**Face Mask Detection**

**To download the dataset**

!git clone https://github.com/misbah4064/face\_mask\_detection.git

%cd face\_mask\_detection

**Train the model**

import cv2

import os

from tensorflow.keras.preprocessing.image import img\_to\_array

from tensorflow.keras.models import load\_model

from tensorflow.keras.applications.mobilenet\_v2 import preprocess\_input

import numpy as np

from google.colab.patches import cv2\_imshow

faceCascade = cv2.CascadeClassifier("haarcascade\_frontalface\_alt2.xml")

model = load\_model("mask\_recog.h5")

def face\_mask\_detector(frame):

  # frame = cv2.imread(fileName)

  gray = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

  faces = faceCascade.detectMultiScale(gray,

                                        scaleFactor=1.1,

                                        minNeighbors=5,

                                        minSize=(60, 60),

                                        flags=cv2.CASCADE\_SCALE\_IMAGE)

  faces\_list=[]

  preds=[]

  for (x, y, w, h) in faces:

      face\_frame = frame[y:y+h,x:x+w]

      face\_frame = cv2.cvtColor(face\_frame, cv2.COLOR\_BGR2RGB)

      face\_frame = cv2.resize(face\_frame, (224, 224))

      face\_frame = img\_to\_array(face\_frame)

      face\_frame = np.expand\_dims(face\_frame, axis=0)

      face\_frame =  preprocess\_input(face\_frame)

      faces\_list.append(face\_frame)

      if len(faces\_list)>0:

          preds = model.predict(faces\_list)

      for pred in preds:

          (mask, withoutMask) = pred

      label = "Mask" if mask > withoutMask else "No Mask"

      color = (0, 255, 0) if label == "Mask" else (0, 0, 255)

      label = "{}: {:.2f}%".format(label, max(mask, withoutMask) \* 100)

      cv2.putText(frame, label, (x, y- 10),

                  cv2.FONT\_HERSHEY\_SIMPLEX, 1, color, 2)

  cv2.rectangle(frame, (x, y), (x + w, y + h),color, 3)

  # cv2\_imshow(frame)

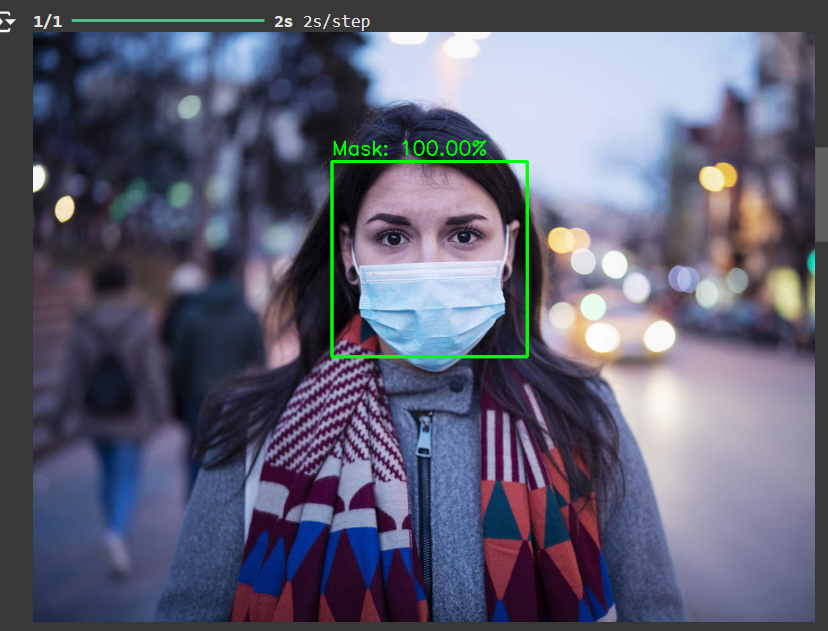
  return frame

**Test the model**

input\_image = cv2.imread("image.jpg")

output = face\_mask\_detector(input\_image)

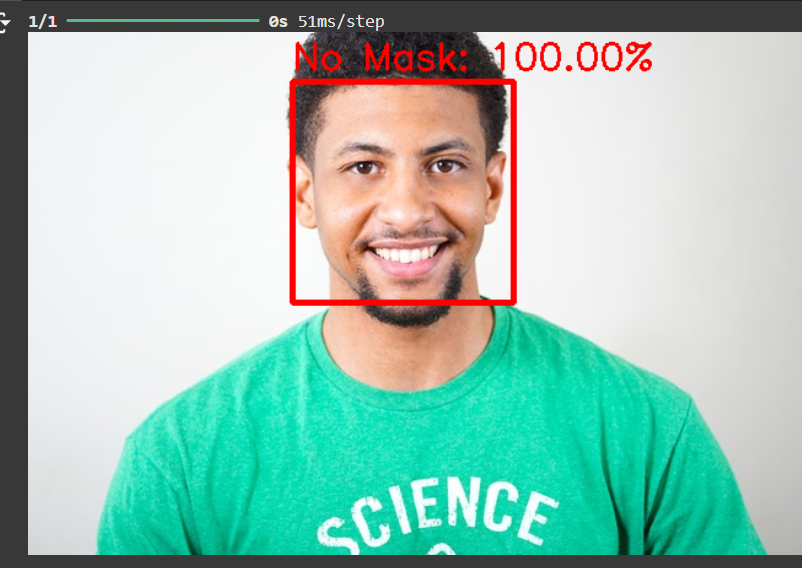
cv2\_imshow(output)

**predicted output**

input\_image = cv2.imread("image4.jpg")

output = face\_mask\_detector(input\_image)

cv2\_imshow(output)

**predicted output**