Face Recognition Using Python

FACE RECOGNITION USING PYTHON
Prepared by : Shubham Singh Chouhan

- What is face recognition?

Face recognition is a way of identifying or confirming an individual's identity using their face. Facial recognition systems can be used to identify people in photos, videos, or in real-time.

Face recognition is a category of biometric security. Other forms of biometric software include voice recognition, fingerprint recognition, and eye retina or iris recognition. The technology is mostly used for security and law enforcement, though there is increasing interest in other areas of use.

- How does face recognition work?

Many people are familiar with face recognition technology through the FaceID used to unlock iPhones (however, this is only one application of face recognition). Typically, facial recognition does not rely on a massive database of photos to determine an individual's identity — it simply identifies and recognizes one person as the sole owner of the device, while limiting access to others.

Beyond unlocking phones, facial recognition works by matching the faces of people walking past special cameras, to images of people on a watch list. The watch lists can contain pictures of anyone, including people who are not suspected of any wrongdoing, and the images can come from anywhere — even from our social media accounts. Facial technology systems can vary, but in general, they tend to operate as follows:

Step 1: Face detection

The camera detects and locates the image of a face, either alone or in a crowd. The image may show the person looking straight ahead or in profile.

Step 2: Face analysis

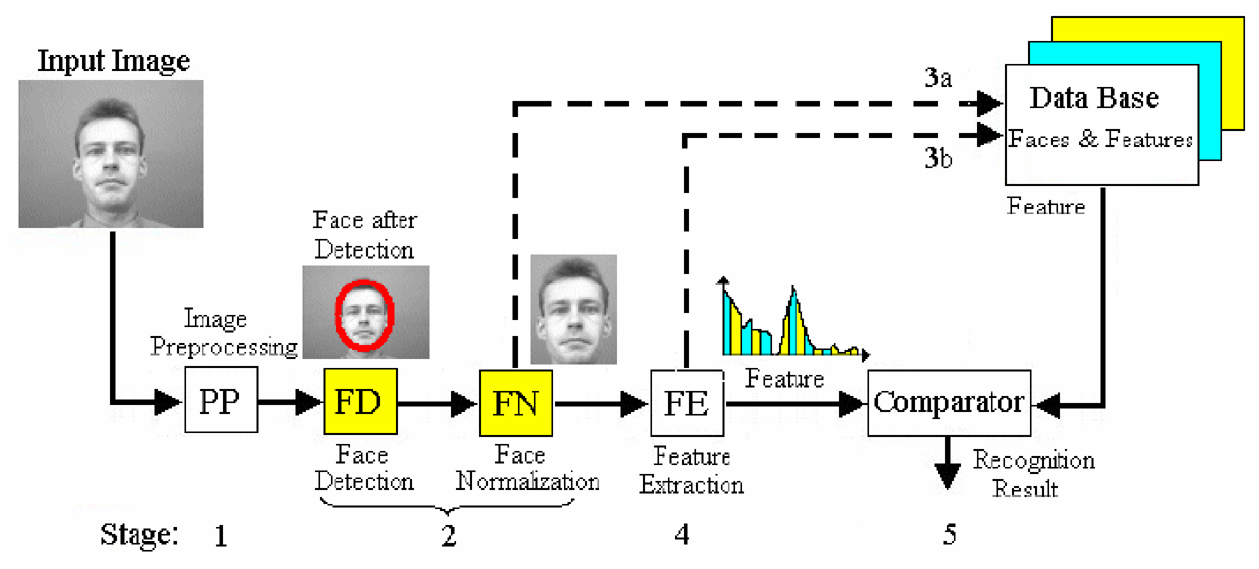
Next, an image of the face is captured and analyzed. Most facial recognition technology relies on 2D rather than 3D images because it can more conveniently match a 2D image with public photos or those in a database. The software reads the geometry of your face. Key factors include the distance between your eyes, the depth of your eye sockets, the distance from forehead to chin, the shape of your cheekbones, and the contour of the lips, ears, and chin. The aim is to identify the facial landmarks that are key to distinguishing your face.

Step 3: Converting the image to data

The face capture process transforms analog information (a face) into a set of digital information (data) based on the person's facial features. Your face's analysis is essentially turned into a mathematical formula. The numerical code is called a faceprint. In the same way that thumbprints are unique, each person has their own faceprint.

Step 4: Finding a match

Your faceprint is then compared against a database of other known faces. For example, the FBI has access to up to 650 million photos, drawn from various state databases. On Facebook, any photo tagged with a person’s name becomes a part of Facebook's database, which may also be used for facial recognition. If your faceprint matches an image in a facial recognition database, then a determination is made.



- How is face recognition achieved?

Face recognition can be achieved using many sources:

- Images from a dataset
- Live webcam feed
- Recorded videos

- How face recognition is used?

The technology is used for a variety of purposes. These include:

1. Unlocking phones

Various phones, including the most recent iPhones, use face recognition to unlock the device. The technology offers a powerful way to protect personal data and ensures that sensitive data remains inaccessible if the phone is stolen. Apple claims that the chance of a random face unlocking your phone is about one in 1 million.

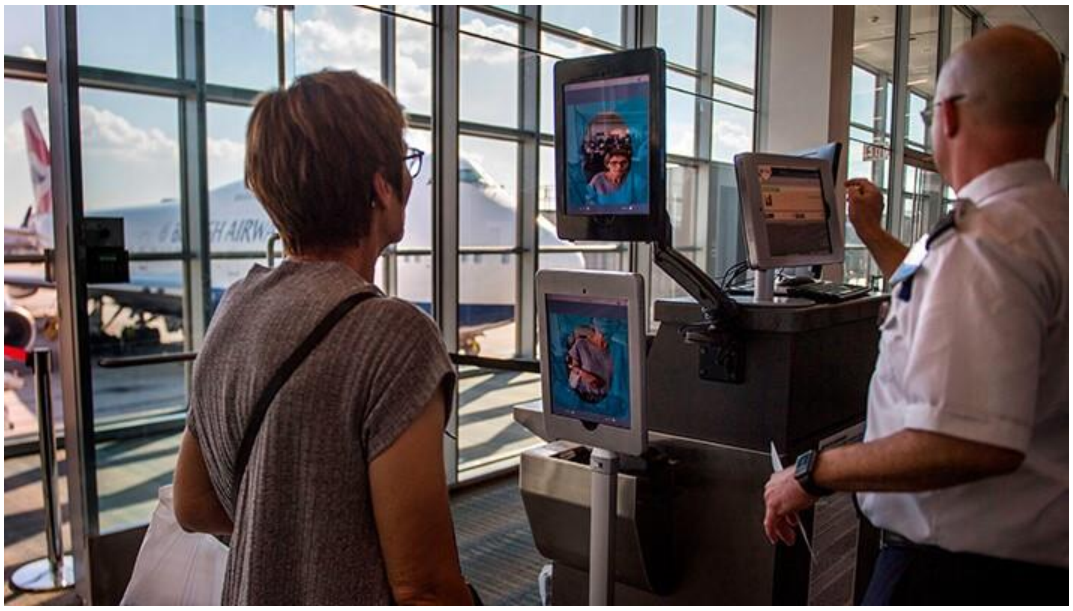
2. Law enforcement

Facial recognition is regularly being used by law enforcement. According to this NBC report, the technology is increasing amongst law enforcement agencies within the US, and the same is true in other countries. Police collects mugshots from arrestees and compare them against local, state, and federal face recognition databases. Once an arrestee's photo has been taken, their picture will be added to databases to be scanned whenever police carry out another criminal search.

Also, mobile face recognition allows officers to use smartphones, tablets, or other portable devices to take a photo of a driver or a pedestrian in the field and immediately compare that photo against to one or more face recognition databases to attempt an identification.

3. Airports and border control

Facial recognition has become a familiar sight at many airports around the world. Increasing numbers of travellers hold biometric passports, which allow them to skip the ordinarily long lines and instead walk through an automated ePassport control to reach the gate faster. Facial recognition not only reduces waiting times but also allows airports to improve security. The US Department of Homeland Security predicts that facial recognition will be used on 97% of travellers by 2023. As well as at airports and border crossings, the technology is used to enhance security at large-scale events such as the Olympics.



4. Finding missing persons

Facial recognition can be used to find missing persons and victims of human trafficking. Suppose missing individuals are added to a database. In that case, law enforcement can be alerted as soon as they are recognized by face recognition — whether it is in an airport, retail store, or other public space.

5. Reducing retail crime

Facial recognition is used to identify when known shoplifters, organized retail criminals, or people with a history of fraud enter stores. Photographs of individuals can be matched against large databases of criminals so that loss prevention and retail security professionals can be notified when shoppers who potentially represent a threat enter the store.

6. Improving retail experiences

The technology offers the potential to improve retail experiences for customers. For example, kiosks in stores could recognize customers, make product suggestions based on their purchase history, and point them in the right direction. “Face pay” technology could allow shoppers to skip long checkout lines with slower payment methods.

7. Banking

Biometric online banking is another benefit of face recognition. Instead of using one-time passwords, customers can authorize transactions by looking at their smartphone or computer. With facial recognition, there are no passwords for hackers to compromise. If hackers steal your photo database, 'liveless' detection – a technique used to determine whether the source of a biometric sample is a live human being or a fake representation – should (in theory) prevent them from using it for impersonation purposes. Face recognition could make debit cards and signatures a thing of the past.

8. Marketing and advertising

Marketers have used facial recognition to enhance consumer experiences. For example, frozen pizza brand DiGiorno used facial recognition for a 2017 marketing campaign where it analyzed the expressions of people at DiGiorno-themed parties to gauge people's emotional reactions to pizza. Media companies also use facial recognition to test audience reaction to movie trailers, characters in TV pilots, and optimal placement of TV promotions. Billboards that incorporate face recognition technology – such as London's Piccadilly Circus – means brands can trigger tailored advertisements.

9. Healthcare

Hospitals use facial recognition to help with patient care. Healthcare providers are testing the use of facial recognition to access patient records, streamline patient registration, detect emotion and pain in patients, and even help to identify specific genetic diseases. AiCure has developed an app that uses facial recognition to ensure that people take their medication as prescribed. As biometric technology becomes less expensive, adoption within the healthcare sector is expected to increase.

10. Tracking student or worker attendance

Some educational institutions in China use face recognition to ensure students are not skipping class. Tablets are used to scan students' faces and match them to photos in a database to validate their identities. More broadly, the technology can be used for workers to sign in and out of their workplaces, so that employers can track attendance.

- Examples of face recognition technology

1. Amazon previously promoted its cloud-based face recognition service named Rekognition to law enforcement agencies. However, in a June 2020 blog post, the company announced it was planning a one-year moratorium on the use of its technology by police. The rationale for this was to allow time for US federal laws to be initiated, to protect human rights and civil liberties.

2. Apple uses facial recognition to help users quickly unlock their phones, log in to apps, and make purchases.

3. Coca-Cola has used facial recognition in several ways across the world. Examples include rewarding customers for recycling at some of its vending machines in China, delivering personalized ads on its vending machines in Australia, and for event marketing in Israel.

4. Facebook began using facial recognition in the US in 2010 when it automatically tagged people in photos using its tag suggestions tool. The tool scans a user's face and offers suggestions about who that person is. Since 2019, Facebook has made the feature opt-in as part of a drive to become more privacy focused. Facebook provides information on how you can opt-in or out of face recognition here.

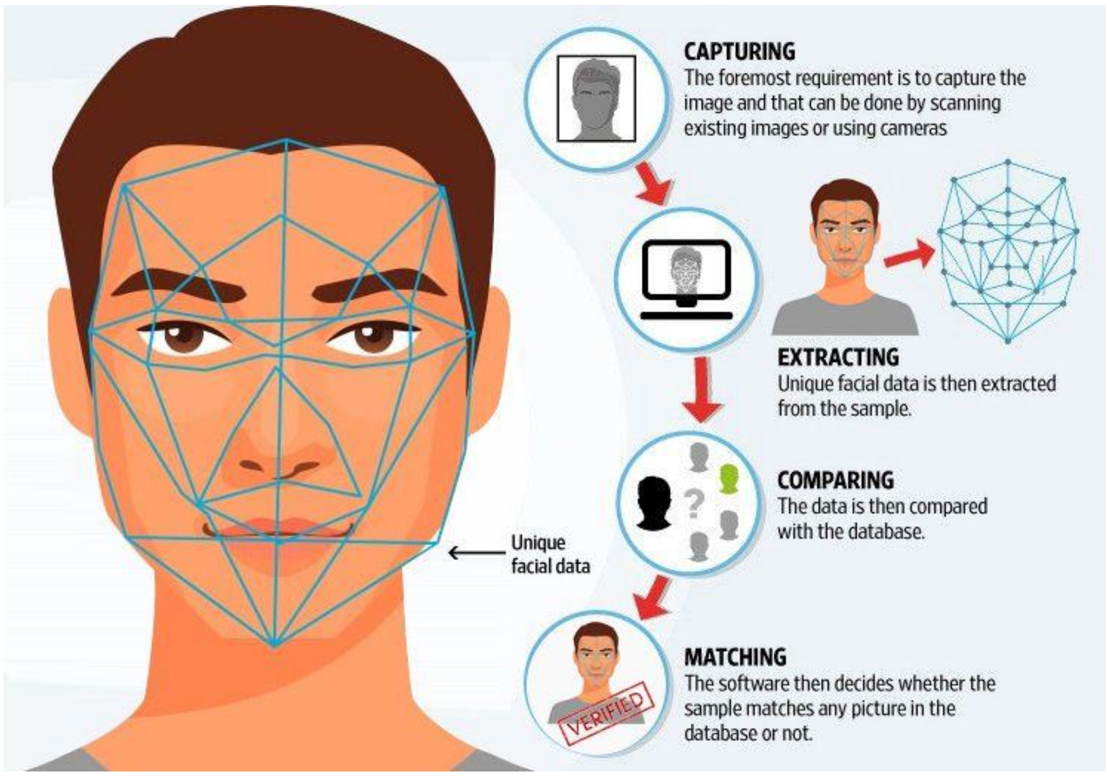
5. Google incorporates the technology into Google Photos and uses it to sort pictures and automatically tag them based on the people recognized.

Face Detection: -

- What is face detection?

Face detection -- also called facial detection -- is an artificial intelligence (AI) based computer technology used to find and identify human faces in digital images. ... It now plays an important role as the first step in many key applications -- including face tracking, face analysis and facial recognition.

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene.


CAPTURING
The foremost requirement is to capture the image and that can be done by scanning existing images or using cameras
EXTRACTING
Unique facial data is then extracted from the sample.
COMPARING
The data is then compared with the database.
MATCHING
The software then decides whether the sample matches any picture in the database or not.
Unique facial data
VERIFIED

Is recognition different from detection?

Detection –

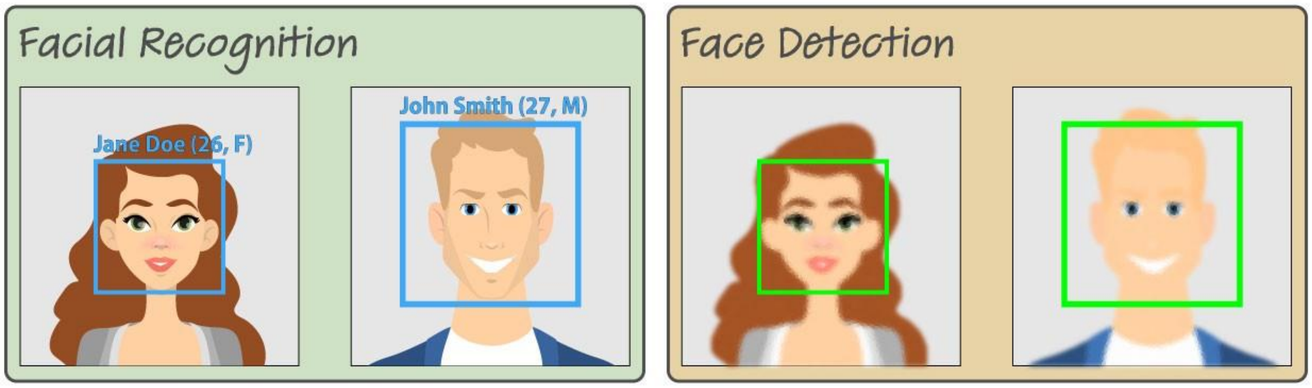
The ability to detect if there is some 'thing' vs nothing.

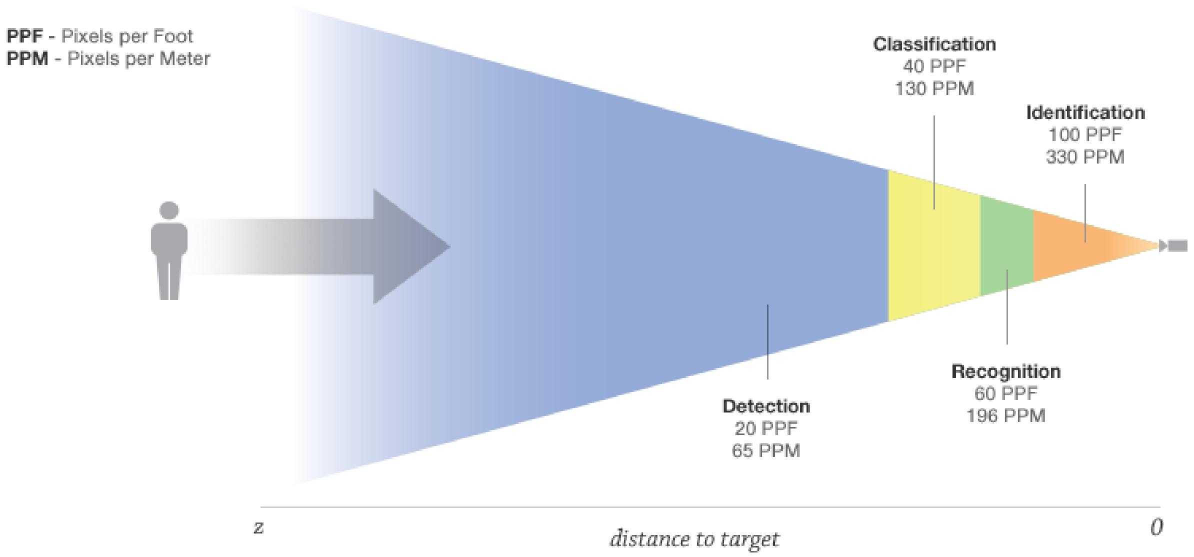
Recognition –

The ability **to recognize what type of** thing it is (person, animal, car, etc.)

Identification –

The ability to identify a specific individual from other people.


Facial Recognition
Jane Doe (26, F)
John Smith (27, M)
Face Detection


PPF - Pixels per Foot
PPM - Pixels per Meter
Classification
40 PPF
130 PPM
Identification
100 PPF
330 PPM
Recognition
60 PPF
196 PPM
Detection
20 PPF
65 PPM
z
distance to target
0

Introduction to OpenCV: -

OpenCV: It stands Computer Vision.

OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.

OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language.

- Computer Vision is a **bridge** between computer software and visualizations.
- It allows computers to **understand** and learn about the visualizations in the surroundings.
- **Example:** Determining the fruit based on the color, shape and size.

Using OpenCV

OpenCV is the most popular library for face recognition.

- OpenCV is an **open-source**library.
- It is supported by **various programming languages** such as R, Python and more.
- It runs on most of the **platforms**such as Windows, Linux and MacOS.

Implementation of Project : Face Recognition using Python

Covered topics in this project:-

- Image processing with basic function of opencv
- Video processing by usning opencv
- Video capture through webcam by using opencv
- Face detection
- Face recognition with dataset
- Face recognition by using webcam

Technology Used

- **Python3**
- **Python Libraries**
- **OpenCV**
- **Face_detection**
- **Face_recognition**

Need to install package:-

1. OpenCV

```
Pip install opencv-python
```

2. Face_Recognition

```
Pip install face_recognition
```

Used Libraries:-

1. cv2

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human.

2. face_recognition

Recognize and manipulate faces from Python or from the command line with the world's simplest face recognition library.

3. os

The OS module in Python provides functions for interacting with the operating system. OS comes under Python's standard utility modules. This module provides a portable way of using operating system-dependent functionality. The *os* and *os.

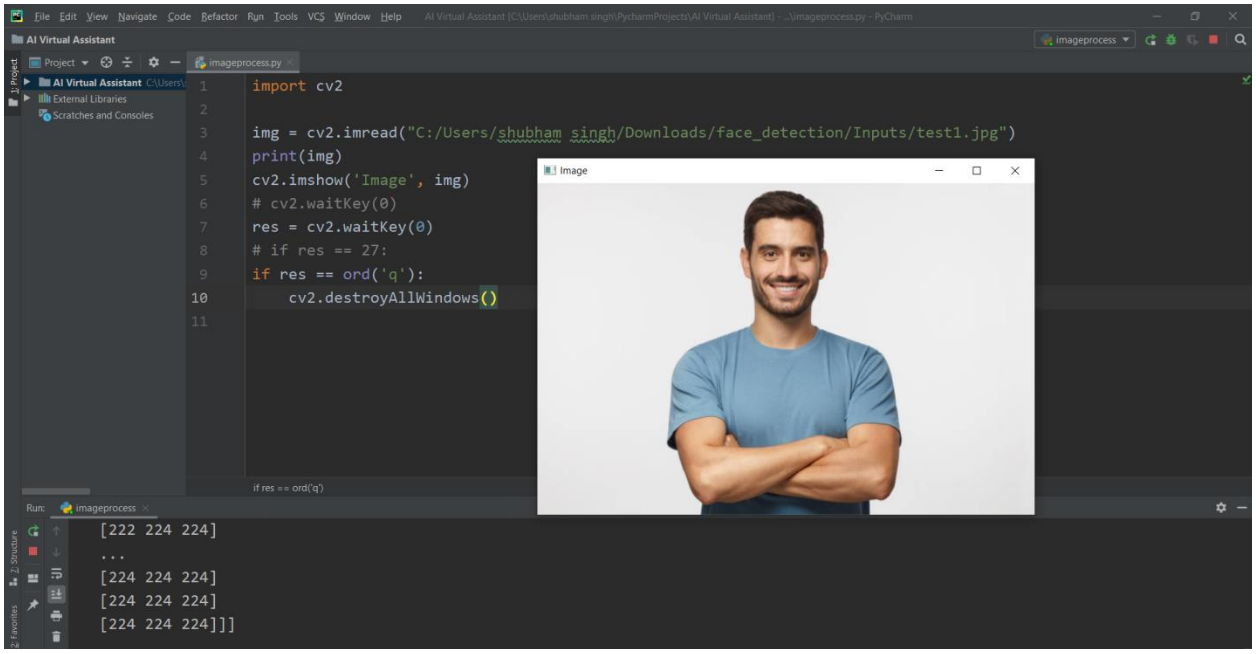
4. cv2_imshow

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. cv2. imshow() method is used to display an image in a window.

5. numpy

NumPy is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices.

1. Image processing with basic function of opencv



```
import cv2

img = cv2.imread("C:/Users/shubham singh/Downloads/face_detection/Inputs/test1.jpg")
print(img)
cv2.imshow('Image', img)
# cv2.waitKey(0)
res = cv2.waitKey(0)
# if res == 27:
if res == ord('q'):
    cv2.destroyAllWindows()
```

```
[222 224 224]
...
[224 224 224]
[224 224 224]
[224 224 224]]]
```

Basic function of OpenCV to test with images -

<u>Code</u>:-

import cv2

import numpy as np

img = cv2.imread("C:/Users/shubham singh/Downloads/face_detection/Inputs/test1.jpg")

imgGray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

imgBlur = cv2.GaussianBlur(imgGray, (7, 7), 0)

imgCanny = cv2.Canny(img, 100, 100)

# im = np.ones((500,500),np.uint8)

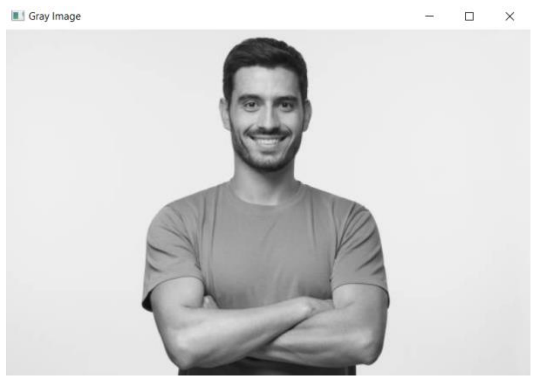
kernal = np.ones((5, 5), np.uint8)

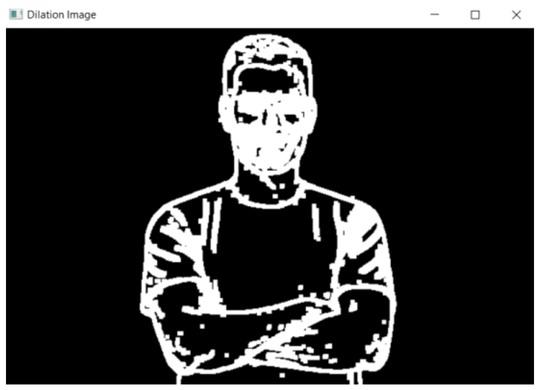
imgDilation = cv2.dilate(imgCanny, kernal, iterations=1)

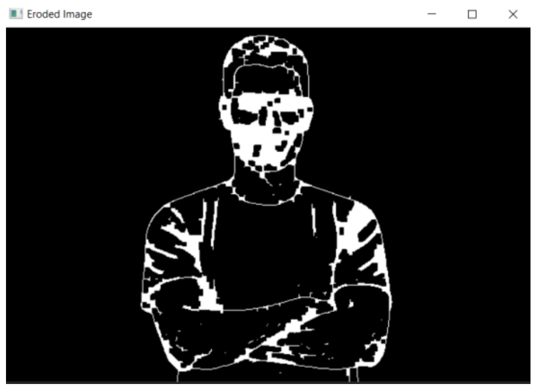
# Erode is opposit of Dilation

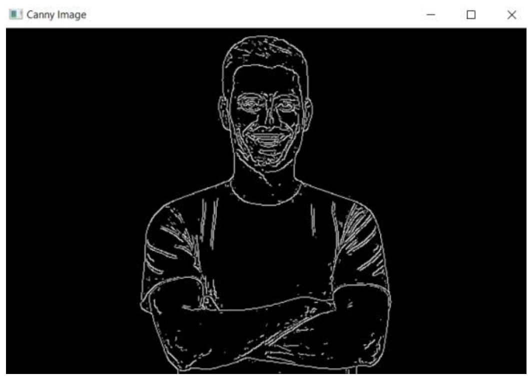
```
imgEroded = cv2.erode(imgDilation, kernal, iterations=1)

# cv2.imshow("IM",im)

cv2.imshow("Image ", img)
cv2.imshow("Gray Image ", imgGray)

cv2.imshow("Blur Image ", imgBlur)
cv2.imshow("Canny Image ", imgCanny)
cv2.imshow("Eroded Image ", imgEroded)

cv2.imshow("Dilation Image ", imgDilation)

cv2.waitKey(0)

cv2.destroyAllWindows()
```

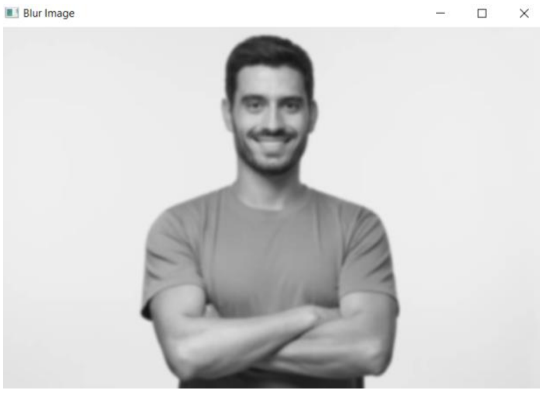
<u>Output images:-</u>

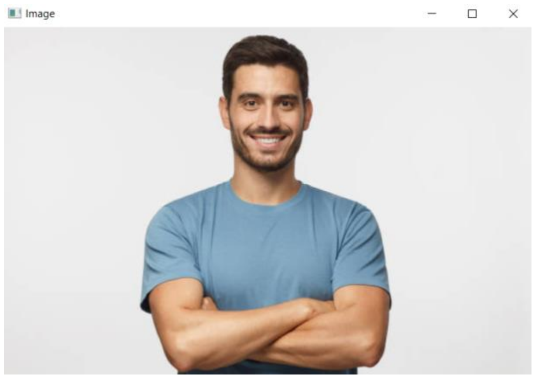




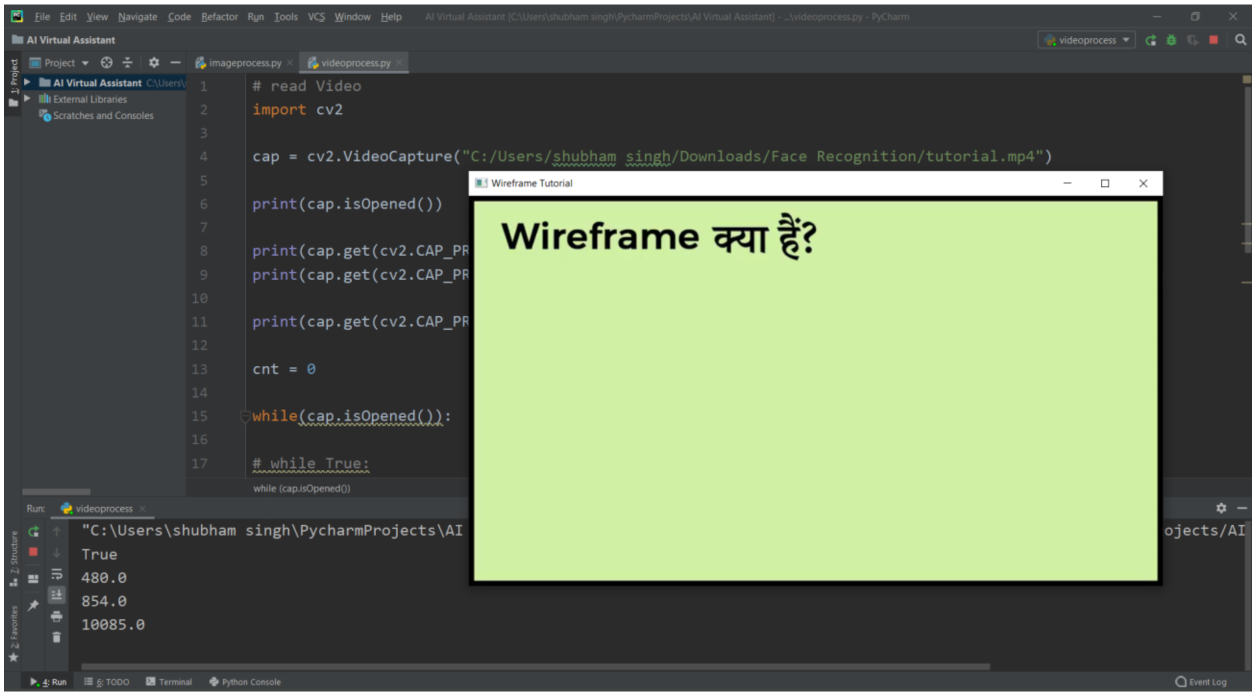








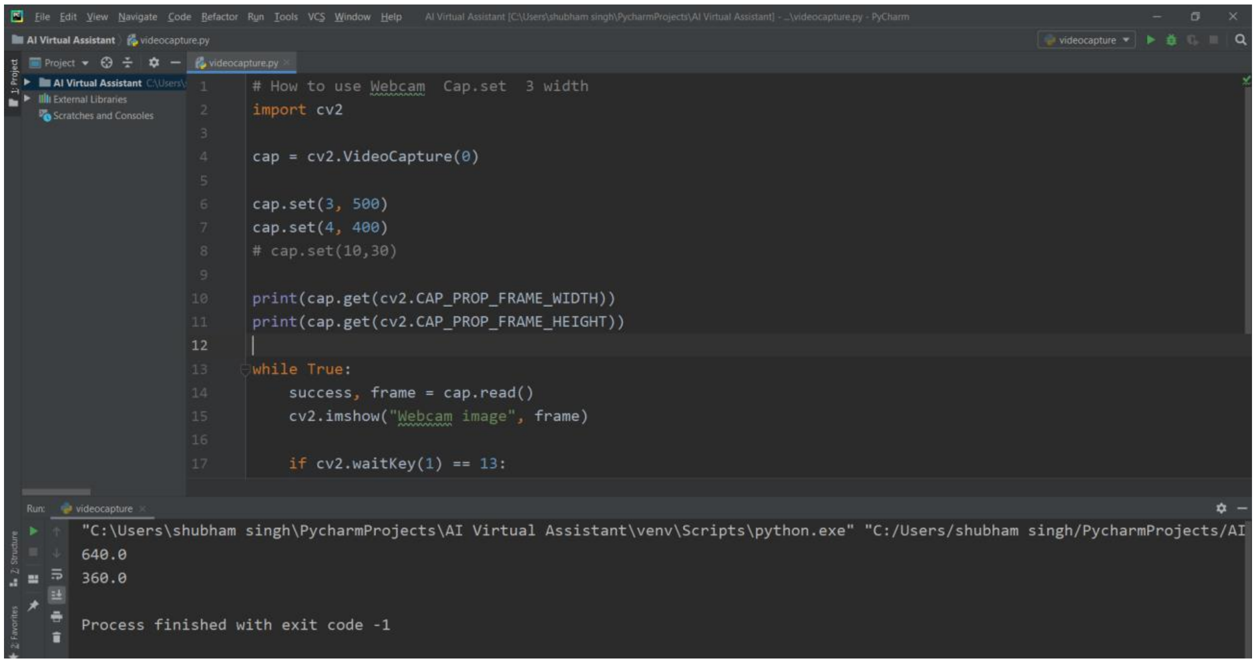
2. Video processing by usning opencv



```
# read Video
import cv2

cap = cv2.VideoCapture("C:/Users/shubham singh/Downloads/Face Recognition/tutorial.mp4")

print(cap.isOpened())

print(cap.get(cv2.CAP_PR
print(cap.get(cv2.CAP_PR

print(cap.get(cv2.CAP_PR

cnt = 0

while(cap.isOpened()):

# while True:
```

```
"C:\Users\shubham singh\PycharmProjects\AI                                  ojects/AI
True
480.0
854.0
10085.0
```

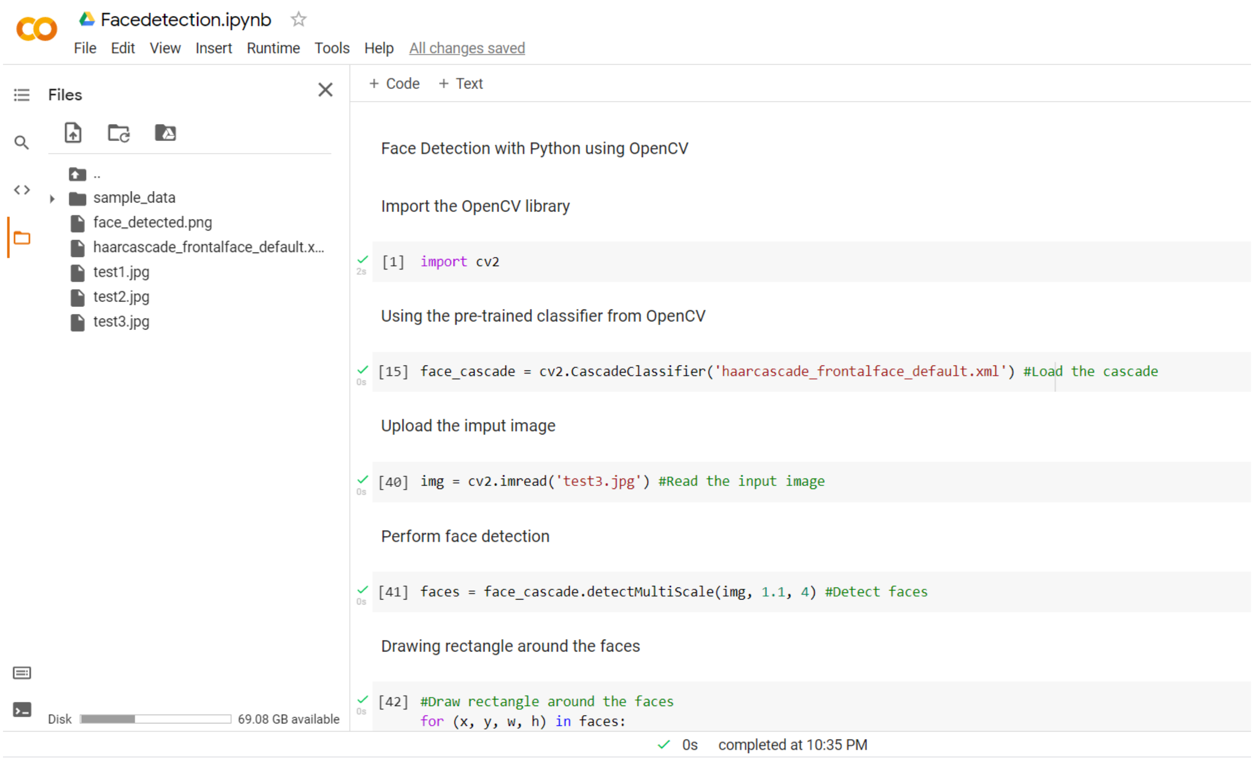
3. Video capture through webcam by using opencv



```
# How to use Webcam  Cap.set  3 width
import cv2

cap = cv2.VideoCapture(0)

cap.set(3, 500)
cap.set(4, 400)
# cap.set(10,30)

print(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
print(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

while True:
    success, frame = cap.read()
    cv2.imshow("Webcam image", frame)

    if cv2.waitKey(1) == 13:
```

```
"C:\Users\shubham singh\PycharmProjects\AI Virtual Assistant\venv\Scripts\python.exe" "C:/Users/shubham singh/PycharmProjects/AI
640.0
360.0

Process finished with exit code -1
```

4. Face detection



Face Detection with Python using OpenCV

Import the OpenCV library

```
[1] import cv2
```

Using the pre-trained classifier from OpenCV

```
[15] face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml') #Load the cascade
```

Upload the imput image

```
[40] img = cv2.imread('test3.jpg') #Read the input image
```

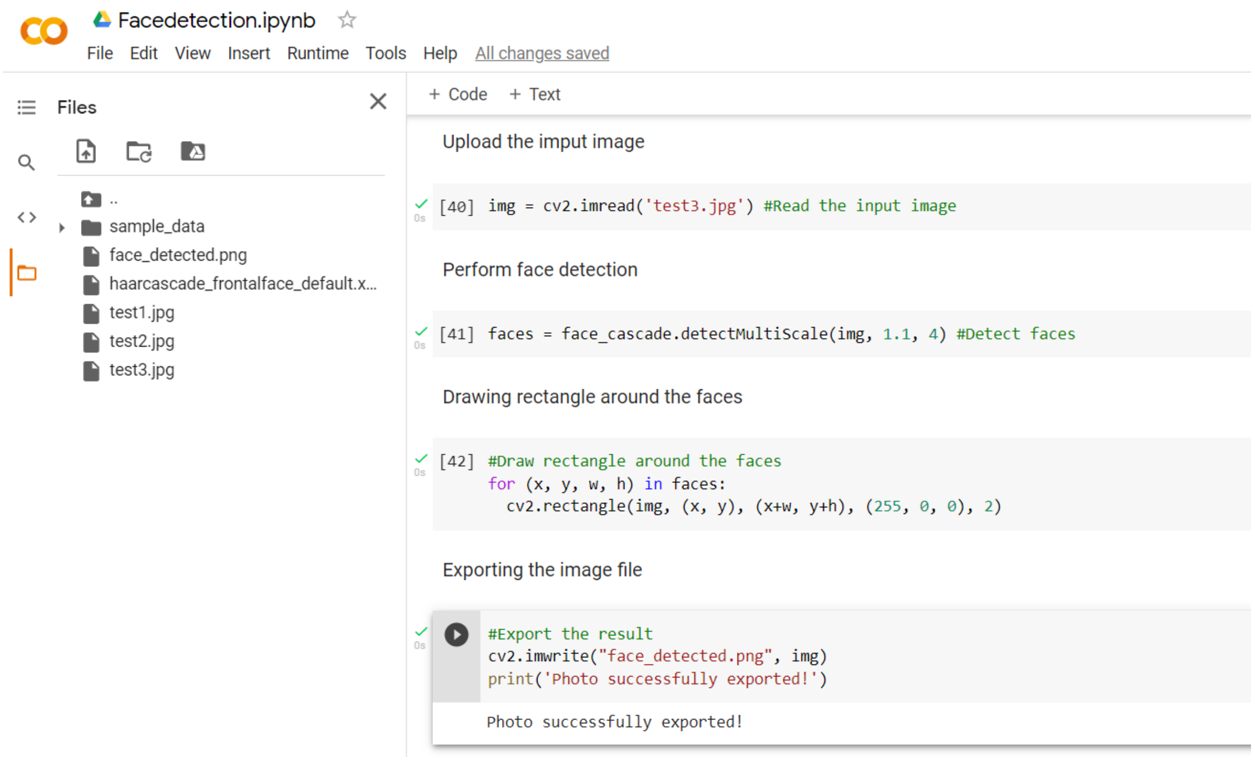
Perform face detection

```
[41] faces = face_cascade.detectMultiScale(img, 1.1, 4) #Detect faces
```

Drawing rectangle around the faces

```
[42] #Draw rectangle around the faces
     for (x, y, w, h) in faces:
```

✓ 0s completed at 10:35 PM



Facedetection.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Files
..
sample_data
face_detected.png
haarcascade_frontalface_default.x...
test1.jpg
test2.jpg
test3.jpg
+ Code + Text

Upload the imput image

```
[40] img = cv2.imread('test3.jpg') #Read the input image
```

Perform face detection

```
[41] faces = face_cascade.detectMultiScale(img, 1.1, 4) #Detect faces
```

Drawing rectangle around the faces

```
[42] #Draw rectangle around the faces
     for (x, y, w, h) in faces:
       cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
```

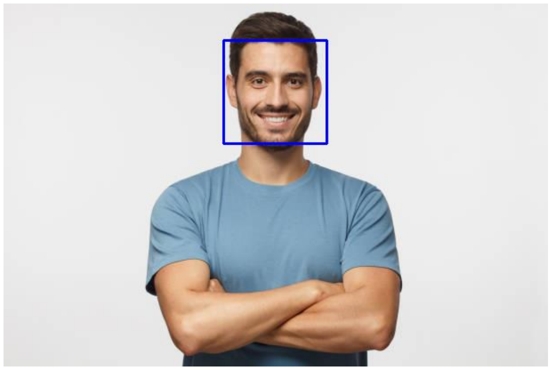
Exporting the image file

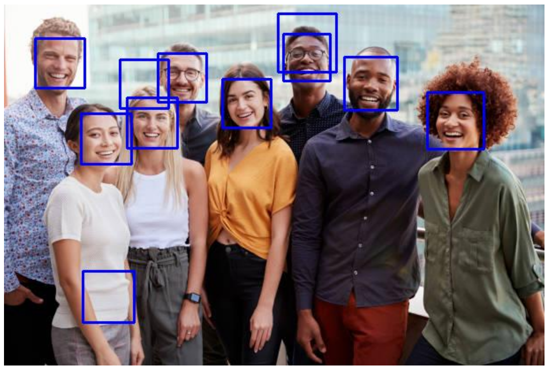
```
#Export the result
cv2.imwrite("face_detected.png", img)
print('Photo successfully exported!')
```

```
Photo successfully exported!
```

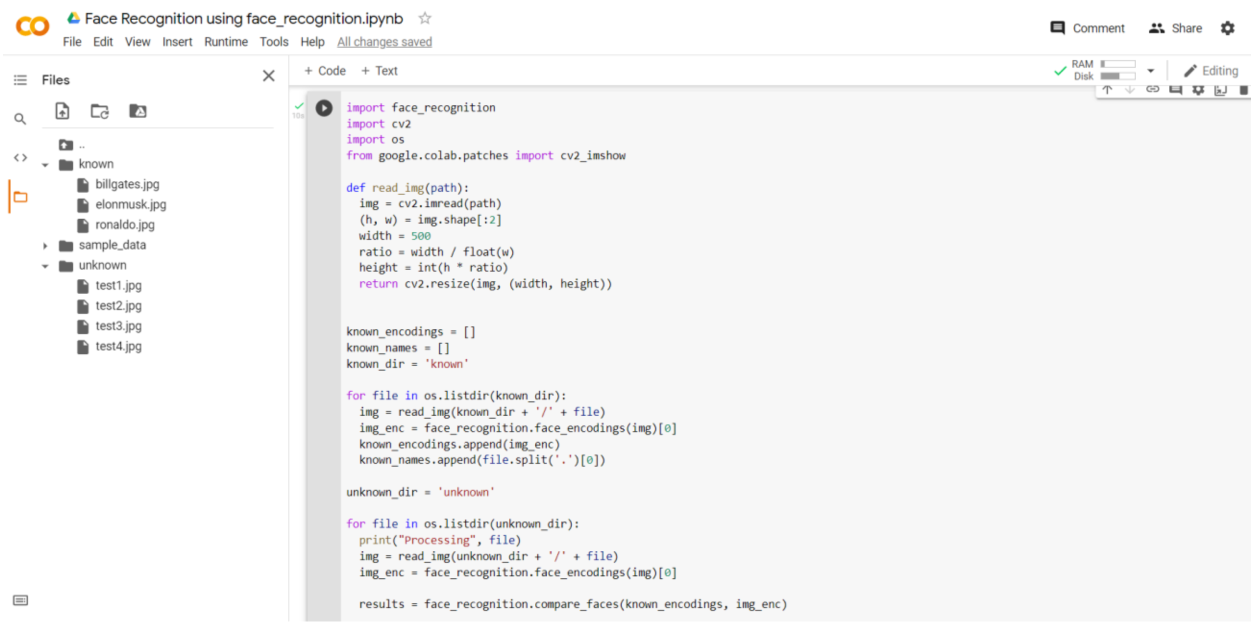
Before

After

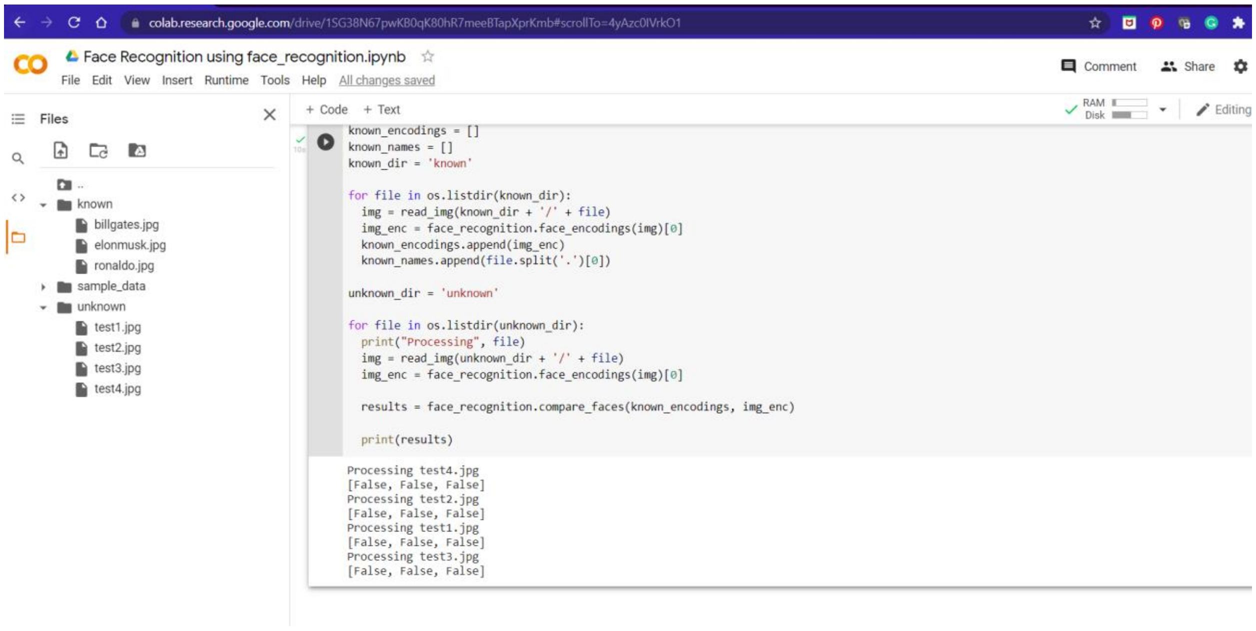




5. Face recognition with dataset

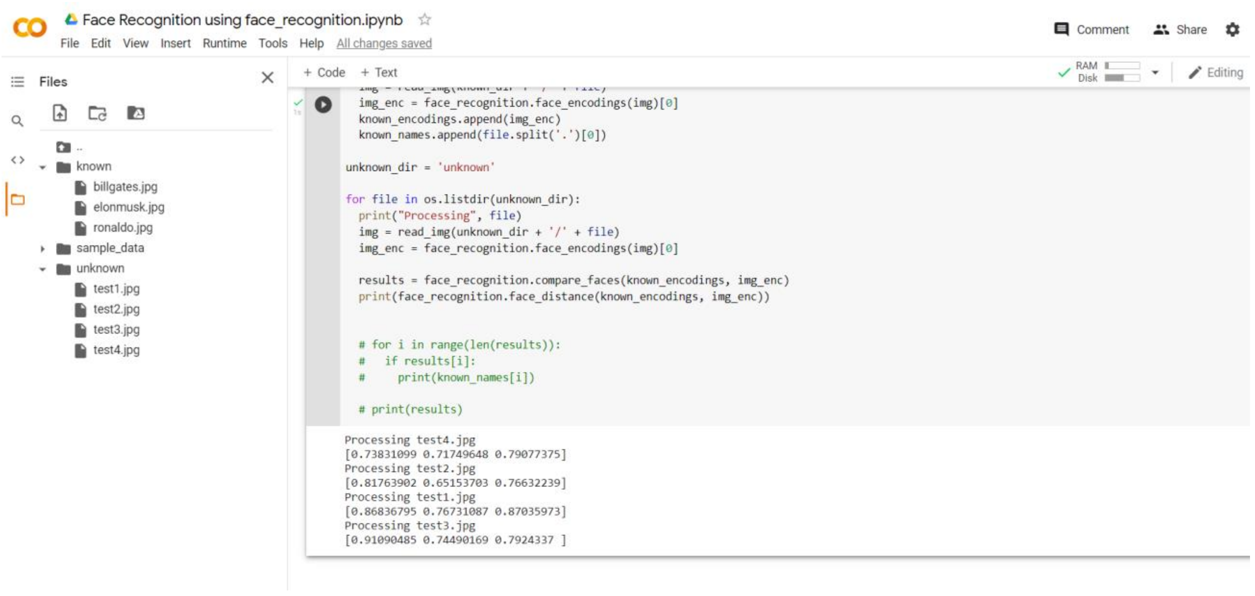


```
import face_recognition
import cv2
import os
from google.colab.patches import cv2_imshow

def read_img(path):
  img = cv2.imread(path)
  (h, w) = img.shape[:2]
  width = 500
  ratio = width / float(w)
  height = int(h * ratio)
  return cv2.resize(img, (width, height))


known_encodings = []
known_names = []
known_dir = 'known'

for file in os.listdir(known_dir):
  img = read_img(known_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]
  known_encodings.append(img_enc)
  known_names.append(file.split('.')[0])

unknown_dir = 'unknown'

for file in os.listdir(unknown_dir):
  print("Processing", file)
  img = read_img(unknown_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]

  results = face_recognition.compare_faces(known_encodings, img_enc)
```


colab.research.google.com/drive/1SG38N67pwKB0qK80hR7meeBTapXprKmb#scrollTo=4yAzc0lVrkO1
Face Recognition using face_recognition.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment Share
+ Code + Text
RAM Disk
Editing
Files
..
known
billgates.jpg
elonmusk.jpg
ronaldo.jpg
sample_data
unknown
test1.jpg
test2.jpg
test3.jpg
test4.jpg

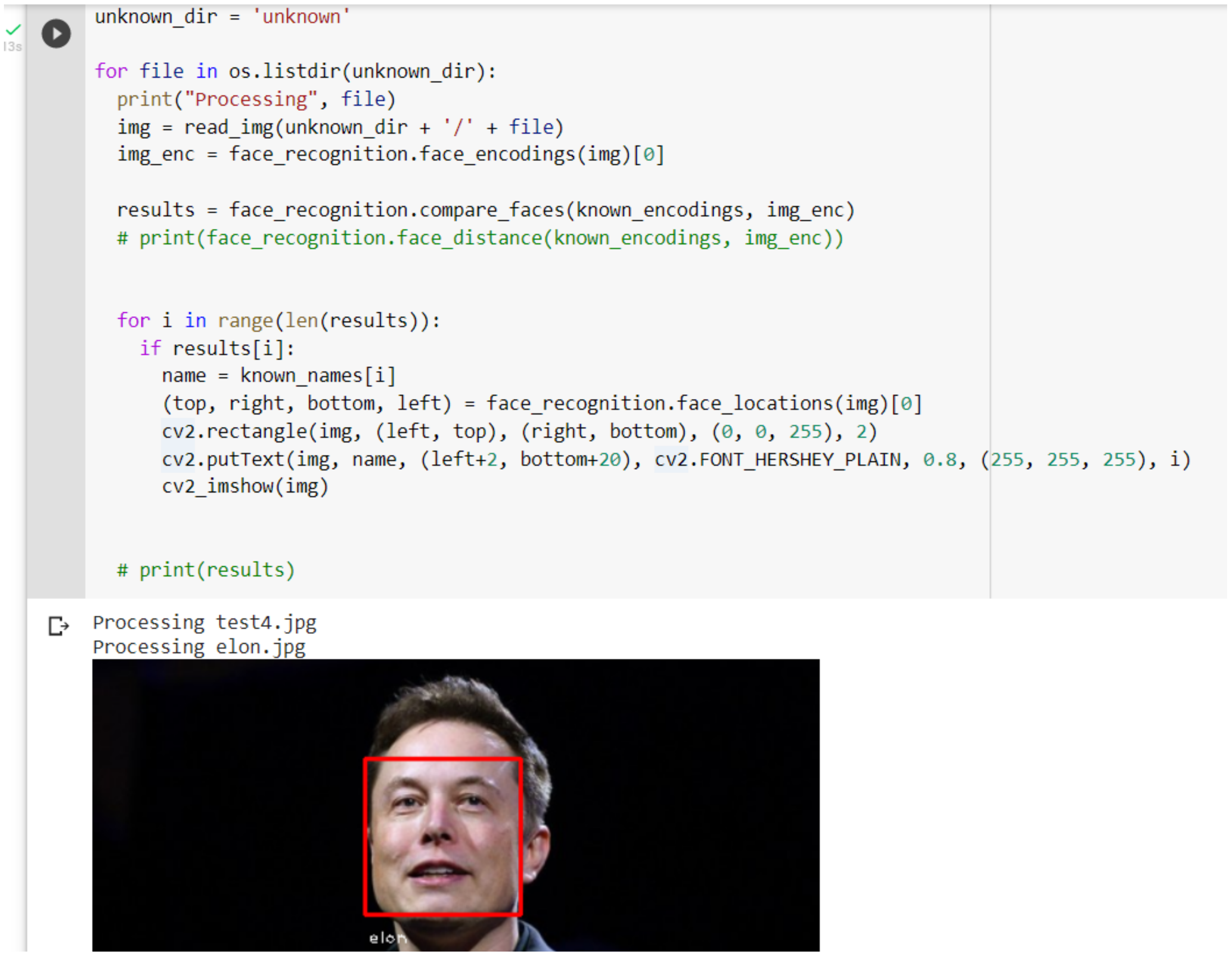
```
known_encodings = []
known_names = []
known_dir = 'known'

for file in os.listdir(known_dir):
  img = read_img(known_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]
  known_encodings.append(img_enc)
  known_names.append(file.split('.')[0])

unknown_dir = 'unknown'

for file in os.listdir(unknown_dir):
  print("Processing", file)
  img = read_img(unknown_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]

  results = face_recognition.compare_faces(known_encodings, img_enc)

  print(results)
```

```
Processing test4.jpg
[False, False, False]
Processing test2.jpg
[False, False, False]
Processing test1.jpg
[False, False, False]
Processing test3.jpg
[False, False, False]
```



```
  img_enc = face_recognition.face_encodings(img)[0]
  known_encodings.append(img_enc)
  known_names.append(file.split('.')[0])

unknown_dir = 'unknown'

for file in os.listdir(unknown_dir):
  print("Processing", file)
  img = read_img(unknown_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]

  results = face_recognition.compare_faces(known_encodings, img_enc)
  print(face_recognition.face_distance(known_encodings, img_enc))


  # for i in range(len(results)):
  #   if results[i]:
  #     print(known_names[i])

  # print(results)
```

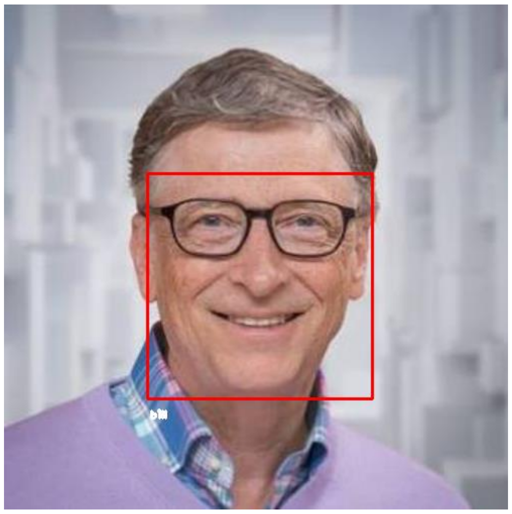
```
Processing test4.jpg
[0.73831099 0.71749648 0.79077375]
Processing test2.jpg
[0.81763902 0.65153703 0.76632239]
Processing test1.jpg
[0.86836795 0.76731087 0.87035973]
Processing test3.jpg
[0.91090485 0.74490169 0.7924337 ]
```

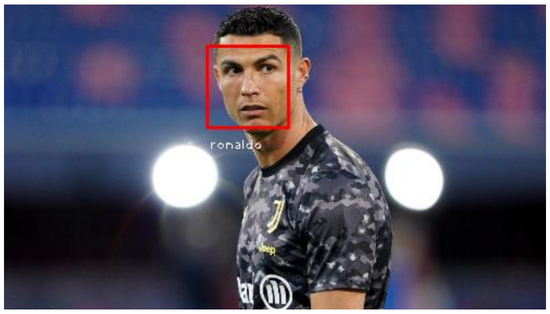


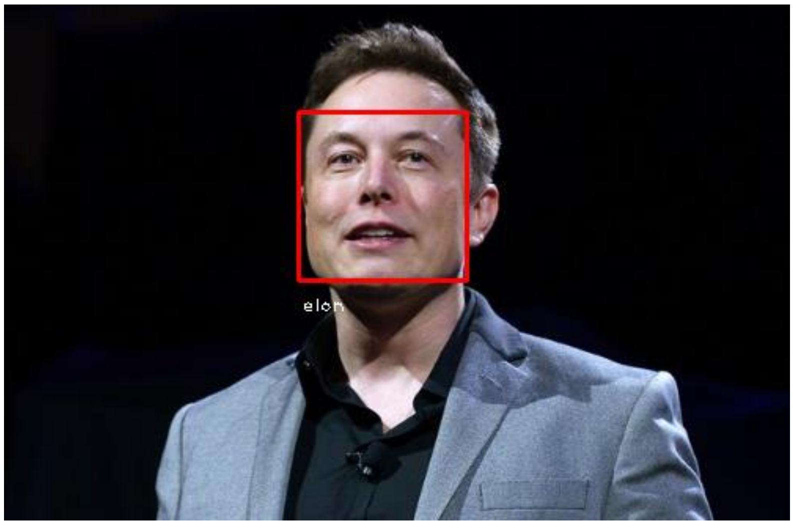
```
unknown_dir = 'unknown'

for file in os.listdir(unknown_dir):
  print("Processing", file)
  img = read_img(unknown_dir + '/' + file)
  img_enc = face_recognition.face_encodings(img)[0]

  results = face_recognition.compare_faces(known_encodings, img_enc)
  # print(face_recognition.face_distance(known_encodings, img_enc))


  for i in range(len(results)):
    if results[i]:
      name = known_names[i]
      (top, right, bottom, left) = face_recognition.face_locations(img)[0]
      cv2.rectangle(img, (left, top), (right, bottom), (0, 0, 255), 2)
      cv2.putText(img, name, (left+2, bottom+20), cv2.FONT_HERSHEY_PLAIN, 0.8, (255, 255, 255), i)
      cv2_imshow(img)


  # print(results)
```

```
Processing test4.jpg
Processing elon.jpg
```

<u>Output image's:-</u>







Project & Reference URL's:-

Project url: -

- https://github.com/shubhamsc11/Face_Recognition

Reference url: -

- https://pypi.org/project/face-recognition/
- https://github.com/ageitgey/face_recognition
- https://www.mygreatlearning.com/blog/real-time-face-detection/
- https://www.youtube.com/watch?v=987QtKPZ-P0&t=53s

Thank You