**Face Mask Detection Using OpenCV in Python**

**Abstract:** In this project I am going to implement face mask detection problem. Today it has become mandatory for all the citizens to wear a face mask to protect themselves from COVID-19. This application can be helpful for all the shop owners, offices, banks or any public place because if anyone is not wearing a mask then he or she must not be allowed in that area. So, to take care of this problem we don’t need any guard or person who keeps a watch on people. We can integrate a camera which continuously clicks pictures of humans and detect from their faces whether they are wearing a face mask or not.

**Introduction:** Before implementing face mask detection problem, first we need to understand that how to handle images. Images are simply a collection of colors in red, green and blue format. As a human we see an image with some object or shape in it, but for computer it is just an array with color values range from 0 to 255. The way computer sees anything is different from the way human see an image. But that’s the good news for us because if we got an array of the image than it becomes simple for us to implement any algorithm on that array.

**Introduction to OpenCV**

* OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library.
* The library has more than 2500 optimized algorithms.
* It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS.
* Will help us to load images in Python and convert them into array.
* Each index of array represents (red, green, blue) color pixel which ranges from 0 to 255.

**Features of OpenCV**

* Face Detection
* Geometric Transformations
* Image Thresholding
* Smoothing Images
* Canny Edge Detection
* Background Removals
* Image Segmentation

**Steps to Perform Image Processing:**

* Load images using Python or any other programming we are working on.
* Convert images into array
* And finally apply some algorithm on that array

Another good thing is that we have a library known as OpenCV which will help us to read the image and return array of color pixels.

**Getting started with OpenCV**

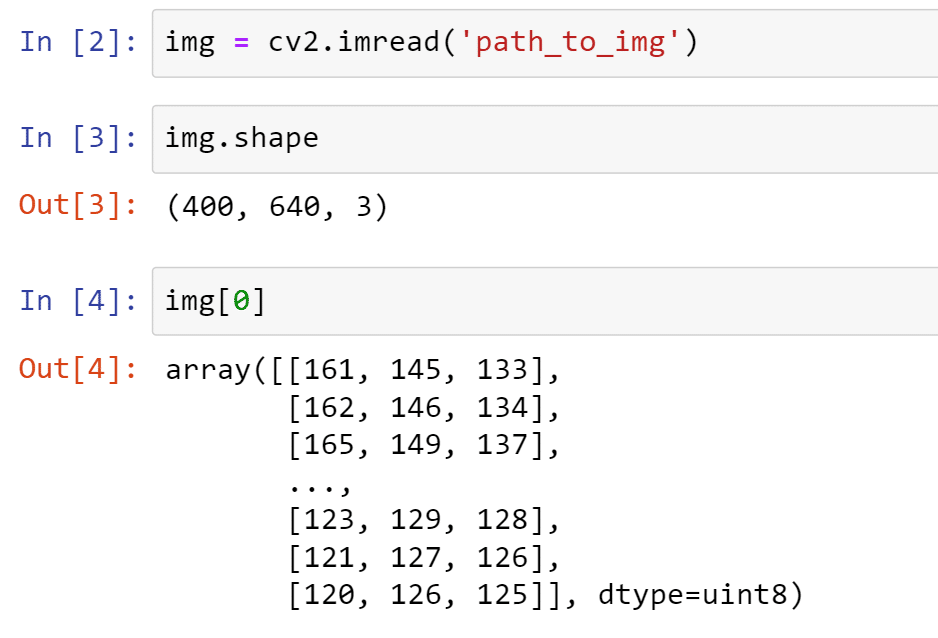
We can install OpenCV directly using Jupyter Notebook



Now we are ready to perform basic image processing using OpenCV Package. So first load OpenCV package like this:

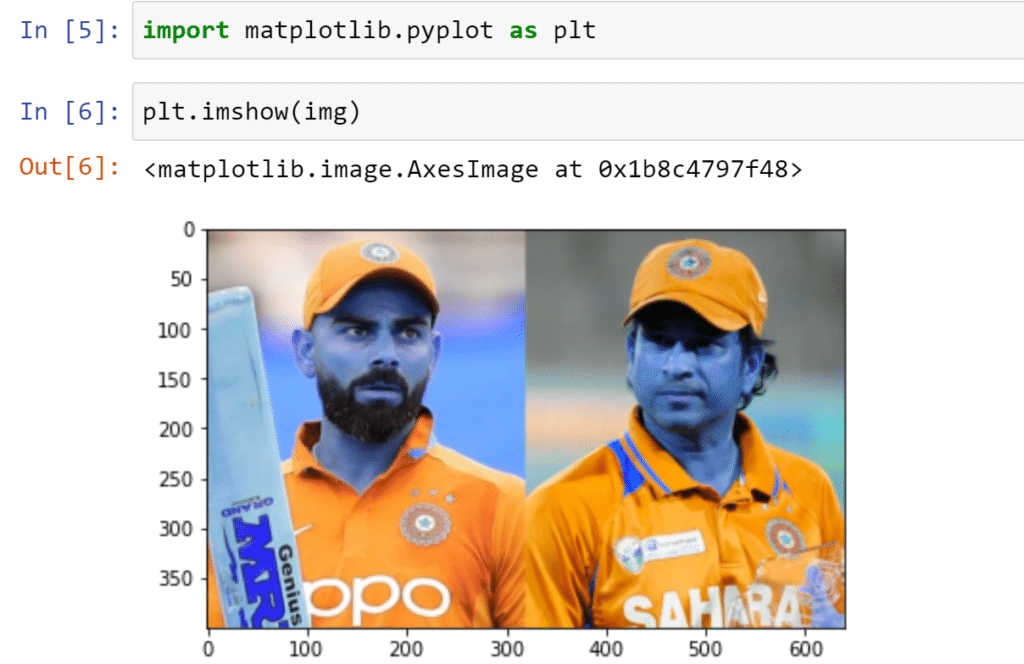


Now after importing OpenCV first let’s read an image:

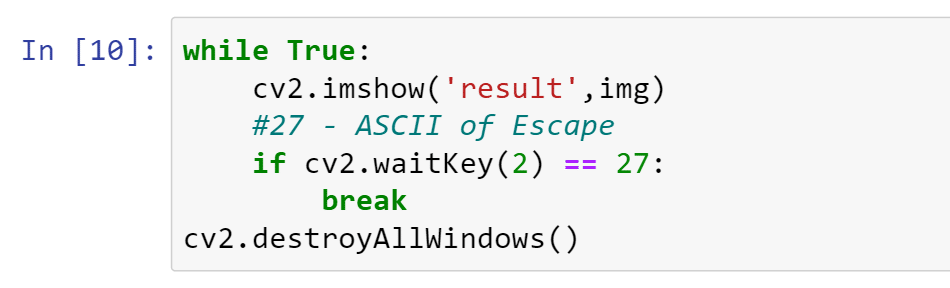


So, when we read any image using OpenCV it returns object of numpy array by default and using img.shape we are checking the height and width of image and also it returns 3 which is the color channel of image. Now we can see the values of array are the color values actually.

Using matplotlib we can visualize the image:



And to open image in new window we can write this code :



Now after running this code you will be able to see image open in a new window.

First we started a while loop then using cv2.waitKey.

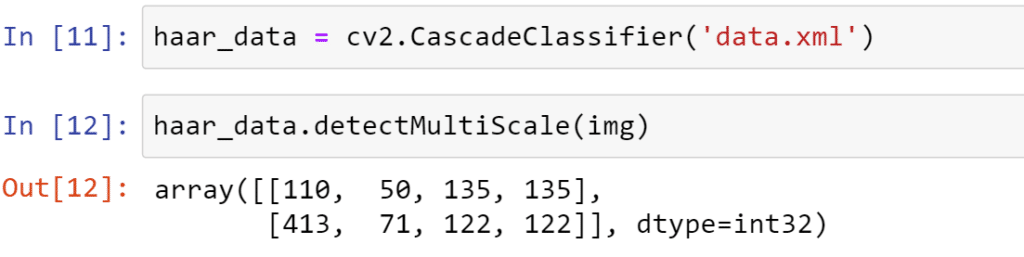
Now what cv2.waitKey is doing? It is mentioned in the below image where 2 is the time in milliseconds and 27 is ASCII number of escape key. So it means if you press escape then loop is going to break and in last line it will destroy or close all the windows that are opened by OpenCV.

**Face Detection Using OpenCV**

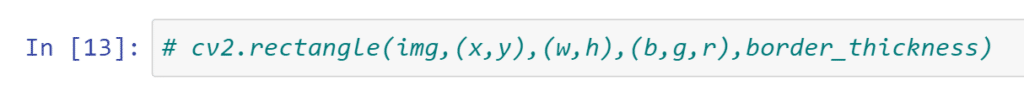
So now we are going to see how to detect face from an image. Face detection algorithm was introduced by Viola and Jones in 2001. They divided this algorithm in four stages:

* Haar Features Selection
* Integral Images
* AdaBoost
* Cascading Classifier

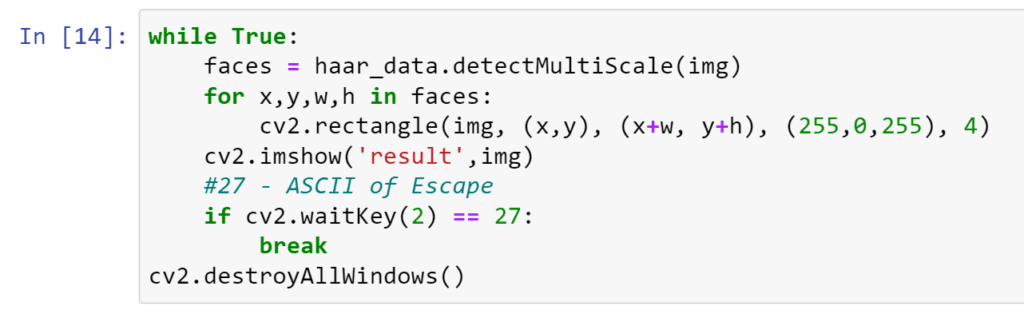
We do not have to worry about these steps right now because we already have a XML file which is going to help us to detect faces from the image.



This code returns x, y, width and height of the face detected in the image. And we can draw a rectangle on the face using this code:



We will iterate over the array returned to us by detectMultiScale method and put x,y,w,h in cv2.rectangle.Now let’s combine all the codes :

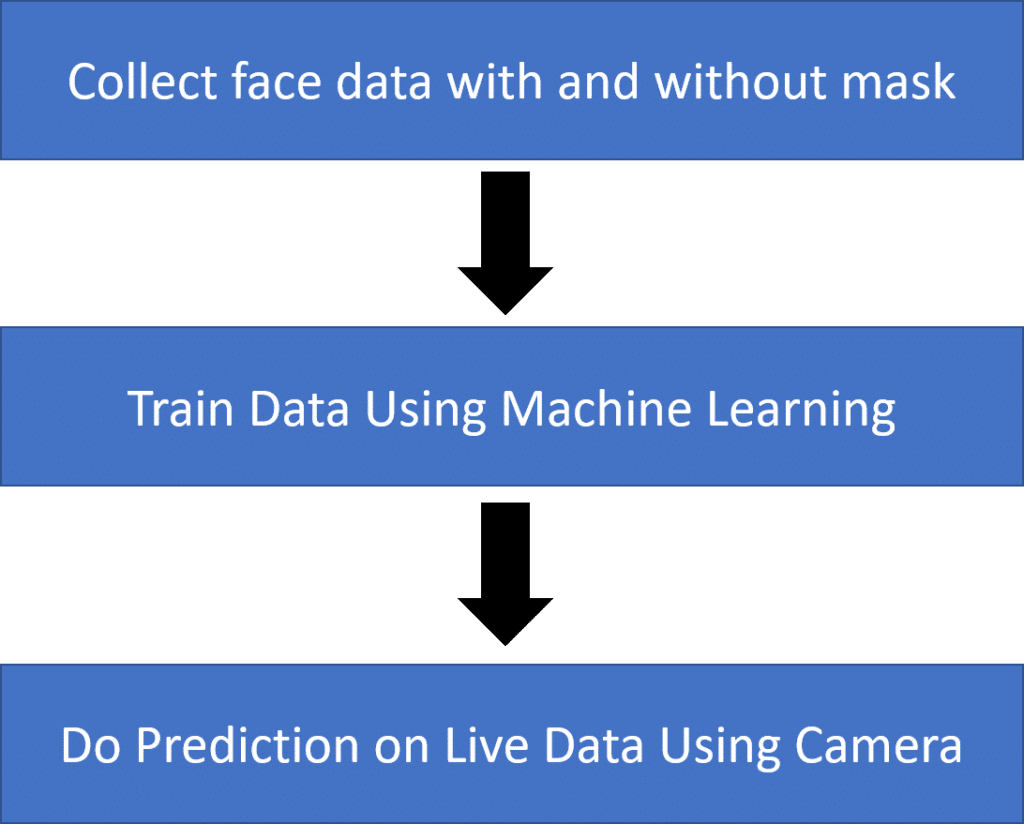


And the output will look like:

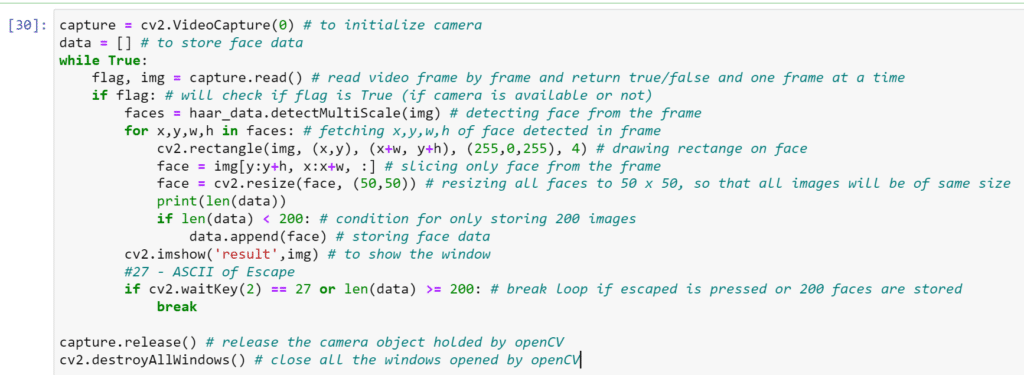


**Face Mask Detection**

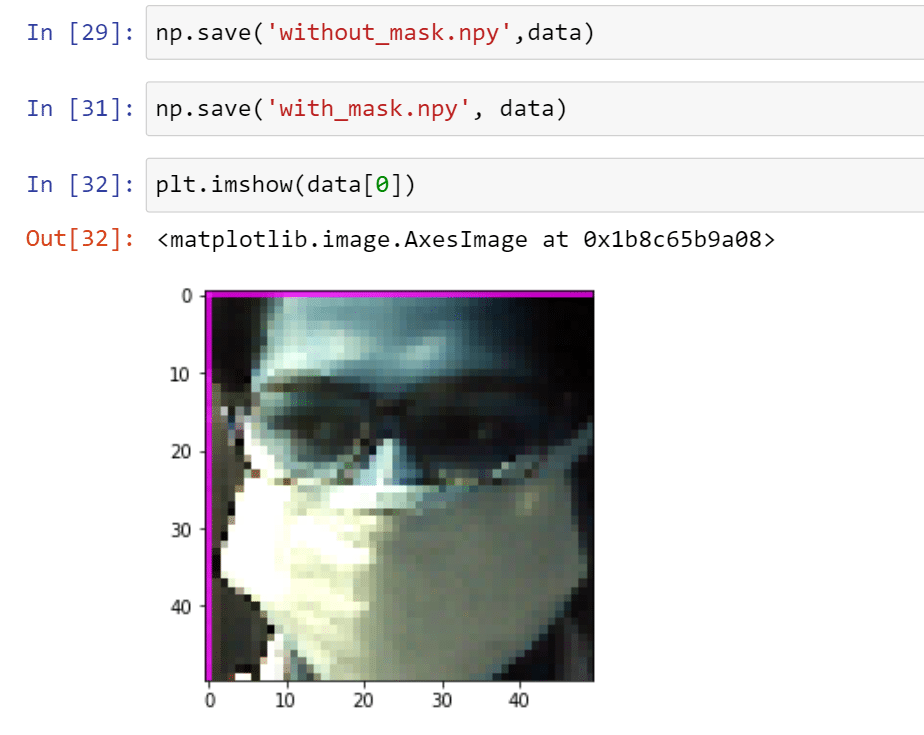
Now we are going to start with face mask detection. First let’s understand the process of it



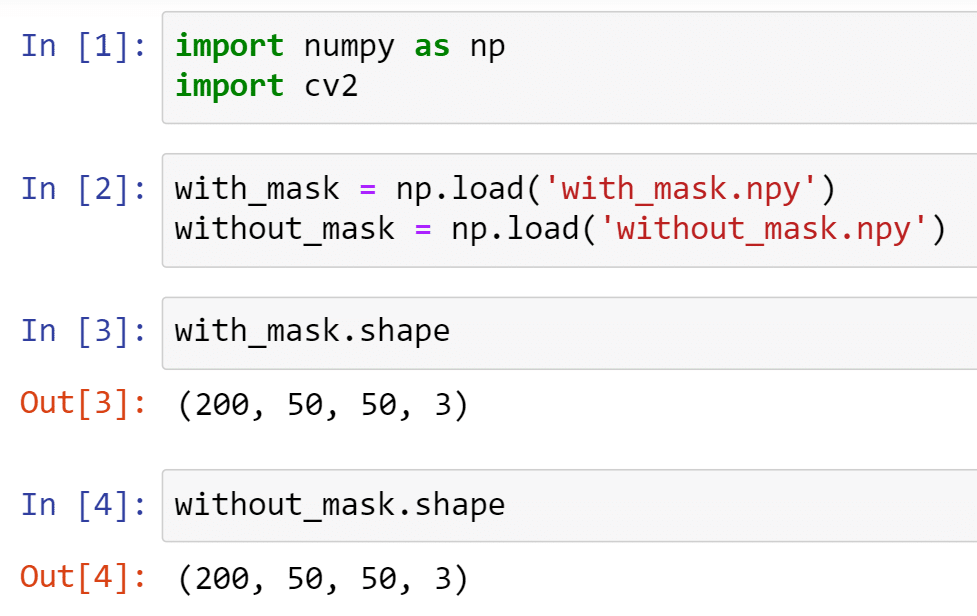
To implement this case study, we need a lot of images of people wearing a mask and not wearing a mask. So first we need to collect data and we are going to collect data using our own camera. Here is the complete code to perform face detection using camera and storing face data only:



Save the data in a numpy file and you can also plot the face data to check the data collected by OpenCV



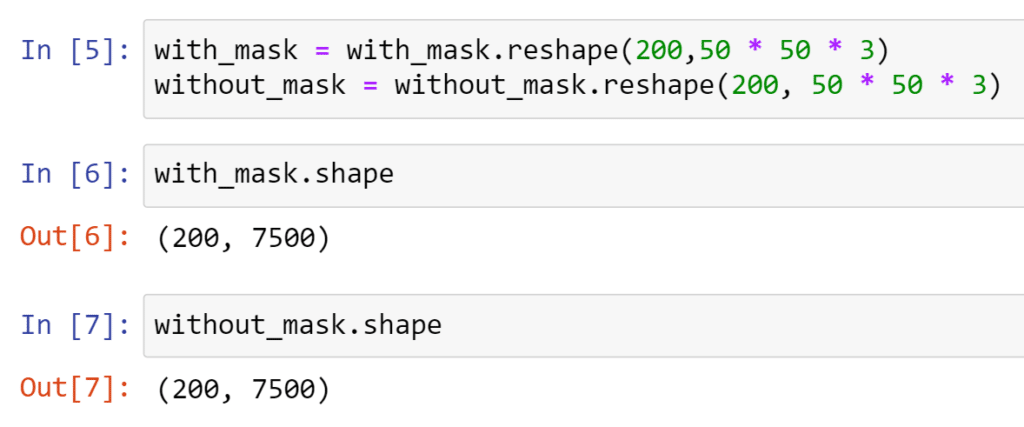
Now we can load the data anywhere and start processing it to apply machine learning on it:



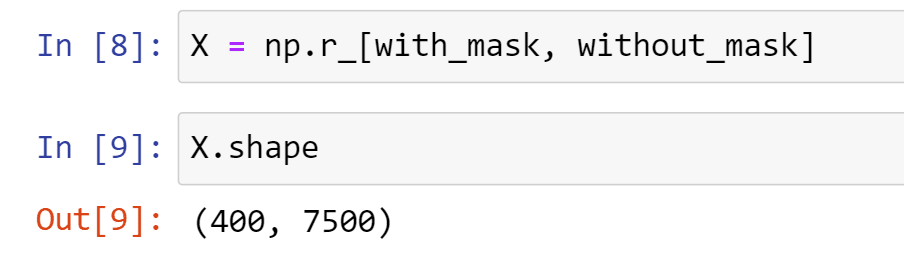
Now we can see that data is loaded with the shape 200, 50, 50, 3.

* Here 200 is the number of images we have collected
* 50, 50 is the size of each image
* 3 is the color channel (red, green, blue)

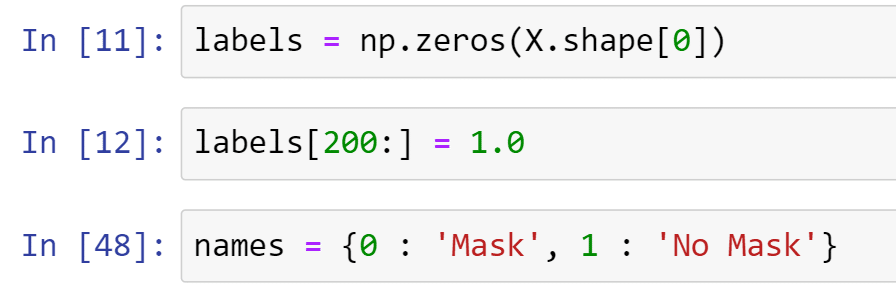
We can reshape the data to make it 2D:



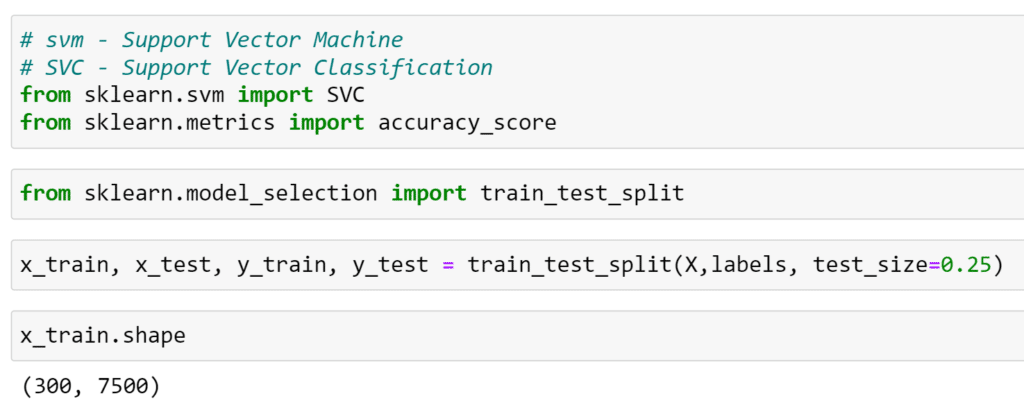
And we will concatenate the data into a single array:



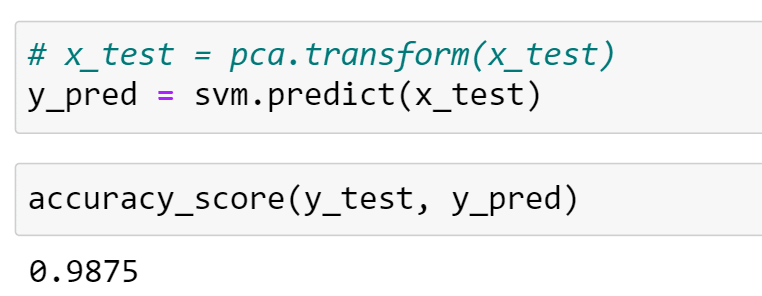
Using NPR will help you to store data row wise. So our features are ready. Now we need target variable. So let’s create one array of zeros and assign first 200 indexes as zero and next 200 indexes as one. Because first 200 images belong to faces with mask and next 200 images belong to faces without mask.



Now we can apply machine learning on our data after dividing it into train and test.



The algorithm we are using is SVM here. And after training this data on SVM we are getting accuracy of 98%.



So finally, we can test our faces with or without mask and check whether this algorithm is able to identity you that you are wearing a mask or not.



