**Project Based Learning Report**

on

**“Age & Gender Detection Using Data Science with Open CV”**

Submitted in the partial fulfillment of the requirements.

For the Project based learning in (**ESSENTIALS OF DATA SCIENCE**)

in

Electronics & Communication Engineering

By

**PRN Name of the Student**

2214110420 Hamza Memon

2214110432 Sameer Abdul Sharief

2214110454 Devasheesh Thakre

Under the guidance of

Prof . Vikas Kaduskar



Department of Electronics & Communication Engineering

Bharati Vidyapeeth

(Deemed to be University)

College of Engineering,

Pune – 4110043

**Academic Year: 2023-24**

**CERTIFICATE**

Certified that the Project Based Learning report entitled, “**Age &** **Gender Detection Using Data Science with Open-CV**” is work done by

**PRN Name of the Student**

2114110420 Hamza Memon

2114110432 Sameer Abdul Sharief

2114110454 Devasheesh Thakre

in partial fulfillment of the requirements for the award of credits for Project Based Learning (PBL) in “**Essentials of Data Science”** Bachelor of Technology Semester IV, in ECE Div1.

**Date:**

**Prof. Vikas Kaduskar Dr. Arundhati Shinde**

**Course In-charge Professor & Head**

**INDEX**

|  |  |  |
| --- | --- | --- |
| Sr. No | Title | age No. |
| 1. | PROBLEM STATEMENT | 1-3 |
| 2. | SOFTWARE USED | 4-5 |
| 3. | INPUT CODE | 6-7 |
| 4. | CODE EXPLAINATION | 8-9 |
| 5. | RESULT | 10-20 |
| 6. | CONCLUSION | 21 |
| 7. | REFERENCES | 22 |

**PROBLEM STATEMENT :**

**Real-time Age-Gender Detection :-**

Age and gender detection is a computer vision task that involves automatically determining the age and gender of individuals depicted in images or videos. It has numerous applications in fields such as marketing, healthcare, security, and entertainment.

**HOW IT WORKS :**

**Face Detection:**

The first step is to detect faces within the input images or video frames. OpenCV provides methods for face detection using pre-trained deep learning models. These models are trained on large datasets to accurately identify faces in various conditions (different lighting, angles, etc.).One popular face detection model is the Single Shot Multibox Detector (SSD) framework, which is capable of real-time face detection. Upon detection, bounding boxes are drawn around the faces to indicate their location in the image.

**Face Extraction:**

Once faces are detected, they are extracted from the input image or frame. This step involves cropping the region of interest (ROI) defined by the bounding box around each detected face.The extracted face regions are then prepared for further analysis and prediction.

**Preprocessing:**

Before feeding the extracted face regions into the age and gender prediction models, preprocessing steps are applied to standardize the data and enhance model performance.

Common preprocessing steps include resizing the face images to a fixed size, normalization (mean subtraction and scaling), and possibly other techniques such as histogram equalization for enhancing image contrast.

**Age and Gender Prediction:**

The preprocessed face images are input to pre-trained deep learning models for age and gender prediction.For age prediction, a deep neural network (DNN) model is trained to classify faces into predefined age groups (e.g., 0-2, 4-6, 8-12, etc.). The model learns to recognize age-related facial features and patterns.

Similarly, for gender prediction, another DNN model is trained to classify faces into male or female categories based on gender-related facial features.

These pre-trained models are typically trained on large-scale datasets containing labeled face images with corresponding age and gender annotations.

**Post-processing and Visualization:**

After obtaining predictions from the age and gender models, post-processing steps may be applied to refine the results or improve interpretability.For example, confidence thresholds may be applied to filter out low-confidence predictions, or smoothing techniques may be used to stabilize predictions over time in video streams.

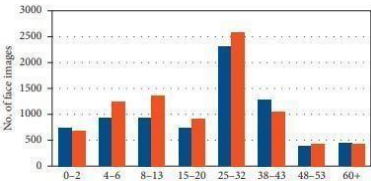
Finally, the predicted age and gender labels are overlaid onto the input images or video frames, often using text annotations or graphical elements such as bounding boxes.

**Real-Time Application:**

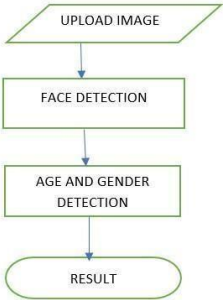
The entire process described above is optimized for real-time performance to enable age and gender detection in live video streams from sources such as webcams or surveillance cameras. Efficient implementation and optimization techniques are employed to ensure that the system can process frames at high speeds without significant latency.

**Evaluation and Fine-tuning:**

The performance of the age and gender detection system is evaluated using metrics such as accuracy, precision, recall, and F1-score.Fine-tuning of the models may be performed to improve performance on specific datasets or address domain-specific challenges.

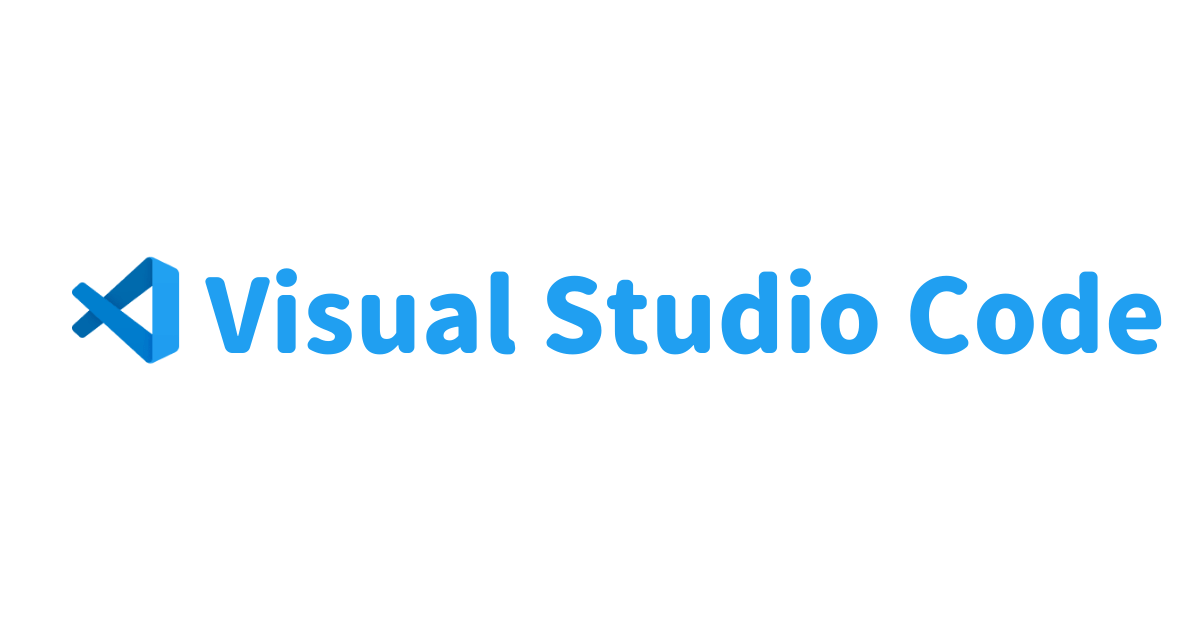


**DATASET OF AGE**



**FLOWCHART**

**ABOUT SOFTWARE :**



Visual Studio Code (VS Code) is a popular source-code editor developed by Microsoft. It is widely used by developers across different platforms and programming languages due to its versatility, ease of use, and extensive customization options. Here are some key features and aspects of Visual Studio Code:

**Cross-Platform:** VS Code is available for Windows, macOS, and Linux, making it accessible to a wide range of developers regardless of their operating system preference.

**Open Source:** Visual Studio Code is open-source software, meaning its source code is freely available for anyone to view, modify, and contribute to. This fosters community collaboration and allows users to tailor the editor to their specific needs.

**Lightweight and Fast:** Compared to other integrated development environments (IDEs), VS Code is lightweight and starts up quickly, making it suitable for various development tasks without consuming excessive system resources.

**Intuitive User Interface:** VS Code provides a clean and intuitive user interface that is easy to navigate. It includes features such as a sidebar for file navigation, a status bar for displaying information, and customizable themes for personalizing the look and feel.

**Rich Language Support:** Visual Studio Code offers robust language support for a wide range of programming languages, including but not limited to JavaScript, Python, Java, C++, and Go. It provides syntax highlighting, code completion, and debugging capabilities tailored to each language.

**Extensions Ecosystem**: One of the standout features of VS Code is its extensive extensions ecosystem. Users can install extensions from the Visual Studio Code Marketplace to add new features, integrate with third-party tools, or enhance language support. There are thousands of extensions available for tasks such as version control, linting, testing, and more.

**Integrated Terminal:** VS Code includes an integrated terminal, allowing developers to run command-line tools and interact with their projects without leaving the editor. This streamlines development workflows and reduces the need to switch between multiple applications.

**Git Integration:** Visual Studio Code seamlessly integrates with Git and other version control systems, providing features such as source control management, commit history visualization, and branch management within the editor itself.

**Customization:** VS Code offers extensive customization options, allowing users to tailor the editor to their preferences. This includes custom key bindings, settings, and themes, as well as the ability to create custom snippets and macros.

**INPUT CODE :**

import cv2

def faceBox(faceNet,frame):

frameHeight=frame.shape[0]

frameWidth=frame.shape[1]

blob= cv2.dnn.blobFromImage(frame, 1.0, (227,227), [104,117,123], swapRB=False )

faceNet.setInput(blob)

detection=faceNet.forward()

bboxs=[]

for i in range(detection.shape[2]):

confidence=detection[0,0,i,2]

if confidence>0.7:

x1=int(detection[0,0,i,3]\*frameWidth)

y1=int(detection[0,0,i,4]\*frameHeight)

x2=int(detection[0,0,i,5]\*frameWidth)

y2=int(detection[0,0,i,6]\*frameHeight)

bboxs.append([x1,y1,x2,y2])

cv2.rectangle(frame,(x1,y1),(x2,y2),(0,255,0), 1)

return frame, bboxs

faceProto = "opencv\_face\_detector.pbtxt"

faceModel = "opencv\_face\_detector\_uint8.pb"

ageProto = "age\_deploy.prototxt"

ageModel = "age\_net.caffemodel"

genderProto = "gender\_deploy.prototxt"

genderModel = "gender\_net.caffemodel"

faceNet=cv2.dnn.readNet(faceModel, faceProto)

ageNet=cv2.dnn.readNet(ageModel,ageProto)

genderNet=cv2.dnn.readNet(genderModel,genderProto)

MODEL\_MEAN\_VALUES = (78.4263377603, 87.7689143744, 114.895847746)

ageList = ['(0-2)', '(4-6)', '(8-12)', '(15-20)', '(25-32)', '(38-43)', '(48-53)', '(60-100)']

genderList = ['Male', 'Female']

video = cv2.VideoCapture(0)

while True:

ret, frame = video.read()

frame,bboxs=faceBox(faceNet,frame)

for bbox in bboxs:

face=frame[bbox[1]:bbox[3], bbox[0]:bbox[2]]

blob=cv2.dnn.blobFromImage(face, 1.0, (227,227), MODEL\_MEAN\_VALUES , swapRB=False)

genderNet.setInput(blob)

genderPred= genderNet.forward()

gender=genderList[genderPred[0].argmax()]

ageNet.setInput(blob)

agePred= ageNet.forward()

age=ageList[agePred[0].argmax()]

label="{},{}".format(gender,age)

cv2.putText(frame, label, (bbox[0], bbox[1]-10), cv2.FONT\_HERSHEY\_SIMPLEX, 0.8, (255,255,255), 2,cv2.LINE\_AA)

cv2.imshow("Age-Gender", frame)

k = cv2.waitKey(1)

if k == ord('q'):

break

video.release()

cv2.destroyAllWindows()

**CODE EXPLANATION :**

Certainly! The provided code is a Python script for real-time age and gender detection using OpenCV, a popular computer vision library. Let's break down the code and explain each part:

**Importing Libraries:**

The script starts by importing the necessary libraries. In this case, it imports the 'cv2' module, which is the OpenCV library.

**FaceBox Function:**

This function takes two arguments: 'faceNet', which is the pre-trained face detection model, and 'frame', which represents a frame from a video feed.

It first extracts the dimensions of the frame and then preprocesses the frame using ‘cv2.dnn.blobFromImage()’ to prepare it for input to the face detection model.

The preprocessed frame is then passed through the face detection model (‘faceNet’) to detect faces.

Bounding boxes are drawn around the detected faces based on the confidence scores of the detections.

The function returns the frame with bounding boxes drawn around faces and a list of bounding box coordinates.

**Loading Pre-trained Models:**

Paths to pre-trained models for face detection, age prediction, and gender prediction are provided.

The script reads these models using ‘cv2.dnn.readNet()’ function, which loads the models into memory.

**Constants and Lists:**

Constants such as’ MODEL\_MEAN\_VALUES’, representing mean values used for preprocessing, and lists like ‘ageList’ and’ genderList’ containing age and gender labels are defined.

**Video Capture:**

The script initializes video capture from the default webcam (‘VideoCapture(0)’).

Main Loop:

The script enters a loop to continuously read frames from the video feed.

For each frame, the ‘faceBox‘ function is called to detect faces and draw bounding boxes around them.

For each detected face, the face region is extracted and preprocessed.

The preprocessed face region is passed through gender and age prediction models (‘genderNet and ageNet’) to obtain predictions. Predicted gender and age labels are overlaid on the frame using cv2.putText().

The annotated frame is displayed in a window titled "Age-Gender".

The loop continues until the user presses the 'q' key.

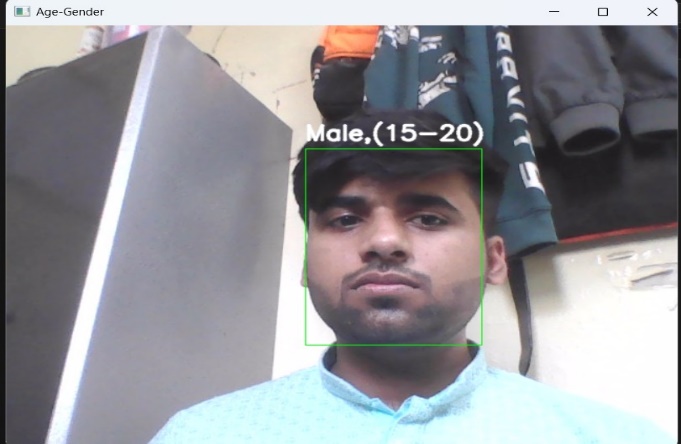
**Release Resources:**

After the loop exits (when 'q' is pressed), the video capture object is released (‘video.release()’) and OpenCV windows are closed (‘cv2.destroyAllWindows()’).

Overall, the script demonstrates a practical application of deep learning models for real-time age and gender detection using OpenCV, showcasing the integration of pre-trained models with computer vision techniques.

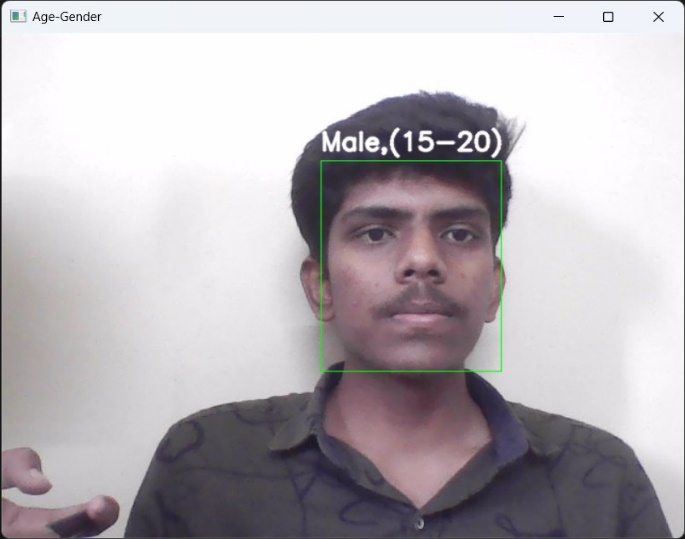
**RESULT :**

**SAMPLE 1**



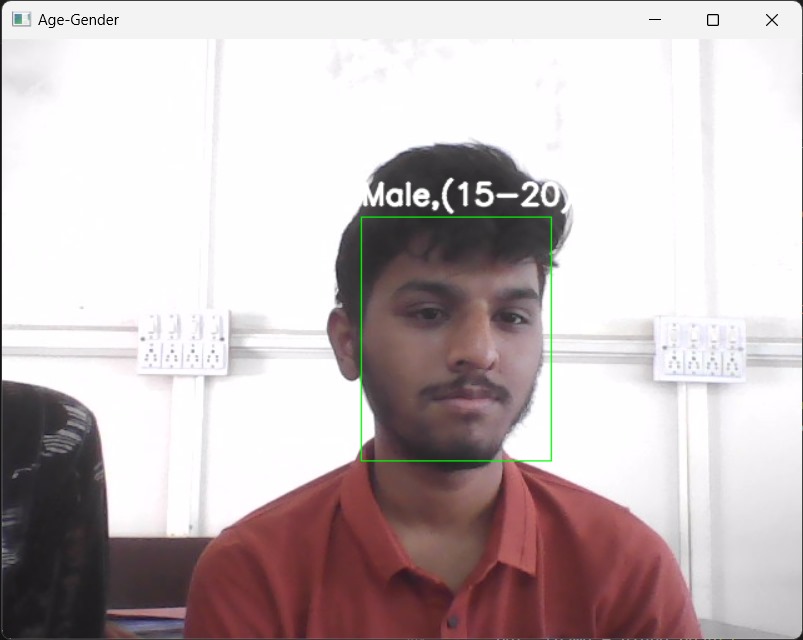
**Real Age = 20 , correct**

**SAMPLE 2**

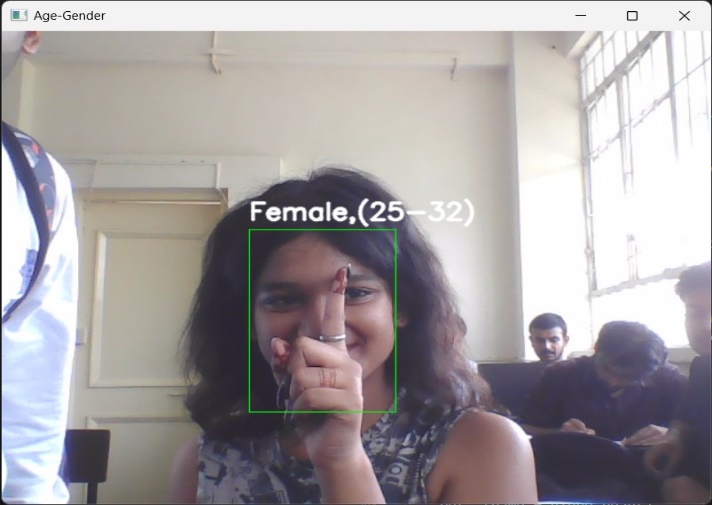


**Real Age = 19 , correct**

**SAMPLE 3**

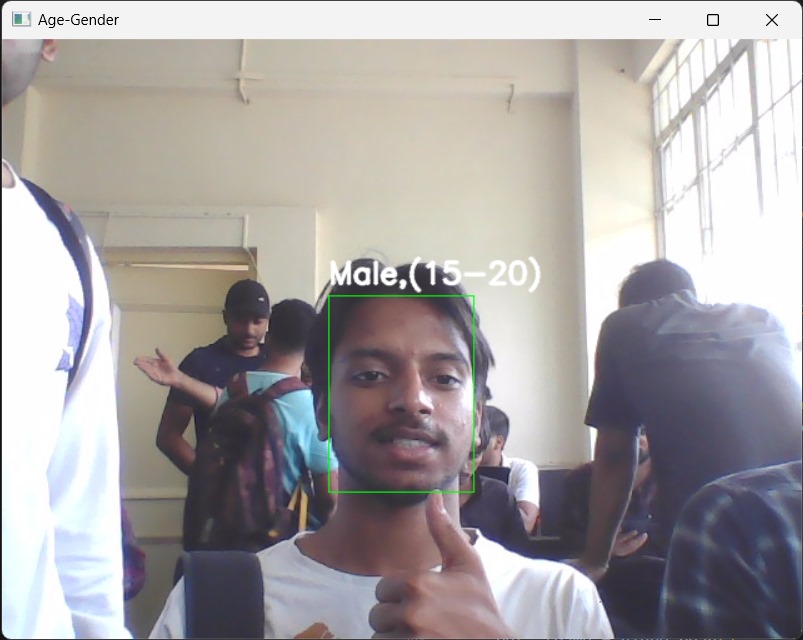
 **Real Age = 20 , correct**

**SAMPLE 4**

****

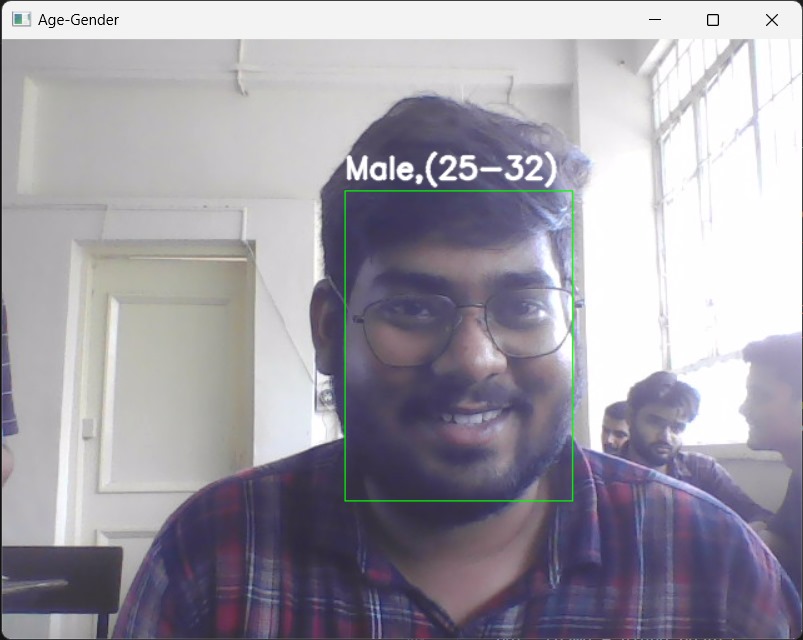
**Real Age = 20 , incorrect**

**SAMPLE 5**

****

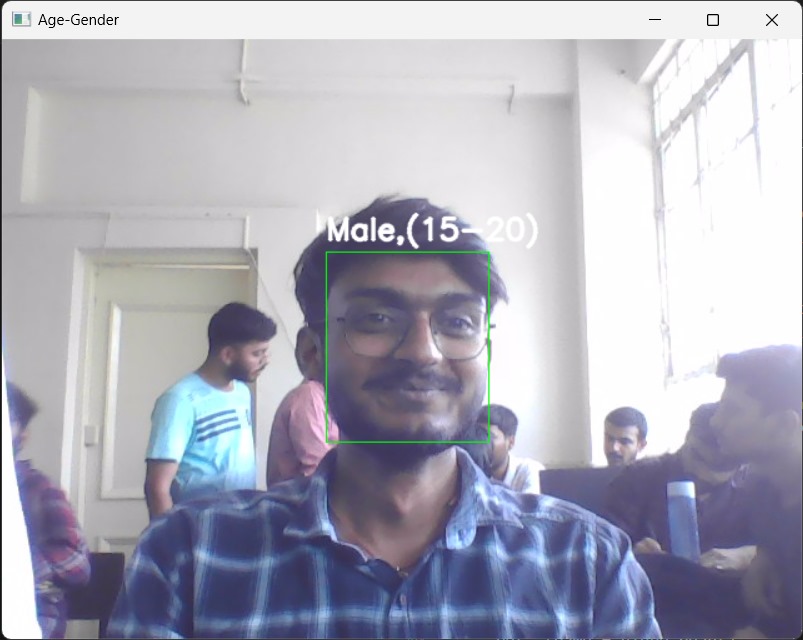
**Real Age = 20 , correct**

**SAMPLE 6**

****

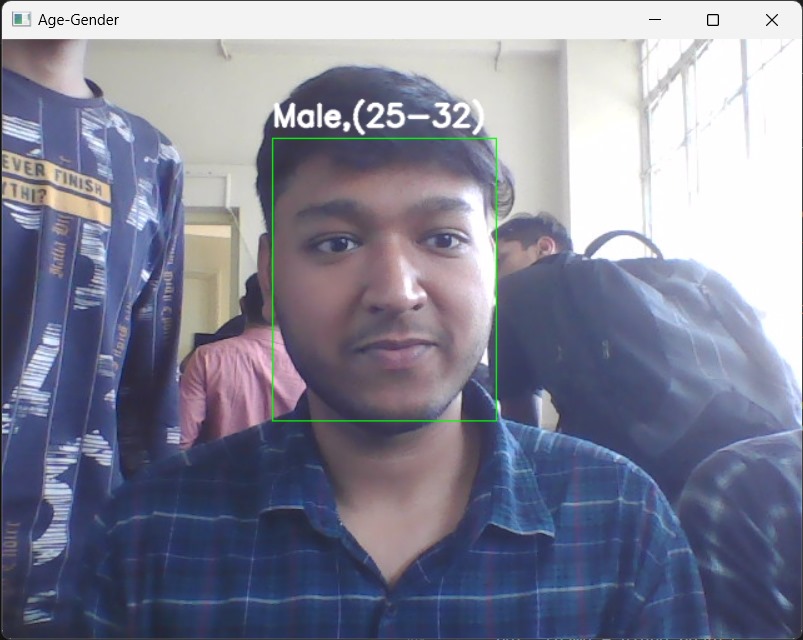
**Real Age = 21 , incorrect**

**SAMPLE 7**



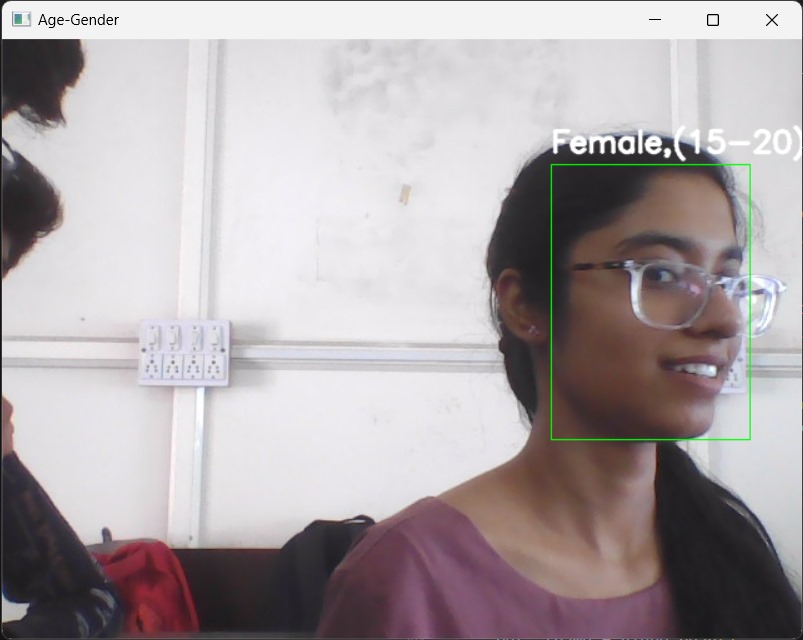
**Real Age = 20 , correct**

**SAMPLE 8**



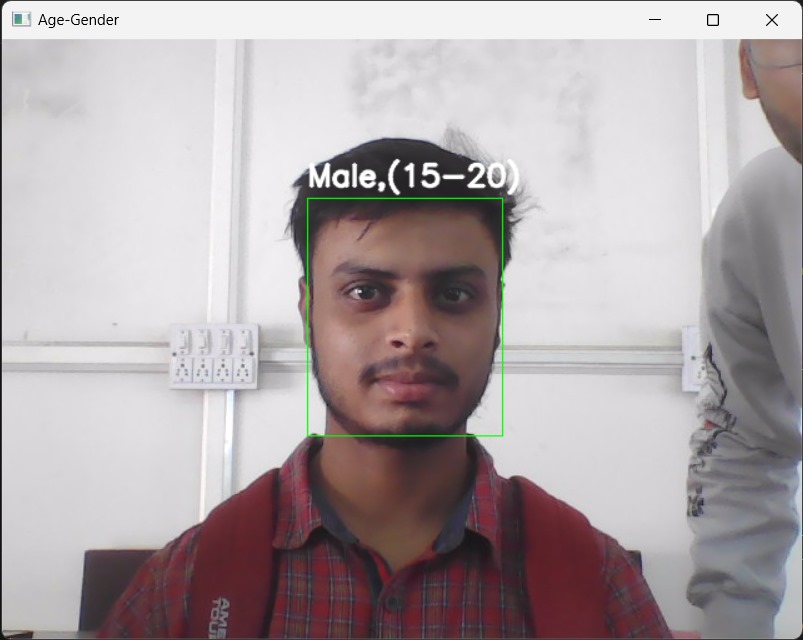
**Real Age = 21 , incorrect**

**SAMPLE 9**



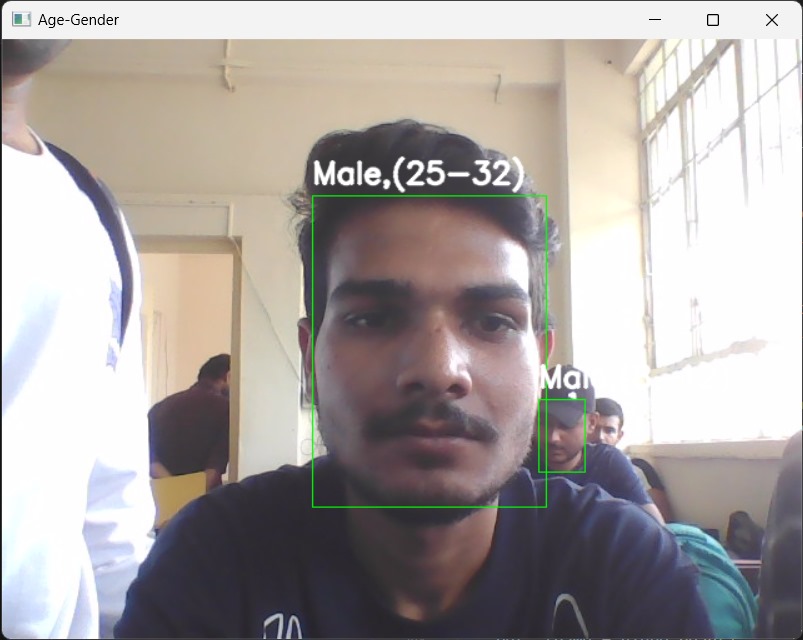
**Real Age = 20 , correct**

**SAMPLE 10**



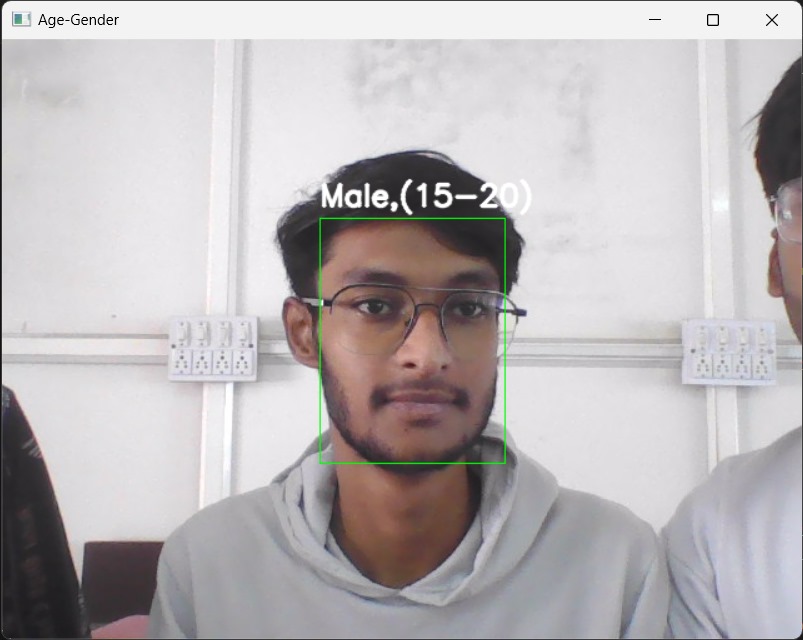
**Real Age = 20 , correct**

**SAMPLE 11**



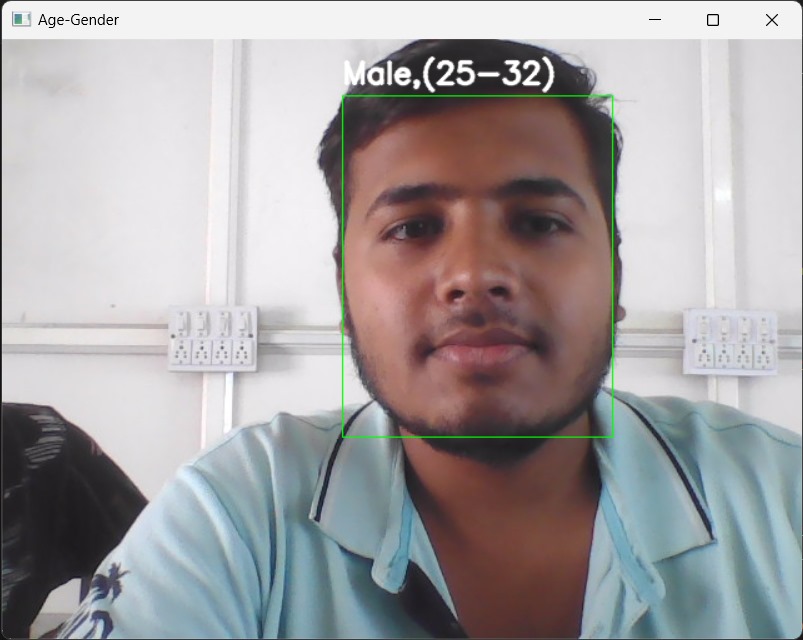
**Real Age = 21 , incorrect**

**SAMPLE 12**



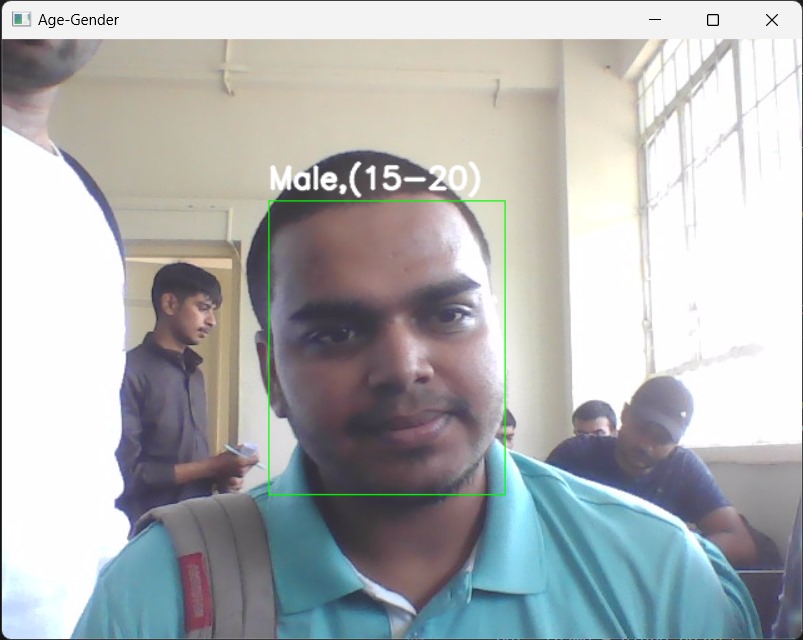
**Real Age = 20 , correct**

**SAMPLE 13**



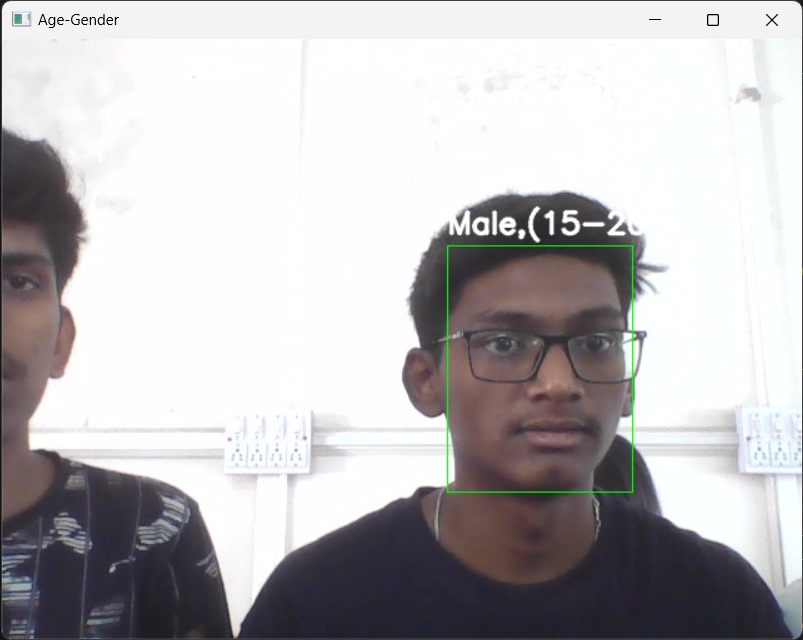
**Real Age = 21 , incorrect**

**SAMPLE 14**



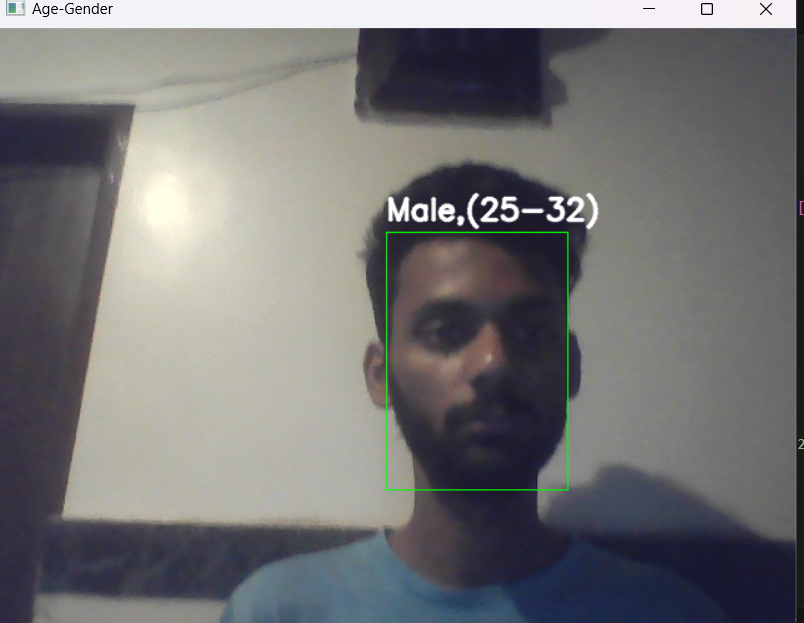
**Real Age = 20 , correct**

**SAMPLE 15**



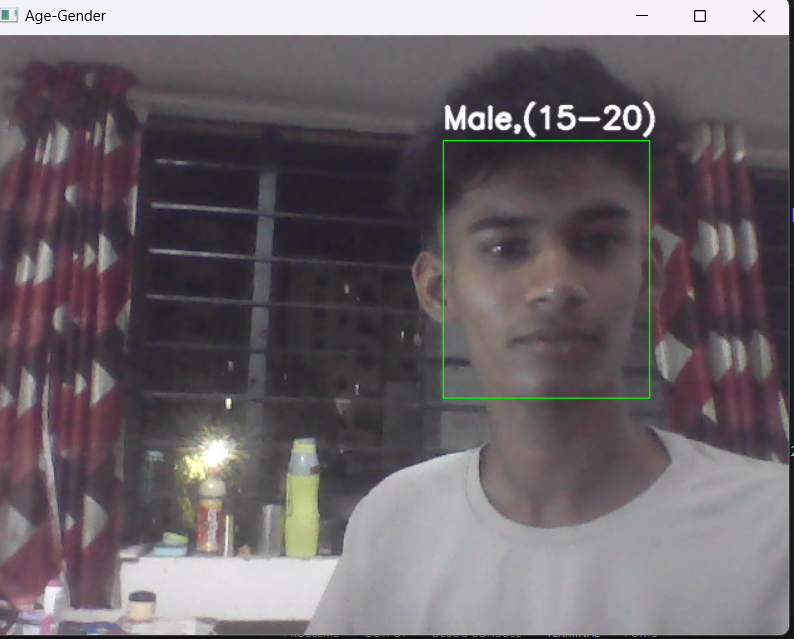
**Real Age = 19 , correct**

**SAMPLE 16**



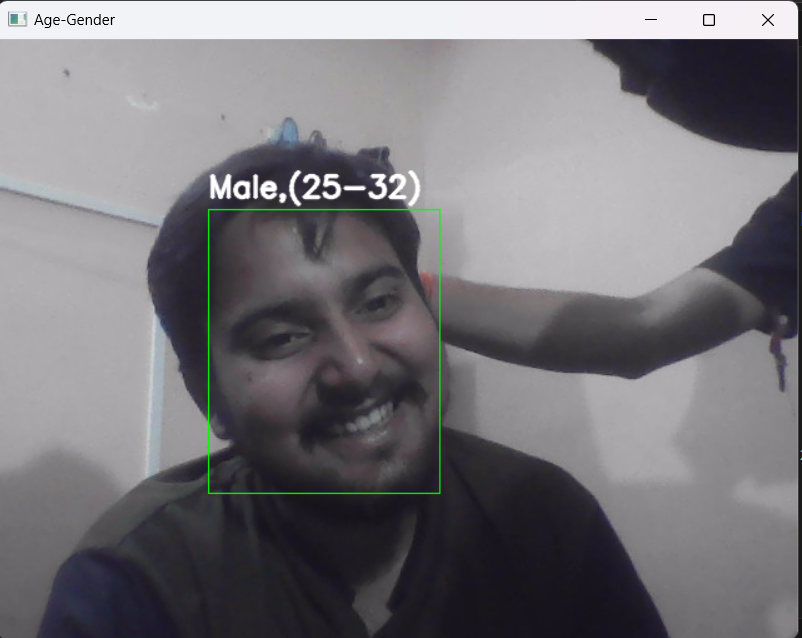
**Real Age = 21 , incorrect**

**SAMPLE 17**



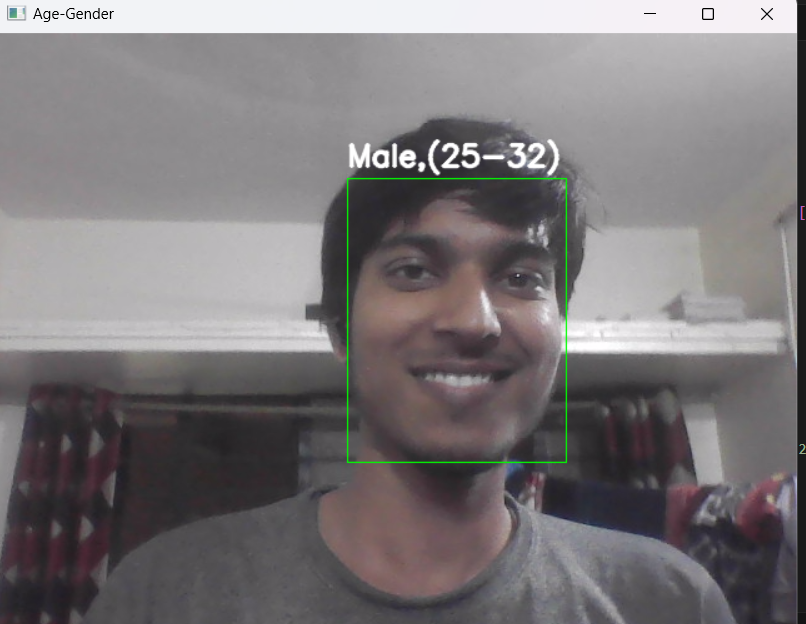
**Real Age = 19 , correct**

**SAMPLE 18**



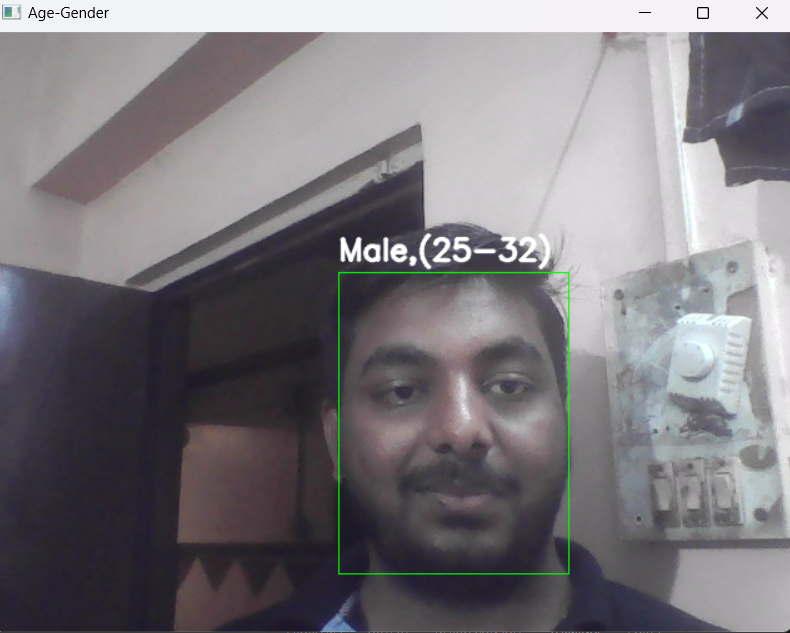
**Real Age = 22 , incorrect**

**SAMPLE 19**



**Real Age = 25 , correct**

**SAMPLE 20**



**Real Age = 23 , incorrect**

**Comparison Table :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample Number** | **Results** | **Actual Results** | **Error** |
|  | 15-20 | 20 | No |
|  | 15-20 | 19 | No |
|  | 15-20 | 20 | No |
|  | 25-32 | 20 | Yes |
|  | 15-20 | 20 | No |
|  | 25-32 | 21 | Yes |
|  | 15-20 | 20 | No |
|  | 25-32 | 21 | Yes |
|  | 15-20 | 20 | No |
|  | 15-20 | 20 | No |
|  | 25-32 | 21 | Yes |
|  | 15-20 | 20 | No |
|  | 25-32 | 21 | Yes |
|  | 15-20 | 20 | No |
|  | 15-20 | 19 | No |
|  | 25-32 | 21 | Yes |
|  | 15-20 | 19 | No |
|  | 25-32 | 22 | Yes |
|  | 25-32 | 25 | No |
|  | 25-32 | 23 | Yes |

**Number of correct predictons = 12**

**Number of incorrect predictions = 8**

**Accuracy = 60 %**

**Error = 40 %**

**CONCLUSION**

In conclusion, the provided Python script demonstrates the capability of OpenCV for real-time age and gender detection. By leveraging pre-trained deep learning models for face detection, age prediction, and gender prediction, the script accurately identifies faces in a live video feed and estimates their age and gender in real-time. This application has various practical uses in fields such as security, marketing, and user experience personalization. The script showcases the versatility and effectiveness of OpenCV in implementing advanced computer vision tasks with minimal coding effort.

**REFERENCES :**

* [www.youtube.com](http://www.youtube.com) (Channel Name – Knowledge Doctor)
* [www.geeksforgeeks.org](http://www.geeksforgeeks.org)
* www.learnopencv.com