**Age and Gender Detection - Age and Gender detection can be an important utilitygoing further to add weightage to VA system. Presently age and gender detection are done on European and American database. Significant opportunities are available foranalysis on Indian / Asian datasets. This could also be extended to generate datasetand publish for future researchers.**

Ans:

**Introduction:**

Age and gender, two of the key facial attributes, play a very foundational role in social interactions, making age and gender estimation from a single face image an important task in intelligent applications, such as access control, human-computer interaction, law enforcement, marketing intelligence and visual surveillance, etc.

**Real World Use Case:**

Recently I came across Quividi which is an AI software application which is used to detect age and gender of users who passes by based on online face analyses and automatically starts playing advertisements based on the targeted audience. Another example could be AgeBot which is an Android App that determines your age from your photos using facial recognition. It can guess your age and gender along with that can also find multiple faces in a picture and estimate the age for each face. Inspired by the above use cases we are going to build a simple Age and Gender detection model in this detailed article. So, let's start with our use-case: Use-case — I will be doing some face recognition, face detection stuff and furthermore, we will be using CNN (Convolutional Neural Networks) for age and gender predictions.

First introducing with the terminologies used in this advanced python project of gender and age detection:

**Computer Vision:**

**Computer Vision** is the field of study that enables computers to see and identify digital images and videos as a human would. The challenges it faces largely follow from the limited understanding of biological vision. Computer Vision involves acquiring, processing, analyzing, and understanding digital images to extract high-dimensional data from the real world in order to generate symbolic or numerical information which can then be used to make decisions. The process often includes practices like object recognition, video tracking, motion estimation, and image restoration.

**OpenCV:**

**OpenCV** is short for Open Source Computer Vision. Intuitively by the name, it is an open-source Computer Vision and Machine Learning library. This library is capable of processing real-time image and video while also boasting analytical capabilities. It supports the Deep Learning frameworks **Tensorflow**, Caffeand PyTorch.

**CNN:**

A CNN is a deep neural network (DNN) widely used for the purposes of image recognition and processing and **NLP**. Also known as a ConvNet, a CNN has input and output layers, and multiple hidden layers, many of which are convolutional. In a way, CNNs are regularized multilayer perceptrons.

**Objective:**

To build a gender and age detector that can approximately guess the gender and age of the person (face) in a picture using Deep Learningon the Adience dataset.

**About the Project:**

In this Project, I’ve used Deep Learning to accurately identify the gender and age of a person from a single image of a face. The predicted gender may be one of ‘Male’ and ‘Female’, and the predicted age may be one of the following ranges- (0 – 2), (4 – 6), (8 – 12), (15 – 20), (20-24), (25 – 32), (38 – 43), (48 – 53), (60 – 100) (8 nodes in the final softmax layer). It is very difficult to accurately guess an exact age from a single image because of factors like makeup, lighting, obstructions, and facial expressions. And so, we make this a classification problem instead of making it one of regression.

**The CNN Architecture:**

The convolutional neural network for this python project has 3 convolutional layers:

* Convolutional layer; 96 nodes, kernel size 7
* Convolutional layer; 256 nodes, kernel size 5
* Convolutional layer; 384 nodes, kernel size 3

It has 2 fully connected layers, each with 512 nodes, and a final output layer of softmax type.

To go about the python project, we’ll:

* Detect faces
* Classify into Male/Female
* Classify into one of the 9 age ranges
* Put the results on the image and display it.

**The Dataset:**

For this python project, I’ve used the Adience dataset; the dataset is available in the public domain and I’ll attach it while submitting. This dataset serves as a benchmark for face photos and is inclusive of various real-world imaging conditions like noise, lighting, pose, and appearance. The images have been collected from Flickr albums and distributed under the Creative Commons (CC) license. It has a total of 26,580 photos of 2,284 subjects in eight age ranges (as mentioned above) and is about 1GB in size. The models we will use have been trained on this dataset.

**Pre-requisites:**

I’ve installed OpenCV (cv2) to run this project and can just pip install opencv-python on the prompt. Other two packages math and argparse comes in the standard python library

In the Dataset the contents are:

* opencv\_face\_detector.pbtxt
* opencv\_face\_detector\_uint8.pb
* age\_deploy.prototxt
* age\_net.caffemodel
* gender\_deploy.prototxt
* gender\_net.caffemodel
* a few pictures to try the project on

For face detection, we have a .pb file- this is a protobuf file (protocol buffer); it holds the graph definition and the trained weights of the model. We can use this to run the trained model. And while a .pb file holds the protobuf in binary format, one with the .pbtxt extension holds it in text format. These are TensorFlow files. For age and gender, the .prototxt files describe the network configuration and the .caffemodel file defines the internal states of the parameters of the layers.

2. We use the argparse library to create an argument parser so we can get the image argument from the command prompt. We make it parse the argument holding the path to the image to classify gender and age for.

3. For face, age, and gender, initialize protocol buffer and model.

4. Initialize the mean values for the model and the lists of age ranges and genders to classify from.

5. Now, use the readNet() method to load the networks. The first parameter holds trained weights and the second carries network configuration.

6. Let’s capture video stream in case you’d like to classify on a webcam’s stream. Set padding to 20.

7. Now until any key is pressed, we read the stream and store the content into the names hasFrame and frame. If it isn’t a video, it must wait, and so we call up waitKey() from cv2, then break.

8. Let’s make a call to the highlightFace() function with the faceNet and frame parameters, and what this returns, we will store in the names resultImg and faceBoxes. And if we got 0 faceBoxes, it means there was no face to detect.  
Here, net is faceNet- this model is the DNN Face Detector and holds only about 2.7MB on disk.

* Create a shallow copy of frame and get its height and width.
* Create a blob from the shallow copy.
* Set the input and make a forward pass to the network.
* faceBoxes is an empty list now. for each value in 0 to 127, define the confidence (between 0 and 1). Wherever we find the confidence greater than the confidence threshold, which is 0.7, we get the x1, y1, x2, and y2 coordinates and append a list of those to faceBoxes.
* Then, we put up rectangles on the image for each such list of coordinates and return two things: the shallow copy and the list of faceBoxes.

9. But if there are indeed faceBoxes, for each of those, we define the face, create a 4-dimensional blob from the image. In doing this, we scale it, resize it, and pass in the mean values.

10. We feed the input and give the network a forward pass to get the confidence of the two class. Whichever is higher, that is the gender of the person in the picture.

11. Then, we do the same thing for age.

12. We’ll add the gender and age texts to the resulting image and display it with imshow().

**Conclusion:**

In this python project, I’ve implemented a CNN to detect gender and age from a single picture of a face.