RAMAKRISHNA MISSION VIVEKANANDA EDUCATIONAL AND RESEARCH INSTITUTE

(Accredited by NAAC with ‘A++’ Grade)
Coimbatore – 641 020

in Collaboration with

Cognizant

SCHOOL OF MATHEMATICAL SCIENCE

SEPTEMBER – 2020

DEPARTMENT OF COMPUTER SCIENCE
CENTRE OF DATA SCIENCE

NAME	**: BHARATH S**
REG. NO	**: H19MSDS003**
PROGRAMME	**: M.Sc. (Data Science)**
SEMESTER	**: IInd Semester**
PROJECT TITLE	**: FACE AND EMOTION DETECTION**
COURSE CODE	**: 19DS2P06**

RAMAKRISHNA MISSION VIVEKANANDA EDUCATIONAL AND RESEARCH INSTITUTE

(Accredited by NAAC with ‘A++’ Grade)
Coimbatore – 641 020

in Collaboration with

Cognizant

SCHOOL OF MATHEMATICAL SCIENCE

SEPTEMBER – 2020

DEPARTMENT OF COMPUTER SCIENCE
CENTRE OF DATA SCIENCE

Bonafide Certificate

This is to certify that the project work done by **BHARATH S** (**H19MSDS003**) entitled "**Face and Emotion detection**" in his Second Semester during the academic year 2019-2020. Submitted for the Semester viva-voice Examination held on 14-09-2020.

Staff In-Charge Head of the Department

Internal Examiner External Examiner

DECLARATION

I hereby declare that this project entitled "**FACE AND EMOTION DETECTION**" submitted to Ramakrishna Mission Vivekananda Educational and Research Institute, Coimbatore-20. is partial fulfillment of the requirement of the degree in M.Sc., (Data Science) is a record of the original project done by me under the guidance of Dr.R.Sridhar M.Sc., MCA., M.Phil., Ph.D., Professor & Head, Ramakrishna Mission Vivekananda Educational And Research Institute, Coimbatore - 641 020.

Place : Coimbatore

Signature of the Candidate

Date : 14-09-2020

BHARATH S

(H19MSDS003)

ACKNOWLEDGEMENT

My pranams to **Rev. Swami Garishthananda**, Secretary, Ramakrishna Mission Vidyalaya and **Rev. Swami Anapekshanada**, Asst. Administrative Head, FyCSAR, Ramakrishna Mission Vivekananda Educational and Research Institute, Coimbatore - 20 for providing me facilities and encouragement to complete the project work.

I express my profound gratitude to **Dr. R.Sridhar,** Professor & Head, FyCSAR, RKMVERI, Coimbatore-20 for his whole hearted encouragement and timely help.

I placed on record my heartfelt thanks and gratitude to **Sri. D. Raja,** Project Consultant and **Sri. V.Dineshkumar,** Assistant Professor, FyCSAR, RKMVERI, Coimbatore-20 under whose guidance the work has been done. It is a matter of joy that without his valuable suggestions, able guidance, scholarly touch and piercing insight that he offered me in each and every stage of this study, coupled with his unreserved, sympathetic and encouraging attitude, this thesis could not have been presented in this manner.

I am thankful to the following members from the industry for their multi-dimensional review and support to complete my project work.

1. Rohini Krishnan, Senior Manager, Cognizant
2. Manjunaath Alagiriswamy, Senior Associate, Cognizant
3. Muralidharan Sivakumar, Senior Associate, Cognizant
4. Kulshum Azmi, Behavioural Trainer, Cognizant
5. Rekha Priyadharshini, Python SME, Cognizant
6. Rajasekar, Pyhton SME, Cognizant
7. Hemnath Muthukrushnun, Data Scientist SME, Cognizant
8. Shabarivasan RC, Data Scientist SME, Cognizant
9. Balasubramanian Mahadevan, Managing Partner, Algolitics India LLP, Coimbatore

SYNOPSIS

Face and Emotion Detection

The face is one of the easiest ways to distinguish the individual identity of each other. Face recognition is a personal identification system that uses personal characteristics of a person to identify the person's identity.

Human face detection and recognition play important roles in many applications such as video surveillance and face image database management. In our project have studied and worked on both face and emotion detection. In this project we have used Open CV which is used to solve computer vision problems in python. In face and emotion detection algorithm can find the human images in front of camera and it can recognize emotion also. Then we have introduced deep face library which is Deep face is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face , Google FaceNet, OpenFace, Facebook DeepFace, DeepID and Dlib.

In our project pre-trained model which is Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video.

The area of this project face detection system with face recognition is Image processing. The software requirements for this project are python jupyter notebook and Google colab software.

Face and Emotion Detection

Object:

To predict Face and emotion detection through the web camera and face analysis

Importing Libraries

In [1]:

```
import numpy as np
import cv2
```

OpenCV

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection. In this tutorial, we explain how you can use OpenCV in your applications.

In [2]:

```
#pip install opencv-python
```

In [3]:

```
# parameters for loading data and images
```

Haar cascades

Face detection using Haar cascades is a machine learning based approach where a cascade function is trained with a set of input data. OpenCV already contains many pre-trained classifiers for face, eyes, smiles, etc.. Today we will be using the face classifier. You can experiment with other classifiers as well.

In [8]:

```
face_cascade = cv2.CascadeClassifier(r'C:\Users\user\Documents\Real-Time-Emotional-Recognition-with-Deep-Learning-master\Real-Time-Emotional-Recognition-with-Deep-Learning-master\haarcascade\haarcascade_frontalface_default.xml')
eye_cascade = cv2.CascadeClassifier(r'C:\Users\user\Documents\Real-Time-Emotional-Recognition-with-Deep-Learning-master\Real-Time-Emotional-Recognition-with-Deep-Learning-master\haarcascade\haarcascade_eye.xml')
smile_cascade = cv2.CascadeClassifier(r'C:\Users\user\Documents\Real-Time-Emotional-Recognition-with-Deep-Learning-master\Real-Time-Emotional-Recognition-with-Deep-Learning-master\haarcascade\haarcascade_smile.xml')
```

In [11]:

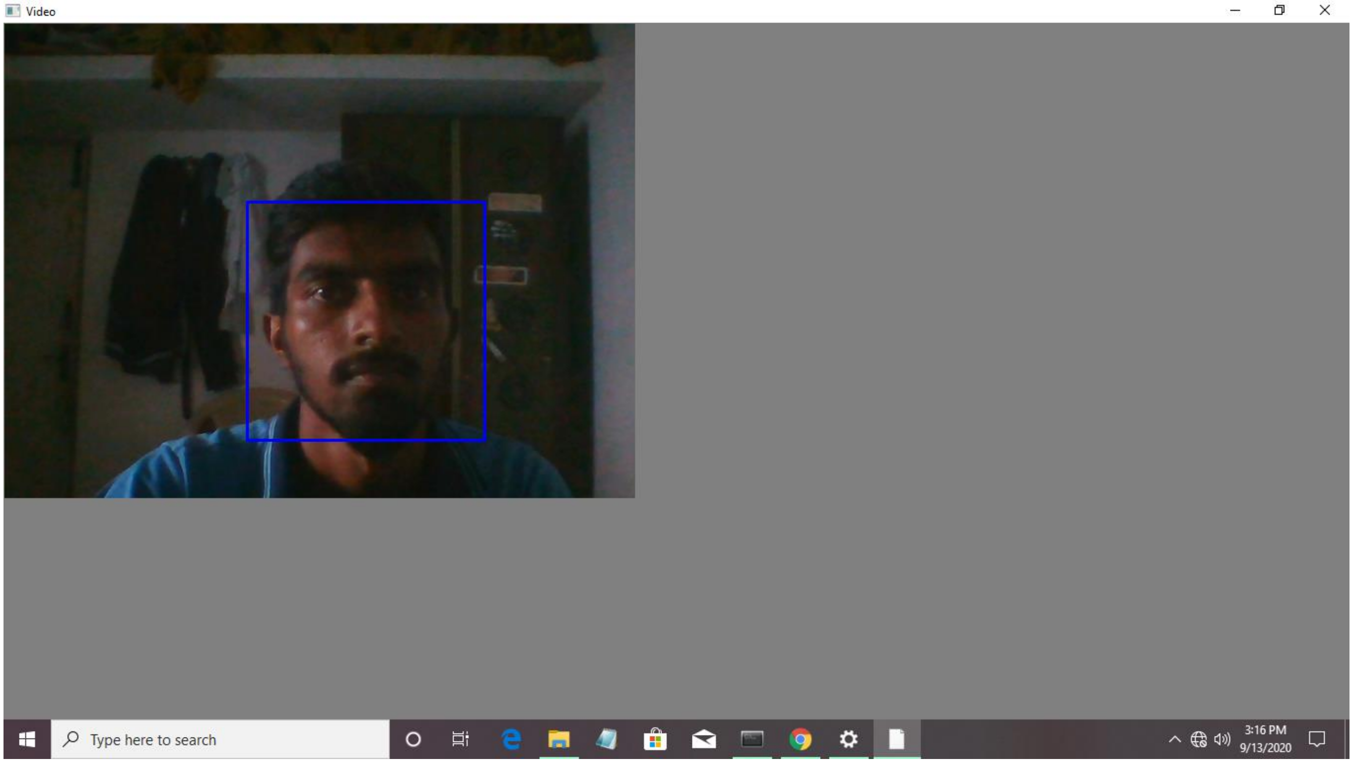
```
def detect(gray, frame):
    faces = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(frame, (x, y), ((x + w), (y + h)), (255, 0, 0), 2)
        roi_gray = gray[y:y + h, x:x + w]
        roi_color = frame[y:y + h, x:x + w]
        smiles = smile_cascade.detectMultiScale(roi_gray, 1.8, 20)

        for (sx, sy, sw, sh) in smiles:
            cv2.rectangle(roi_color, (sx, sy), ((sx + sw), (sy + sh)), (0, 0, 
255), 2)
    return frame
```

In [12]:

```
# To use a video file as input
# cap = cv2.VideoCapture('filename.)
video_capture = cv2.VideoCapture(0)
while True:
   # Captures video_capture frame by frame
    _, frame = video_capture.read()

    # To capture image in monochrome
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    #The detection works only on grayscale images. So it is important to conve
rt the color image to grayscale.

    # calls the detect() function
    canvas = detect(gray, frame)
    #It takes 3 arguments — the input image, scaleFactor and minNeighbours.
    #scaleFactor specifies how much the image size is reduced with each scale.
    #minNeighbours specifies how many neighbors each candidate rectangle shoul
d have to retain it.
    #You can read about it in detail here. You may have to tweak these values
 to get the best results.


    # Displays the result on camera feed
    cv2.imshow('Video', canvas)

    # The control breaks once q key is pressed
    if cv2.waitKey(1) & 0xff == ord('q'):
        break

# Release the capture once all the processing is done.
video_capture.release()
cv2.destroyAllWindows()
```

In [ ]:

In [ ]:


Video
Type here to search
3:16 PM
9/13/2020

In [1]:

```
from keras.preprocessing.image import img_to_array
import imutils
import cv2
from keras.models import load_model
import numpy as np
```

```
Using TensorFlow backend.
```

In [2]:

```
#pip install imutils
```

In [3]:

```
#pip install tensorflow
```

In [4]:

```
#pip install keras
```

Load Haarcascade file and hdf5 file

In [5]:

```
# parameters for loading data and images
detection_model_path = r'C:\Users\user\Documents\Real-Time-Emotional-Recogniti
on-with-Deep-Learning-master\Real-Time-Emotional-Recognition-with-Deep-Learnin
g-master\haarcascade/haarcascade_frontalface_default.xml'
emotion_model_path = r'C:\Users\user\Documents\Real-Time-Emotional-Recognition
-with-Deep-Learning-master\Real-Time-Emotional-Recognition-with-Deep-Learning-
master\pretrained_models/cnn.hdf5'
```

HDF file

The Hierarchical Data Format version 5 (HDF5), is an open source file format that supports large, complex, heterogeneous data. HDF5 uses a "file directory" like structure that allows you to organize data within the file in many different structured ways, as you might do with files on your computer.

In [6]:

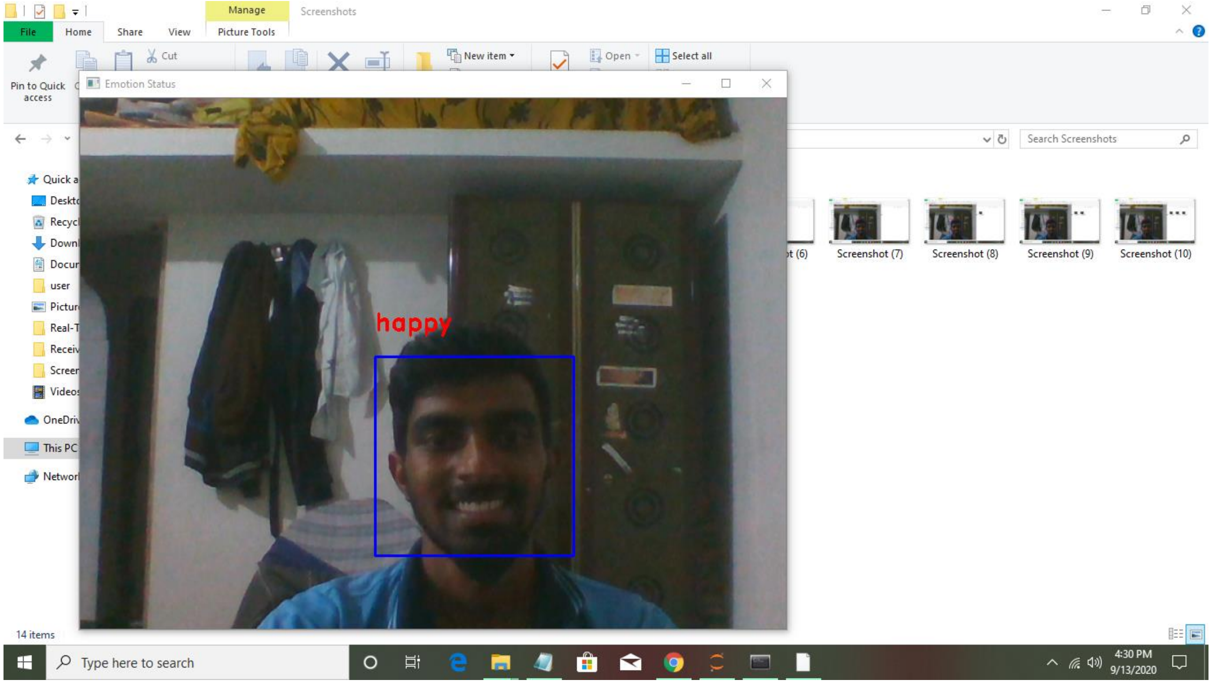
```
# hyper-parameters for bounding boxes shape
# loading models
face_detection = cv2.CascadeClassifier(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)
EMOTIONS = ["angry" ,"disgust","scared", "happy", "sad", "surprised",
 "neutral"]
```

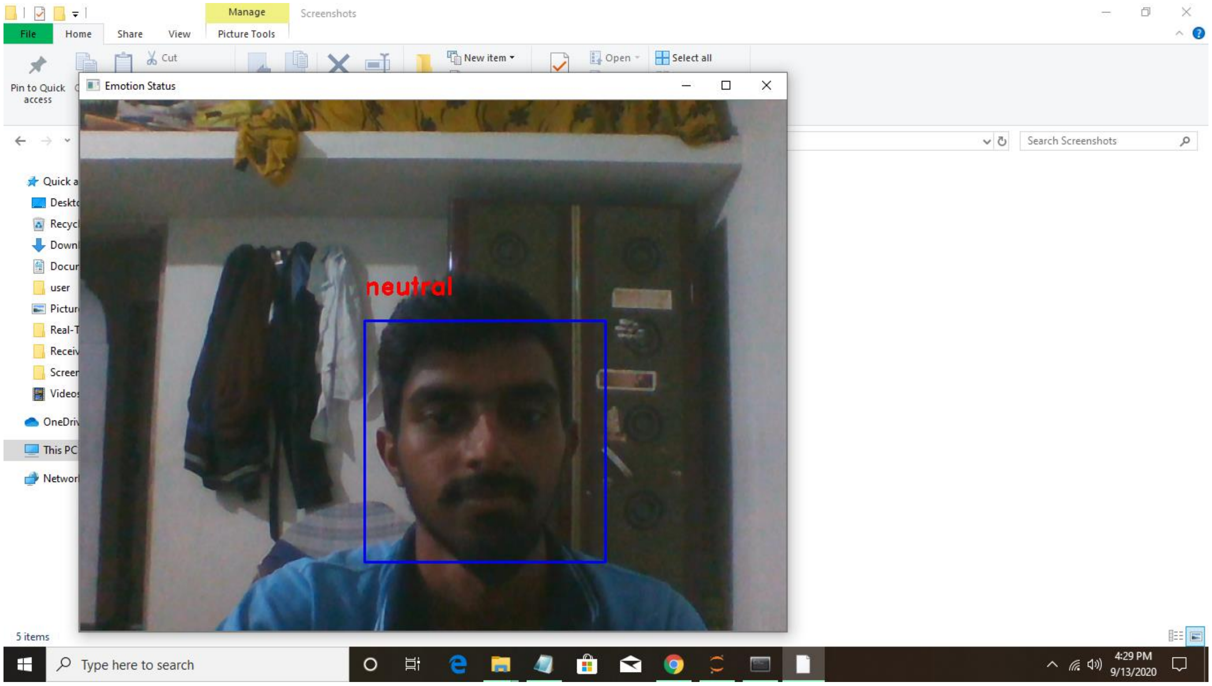
```
WARNING:tensorflow:From C:\Users\user\Anaconda3\lib\site-packages\keras\backe
nd\tensorflow_backend.py:4070: The name tf.nn.max_pool is deprecated. Please
use tf.nn.max_pool2d instead.
```

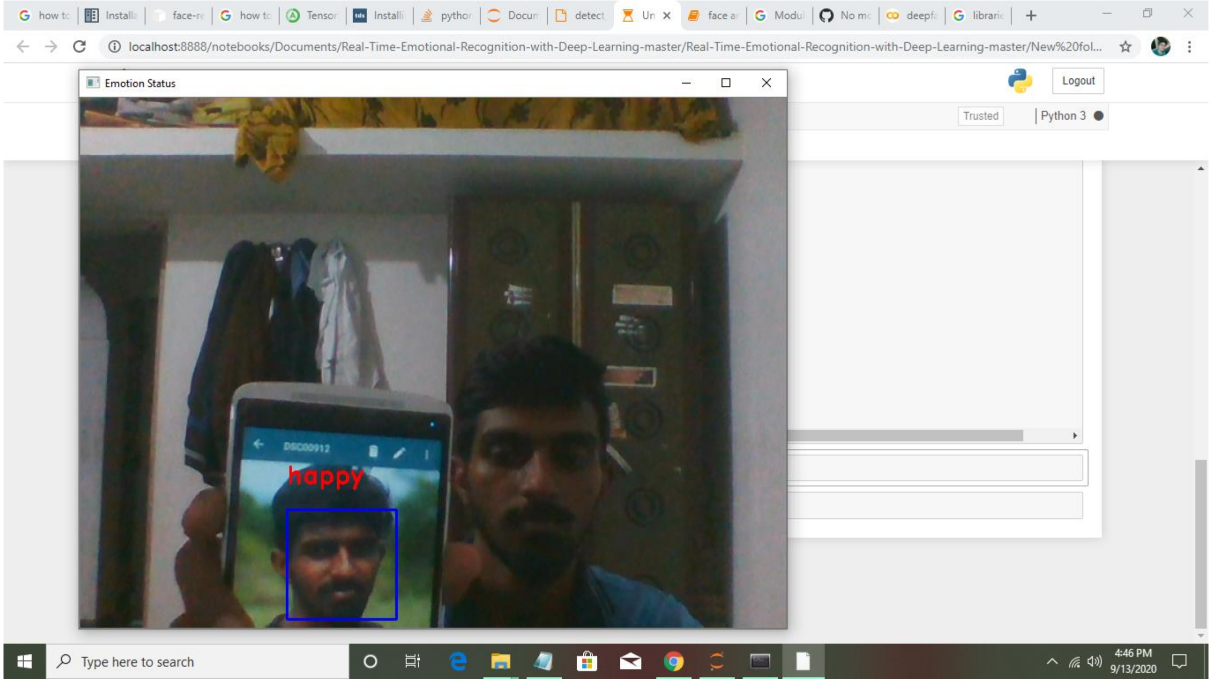
In [ ]:

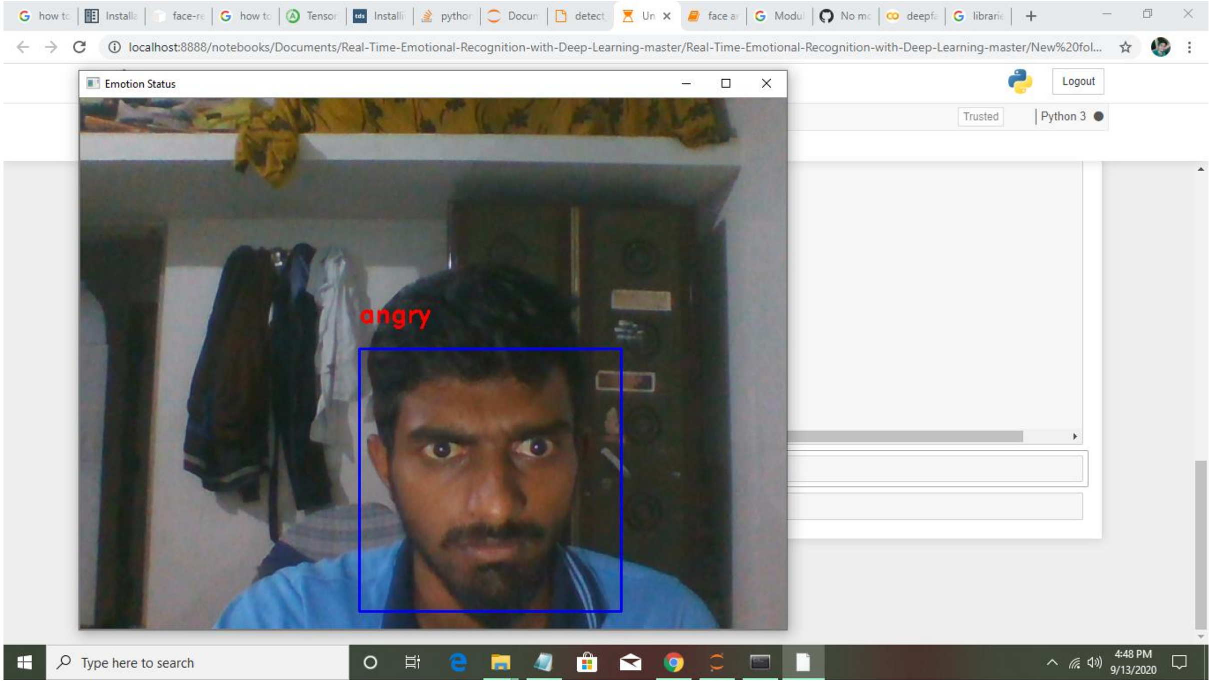
```
feelings_faces = []

camera = cv2.VideoCapture(0)
while True:
    frame = camera.read()[1]
    #reading the frame
    frame = imutils.resize(frame,width=800)
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_detection.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=
5,minSize=(30,30),flags=cv2.CASCADE_SCALE_IMAGE)

    canvas = np.zeros((250, 300, 3), dtype="uint8")
    frameClone = frame.copy()
    if len(faces) > 0:
        faces = sorted(faces, reverse=True,
        key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
        (X, Y, W, H) = faces

        # Extract the facial key point of the face from the grayscale image, r
esize it to a fixed 64x64 pixels(pre-trained model shape)
        # the facial for classification via the CNN
        facial = gray[Y:Y + H, X:X + W]
        facial = cv2.resize(facial, (64, 64))
        facial = facial.astype("float") / 255.0
        facial = img_to_array(facial)
        facial = np.expand_dims(facial, axis=0)


        preds = emotion_classifier.predict(facial)[0]
        emotion_probability = np.max(preds)
        label = EMOTIONS[preds.argmax()]

    for (i, (emotion, prob)) in enumerate(zip(EMOTIONS, preds)):
                # construct the label text
                text = "{}: {:.2f}% ".format(emotion, prob * 100)
                w = int(prob * 300)
                cv2.rectangle(canvas, (7, (i * 35) + 5),
                (w, (i * 35) + 35), (255, 0, 0), -1)
                cv2.putText(frameClone,label,(X, Y - 30),
                cv2.FONT_HERSHEY_DUPLEX, 0.9, (0, 0, 255), 2)
                cv2.rectangle(frameClone, (X, Y), (X + W, Y + H),
                              (255, 0, 0), 2)


    cv2.imshow('Emotion Status', frameClone)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

camera.release()
cv2.destroyAllWindows()
```

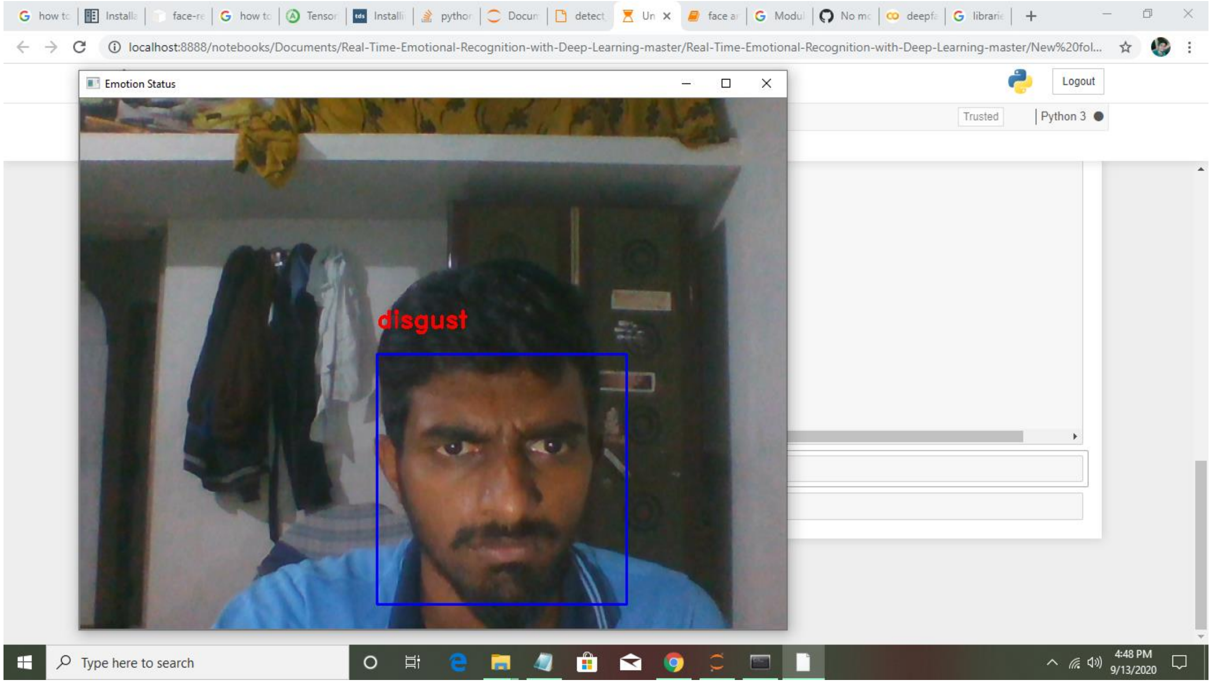
In [ ]:

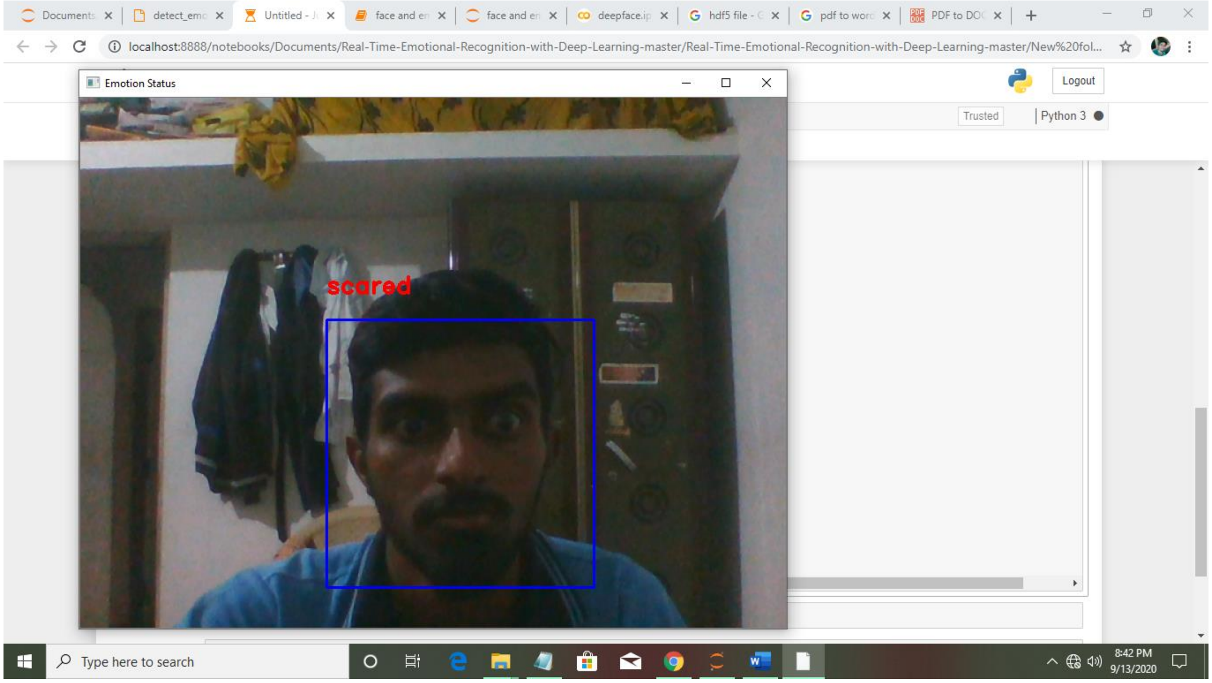

Manage
Screenshots
File
Home
Share
View
Picture Tools
Cut
New item
Open
Select all
Pin to Quick access
Emotion Status
happy
Search Screenshots
Quick a
Deskto
Recycl
Downl
Docur
user
Pictur
Real-T
Receiv
Screen
Videos
OneDriv
This PC
Networ
t (6)
Screenshot (7)
Screenshot (8)
Screenshot (9)
Screenshot (10)
14 items
Type here to search
4:30 PM
9/13/2020


Manage
Screenshots
File
Home
Share
View
Picture Tools
Pin to Quick access
Cut
New item
Open
Select all
Emotion Status
neutral
Search Screenshots
Quick access
user
This PC
5 items
Type here to search
4:29 PM
9/13/2020


localhost:8888/notebooks/Documents/Real-Time-Emotional-Recognition-with-Deep-Learning-master/Real-Time-Emotional-Recognition-with-Deep-Learning-master/New%20fol...
Emotion Status
Logout
Trusted
Python 3
happy
Type here to search
4:46 PM
9/13/2020


how to
Installa
face-re
how to
Tensor
Installi
python
Docum
detect
Un
face ar
Modul
No mo
deepfa
librarie
localhost:8888/notebooks/Documents/Real-Time-Emotional-Recognition-with-Deep-Learning-master/Real-Time-Emotional-Recognition-with-Deep-Learning-master/New%20fol...
Logout
Trusted
Python 3
Emotion Status
angry
Type here to search
4:48 PM
9/13/2020


localhost:8888/notebooks/Documents/Real-Time-Emotional-Recognition-with-Deep-Learning-master/Real-Time-Emotional-Recognition-with-Deep-Learning-master/New%20fol...
Emotion Status
disgust
Logout
Trusted
Python 3
Type here to search
4:48 PM
9/13/2020


Documents
detect_emo
Untitled - J
face and en
face and en
deepface.ip
hdf5 file - C
pdf to word
PDF to DOC
localhost:8888/notebooks/Documents/Real-Time-Emotional-Recognition-with-Deep-Learning-master/Real-Time-Emotional-Recognition-with-Deep-Learning-master/New%20fol...
Emotion Status
scared
Logout
Trusted
Python 3
Type here to search
8:42 PM
9/13/2020

Face recognition

In [ ]:

```
#pip install face-recognition==0.1.7
```

Import Libraries

In [2]:

```
import face_recognition
import os
import cv2
```

Load the image files in the directory

In [2]:

```
KNOWN_FACES_DIR = (r'C:\Users\user\Documents\database\Bharath')
UNKNOWN_FACES_DIR = (r'C:\Users\user\Documents\database\mark')
TOLERANCE = 0.6
FRAME_THICKNESS = 3
FONT_THICKNESS = 2
MODEL = 'cnn'  # default: 'hog', other one can be 'cnn' - CUDA accelerated (if
available) deep-learning pretrained model
```

```
MODEL = 'cnn'  # default: 'hog', other one can be 'cnn' - CUDA accelerated (if
available) deep-learning pretrained model


# Returns (R, G, B) from name
def name_to_color(name):
    # Take 3 first letters, tolower()
    # lowercased character ord() value rage is 97 to 122, substract 97, multip
ly by 8
    color = [(ord(c.lower())-97)*8 for c in name[:3]]
    return color


print('Loading known faces...')
known_faces = []
known_names = []

# We oranize known faces as subfolders of KNOWN_FACES_DIR
# Each subfolder's name becomes our label (name)
for name in os.listdir(KNOWN_FACES_DIR):

    # Next we load every file of faces of known person
    for filename in os.listdir(f'{KNOWN_FACES_DIR}/{name}'):

        # Load an image
        image = face_recognition.load_image_file(f'{KNOWN_FACES_DIR}/{name}/{f
ilename}')

        # Get 128-dimension face encoding
        # Always returns a list of found faces, for this purpose we take first
face only (assuming one face per image as you can't be twice on one image)
        encoding = face_recognition.face_encodings(image)[0]

        # Append encodings and name
        known_faces.append(encoding)
        known_names.append(name)


print('Processing unknown faces...')
# Now let's loop over a folder of faces we want to label
for filename in os.listdir(UNKNOWN_FACES_DIR):

    # Load image
    print(f'Filename {filename}', end='')
    image = face_recognition.load_image_file(f'{UNKNOWN_FACES_DIR}/{filename}'
)

    # This time we first grab face locations - we'll need them to draw boxes
    locations = face_recognition.face_locations(image, model=MODEL)

    # Now since we know loctions, we can pass them to face_encodings as second
```

```
argument
    # Without that it will search for faces once again slowing down whole proc
ess
    encodings = face_recognition.face_encodings(image, locations)

    # We passed our image through face_locations and face_encodings, so we can
modify it
    # First we need to convert it from RGB to BGR as we are going to work with
cv2
    image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

    # But this time we assume that there might be more faces in an image - we
 can find faces of dirrerent people
    print(f', found {len(encodings)} face(s)')
    for face_encoding, face_location in zip(encodings, locations):

        # We use compare_faces (but might use face_distance as well)
        # Returns array of True/False values in order of passed known_faces
        results = face_recognition.compare_faces(known_faces, face_encoding, T
OLERANCE)

        # Since order is being preserved, we check if any face was found then
 grab index
        # then label (name) of first matching known face withing a tolerance
        match = None
        if True in results:  # If at least one is true, get a name of first of
found labels
            match = known_names[results.index(True)]
            print(f' - {match} from {results}')

            # Each location contains positions in order: top, right, bottom, l
eft
            top_left = (face_location[3], face_location[0])
            bottom_right = (face_location[1], face_location[2])

            # Get color by name using our fancy function
            color = name_to_color(match)

            # Paint frame
            cv2.rectangle(image, top_left, bottom_right, color, FRAME_THICKNES
S)

            # Now we need smaller, filled grame below for a name
            # This time we use bottom in both corners - to start from bottom a
nd move 50 pixels down
            top_left = (face_location[3], face_location[2])
            bottom_right = (face_location[1], face_location[2] + 22)

            # Paint frame
            cv2.rectangle(image, top_left, bottom_right, color, cv2.FILLED)

            # Wite a name
            cv2.putText(image, match, (face_location[3] + 10, face_location[2]
+ 15), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (200, 200, 200), FONT_THICKNESS)

    # Show image
    cv2.imshow(filename, image)
```

```
cv2.waitKey(0)
cv2.destroyWindow(filename
```

DeepFace

Deepface is a lightweight face recognition and facial attribute analysis (age, gender, emotion and race) framework for python. It is a hybrid face recognition framework wrapping state-of-the-art models: VGG-Face , Google FaceNet, OpenFace, Facebook DeepFace, DeepID and Dlib.

Importing Libraries

In [4]:

```
from deepface import DeepFace
import numpy as np
import cv2
from matplotlib import pyplot as plt
```

```
Directory  /root /.deepface created
Directory  /root /.deepface/weights created
```

In [3]:

```
#pip install deepface
```

Import images

In [ ]:

```
img_path=('bharath.JPG')
img_path1=('nirmal.jpg')
```

In [ ]:

```
img = cv2.imread(img_path)

img1=cv2.imread(img_path1)
```

In [ ]:

```
face=DeepFace.detectFace('bharath.JPG')
```

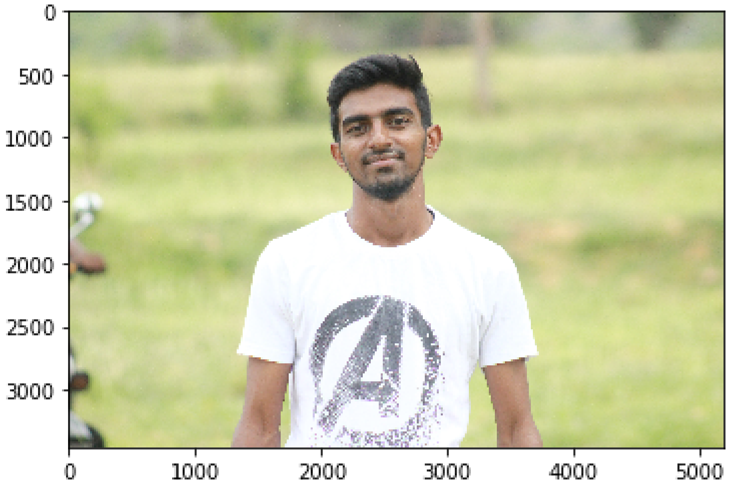
Show images

In [ ]:

```
plt.imshow(img[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fba7e6b71d0>
```



In [ ]:

```
img = cv2.imread(img_path)
```

In [ ]:

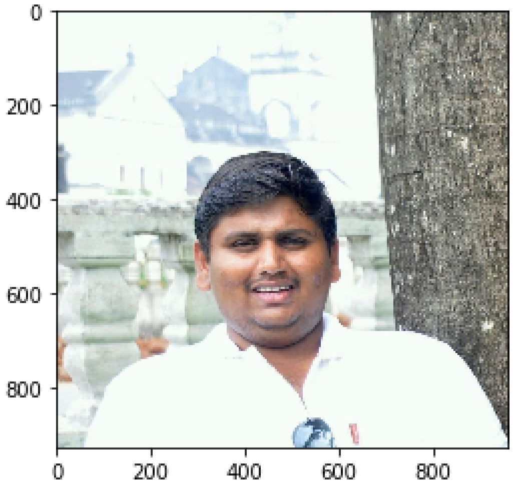
```
img1=cv2.imread(img_path1)
```

In [ ]:

```
plt.imshow(img1[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fba7bedb550>
```



Facial Attribute Analysis

Deepface also offers facial attribute analysis including age, gender, facial expression (including angry, fear, neutral, sad, disgust, happy and surprise) and race (including asian, white, middle eastern, indian, latino and black) predictions. Analysis function under the DeepFace interface is used to find demography of a face.

In [ ]:

```
DeepFace.analyze(img_path)
```

```
Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]

Actions to do:  ['emotion', 'age', 'gender', 'race']
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated.
Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Pleas
e use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. P
lease use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Plea
se use tf.nn.max_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:4271: The name tf.nn.avg_pool is deprecated. Plea
se use tf.nn.avg_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:148: The name tf.placeholder_with_default is depr
ecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:3733: calling dropout (from tensorflow.python.op
s.nn_ops) with keep_prob is deprecated and will be removed in a future ver
sion.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1
- keep_prob`.
facial_expression_model_weights.h5 will be downloaded...

Downloading...
From: https://drive.google.com/uc?id=13iUHHP3SlNg53qSuQZDdHDSDNdBP9nwy
To: /root/.deepface/weights/facial_expression_model_weights.zip

0.00B [00:00, ?B/s]
5.54MB [00:00, 8.13MB/s]
```

```
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:190: The name tf.get_default_session is deprecate
d. Please use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Pleas
e use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:203: The name tf.Session is deprecated. Please us
e tf.compat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:207: The name tf.global_variables is deprecated.
Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:216: The name tf.is_variable_initialized is depre
cated. Please use tf.compat.v1.is_variable_initialized instead.



WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:223: The name tf.variables_initializer is depreca
ted. Please use tf.compat.v1.variables_initializer instead.


Action: age:  25%|██        | 1/4 [00:08<00:24,  8.19s/it]

age_model_weights.h5 will be downloaded...
```

```
Downloading...
From: https://drive.google.com/uc?id=1YCox_4kJ-BYeXq27uUbasu--yz28zUMV
To: /root/.deepface/weights/age_model_weights.h5

0.00B [00:00, ?B/s]
4.72MB [00:00, 27.5MB/s]
8.91MB [00:00, 13.0MB/s]
34.1MB [00:01, 17.6MB/s]
42.5MB [00:01, 17.8MB/s]
57.7MB [00:01, 24.2MB/s]
67.6MB [00:02, 24.6MB/s]
76.0MB [00:02, 30.0MB/s]
97.0MB [00:02, 40.4MB/s]
107MB [00:02, 33.4MB/s]
116MB [00:02, 37.2MB/s]
128MB [00:03, 47.1MB/s]
137MB [00:03, 47.9MB/s]
145MB [00:03, 35.9MB/s]
160MB [00:03, 35.6MB/s]
177MB [00:04, 41.8MB/s]
193MB [00:04, 47.8MB/s]
202MB [00:04, 42.4MB/s]
210MB [00:04, 37.2MB/s]
216MB [00:05, 27.7MB/s]
245MB [00:05, 38.0MB/s]
256MB [00:05, 34.7MB/s]
269MB [00:06, 40.0MB/s]
286MB [00:06, 45.8MB/s]
297MB [00:06, 55.8MB/s]
306MB [00:06, 29.4MB/s]
319MB [00:07, 37.5MB/s]
336MB [00:07, 36.1MB/s]
344MB [00:07, 36.6MB/s]
361MB [00:08, 41.9MB/s]
378MB [00:08, 44.7MB/s]
386MB [00:08, 37.1MB/s]
412MB [00:08, 49.5MB/s]
422MB [00:09, 30.1MB/s]
434MB [00:09, 38.8MB/s]
454MB [00:09, 44.8MB/s]
462MB [00:10, 34.1MB/s]
489MB [00:10, 46.2MB/s]
502MB [00:10, 40.6MB/s]
512MB [00:10, 45.3MB/s]
539MB [00:11, 48.0MB/s]
Action: gender:  50%|█████     | 2/4 [00:26<00:22, 11.20s/it]

gender_model_weights.h5 will be downloaded...
```

```
Downloading...
From: https://drive.google.com/uc?id=1wUXRVlbsni2FN9-jkS_f4UTUrm1bRLyk
To: /root/.deepface/weights/gender_model_weights.h5

0.00B [00:00, ?B/s]
4.72MB [00:00, 7.88MB/s]
34.1MB [00:00, 10.9MB/s]
52.4MB [00:00, 15.2MB/s]
67.6MB [00:01, 20.4MB/s]
84.4MB [00:01, 26.6MB/s]
101MB [00:01, 35.1MB/s]
119MB [00:01, 46.1MB/s]
132MB [00:01, 48.0MB/s]
160MB [00:02, 51.8MB/s]
192MB [00:02, 69.2MB/s]
209MB [00:02, 71.9MB/s]
222MB [00:02, 76.4MB/s]
235MB [00:02, 85.6MB/s]
248MB [00:02, 84.4MB/s]
262MB [00:03, 94.9MB/s]
274MB [00:03, 74.6MB/s]
294MB [00:03, 87.9MB/s]
306MB [00:03, 66.5MB/s]
319MB [00:03, 70.8MB/s]
336MB [00:04, 76.5MB/s]
353MB [00:04, 66.5MB/s]
370MB [00:04, 64.1MB/s]
395MB [00:04, 80.8MB/s]
406MB [00:05, 67.3MB/s]
437MB [00:05, 80.5MB/s]
448MB [00:05, 53.5MB/s]
479MB [00:05, 59.9MB/s]
504MB [00:06, 64.5MB/s]
537MB [00:06, 82.4MB/s]
Action: race:  75%|███████▌  | 3/4 [00:40<00:11, 11.92s/it]

race_model_single_batch.h5 will be downloaded...
```

```
Downloading...
From: https://drive.google.com/uc?id=1nz-WDhghGQBC4biwShQ9kYjvQMpO6smj
To: /root/.deepface/weights/race_model_single_batch.zip

0.00B [00:00, ?B/s]
4.72MB [00:00, 4.80MB/s]
34.1MB [00:01, 6.76MB/s]
42.5MB [00:01, 9.30MB/s]
65.0MB [00:01, 13.0MB/s]
76.0MB [00:01, 17.2MB/s]
92.8MB [00:01, 23.2MB/s]
110MB [00:01, 29.8MB/s]
126MB [00:02, 38.8MB/s]
143MB [00:02, 49.3MB/s]
155MB [00:02, 56.1MB/s]
168MB [00:02, 67.2MB/s]
185MB [00:02, 71.3MB/s]
202MB [00:02, 83.0MB/s]
219MB [00:02, 91.7MB/s]
235MB [00:03, 74.0MB/s]
269MB [00:03, 90.8MB/s]
282MB [00:03, 74.4MB/s]
298MB [00:03, 89.1MB/s]
311MB [00:03, 96.9MB/s]
328MB [00:03, 94.5MB/s]
344MB [00:04, 104MB/s]
361MB [00:04, 111MB/s]
374MB [00:04, 98.3MB/s]
386MB [00:04, 105MB/s]
398MB [00:04, 69.9MB/s]
423MB [00:04, 88.8MB/s]
437MB [00:05, 78.9MB/s]
454MB [00:05, 90.6MB/s]
466MB [00:05, 65.3MB/s]
511MB [00:05, 87.2MB/s]
Action: race: 100%|██████████| 4/4 [00:58<00:00, 14.58s/it]
```

Out[ ]:

```
{'age': 26.41808512567964,
 'dominant_emotion': 'happy',
 'dominant_race': 'indian',
 'emotion': {'angry': 0.7454432632462351,
  'disgust': 0.0011935266112451791,
  'fear': 0.34658413964739093,
  'happy': 84.05967138775077,
  'neutral': 14.331475953703892,
  'sad': 0.22008809601650334,
  'surprise': 0.29553510187474447},
 'gender': 'Man',
 'race': {'asian': 0.1530987883902364,
  'black': 2.0049493092268293,
  'indian': 90.01411736095652,
  'latino hispanic': 6.417762125597127,
  'middle eastern': 0.7224719470722644,
  'white': 0.6875927504207343}}
```

In [ ]:

```
DeepFace.analyze(img_path1)
```

```
Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]

Actions to do:  ['emotion', 'age', 'gender', 'race']

Action: race: 100%|██████████| 4/4 [00:11<00:00,  2.94s/it]
```

Out[ ]:

```
{'age': 27.71529435253023,
 'dominant_emotion': 'happy',
 'dominant_race': 'indian',
 'emotion': {'angry': 1.0521309450268745,
  'disgust': 9.464661943070496e-07,
  'fear': 0.005371573570300825,
  'happy': 98.71529936790466,
  'neutral': 0.2112343441694975,
  'sad': 0.015786771837156266,
  'surprise': 0.00017549884488755771},
 'gender': 'Man',
 'race': {'asian': 0.0002652298007888021,
  'black': 0.058505049673840404,
  'indian': 99.79440569877625,
  'latino hispanic': 0.1443214132450521,
  'middle eastern': 0.0014743599422217812,
  'white': 0.0010294349522155244}}
```

In [ ]:

```
img_path2='dineshkumar.png'
```

In [ ]:

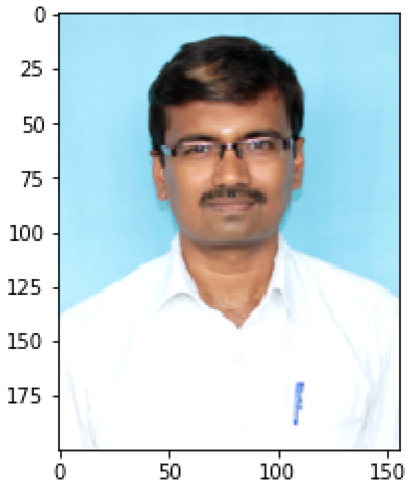
```
img2=cv2.imread(img_path2)
```

In [ ]:

```
plt.imshow(img2[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fba7bd8acc0>
```



In [ ]:

```
DeepFace.analyze(img_path2)
```

```
Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]

Actions to do:  ['emotion', 'age', 'gender', 'race']

Action: race: 100%|██████████| 4/4 [00:10<00:00,  2.64s/it]
```

Out[ ]:

```
{'age': 25.859603176765408,
 'dominant_emotion': 'happy',
 'dominant_race': 'indian',
 'emotion': {'angry': 5.340279329857367e-08,
  'disgust': 3.340865431522566e-09,
  'fear': 4.732376268634653e-06,
  'happy': 93.60448122024536,
  'neutral': 6.3954271376132965,
  'sad': 8.604337153883534e-05,
  'surprise': 7.27333970829136e-07},
 'gender': 'Man',
 'race': {'asian': 0.0049813130317488685,
  'black': 0.12425618479028344,
  'indian': 99.655020236969,
  'latino hispanic': 0.21273961756378412,
  'middle eastern': 0.0016008931197575293,
  'white': 0.0013974002285976894}}
```

In [12]:

```
africa='africanyoungman.jpg'
```

In [13]:

```
img_africa1=cv2.imread(africa)
```

In [14]:

```
plt.imshow(img_africa1[:,:,::-1])
```

Out[14]:

```
<matplotlib.image.AxesImage at 0x7f81de7b44a8>
```

In [15]:

```
DeepFace.analyze(africa)
```

```
Actions to do:  ['emotion', 'age', 'gender', 'race']


Analyzing:   0%|          | 0/1 [00:00<?, ?it/s]


Finding actions:   0%|          | 0/4 [00:00<?, ?it/s]


Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]


Action: emotion:  25%|██▌       | 1/4 [00:06<00:20,  7.00s/it]


Action: age:  25%|██▌       | 1/4 [00:07<00:20,  7.00s/it]


Action: age:  50%|█████     | 2/4 [00:07<00:10,  5.17s/it]


Action: gender:  50%|█████     | 2/4 [00:07<00:10,  5.17s/it]


Action: gender:  75%|███████▌  | 3/4 [00:08<00:03,  3.69s/it]


Action: race:  75%|███████▌  | 3/4 [00:08<00:03,  3.69s/it]


Action: race: 100%|██████████| 4/4 [00:08<00:00,  2.10s/it]
```

Out[15]:

```
{'age': 29.197178275146026,
 'dominant_emotion': 'neutral',
 'dominant_race': 'asian',
 'emotion': {'angry': 0.005663492993335524,
  'disgust': 8.253663850155135e-05,
  'fear': 0.15647601103410125,
  'happy': 1.8161613494157791,
  'neutral': 91.85335040092468,
  'sad': 6.168239936232567,
  'surprise': 2.8077045044483384e-05},
 'gender': 'Man',
 'race': {'asian': 30.515384674072266,
  'black': 14.639833569526672,
  'indian': 10.802464932203293,
  'latino hispanic': 15.080086886882782,
  'middle eastern': 8.77605602145195,
  'white': 20.186172425746918}}
```

In [ ]:

```
img_angry2='angry2.jpg'
```

In [ ]:

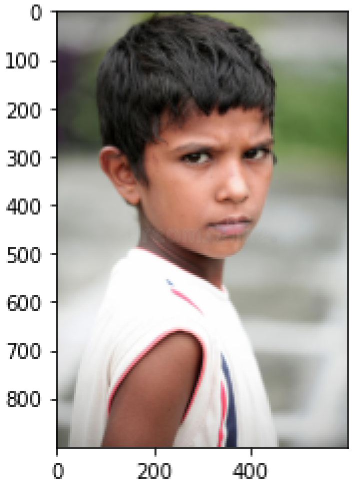
```
img_angry2=cv2.imread(img_angry2)
```

In [ ]:

```
plt.imshow(img_angry2[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7f604562d2b0>
```



In [ ]:

```
DeepFace.analyze(img_angry2)
```

```
Actions to do:  ['emotion', 'age', 'gender', 'race']

Analyzing:   0%|          | 0/1 [00:00<?, ?it/s]
Finding actions:   0%|          | 0/4 [00:00<?, ?it/s]
Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]

WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f5fd48a12f0> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings 
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For 
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer 
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.

Action: emotion:  25%|██▌       | 1/4 [00:00<00:00,  3.06it/s]
Action: age:  25%|██▌       | 1/4 [00:00<00:00,  3.06it/s]

WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f5fd4861bf8> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings 
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For 
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer 
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.


Action: age:  50%|█████     | 2/4 [00:00<00:00,  2.48it/s]
Action: gender:  50%|█████     | 2/4 [00:00<00:00,  2.48it/s]

WARNING:tensorflow:7 out of the last 7 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f5fd3fd2598> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings 
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For 
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer 
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.

Action: gender:  75%|███████▌  | 3/4 [00:01<00:00,  2.56it/s]
Action: race:  75%|███████▌  | 3/4 [00:01<00:00,  2.56it/s]
```

```
WARNING:tensorflow:8 out of the last 8 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f5fd3778f28> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.
```

```
Action: race: 100%|██████████| 4/4 [00:01<00:00,  2.45it/s]
Analyzing:   0%|          | 0/1 [00:01<?, ?it/s]
```

Out[ ]:

```
{'age': 27.218469101917837,
 'dominant_emotion': 'angry',
 'dominant_race': 'latino hispanic',
 'emotion': {'angry': 98.04232143851263,
  'disgust': 3.3801128436490824e-10,
  'fear': 0.03752286273247895,
  'happy': 0.018511023494032205,
  'neutral': 0.15784131111233698,
  'sad': 1.7433626361653047,
  'surprise': 0.0004334505712784146},
 'gender': 'Woman',
 'race': {'asian': 8.717638999223709,
  'black': 1.5856485813856125,
  'indian': 10.173255205154419,
  'latino hispanic': 37.24655210971832,
  'middle eastern': 15.7467320561409,
  'white': 26.530173420906067}}
```

In [ ]:

```
Leonardo_DiCaprio='Leonardo-DiCaprio.jpg'
```

In [ ]:

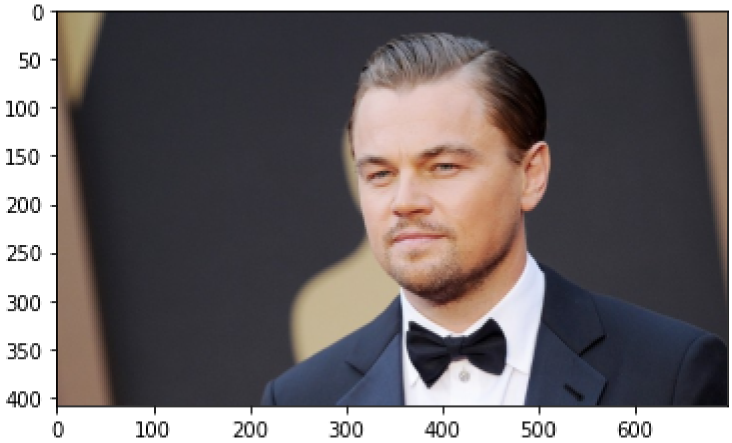
```
l_d=cv2.imread(Leonardo_DiCaprio)
```

In [ ]:

```
plt.imshow(l_d[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7f968c042438>
```



In [ ]:

```
DeepFace.analyze(Leonardo_DiCaprio)
```

```
Actions to do:  ['emotion', 'age', 'gender', 'race']
```

```
Analyzing:   0%|          | 0/1 [00:00<?, ?it/s]
```

```
Finding actions:   0%|          | 0/4 [00:00<?, ?it/s]
```

```
Action: emotion:   0%|          | 0/4 [00:00<?, ?it/s]
```

```
WARNING:tensorflow:5 out of the last 5 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f93e850cf28> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.
```

```
Action: emotion:  25%|██▌       | 1/4 [00:00<00:00,  5.77it/s]
```

```
Action: age:  25%|██▌       | 1/4 [00:00<00:00,  5.77it/s]
```

```
WARNING:tensorflow:6 out of the last 6 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f93e84d67b8> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.
```

```
Action: age:  50%|█████     | 2/4 [00:00<00:00,  5.00it/s]
```

```
Action: gender:  50%|█████     | 2/4 [00:00<00:00,  5.00it/s]
```

```
WARNING:tensorflow:7 out of the last 7 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f93e847a268> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings 
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For 
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.
```

```
Action: gender:  75%|███████▌  | 3/4 [00:00<00:00,  5.22it/s]
```

```
Action: race:  75%|███████▌  | 3/4 [00:00<00:00,  5.22it/s]
```

```
WARNING:tensorflow:8 out of the last 8 calls to <function Model.make_predi
ct_function.<locals>.predict_function at 0x7f93e8417f28> triggered tf.func
tion retracing. Tracing is expensive and the excessive number of tracings
could be due to (1) creating @tf.function repeatedly in a loop, (2) passin
g tensors with different shapes, (3) passing Python objects instead of ten
sors. For (1), please define your @tf.function outside of the loop. For
(2), @tf.function has experimental_relax_shapes=True option that relaxes a
rgument shapes that can avoid unnecessary retracing. For (3), please refer
to https://www.tensorflow.org/tutorials/customization/performance#python_o
r_tensor_args and https://www.tensorflow.org/api_docs/python/tf/function f
or  more details.
```

```
Action: race: 100%|██████████| 4/4 [00:00<00:00,  5.11it/s]
Analyzing:   0%|          | 0/1 [00:00<?, ?it/s]
```

Out[ ]:

```
{'age': 26.81033398469586,
 'dominant_emotion': 'neutral',
 'dominant_race': 'white',
 'emotion': {'angry': 2.9125839471817017,
  'disgust': 0.0020122517526033334,
  'fear': 3.896341845393181,
  'happy': 0.9911656379699707,
  'neutral': 75.00253915786743,
  'sad': 17.026448249816895,
  'surprise': 0.16890825936570764},
 'gender': 'Man',
 'race': {'asian': 5.547038093209267,
  'black': 0.8930975571274757,
  'indian': 1.4937848784029484,
  'latino hispanic': 27.56403088569641,
  'middle eastern': 10.531724989414215,
  'white': 53.970324993133545}}
```

In [ ]:

Face Verification

Verification function under the DeepFace interface offers a single face recognition. Each call of the function builds a face recognition model and this is very costly. If you are going to verify several faces sequentially, then you should pass an array of faces to the function instead of calling the function in a for loop. In this way, complex face recognition models will be built once and this will speed the function up dramatically. Besides, calling the function in a for loop might cause memory problems as well.

In [ ]:

```
img_path2=('bharath1.jpg')
```

In [ ]:

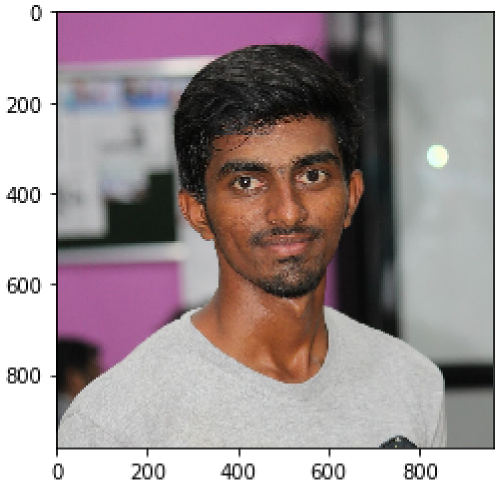
```
img3=cv2.imread(img_path2)
```

In [ ]:

```
plt.imshow(img3[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fba7bce4e48>
```



Bharath vs Bharath1

In [ ]:

```
result = DeepFace.verify(img_path, img_path2)
```

```
Using VGG-Face model backend and cosine distance.
WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:66: The name tf.get_default_graph is deprecated.
Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:541: The name tf.placeholder is deprecated. Pleas
e use tf.compat.v1.placeholder instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:4432: The name tf.random_uniform is deprecated. P
lease use tf.random.uniform instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:4267: The name tf.nn.max_pool is deprecated. Plea
se use tf.nn.max_pool2d instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:148: The name tf.placeholder_with_default is depr
ecated. Please use tf.compat.v1.placeholder_with_default instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:3733: calling dropout (from tensorflow.python.op
s.nn_ops) with keep_prob is deprecated and will be removed in a future ver
sion.
Instructions for updating:
Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1
- keep_prob`.
vgg_face_weights.h5 will be downloaded...

Downloading...
From: https://drive.google.com/uc?id=1CPSeum3HpopfomUEK1gybeuIVoeJT_Eo
To: /root/.deepface/weights/vgg_face_weights.h5
580MB [00:05, 109MB/s]

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:190: The name tf.get_default_session is deprecate
d. Please use tf.compat.v1.get_default_session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:197: The name tf.ConfigProto is deprecated. Pleas
e use tf.compat.v1.ConfigProto instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:203: The name tf.Session is deprecated. Please us
e tf.compat.v1.Session instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:207: The name tf.global_variables is deprecated.
Please use tf.compat.v1.global_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:216: The name tf.is_variable_initialized is depre
cated. Please use tf.compat.v1.is_variable_initialized instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/keras/backe
nd/tensorflow_backend.py:223: The name tf.variables_initializer is depreca
ted. Please use tf.compat.v1.variables_initializer instead.
```

In [ ]:

```

```

Deepface is a hybrid face recognition package. It currently wraps the state-of-the-art face recognition models: VGG-Face , Google FaceNet, OpenFace, Facebook DeepFace, DeepID and Dlib. The default configuration verifies faces with VGG-Face model.

In [ ]:

```
result
```

Out[ ]:

```
{'distance': 0.20981597900390625,
 'max_threshold_to_verify': 0.4,
 'model': 'VGG-Face',
 'similarity_metric': 'cosine',
 'verified': True}
```

Similarity

Face recognition models are regular convolutional neural networks and they are responsible to represent face photos as vectors. Decision of verification is based on the distance between vectors. We can classify pairs if its distance is less than a threshold.

Bharath vs Nirmal

In [ ]:

```
result1 = DeepFace.verify(img_path, img_path1)
```

```
Using VGG-Face model backend and cosine distance.
```

In [ ]:

```
result1
```

Out[ ]:

```
{'distance': 0.4386165738105774,
 'max_threshold_to_verify': 0.4,
 'model': 'VGG-Face',
 'similarity_metric': 'cosine',
 'verified': False}
```

Facial Attribute Analysis

Deepface also offers facial attribute analysis including age, gender, facial expression (including angry, fear, neutral, sad, disgust, happy and surprise) and race (including asian, white, middle eastern, indian, latino and black) predictions. Analysis function under the DeepFace interface is used to find demography of a face.

In [ ]:

```
img1=('Bharath.JPG')
img2=('Bharath1.jpg')
img3=('Bharath2.JPG')
```

In [ ]:

```
i_1=cv2.imread(img1)
i_2=cv2.imread(img2)
i_3=cv2.imread(img3)
```

In [ ]:

```
plt.imshow(i_1[:,:,::-1])
```

Out[ ]:

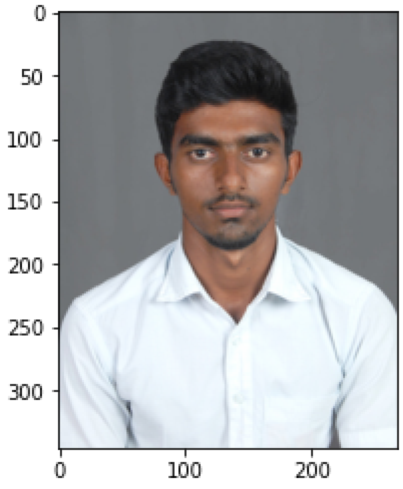
```
<matplotlib.image.AxesImage at 0x7fb66c94ceb8>
```

In [ ]:

```
plt.imshow(i_2[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fb66c42c278>
```

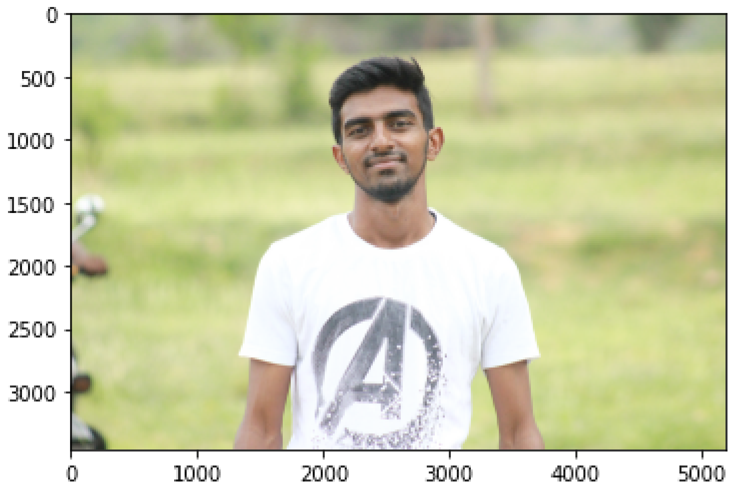


In [ ]:

```
plt.imshow(i_3[:,:,::-1])
```

Out[ ]:

```
<matplotlib.image.AxesImage at 0x7fb6a60cb0b8>
```



In [ ]:

```
img_path=('Bharath.JPG')
img_path1=('Bharath2.JPG')
```

In [ ]:

```
from deepface.basemodels import VGGFace, OpenFace, Facenet, FbDeepFace, DeepID
```

In [ ]:

```
model = VGGFace.loadModel() #all face recognition models have loadModel() function in t
heir interfaces
DeepFace.verify(img_path,img_path1, model_name = "VGG-Face", model = model)
```

```
Verification:   0%|          | 0/1 [00:00<?, ?it/s]

Already built model is passed

Verification:   0%|          | 0/1 [00:08<?, ?it/s]
```

Out[ ]:

```
{'distance': 0.2933734059333801,
 'max_threshold_to_verify': 0.4,
 'model': 'VGG-Face',
 'similarity_metric': 'cosine',
 'verified': True}
```

In [ ]:

```
result = DeepFace.verify('Bharath1.jpg','Bharath.JPG')
```

```
Using VGG-Face model backend and cosine distance.

Verification:   0%|          | 0/1 [00:04<?, ?it/s]
```

In [ ]:

```
print(result)
```

```
{'verified': True, 'distance': 0.30417782068252563, 'max_threshold_to_veri
fy': 0.4, 'model': 'VGG-Face', 'similarity_metric': 'cosine'}
```

In [ ]:

```

```

In [ ]:

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, c
all drive.mount("/content/drive", force_remount=True).
```

In [ ]:

```
!unzip "/database.zip" -d "/"
```

```
Archive:  /database.zip
replace /database/Bharath/bharath (2).JPG? [y]es, [n]o, [A]ll, [N]one, [r]
ename:
```

In [ ]:

```

```

In [ ]:

```
#from deepface import DeepFace
#DeepFace.stream("/database")
```

In [ ]:

```
from deepface import DeepFace
import pandas as pd
#df = DeepFace.find(img_path = "Bharath1.jpg", db_path = r"C:\Users\user\Documents/data
base")
#dfs = DeepFace.find(img_path = ["img1.jpg", "img2.jpg"], db_path = "C:/workspace/my_d
b")
```

In [ ]:

```

```