**Face recognition using CNN:**

**Program:**

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.preprocessing import image

from tensorflow.keras.optimizers import RMSprop

import tensorflow as tf

import matplotlib.pyplot as plt

import cv2

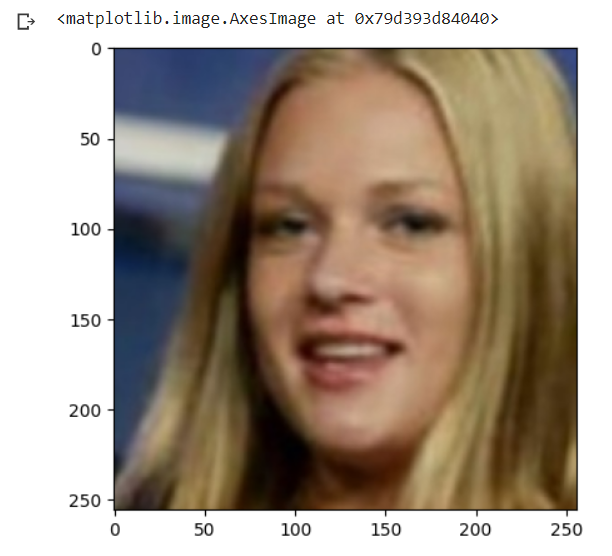
import os

import numpy as np

img = image.load\_img("/content/drive/MyDrive/deep learning/archive (1)/basedata/training/face/person\_0000.jpg")

plt.imshow(img)

**Output:**

****

cv2.imread("/content/drive/MyDrive/deep learning/archive (1)/basedata/training/face/person\_0000.jpg").shape

**Output:**

****

train = ImageDataGenerator(rescale = 1/255)

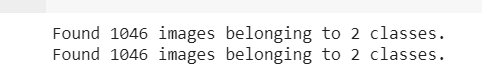
validation =ImageDataGenerator(rescale = 1/255)

trained\_dataset = train.flow\_from\_directory("/content/drive/MyDrive/deep learning/archive (1)/basedata/training",target\_size=(200,200),batch\_size= 3,

                                                                            class\_mode ='binary')

validation\_dataset = train.flow\_from\_directory("/content/drive/MyDrive/deep learning/archive (1)/basedata/validation",target\_size=(200,200),batch\_size= 3,class\_mode ='binary')

**Output:**



trained\_dataset.class\_indices

**Output:**



trained\_dataset.classes

**Output:**



model = tf.keras.models.Sequential([tf.keras.layers.Conv2D(16,(3,3),activation='relu',input\_shape=(200,200,3)),

                                    tf.keras.layers.MaxPool2D(2,2),

                                    tf.keras.layers.Conv2D(32,(3,3),activation='relu',input\_shape=(200,200,3)),

                                    tf.keras.layers.MaxPool2D(2,2),

                                    tf.keras.layers.Conv2D(64,(3,3),activation='relu',input\_shape=(200,200,3)),

                                    tf.keras.layers.MaxPool2D(2,2),

                                    tf.keras.layers.Flatten(),

                                    tf.keras.layers.Dense(512,activation='relu'),

                                    tf.keras.layers.Dense(1,activation='sigmoid')

                                   ])

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

             optimizer = RMSprop(learning\_rate=0.001),

             metrics=['accuracy'])

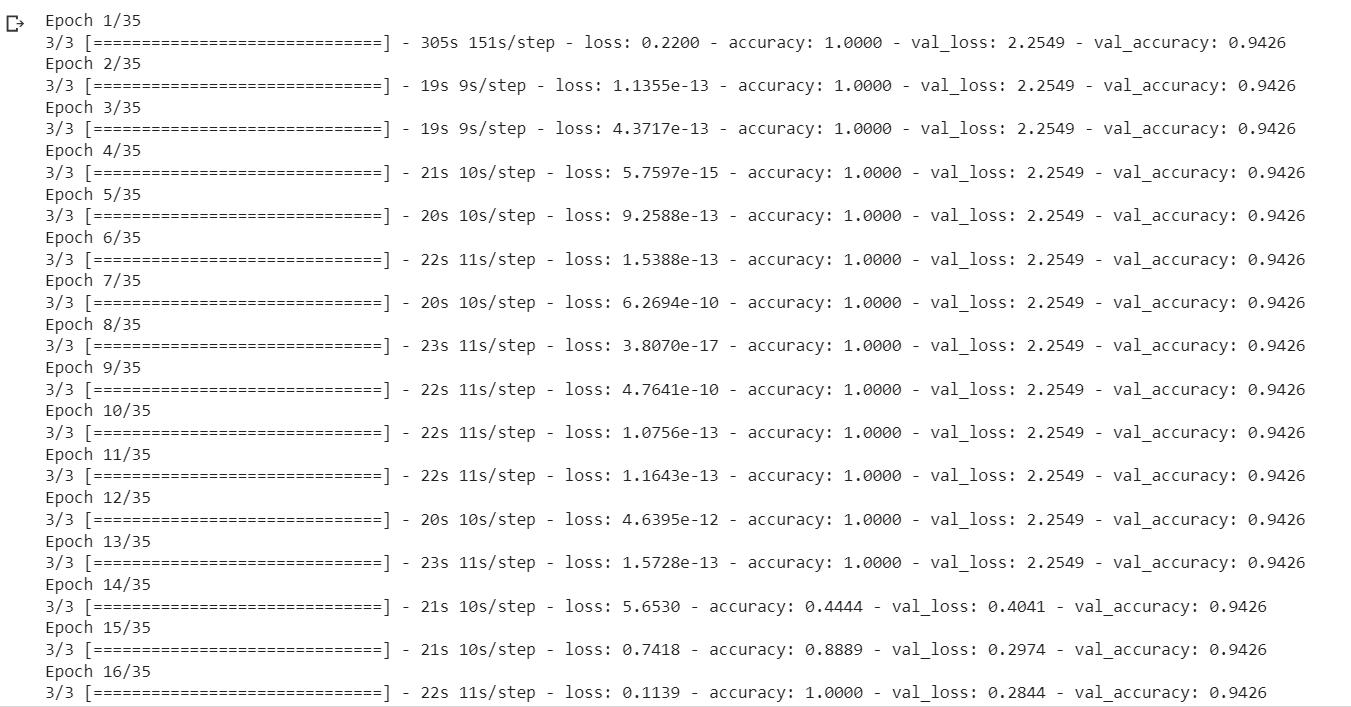
model\_fit = model.fit(trained\_dataset,

                     steps\_per\_epoch = 3,

                     epochs =35,

                     validation\_data = validation\_dataset)

**Output:**

****

dir\_path = "/content/drive/MyDrive/deep learning/testing"

for i in os.listdir(dir\_path):

    img = image.load\_img(dir\_path + '//' + i,target\_size=(200,200))

    plt.imshow(img)

    plt.show()

    X= image.img\_to\_array(img)

    X = np.expand\_dims(X,axis=0)

    images = np.vstack([X])

    val = model.predict(images)

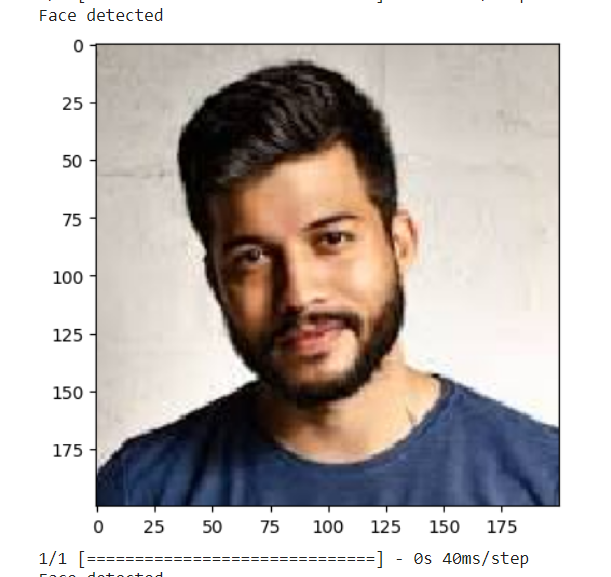
    if val == 0:

        print("Face detected")

    else:

        print("Face not detected")

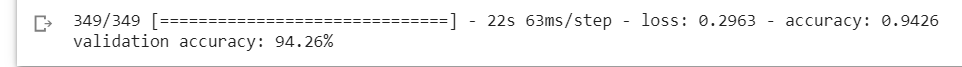
**Output:**

****

validation\_accuracy = model.evaluate(validation\_dataset)

print(f"validation accuracy: {validation\_accuracy[1] \* 100:.2f}%")

**Output:**

****

import tensorflow.keras.models

model.save('detection\_model.h5')

loaded\_model = tf.keras.models.load\_model('detection\_model.h5')

loaded\_model