EX:No. 6 **DATE: 8/4/25**	**Face Detection Using OPENCV**

AIM:

To build and train a model for face detection using opencv.

ALGORITHM:

 Import Libraries

- TensorFlow/Keras, OpenCV, NumPy, etc.

 Load & Preprocess Data

- Load face images and labels.
- Convert to grayscale or normalize color.
- Resize images to a fixed size.
- Encode labels (e.g., one-hot).
- Split into training/testing sets.

 Load YOLOv5 model (can change to yolov5s, yolov5m, yolov5l, yolov5x)

 Evaluate Model

- Test on unseen camera realtime data.
- Preprocess new image
- Use model to predict identity (highest softmax score)

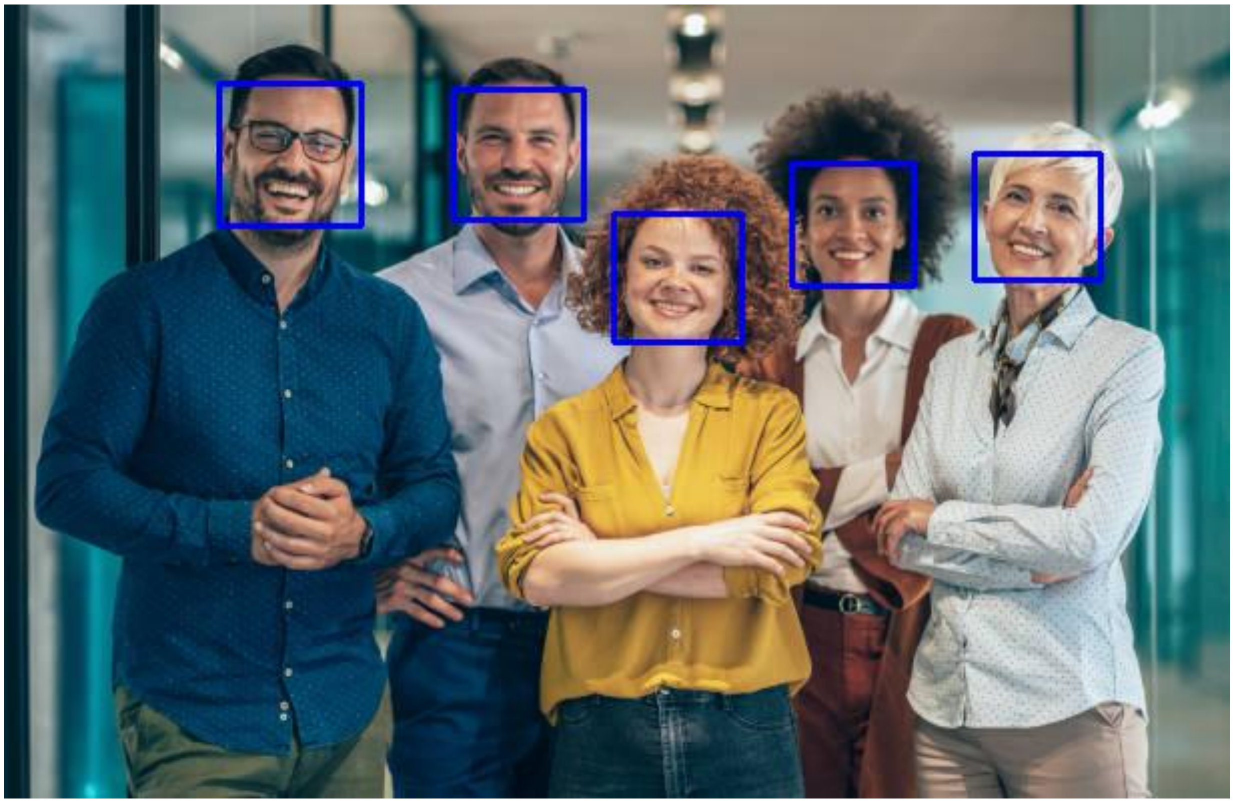
CODE:

```
!pip install opencv-python
import cv2
from google.colab.patches import cv2_imshow # Import the patched cv2_imshow
# Load the pre-trained face detection model
face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 'haarcascade_frontalface_default.xml')
# Read an image
image = cv2.imread('/content/istockphoto-1413873774-612x612.jpg')  # Replace with your image path
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)  # Convert to grayscale
# Detect faces
faces = face_cascade.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5, minSize=(30, 30))
```

```
# Draw rectangles around detected faces
for (x, y, w, h) in faces:
    cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)

# Display the image with detected faces
cv2_imshow(image) # Use cv2_imshow instead of cv2.imshow
cv2.waitKey(0)
cv2.destroyAllWindows()
```

OUTPUT:



RESULT:

Thus the program has been completed and verified successfully.