabel("fire",fontsize=30)  
  elif val==1:  
    plt.xlabel("NO fire",fontsize=30)

[23]

1s

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/fire/fire\_0027.jpg")

[24]

0s

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/nofire/nofire\_0012.jpg")

[25]

6s

pip install twilio

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 7.2 MB/s

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

Collecting PyJWT<3.0.0,>=2.0.0

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

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: 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)

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)

Installing collected packages: PyJWT, twilio

Successfully installed PyJWT-2.6.0 twilio-7.15.2

[26]

7s

pip install playsound

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

Collecting playsound

Downloading playsound-1.3.0.tar.gz (7.7 kB)

Building wheels for collected packages: playsound

Building wheel for playsound (setup.py) ... done

Created wheel for playsound: filename=playsound-1.3.0-py3-none-any.whl size=7035 sha256=3ce1637bf056fee9315ff557393641273f3793c244c71dc4313230f5044f055e

Stored in directory: /root/.cache/pip/wheels/ba/f8/bb/ea57c0146b664dca3a0ada4199b0ecb5f9dfcb7b7e22b65ba2

Successfully built playsound

Installing collected packages: playsound

Successfully installed playsound-1.3.0

[27]

5s

pip install opencv-python

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

[28]

0s

#import opencv librariy  
import cv2  
#import numpy  
import numpy as np  
#import image function from keras  
from keras.preprocessing import image  
#import load\_model from keras  
from keras.models import load\_model  
#import client from twilio API  
from twilio.rest import Client  
#imort playsound package  
from playsound import playsound

WARNING:playsound:playsound is relying on another python subprocess. Please use `pip install pygobject` if you want playsound to run more efficiently.

[31]

#load the saved model  
model = load\_model(r'/content/forest1.h5')  
#define video  
video = cv2.VideoCapture('/content/drive/MyDrive/forest fire.mp4')  
#define the features  
name = ['forest','with forest']

[32]

1s

video.isOpened()

True

[33]

from tensorflow.keras.preprocessing import image

[34]

from IPython.display import Audio

[ ]

while(video.isOpened()):

  success,frame=video.read()

  cv2.imwrite("image.jpg",frame)

  img=image.load\_img("/content/image.jpg",target\_size=(128,128))

  x=image.img\_to\_array(img)

  x=np.expand\_dims(x,axis=0)

  pred=model.predict(x)

  p=pred[0]

  print(pred)

  cv2.putText(frame,"predicted class = ",(100,100),cv2.FONT\_HERSHEY\_SIMPLEX, 1, (0,0,0), 1)

  if pred[0]==1:

    account\_sid='AC1a92521871480f58548ab47433527298'

    auth\_token='596097e117fb2295a39c05a192353001'

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

    message=client.messages \

    .create(

        body="Forest fire is detected ,stay alert",

        from\_='+14258421887',

        to='+919150947787')

    print(message.sid)

    print('Fire detected')

    print('SMS sent')

    wn=Audio('/content/tornado-siren.mp3',autoplay=True)

    display(wn)

    break

  else:

    print('No danger')

    break

  if cv2.waitKey(1) & 0xFF==ord('a'):

    break

video.release()

cv2.destroyAllWindows()



**7.2 FEATURE2**

**import**keras

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

In [ ]:

*#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)

In [ ]:

*#Applying ImageDataGenerator functionality to trainset*

x\_train**=**train\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/Forest Fire Dataset',target\_size**=**(128,128),batch\_size**=**32,class\_mode**=**'binary')

Found 1903 images belonging to 2 classes.

In [ ]:

*#Applying ImageDataGenerator functionality to testset*

x\_test**=**test\_datagen**.**flow\_from\_directory('/content/drive/MyDrive/Forest Fire Dataset',target\_size**=**(128,128),batch\_size**=**32,class\_mode**=**'binary')

Found 1903 images belonging to 2 classes.

In [ ]:

*#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

*#importMaxpooling layer*

**from**keras.layers**import** MaxPooling2D

*#import flatten layer*

**from**keras.layers**import** Flatten

**import** warnings

warnings**.**filterwarnings('ignore')

In [ ]:

*#initializing the model*

model**=**Sequential()

In [ ]:

*#add convolutional layer*

model**.**add(Convolution2D(32,(3,3),input\_shape**=**(128,128,3),activation**=**'relu'))

*#addmaxpooling layer*

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

*#add flatten layer*

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

In [ ]:

*#add hidden layer*

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

*#add output layer*

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

In [ ]:

*#configure the learning process*

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

In [ ]:

*#Training the model*

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

Epoch 1/10

14/14 [==============================] - 125s 9s/step - loss: 1.8303 - accuracy: 0.6942 - val\_loss: 0.8162 - val\_accuracy: 0.5625

Epoch 2/10

14/14 [==============================] - 84s 6s/step - loss: 0.7480 - accuracy: 0.7098 - val\_loss: 0.5381 - val\_accuracy: 0.7578

Epoch 3/10

14/14 [==============================] - 59s 4s/step - loss: 0.5674 - accuracy: 0.7612 - val\_loss: 0.4635 - val\_accuracy: 0.8203

Epoch 4/10

14/14 [==============================] - 43s 3s/step - loss: 0.5568 - accuracy: 0.8036 - val\_loss: 0.5385 - val\_accuracy: 0.8125

Epoch 5/10

14/14 [==============================] - 32s 2s/step - loss: 0.5617 - accuracy: 0.7879 - val\_loss: 0.4540 - val\_accuracy: 0.8359

Epoch 6/10

14/14 [==============================] - 25s 2s/step - loss: 0.5714 - accuracy: 0.7723 - val\_loss: 0.5967 - val\_accuracy: 0.7188

Epoch 7/10

14/14 [==============================] - 20s 1s/step - loss: 0.5346 - accuracy: 0.8103 - val\_loss: 0.5203 - val\_accuracy: 0.8047

Epoch 8/10

14/14 [==============================] - 15s 1s/step - loss: 0.5480 - accuracy: 0.7768 - val\_loss: 0.4697 - val\_accuracy: 0.8516

Epoch 9/10

14/14 [==============================] - 14s 1s/step - loss: 0.5030 - accuracy: 0.7991 - val\_loss: 0.4654 - val\_accuracy: 0.8516

Epoch 10/10

14/14 [==============================] - 13s 883ms/step - loss: 0.5373 - accuracy: 0.7879 - val\_loss: 0.5347 - val\_accuracy: 0.8359

Out[ ]:

In [ ]:

*#import load\_model from keras.model*

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

*#import image class from keras*

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

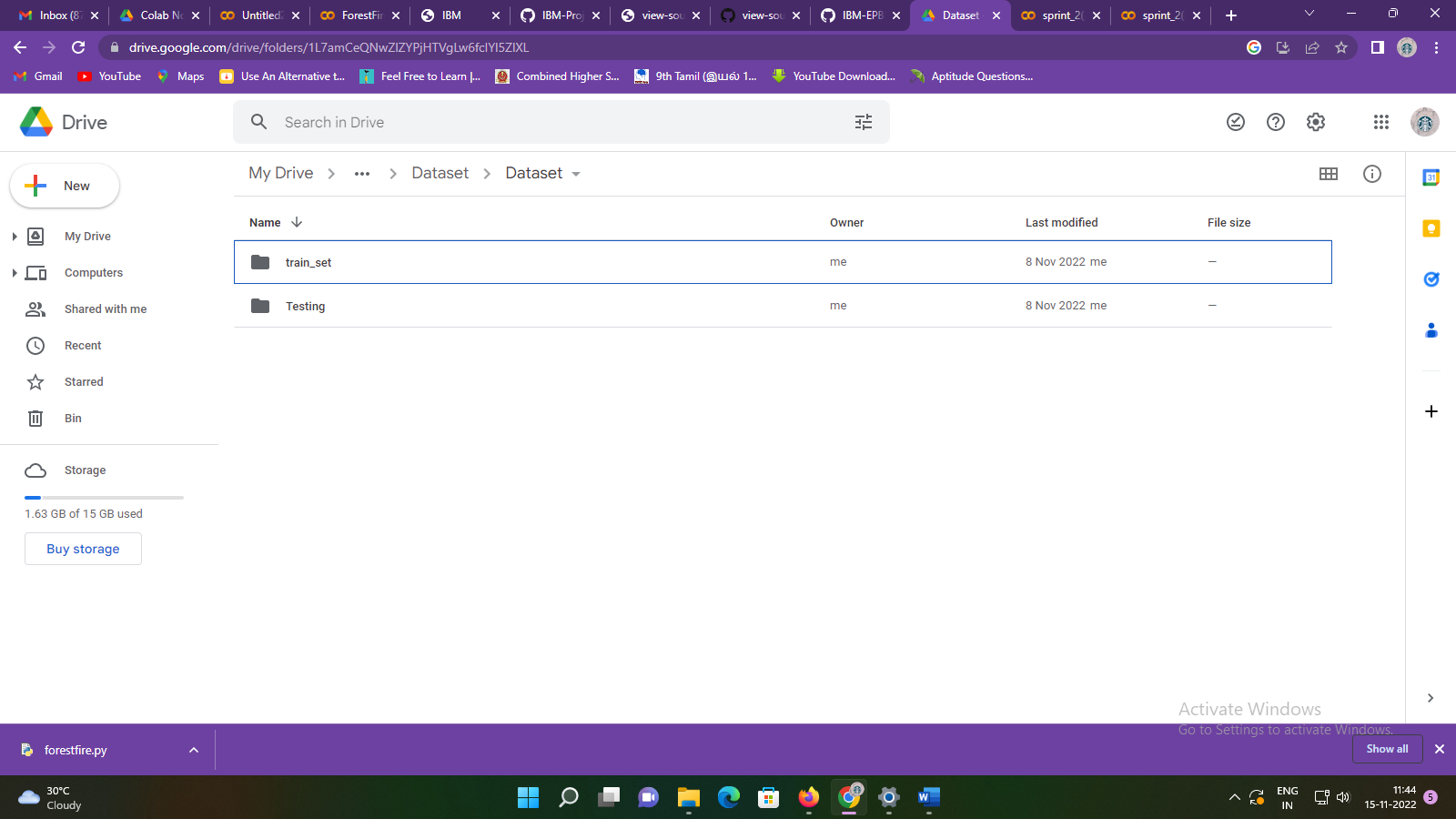
**import**numpy**as** np

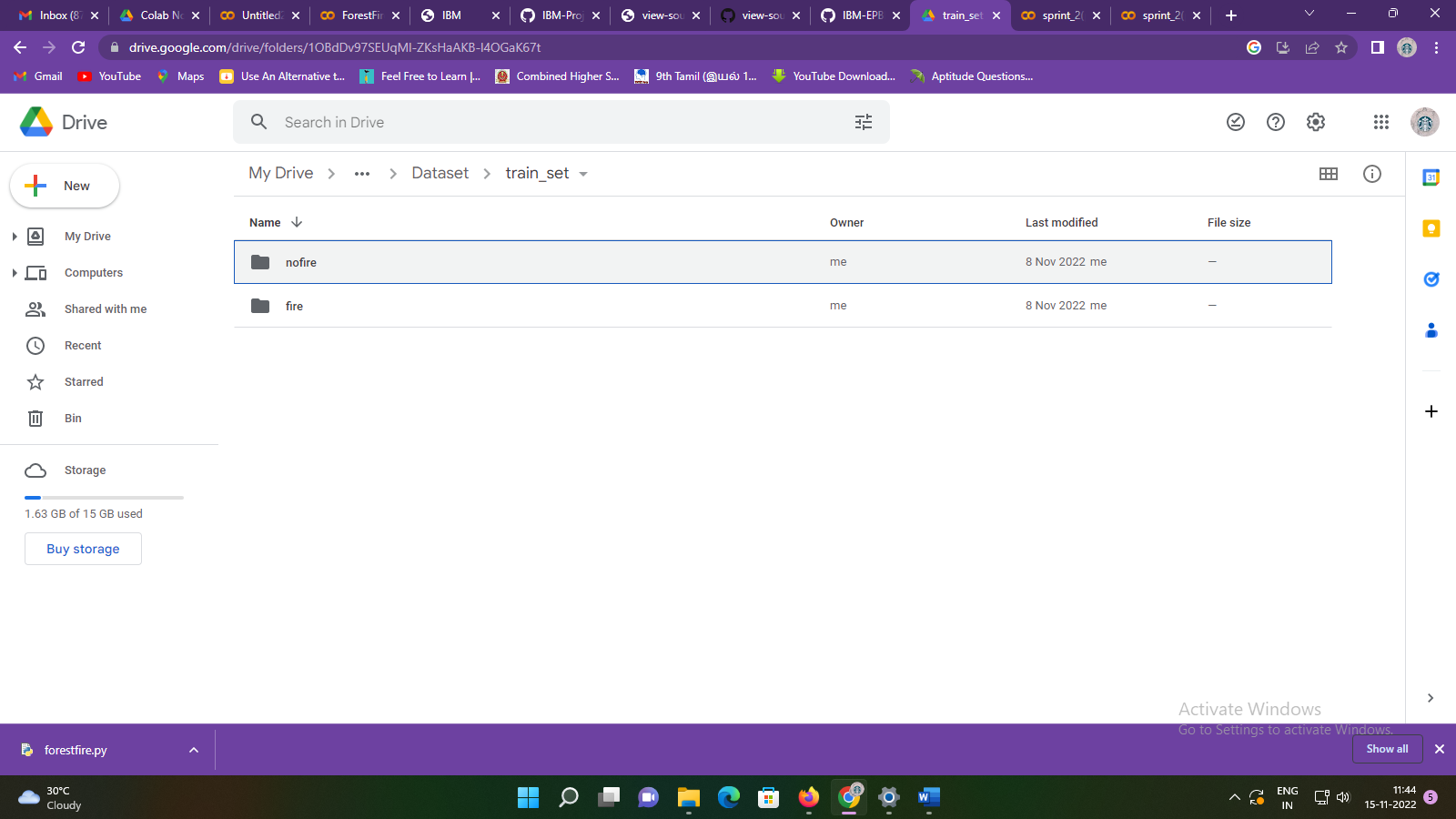
*#import cv2*

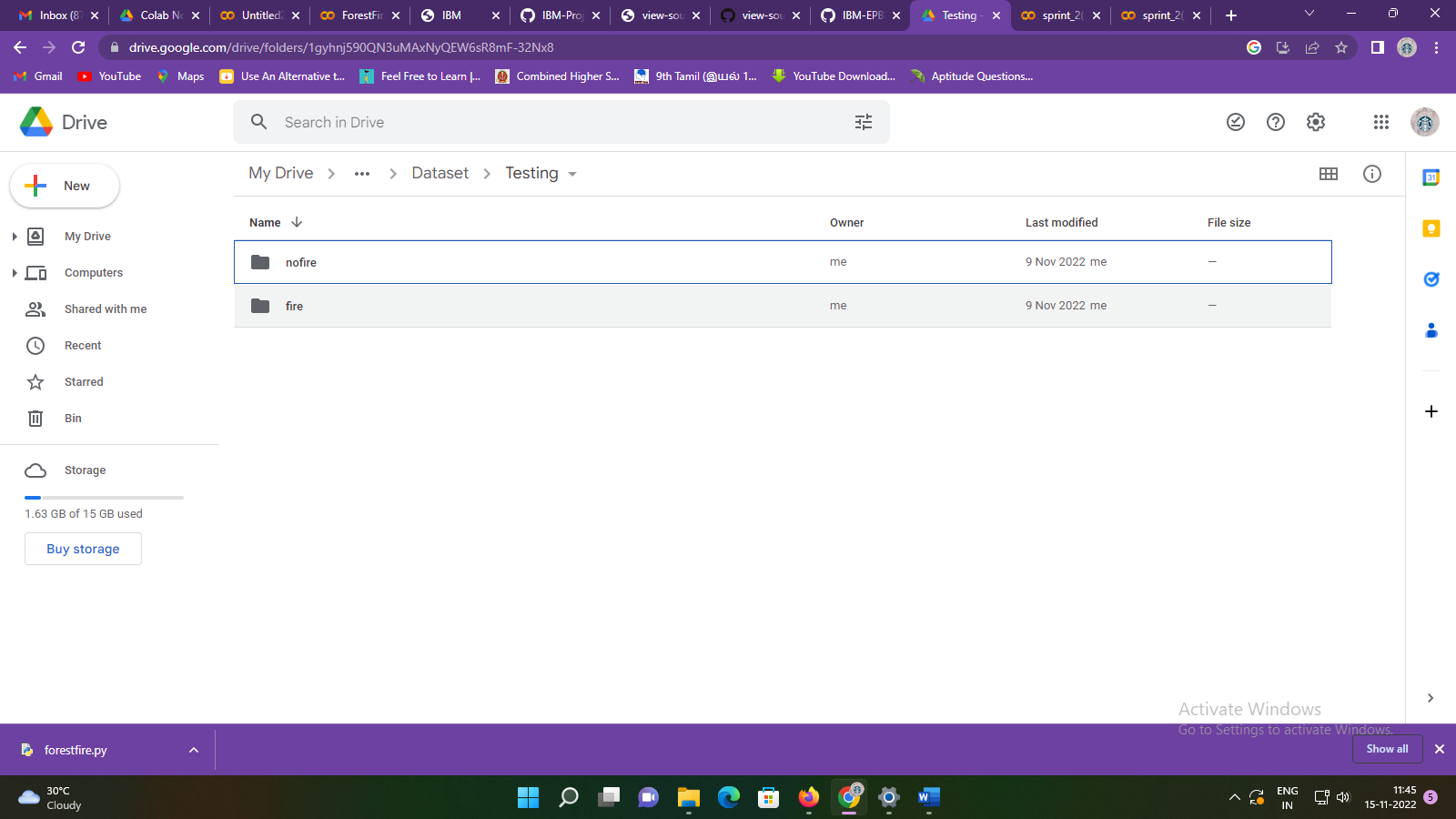
**import** cv2

7.3 DATABASE SCHEMA

**https://drive.google.com/drive/folders/1hyScM8alj3Nfr5jwrN4-Hqyi4XxouaR1?usp=share\_link**







**8.TESTING**

!pip install tensorflow

!pip install opencv**-**python

!pip install opencv**-**contrib**-**python

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

Requirement already satisfied: tensorflow in /usr/local/lib/python3.7/dist-packages (2.9.2)

Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (14.0.6)

Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.1.0)

Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.27.0)

Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from tensorflow) (21.3)

Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.6.3)

Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.2.0)

Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.15.0)

Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.50.0)

Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from tensorflow) (57.4.0)

Requirement already satisfied: keras-preprocessing>=1.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.1.2)

Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.4.0)

Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.3.0)

Requirement already satisfied: tensorboard<2.10,>=2.9 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.1)

Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.21.6)

Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (4.1.1)

Requirement already satisfied: protobuf<3.20,>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.19.6)

Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.3.0)

Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.1.0)

Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.14.1)

Requirement already satisfied: tensorflow-estimator<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.0)

Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.0)

Requirement already satisfied: flatbuffers<2,>=1.12 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.12)

Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (from astunparse>=1.6.0->tensorflow) (0.38.3)

Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h5py>=2.9.0->tensorflow) (1.5.2)

Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (3.4.1)

Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (1.0.1)

Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.6.1)

Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (2.23.0)

Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (1.8.1)

Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.4.6)

Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (2.14.1)

Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (4.9)

Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (5.2.0)

Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.2.8)

Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (1.3.1)

Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist-packages (from markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (4.13.0)

Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (3.10.0)

Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.4.8)

Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (3.0.4)

Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2.10)

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<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (1.24.3)

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2022.9.24)

Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (3.2.2)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->tensorflow) (3.0.9)

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

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

Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-contrib-python) (1.21.6)

In [2]:

pip install opencv**-**python

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

In [3]:

**import**tensorflow**as**tf

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

**from**tensorflow**import**keras

**import**os

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

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

In [4]:

train**=**ImageDataGenerator(rescale**=**1.**/**255,

shear\_range**=**0.2,

rotation\_range**=**180,

zoom\_range**=**0.2,

horizontal\_flip**=True**)

train **=**ImageDataGenerator(rescale**=**1**/**255)

test **=**ImageDataGenerator(rescale**=**1**/**255)

In [5]:

train\_dataset**=**train**.**flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing",

target\_size**=**(128,128),

batch\_size**=** 32,

class\_mode**=** 'binary' )

Found 380 images belonging to 2 classes.

In [6]:

test\_dataset**=**test**.**flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/train\_set",

target\_size**=**(128,128),

batch\_size**=** 32,

class\_mode**=** 'binary' )

Found 1520 images belonging to 2 classes.

In [7]:

test\_dataset**.**class\_indices

Out[7]:

{'fire': 0, 'nofire': 1}

In [8]:

*#to define linear initialisation import sequential*

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

*#to add layer import Dense*

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

*#to create convolution kernel import convolution2D*

**from**keras.layers**import** Convolution2D

*#importMaxpooling layer*

**from**keras.layers**import** MaxPooling2D

*#import flatten layer*

**from**keras.layers**import** Flatten

**import** warnings

warnings**.**filterwarnings('ignore')

In [9]:

model **=**keras**.**Sequential()

model**.**add(Convolution2D(32,(3,3),input\_shape**=**(128,128,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

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

In [10]:

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

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

In [11]:

model**.**compile(loss **=** 'binary\_crossentropy',

optimizer **=** "adam",

metrics **=** ["accuracy"])

In [12]:

r **=**model**.**fit(train\_dataset, epochs **=** 5, validation\_data**=**test\_dataset)

Epoch 1/5

12/12 [==============================] - 724s 66s/step - loss: 0.5401 - accuracy: 0.7868 - val\_loss: 0.3640 - val\_accuracy: 0.8329

Epoch 2/5

12/12 [==============================] - 19s 2s/step - loss: 0.3495 - accuracy: 0.8684 - val\_loss: 0.2594 - val\_accuracy: 0.9171

Epoch 3/5

12/12 [==============================] - 20s 2s/step - loss: 0.2189 - accuracy: 0.9237 - val\_loss: 0.1627 - val\_accuracy: 0.9454

Epoch 4/5

12/12 [==============================] - 19s 2s/step - loss: 0.1603 - accuracy: 0.9526 - val\_loss: 0.1404 - val\_accuracy: 0.9520

Epoch 5/5

12/12 [==============================] - 21s 2s/step - loss: 0.1412 - accuracy: 0.9421 - val\_loss: 0.1792 - val\_accuracy: 0.9257

In [13]:

predictions **=**model**.**predict(test\_dataset)

predictions **=**np**.**round(predictions)

48/48 [==============================] - 12s 239ms/step

In [14]:

print(len(predictions))

1520

In [15]:

model**.**save("/content/forest1.h5")

In [16]:

*#import load\_model from keras.model*

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

*#import image class from keras*

**import**tensorflow**as**tf

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

*#importnumpy*

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

*#import cv2*

**import** cv2

In [17]:

model **=**load\_model("/content/forest1.h5")

In [18]:

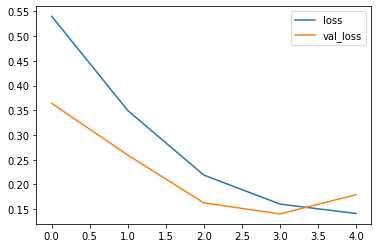
**import**matplotlib.pyplot**as**plt

plt**.**plot(r**.**history['loss'],label**=**'loss')

plt**.**plot(r**.**history['val\_loss'],label**=**'val\_loss')

plt**.**legend()

Out[18]:

****

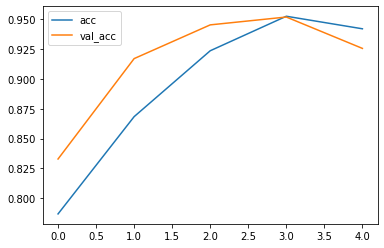
In [19]:

plt**.**plot(r**.**history['accuracy'],label**=**'acc')

plt**.**plot(r**.**history['val\_accuracy'],label**=**'val\_acc')

plt**.**legend()

Out[19]:

****

In [22]:

**def**predictImage(filename):

img1**=**image**.**load\_img(filename,target\_size**=**(128,128))

plt**.**imshow(img1)

y**=**image**.**img\_to\_array(img1)

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

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

print(val)

**if**val**==**0:

plt**.**xlabel("fire",fontsize**=**30)

**elif**val**==**1:

plt**.**xlabel("NO fire",fontsize**=**30)

In [23]:

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/fire/fire\_0027.jpg")

1/1 [==============================] - 0s 25ms/step

[[0.]]

****

In [24]:

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/nofire/nofire\_0012.jpg")

1/1 [==============================] - 0s 23ms/step

[[1.]]

****

In [25]:

pip install twilio

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 7.2 MB/s

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

Collecting PyJWT<3.0.0,>=2.0.0

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

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: 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)

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)

Installing collected packages: PyJWT, twilio

Successfully installed PyJWT-2.6.0 twilio-7.15.2

In [26]:

pip install playsound

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

Collecting playsound

Downloading playsound-1.3.0.tar.gz (7.7 kB)

Building wheels for collected packages: playsound

Building wheel for playsound (setup.py) ... done

Created wheel for playsound: filename=playsound-1.3.0-py3-none-any.whl size=7035 sha256=3ce1637bf056fee9315ff557393641273f3793c244c71dc4313230f5044f055e

Stored in directory: /root/.cache/pip/wheels/ba/f8/bb/ea57c0146b664dca3a0ada4199b0ecb5f9dfcb7b7e22b65ba2

Successfully built playsound

Installing collected packages: playsound

Successfully installed playsound-1.3.0

In [27]:

pip install opencv**-**python

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

In [28]:

*#importopencvlibrariy*

**import** cv2

*#importnumpy*

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

*#import image function from keras*

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

*#importload\_model from keras*

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

*#import client from twilio API*

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

*#imortplaysound package*

**from**playsound**import**playsound

WARNING:playsound:playsound is relying on another python subprocess. Please use `pip install pygobject` if you want playsound to run more efficiently.

In [31]:

*#load the saved model*

model **=**load\_model(r'/content/forest1.h5')

*#define video*

video **=** cv2**.**VideoCapture('/content/drive/MyDrive/forest fire.mp4')

*#define the features*

name **=** ['forest','with forest']

In [32]:

video**.**isOpened()

Out[32]:

True

In [33]:

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

In [34]:

**from**IPython.display**import** Audio

In [ ]:

**while**(video**.**isOpened()):

success,frame**=**video**.**read()

cv2**.**imwrite("image.jpg",frame)

img**=**image**.**load\_img("/content/image.jpg",target\_size**=**(128,128))

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

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

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

p**=**pred[0]

print(pred)

cv2**.**putText(frame,"predicted class = ",(100,100),cv2**.**FONT\_HERSHEY\_SIMPLEX, 1, (0,0,0), 1)

**if**pred[0]**==**1:

account\_sid**=**'AC1a92521871480f58548ab47433527298'

auth\_token**=**'596097e117fb2295a39c05a192353001'

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

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

**.**create(

body**=**"Forest fire is detected ,stay alert",

from\_**=**'+14258421887',

to**=**'+919150947787')

print(message**.**sid)

print('Fire detected')

print('SMS sent')

wn**=**Audio('/content/tornado-siren.mp3',autoplay**=True**)

display(wn)

**break**

**else**:

print('No danger')

**break**

**if** cv2**.**waitKey(1) **&** 0xFF**==**ord('a'):

**break**

video**.**release()

cv2**.**destroyAllWindows()

**9.RESULT:**

**9.1 PERFORMANCE METRICES**

IX PERFORMANCE EVALUATION In this project, as of now, we have worked with two different machine learning models. We calculated the accuracy of these models. The comparison of these models is as follows: SVM Decision Tree Accuracy 0.62 0.99 Precision 0 0.54 0.98 Precision 1 0.76 1 Recall 0 0.78 0.99 Recall 1 0.51 0.98 Based on these observations after our experiment and analysis we can clearly compare the performance of the models to predict the chances of fire.

10.ADVANTAGE AND DISADVANTAGE:

Every year it seems like there’s another disastrous wildfire in the American West. In 2018, nearly 9 million acres were burned in the US alone. Uncontrolled fires often started accidentally by people, rampage and decimate forests. For most people, a forest fire is synonymous with disaster. But there are some kinds of forest fires that actually benefit the environment.

From forests to deserts, wildfires affect air quality, vegetation, human and animal habitats, and climate around the world. Fire managers and researchers are finding ways to use NASA data to battle fires and measure their effects. Burning fires produce both ashes, which falls to the ground like snow but can also get caught up in winds, and smoke, a mixture of gases and particulate matter. These get into the atmosphere and can travel long distances impacting air quality regionally. Wildfires are unplanned fires that start in forests or wildland areas. There are numerous post-fire impacts, including an increase in air pollution and less infiltration of precipitation, contributing to flooding hazards even long after the burn.

A controlled burn is a wildfire that people set intentionally for a specific purpose. Well-thought-out and well-managed controlled burns can be incredibly beneficial for forest management, in part because they can help stop an out-of-control wildfire. The technique is called backburning, and it involves setting a controlled fire in the path of the approaching wildfire. All the flammable material is burnt up and extinguished. When the wildfire approaches, there’s no more fuel left for it to keep going, and it dies out. Forest fire science entails understanding how a fire starts, what contributes to the fire and how the fire might impact future Earth processes. Understanding climatological changes are important to understand how these changes may contribute to fires in the future.

Controlled burns are also used to prevent forest fires. Even before human involvement, natural, low-intensity wildfires occurred every few years to burn up fuel, plant debris, and dead trees, making way for young, healthy trees and vegetation to thrive. That new growth in turn supports forest wildlife. Forest managers are now replicating this natural strategy when appropriate, starting manageable, slow-burning fires to make room for the new life that will help keep the forest healthy in the long term.

The same method is one of WWF’s strategies for maintaining grassland habitats in the Northern Great Plains. Working with partners such as the U.S. Fish and Wildlife Service, WWF has intentionally burned hundreds of acres of prairie land to revitalize these key habitats. The fire burns off tall, aggressive vegetation that isn’t as hospitable to wildlife, and makes room for new growth that attracts bison, birds, and prairie dogs.

This doesn’t mean all intentional wildfires are good. Many of the fires intentionally set for agriculture and land clearing are at best ill-advised, and at worst devastating. Slash and burn fires are set every day to destroy large sections of forests. Of course, these forests don’t just remove trees; they kill and displace wildlife, alter water cycles and soil fertility, and endanger the lives and livelihoods of local communities. They also can rage out of control. In 1997, fires set intentionally to clear forests in Indonesia escalated into one of the largest wildfires in recorded history. Hundreds of people died; millions of acres burned; already at-risk species like orangutans perished by the hundreds; and a smoke and ash haze hung over Southeast Asia for months, reducing visibility and causing acute health conditions.

When scientists think a fire could be the best solution for revitalizing wild areas, WWF brings the right experts to the table to study the situation and come up with a plan. All fire is risky. To minimize that risk as much as possible, controlled burns must be well-considered, well-planned, and ignited and maintained by trained professionals. Fire can be a tool for conservation, but only when used the right way.

Since 1990, “Wildland Fires” across Canada have consumed an average of 2.5 million hectares a year. These fires occur in forests, shrublands and grasslands. Some are uncontrolled wildfires started by lightning or human carelessness. A small number are prescribed fires set by authorized forest managers to mimic natural fire processes that renew and maintain healthy ecosystems. Wildland fires present a challenge for forest management because they have the potential to be at once harmful and beneficial. They can threaten communities and destroy vast amounts of timber resources, resulting in costly losses. However, wildland fires are a natural part of the forest ecosystem and important in many parts of Canada for maintaining the health and diversity of the forest. In this way, prescribed fires offer a valuable resource management tool for enhancing ecological conditions and eliminating excessive fuel build-up.

Not all wildland fires should or can be controlled. Forest agencies work to harness the force of natural fire to take advantage of its ecological benefits while at the same time limiting its potential damage and costs. This makes fire control strategies a vital component of forest management and emergency management in Canada. Understanding the complex phenomenon of wildland fire begins with understanding the basic physical aspects of fire and the ecological role of fire in forests and other wildland areas. Increasingly accurate assessments of the fire situation across Canada are now helping land managers use forest science to reduce fire risk and optimize the benefits.

CONCLUSION

From this project we came to the conclusion that decision tree has a remarkable accuracy of 99% in predicting fires in forest areas. This reduces the chances of false alarm to a great extent. Our system is able to differentiate various forest fire scenarios, from initial case (no fire) to detection of fire, fairly accurately. It can accurately determine the growth of fire. This will help in early stages of fire detection and help to confine fire to limited areas before much damage occurs. The system will be very effective in preventing occurrence of false alarms. We aim at monitoring the forests without constant human supervision.

FUTURE SCOPE

This project carries a broad prospective for future. Moreover it is a need for great research to be done in this field in the coming years. In future, our project can be extended towards finding an efficient way of localization of the fire, gravity of fire, direction of spread, area burnt and many more. In our experiment, the process of simulation of forest fire was done by burning the dried leaves directly. We could come up with ways to make this simulation more close to actual forest fires. Moreover, we can include the region

**13.APPENDIX**

**SOURCE CODE**

!pip install tensorflow

!pip install opencv**-**python

!pip install opencv**-**contrib**-**python

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

Requirement already satisfied: tensorflow in /usr/local/lib/python3.7/dist-packages (2.9.2)

Requirement already satisfied: libclang>=13.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (14.0.6)

Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.1.0)

Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.27.0)

Requirement already satisfied: packaging in /usr/local/lib/python3.7/dist-packages (from tensorflow) (21.3)

Requirement already satisfied: astunparse>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.6.3)

Requirement already satisfied: google-pasta>=0.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.2.0)

Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.15.0)

Requirement already satisfied: grpcio<2.0,>=1.24.3 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.50.0)

Requirement already satisfied: setuptools in /usr/local/lib/python3.7/dist-packages (from tensorflow) (57.4.0)

Requirement already satisfied: keras-preprocessing>=1.1.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.1.2)

Requirement already satisfied: gast<=0.4.0,>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (0.4.0)

Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.3.0)

Requirement already satisfied: tensorboard<2.10,>=2.9 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.1)

Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.21.6)

Requirement already satisfied: typing-extensions>=3.6.6 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (4.1.1)

Requirement already satisfied: protobuf<3.20,>=3.9.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.19.6)

Requirement already satisfied: opt-einsum>=2.3.2 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (3.3.0)

Requirement already satisfied: termcolor>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.1.0)

Requirement already satisfied: wrapt>=1.11.0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.14.1)

Requirement already satisfied: tensorflow-estimator<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.0)

Requirement already satisfied: keras<2.10.0,>=2.9.0rc0 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (2.9.0)

Requirement already satisfied: flatbuffers<2,>=1.12 in /usr/local/lib/python3.7/dist-packages (from tensorflow) (1.12)

Requirement already satisfied: wheel<1.0,>=0.23.0 in /usr/local/lib/python3.7/dist-packages (from astunparse>=1.6.0->tensorflow) (0.38.3)

Requirement already satisfied: cached-property in /usr/local/lib/python3.7/dist-packages (from h5py>=2.9.0->tensorflow) (1.5.2)

Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (3.4.1)

Requirement already satisfied: werkzeug>=1.0.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (1.0.1)

Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.6.1)

Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (2.23.0)

Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (1.8.1)

Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (0.4.6)

Requirement already satisfied: google-auth<3,>=1.6.3 in /usr/local/lib/python3.7/dist-packages (from tensorboard<2.10,>=2.9->tensorflow) (2.14.1)

Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (4.9)

Requirement already satisfied: cachetools<6.0,>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (5.2.0)

Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist-packages (from google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.2.8)

Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/dist-packages (from google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (1.3.1)

Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/dist-packages (from markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (4.13.0)

Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (from importlib-metadata>=4.4->markdown>=2.6.8->tensorboard<2.10,>=2.9->tensorflow) (3.10.0)

Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-packages (from pyasn1-modules>=0.2.1->google-auth<3,>=1.6.3->tensorboard<2.10,>=2.9->tensorflow) (0.4.8)

Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (3.0.4)

Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2.10)

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<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (1.24.3)

Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packages (from requests<3,>=2.21.0->tensorboard<2.10,>=2.9->tensorflow) (2022.9.24)

Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packages (from requests-oauthlib>=0.7.0->google-auth-oauthlib<0.5,>=0.4.1->tensorboard<2.10,>=2.9->tensorflow) (3.2.2)

Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/dist-packages (from packaging->tensorflow) (3.0.9)

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

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

Requirement already satisfied: opencv-contrib-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-contrib-python) (1.21.6)

In [2]:

pip install opencv**-**python

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

In [3]:

**import**tensorflow**as**tf

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

**from**tensorflow**import**keras

**import**os

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

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

In [4]:

train**=**ImageDataGenerator(rescale**=**1.**/**255,

shear\_range**=**0.2,

rotation\_range**=**180,

zoom\_range**=**0.2,

horizontal\_flip**=True**)

train **=**ImageDataGenerator(rescale**=**1**/**255)

test **=**ImageDataGenerator(rescale**=**1**/**255)

In [5]:

train\_dataset**=**train**.**flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing",

target\_size**=**(128,128),

batch\_size**=** 32,

class\_mode**=** 'binary' )

Found 380 images belonging to 2 classes.

In [6]:

test\_dataset**=**test**.**flow\_from\_directory("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/train\_set",

target\_size**=**(128,128),

batch\_size**=** 32,

class\_mode**=** 'binary' )

Found 1520 images belonging to 2 classes.

In [7]:

test\_dataset**.**class\_indices

Out[7]:

{'fire': 0, 'nofire': 1}

In [8]:

*#to define linear initialisation import sequential*

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

*#to add layer import Dense*

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

*#to create convolution kernel import convolution2D*

**from**keras.layers**import** Convolution2D

*#importMaxpooling layer*

**from**keras.layers**import** MaxPooling2D

*#import flatten layer*

**from**keras.layers**import** Flatten

**import** warnings

warnings**.**filterwarnings('ignore')

In [9]:

model **=**keras**.**Sequential()

model**.**add(Convolution2D(32,(3,3),input\_shape**=**(128,128,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

model**.**add(Convolution2D(32,(3,3),activation**=**'relu'))

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

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

In [10]:

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

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

In [11]:

model**.**compile(loss **=** 'binary\_crossentropy',

optimizer **=** "adam",

metrics **=** ["accuracy"])

In [12]:

r **=**model**.**fit(train\_dataset, epochs **=** 5, validation\_data**=**test\_dataset)

Epoch 1/5

12/12 [==============================] - 724s 66s/step - loss: 0.5401 - accuracy: 0.7868 - val\_loss: 0.3640 - val\_accuracy: 0.8329

Epoch 2/5

12/12 [==============================] - 19s 2s/step - loss: 0.3495 - accuracy: 0.8684 - val\_loss: 0.2594 - val\_accuracy: 0.9171

Epoch 3/5

12/12 [==============================] - 20s 2s/step - loss: 0.2189 - accuracy: 0.9237 - val\_loss: 0.1627 - val\_accuracy: 0.9454

Epoch 4/5

12/12 [==============================] - 19s 2s/step - loss: 0.1603 - accuracy: 0.9526 - val\_loss: 0.1404 - val\_accuracy: 0.9520

Epoch 5/5

12/12 [==============================] - 21s 2s/step - loss: 0.1412 - accuracy: 0.9421 - val\_loss: 0.1792 - val\_accuracy: 0.9257

In [13]:

predictions **=**model**.**predict(test\_dataset)

predictions **=**np**.**round(predictions)

48/48 [==============================] - 12s 239ms/step

In [14]:

print(len(predictions))

1520

In [15]:

model**.**save("/content/forest1.h5")

In [16]:

*#import load\_model from keras.model*

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

*#import image class from keras*

**import**tensorflow**as**tf

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

*#importnumpy*

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

*#import cv2*

**import** cv2

In [17]:

model **=**load\_model("/content/forest1.h5")

In [18]:

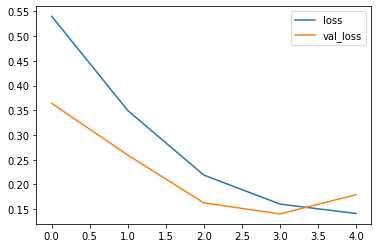
**import**matplotlib.pyplot**as**plt

plt**.**plot(r**.**history['loss'],label**=**'loss')

plt**.**plot(r**.**history['val\_loss'],label**=**'val\_loss')

plt**.**legend()

Out[18]:

****

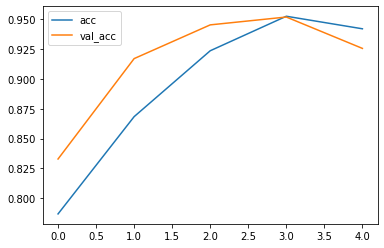
In [19]:

plt**.**plot(r**.**history['accuracy'],label**=**'acc')

plt**.**plot(r**.**history['val\_accuracy'],label**=**'val\_acc')

plt**.**legend()

Out[19]:

****

In [22]:

**def**predictImage(filename):

img1**=**image**.**load\_img(filename,target\_size**=**(128,128))

plt**.**imshow(img1)

y**=**image**.**img\_to\_array(img1)

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

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

print(val)

**if**val**==**0:

plt**.**xlabel("fire",fontsize**=**30)

**elif**val**==**1:

plt**.**xlabel("NO fire",fontsize**=**30)

In [23]:

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/fire/fire\_0027.jpg")

1/1 [==============================] - 0s 25ms/step

[[0.]]

****

In [24]:

predictImage("/content/drive/MyDrive/IBM/final deliverables/Dataset/Dataset/Testing/nofire/nofire\_0012.jpg")

1/1 [==============================] - 0s 23ms/step

[[1.]]

****

In [25]:

pip install twilio

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 7.2 MB/s

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

Collecting PyJWT<3.0.0,>=2.0.0

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

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: 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)

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)

Installing collected packages: PyJWT, twilio

Successfully installed PyJWT-2.6.0 twilio-7.15.2

In [26]:

pip install playsound

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

Collecting playsound

Downloading playsound-1.3.0.tar.gz (7.7 kB)

Building wheels for collected packages: playsound

Building wheel for playsound (setup.py) ... done

Created wheel for playsound: filename=playsound-1.3.0-py3-none-any.whl size=7035 sha256=3ce1637bf056fee9315ff557393641273f3793c244c71dc4313230f5044f055e

Stored in directory: /root/.cache/pip/wheels/ba/f8/bb/ea57c0146b664dca3a0ada4199b0ecb5f9dfcb7b7e22b65ba2

Successfully built playsound

Installing collected packages: playsound

Successfully installed playsound-1.3.0

In [27]:

pip install opencv**-**python

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

Requirement already satisfied: opencv-python in /usr/local/lib/python3.7/dist-packages (4.6.0.66)

Requirement already satisfied: numpy>=1.14.5 in /usr/local/lib/python3.7/dist-packages (from opencv-python) (1.21.6)

In [28]:

*#importopencvlibrariy*

**import** cv2

*#importnumpy*

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

*#import image function from keras*

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

*#importload\_model from keras*

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

*#import client from twilio API*

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

*#imortplaysound package*

**from**playsound**import**playsound

WARNING:playsound:playsound is relying on another python subprocess. Please use `pip install pygobject` if you want playsound to run more efficiently.

In [31]:

*#load the saved model*

model **=**load\_model(r'/content/forest1.h5')

*#define video*

video **=** cv2**.**VideoCapture('/content/drive/MyDrive/forest fire.mp4')

*#define the features*

name **=** ['forest','with forest']

In [32]:

video**.**isOpened()

Out[32]:

True

In [33]:

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

In [34]:

**from**IPython.display**import** Audio

In [ ]:

**while**(video**.**isOpened()):

success,frame**=**video**.**read()

cv2**.**imwrite("image.jpg",frame)

img**=**image**.**load\_img("/content/image.jpg",target\_size**=**(128,128))

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

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

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

p**=**pred[0]

print(pred)

cv2**.**putText(frame,"predicted class = ",(100,100),cv2**.**FONT\_HERSHEY\_SIMPLEX, 1, (0,0,0), 1)

**if**pred[0]**==**1:

account\_sid**=**'AC1a92521871480f58548ab47433527298'

auth\_token**=**'596097e117fb2295a39c05a192353001'

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

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

**.**create(

body**=**"Forest fire is detected ,stay alert",

from\_**=**'+14258421887',

to**=**'+919150947787')

print(message**.**sid)

print('Fire detected')

print('SMS sent')

wn**=**Audio('/content/tornado-siren.mp3',autoplay**=True**)

display(wn)

**break**

**else**:

print('No danger')

**break**

**if** cv2**.**waitKey(1) **&** 0xFF**==**ord('a'):

**break**

video**.**release()

cv2**.**destroyAllWindows()

GITHUB & PROJECT REPORT DEMO LINK:

GITHUB:

[IBM-EPBL](https://github.com/IBM-EPBL)/[**IBM-Project-23061-1659865670**](https://github.com/IBM-EPBL/IBM-Project-23061-1659865670)

PROCJECT REPORT DEMO LINK:

https://youtu.be/1DieUoOvxWY?t=33