from sklearn.metrics import accuracy\_score

from sklearn.svm import SVC

from sklearn.model\_selection import train\_test\_split

from sklearn.decomposition import PCA

warnings.filterwarnings("ignore", category=FutureWarning)

#Function that returns the co-ordinates of faces if detected

def Face\_Recognition(img):

@@ -46,23 +45,18 @@ def Face\_Recognition(img):

nomask = nomask.reshape((250,50\*50\*3))

X = np.concatenate((mask,nomask))

labels = np.zeros(X.shape[0])

labels[100:] = 1

labels[250:] = 1

mask\_nomask = {"Mask":0,"No Mask":1}

print(X.shape,labels.shape)

#Creating a Support Vector Model

svm = SVC(kernel='linear', C = 1.0)

x\_train,x\_test,y\_train,y\_test = train\_test\_split(X,labels,test\_size = 0.3)

print(x\_train.shape,y\_train.shape,x\_test.shape,y\_test.shape)

#Dimentionality Reduciton(Reduction of feature variables)

pca = PCA(n\_components=3)

x\_train = pca.fit\_transform(x\_train)

x\_test = pca.transform(x\_test)

print(x\_test.shape)

svm.fit(x\_train,y\_train)

y\_pred = svm.predict(x\_test)

print(accuracy\_score(y\_pred,y\_test))

img = cv2.imread("D:/Data/Downloads/download (1).jpg")

img = cv2.imread("Images\download (1).jpg")

faces = Face\_Recognition(img)

for (x,y,w,h) in faces:

cv2.rectangle(img,(x,y),(x+w,y+h),(255,255,0))

@@ -74,7 +68,5 @@ def Face\_Recognition(img):

print(img.shape)

img = img.reshape((1,-1))

print(img.shape)

img = pca.transform(img)

print(img.shape)

y\_pred = svm.predict(img)

print(y\_pred)