Use Jupyter

```
import cv2
face_classifier = cv2.CascadeClassifier(
cv2.data.haarcascades + "haarcascade_frontalface_default.xml"
)
video_capture = cv2.VideoCapture(0)
def detect_bounding_box(vid):
gray_image = cv2.cvtColor(vid, cv2.COLOR_BGR2GRAY)
faces = face_classifier.detectMultiScale(gray_image, 1.1, 5, minSize=(40, 40))
for (x, y, w, h) in faces:
cv2.rectangle(vid, (x, y), (x + w, y + h), (0, 255, 0), 4)
return faces
while True:
result, video_frame = video_capture.read() # read frames from the video
if result is False:
break # terminate the loop if the frame is not read successfully
faces = detect_bounding_box(
video_frame
) # apply the function we created to the video frame
cv2.imshow(
"My Face Detection Project", video_frame
) # display the processed frame in a window named "My Face Detection Project"
if cv2.waitKey(1) & 0xFF == ord("q"):
break
video_capture.release()
cv2.destroyAllWindows()
```

Real-Time Face Detection-2

```
import cv2
# Load the cascade
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
# To capture video from webcam.
cap = cv2.VideoCapture(0)
# To use a video file as input
# cap = cv2.VideoCapture('filename.mp4')
while True:
# Read the frame
_, img = cap.read()
# Convert to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# Detect the faces
faces = face_cascade.detectMultiScale(gray, 1.1, 4)
# Draw the rectangle around each face
for (x, y, w, h) in faces:
cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
# Display
cv2.imshow('img', img)
# Stop if escape key is pressed
k = cv2.waitKey(30) & 0xff
if k==27:
break
# Release the VideoCapture object
cap.release()
```