import os

from tkinter import \*

import tkinter.font as font

import tkinter.simpledialog as simpledialog

from PIL import Image, ImageTk

from tkinter import messagebox as mbox

import cv2

import face\_recognition

from numpy import info

from sklearn import svm

def test\_svm():

# Training the SVC classifier

# The training data would be all the face encodings from all the known images and the labels are their names

encodings = []

names = []

train\_dir='C:\\Users\\91906\\Images\\train'

for person in train\_dir:

pix = os.listdir("train/" + person)

# Loop through each training image for the current person

for person\_img in pix:

# Get the face encodings for the face in each image file

face = face\_recognition.load\_image\_file(

"train/" + person + "/" + person\_img)

face\_bounding\_boxes = face\_recognition.face\_locations(face)

# If training image contains exactly one face

if len(face\_bounding\_boxes) == 1:

face\_enc = face\_recognition.face\_encodings(face)[0]

# Add face encoding for current image with corresponding label (name) to the training data

encodings.append(face\_enc)

names.append(person)

else:

print(person + "/" + person\_img +" was skipped and can't be used for training")

# Create and train the SVC classifier

clf = svm.SVC(gamma='scale')

clf.fit(encodings, names)

# Load the test image with unknown faces into a numpy array

test\_image = face\_recognition.load\_image\_file('C:\\Users\\91906\\Images\\test')

# Find all the faces in the test image using the default HOG-based model

face\_locations = face\_recognition.face\_locations(test\_image)

num = len(face\_locations)

print("Number of faces detected: ", num)

# Predict all the faces in the test image using the trained classifier

list\_names = []

print("Found:")

for i in range(num):

test\_image\_enc = face\_recognition.face\_encodings(test\_image)[i]

name = clf.predict([test\_image\_enc])

print(name)

list\_names.append(\*name)

display\_name(list\_names)