AGE PREDICTION USING BIOMETRIC FEATURES

A PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMETNS

FOR THE AWARD OF THE DEGREE OF

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

SUBMITTED BY :

APOORV SHARMA (2K15/IT/015)

HITESH KUMAR (2K15/IT/032)

UNDER THE SUPERVISION OF

DR. DINESH K. VISHWAKARMA

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

AUGUST – DECEMBER 2018

DEPARTMENT OF INFORMATION TECHNOLOGY, DELHI

TECHNOLOGICAL UNIVERSITY (Formerly Delhi College of Engineering)

BAWANA ROAD, NEW DELHI -110043

CANDIDATE'S DECLARATION

We, Apoorv Sharma (2K15/IT/015) and Hitesh Kumar (2K15/IT/032), students of B.Tech (Information Technology) hereby declare that the project dissertation titled- 'Age Predictor using Biometric Features' which is submitted by us to the Department of Information Technology, Delhi Technological University, Delhi in partial fulfilment of the requirement for the award of he degree of Bachelor of Technology, is original and not copied from any source with proper citation. This work has not previously formed the basis for the award of any degree, diploma associateship, fellowship or other similar title or recognition.

Place: Delhi

Date: 07/12/2018

Apoorv Sharma

2K15/IT/015

Hitesh Kumar

2K15/IT/032

DEPARTMENT OF INFORMATION TECHNOLOGY, DELHI

TECHNOLOGICAL UNIVERSITY (Formerly Delhi College of Engineering)

BAWANA ROAD, NEW DELHI -110043

CERTIFICATE

I hereby certify that the Project Dissertation titled '**Age Predicator using Biometric features**' which was submitted by Apoorv Sharma (2K15/IT/015) and Hitesh Kumar (2K15/IT/032) is partial fulfilment of the requirement for the award of the degree of Bachelor of Technology, is a record of the project carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full Degree or diploma to this university or elsewhere.

Place: Delhi

Date: 07/12/2018

DR. DINESH K VISHWAKARMA

SUPERVISOR

Associate Professor

Department of Information Technology

Delhi Technological University

ABSTRACT

Usage of Biometrics to get the age is one of the tasks which can have a wide number of applications. It ranges from getting to know about the person's age for providing customized information and content to a user on a web application to selected access of user to some special content.

Our objective was to make an application that can be used to predict age of a person using his/her biometric features which includes facial features. This may include image processing to get the information to finally predicting the age of the person.

Hence, we came up with an application **Age Prediction using Biometric Features,** it helps to find the age of a person using facial features. It uses tools and techniques such as Keras, TensorFlow, flask, JavaScript etc.

ACKNOWLEDGEMENT

The successful culmination of any task is incomplete without giving the acknowledgement to the people who made it possible and without whose constant guidance and encouragement, the project would not have been successful.

We owe a lot to our supervisor, Dr. Dinesh K Vishwakarma, Assistant Professor, Department Of Information Technology, for igniting in us the idea of a creatively performed Major Project, for helping us in undertaking this endeavor and also for being there whenever we needed her assistance.

We would also like to take this moment to show our sense of gratitude to one and all, who indirectly or directly have lent us their hand in this task. We feel happy and content, in expressing our deep sense of gratitude to all those who have helped us in presenting this project.

Last, but never the least, we thank our parents for always being with us, in every sense.

CONTENT

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Age Predictor using Biometric Features, as the name suggests, is a platform that is used to predict the age of a person using facial features. It makes a feature matrix of face and then applies machine learning algorithms and techniques to predict age. It takes help of convolutional neural network which comes inside deep learning.

The main reason behind developing such an application is that it can be used in various real life aspects to make matters more efficient and quick. For instance , this application could be used in election polling stations and thus it will only allow eligible candidates to cast their votes. This will help in cutting down the labour cost.

Another example being restricting access to some specified websites . At present , the way to confirm age of a person trying to access a particular website is very inefficient and weak. Any non-eligible person can intrude into the website. But what if this inefficient method of verifying, which generally include the usage of some email id for confirmation or just a dialog box which ask 'Are you more than 18 years old?', is replaced through live age predictor and thus only allowing persons that meet some threshold.

Before diving into the concepts of how we made the project, there are some basics needs to be mentioned. These are :

OpenCV (CV2) :

It is a library which contains the functions for real-time computer vision. It is an Open Source Computer Vision is a library. The version used in the project is cv2 which returns everything as NumPy objects, a python library, like ndarray and native python objects like lists, tuples, dictionary etc.

Python's OS Module :

Python's OS module is used for using Operating System dependent functionality. These functions allows us to interface with the underlying operating system that the python is running on – be that Windows, Linux, Mac OS, etc.

NumPy:

It is a Python Library which adds support for large, multi-dimensional matrices and arrays, along with a large collection of high-level mathematical functions to operate on these arrays.

Keras :

Keras is an Open Source Neural Network library written in python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, or Theano. It is designed to enable experimentation with Artificial Neural Networks and Deep Neural Network.

TensorFlow :

It is also an Open Source software library for dataflow programming across a range of tasks. It is a symbolic math library , and is also used for machine learning applications such as neural networks.

Python GUI – tkinter :

It is a standard python interface to the Tk GUI toolkit shipped with python. It is one of the most popular way for creating high performance and nice looking GUI applications.

Flask :

Flask is a web application written in python. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine.

Chapter 2

PROPOSAL: CNN FOR AGE PREDICTION

In this project, we have proposed a CNN i.e. Convolutional Neural Network to predict the age of an image of person consisting of his/her clear facial characteristics. Therefore, image processing is necessary initial preprocessing part of this project which is later succeeded by creation of model and its training and then finally prediction of age.

The technologies used are-

- Language used – Python, JavaScript (for website frontend)
- Language version for python – 3.5.2

2.1 Image Processing :

Image can be taken as input either through the front camera of the laptop or as input file.
The first step is to find the face in the picture. For this purpose, Haar Cascade Classifier is used. It detects the facial region using superimposition of pre-trained image data. After facial region is detected and the rest of the region is left as it is of no use.

The facial region is then gray-scaled. Gray-scaling is one in which the value of each pixel is a single sample representing only an amount of light , that is , it carries only intensity information. Image is usually grey-scaled since color increases complexity, thus , greyscaling helps in resource efficiency.

The next step is to normalize the grey-scaled image . Normalization is a process that changes the ratio of pixel intensity values. It is sometimes even called as contrast stretching or histogram stretching. The benefit of normalization is that it helps in cases of poor contrast and brings the image into a range that is more familiar to the senses. In addition, it only helps to remove noise.

Image is also resized to predefined dimensions using the opencv function called resize().

Now this processed image is ready to be given as input to the model.

2.2 Model Preparation :

In this step , the main work is being done , that is , to make a model that predicts the age using the processed image from the previous step.

Age Predictor's model is created using Keras with TensorFlow as the backend. The model that we have made is a multi-layer Convolutional Neural Network. Few model architectures like 'AlexNet' and 'GoogLeNet' were also tried but didn't show results as good as our Neural Network
We have used a **sequential model** that is a linear stack of layers. Each layer is a 2D convolutional layer.
This layer creates a convolutional kernel that is convolved with the layer input to produce a tensor of outputs.

The **activation function** that is used in each layer is '**relu'.**

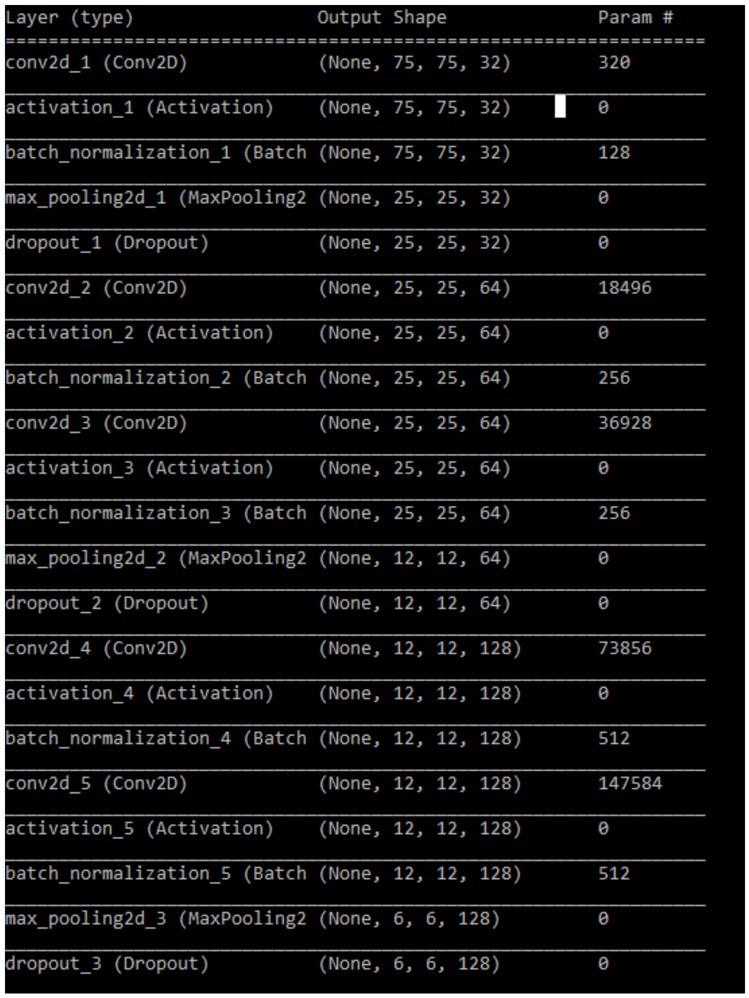
Batch normalization layer is also added. It normalizes the activations of previous layer at each batch , that is , it applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

MaxPooling2D is another feature that is added to each layer. Max pooling operation is used for spatial data.

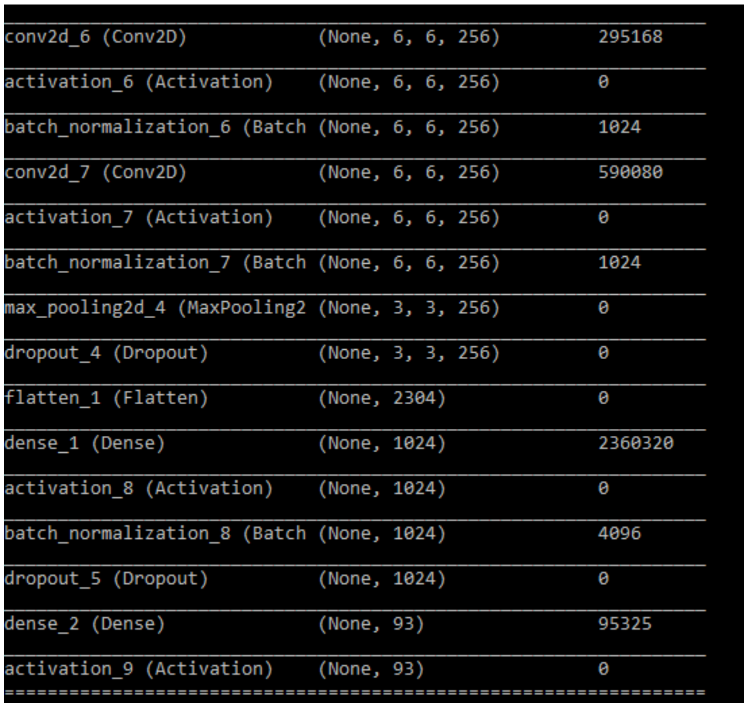
Dropout feature is applied to the input. Dropout consists in randomly setting a fraction rate of input units to 0 at each update during training time, which helps prevent overfitting.

Adam is used as the optimizer.

In the last layer , **Dense** feature is added , it is a regular densely-connected NN layer.

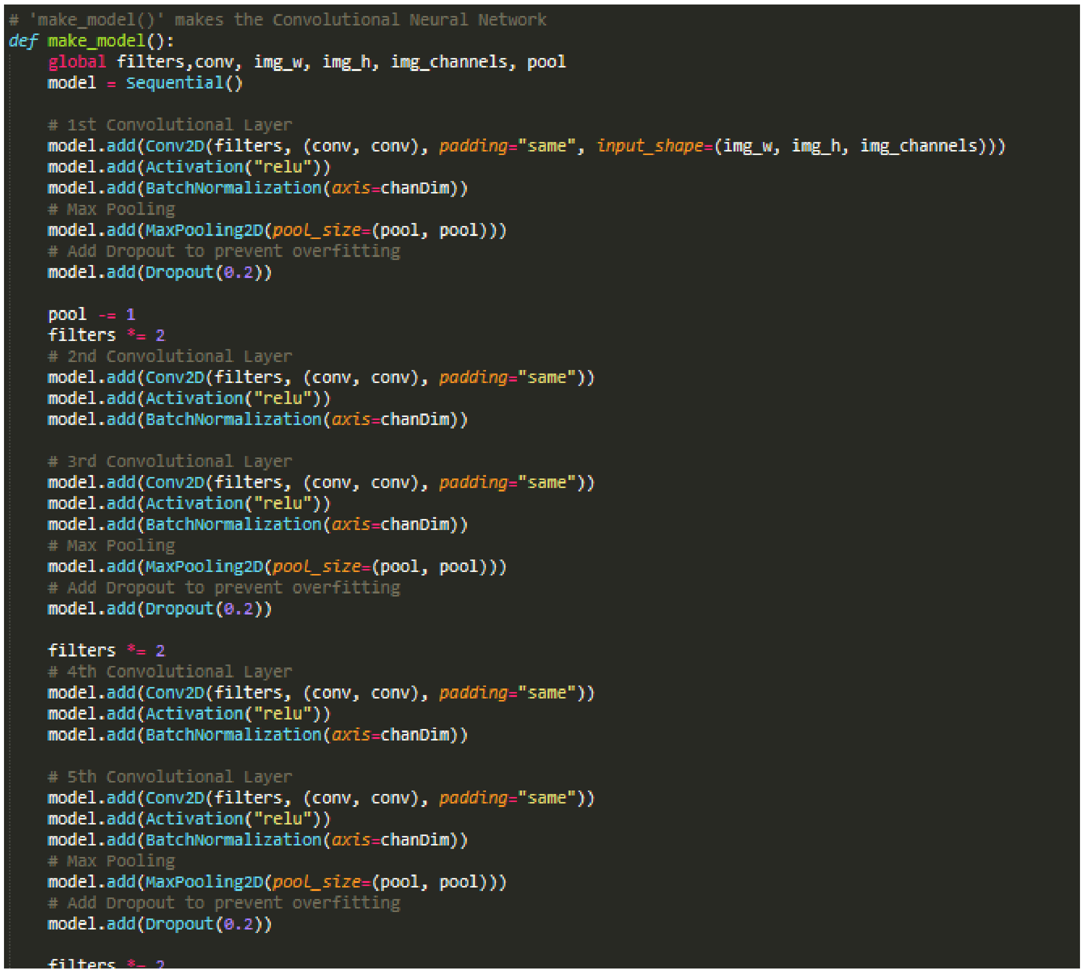


```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 75, 75, 32)        320
_________________________________________________________________
activation_1 (Activation)    (None, 75, 75, 32)        0
_________________________________________________________________
batch_normalization_1 (Batch (None, 75, 75, 32)        128
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 25, 25, 32)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 25, 25, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 25, 25, 64)        18496
_________________________________________________________________
activation_2 (Activation)    (None, 25, 25, 64)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 25, 25, 64)        256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 25, 25, 64)        36928
_________________________________________________________________
activation_3 (Activation)    (None, 25, 25, 64)        0
_________________________________________________________________
batch_normalization_3 (Batch (None, 25, 25, 64)        256
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 12, 12, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 12, 12, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
activation_4 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_4 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 12, 12, 128)       147584
_________________________________________________________________
activation_5 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_5 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 6, 6, 128)         0
```

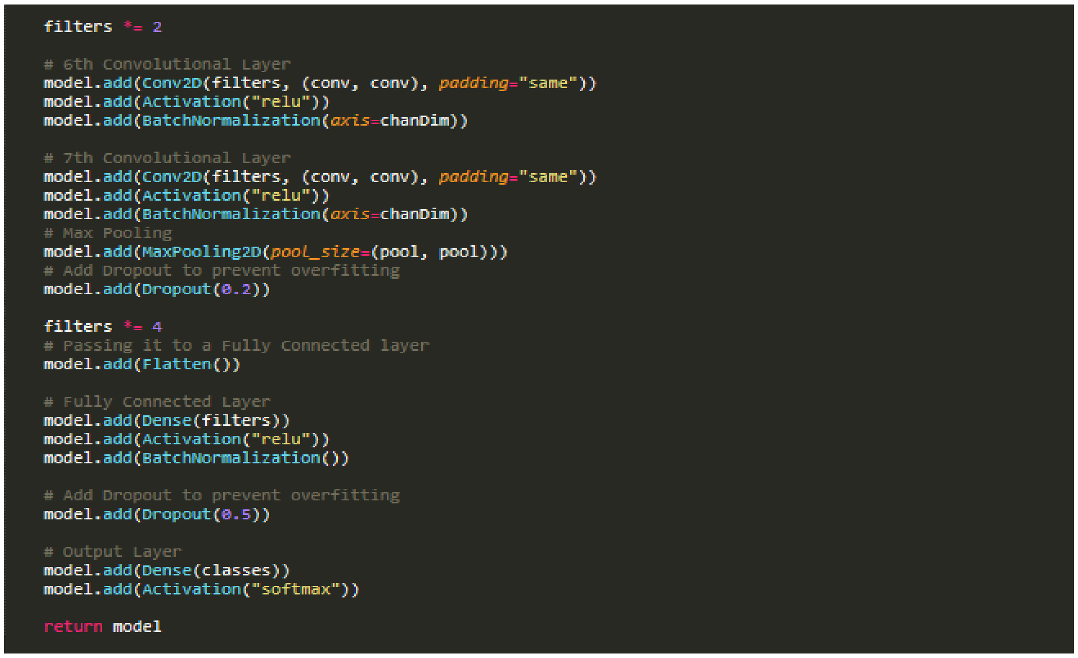


```
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 6, 6, 256)         295168
_________________________________________________________________
activation_6 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_6 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 6, 6, 256)         590080
_________________________________________________________________
activation_7 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_7 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 3, 3, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 2304)              0
_________________________________________________________________
dense_1 (Dense)              (None, 1024)              2360320
_________________________________________________________________
activation_8 (Activation)    (None, 1024)              0
_________________________________________________________________
batch_normalization_8 (Batch (None, 1024)              4096
_________________________________________________________________
dropout_5 (Dropout)          (None, 1024)              0
_________________________________________________________________
dense_2 (Dense)              (None, 93)                95325
_________________________________________________________________
activation_9 (Activation)    (None, 93)                0
=================================================================
```

Fig 1: Convolutional Neural Network Architecture



```python
# 'make_model()' makes the Convolutional Neural Network
def make_model():
    global filters,conv, img_w, img_h, img_channels, pool
    model = Sequential()

    # 1st Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same", input_shape=(img_w, img_h, img_channels)))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))
    # Max Pooling
    model.add(MaxPooling2D(pool_size=(pool, pool)))
    # Add Dropout to prevent overfitting
    model.add(Dropout(0.2))

    pool -= 1
    filters *= 2
    # 2nd Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))

    # 3rd Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))
    # Max Pooling
    model.add(MaxPooling2D(pool_size=(pool, pool)))
    # Add Dropout to prevent overfitting
    model.add(Dropout(0.2))

    filters *= 2
    # 4th Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))

    # 5th Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))
    # Max Pooling
    model.add(MaxPooling2D(pool_size=(pool, pool)))
    # Add Dropout to prevent overfitting
    model.add(Dropout(0.2))

    filters *= 2
```



```python
    filters *= 2

    # 6th Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))

    # 7th Convolutional Layer
    model.add(Conv2D(filters, (conv, conv), padding="same"))
    model.add(Activation("relu"))
    model.add(BatchNormalization(axis=chanDim))
    # Max Pooling
    model.add(MaxPooling2D(pool_size=(pool, pool)))
    # Add Dropout to prevent overfitting
    model.add(Dropout(0.2))

    filters *= 4
    # Passing it to a Fully Connected layer
    model.add(Flatten())

    # Fully Connected Layer
    model.add(Dense(filters))
    model.add(Activation("relu"))
    model.add(BatchNormalization())

    # Add Dropout to prevent overfitting
    model.add(Dropout(0.5))

    # Output Layer
    model.add(Dense(classes))
    model.add(Activation("softmax"))

    return model
```

Fig 2: Preparing of Keras Model

2.3 TRAINING THE MODEL AND PREDICTING THE AGE

Once the model is prepared , the next step is to train the model. The dataset of images that we have used to train the model is **UTKFace .** It is a large-scale dataset with long age span (range from 1 to 116 years old) although we have used images of person having age less than or equal to 90 years old. The dataset contsists of over 20,000 face images with annotations of age, gender. The images cover large variation in pose, facial expression, illumination, occlusion, resolution etc. This dataset could be used on a variety of tasks, e.g., face detection, age estimation, age progression/regression etc.

The images are then pre-processed, that is, passed through various steps such as grey-scaling, resizing, expanding dimensions.

80% of the images are used as the training dataset and the remaining 20% images are used for testing the model.

Best weights are also saved when the model is being trained to be used at the time of prediction.

The actual age and the predicted age is compared and the absolute error is mentioned alongside the output.

At the end, Mean Absolute Error (mea) is displayed for all the images that have been predicted.

2.4 BUILDING OF GUI :

Now we are ready with our model that predicts the age of person with utmost accuracy , the next step is to build a graphical user interface so as to improve the user experience.

To build a GUI we used tkinter. Tkinter is a standard python interface to the Tk GUI toolkit shipped with python. Python with tkinter outputs the fastest and easiest way to create GUI applications.

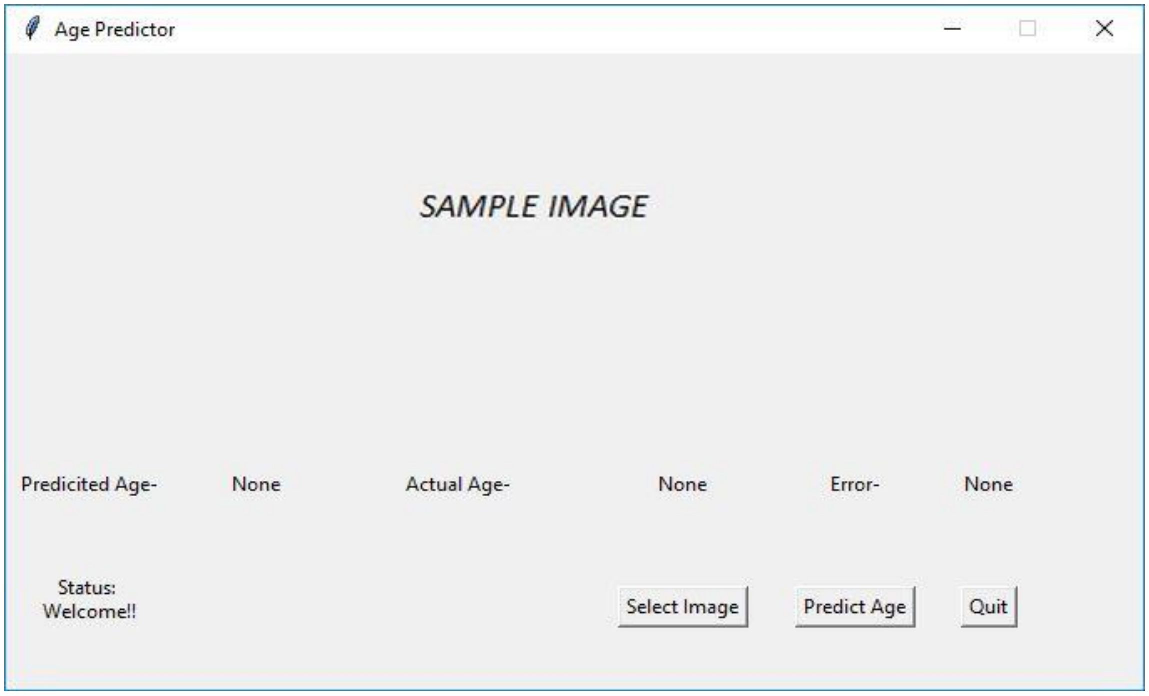
To create a tkinter, steps are :

a) Importing the module – tkinter
b) Create the main window

c) Add any number of widgets to the main window

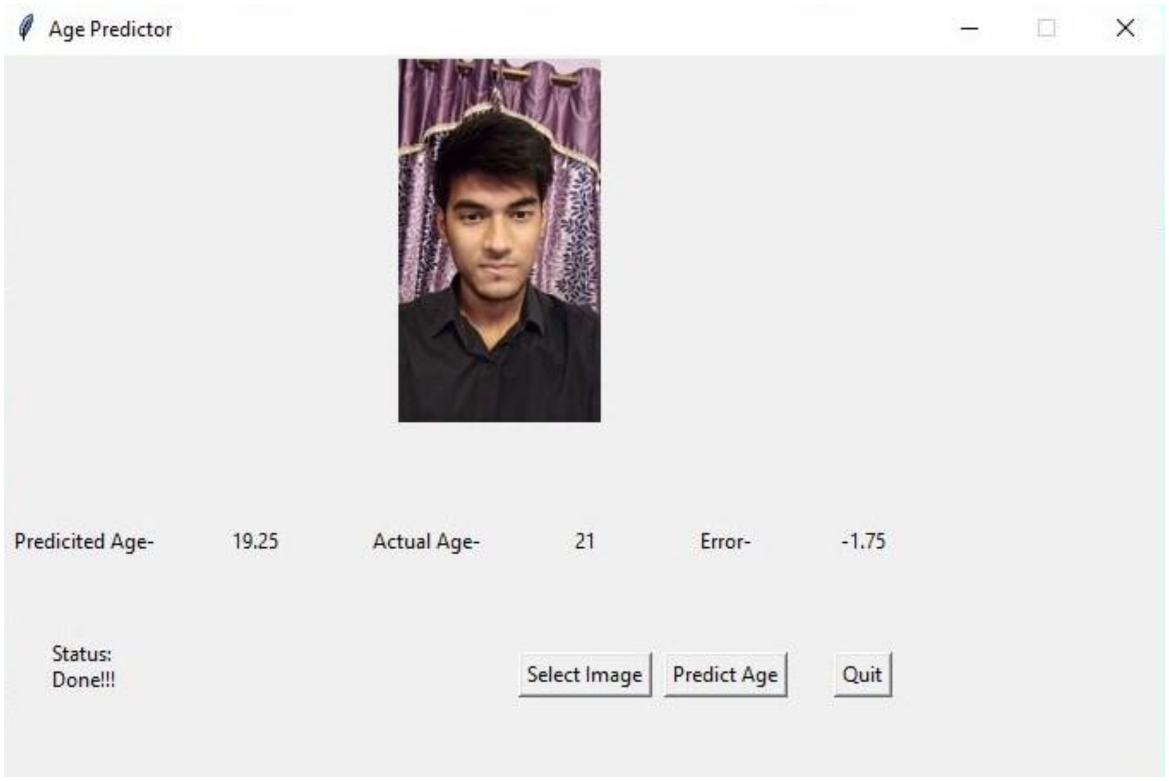
d) Apply the event trigger on the widgets

Importing tkinter is same as importing any other module in the python code.



Predicited Age- None Actual Age- None Error- None

Status:
Welcome!!



Predicited Age- 19.25 Actual Age- 21 Error- -1.75

Status:
Done!!!

Fig 3: GUI features

2.5 MAKING A WEBPAGE :

The final step was to make a webpage that incorporates the **Age Predicator** application. We made a webpage using flask as backend and JS as frontend.

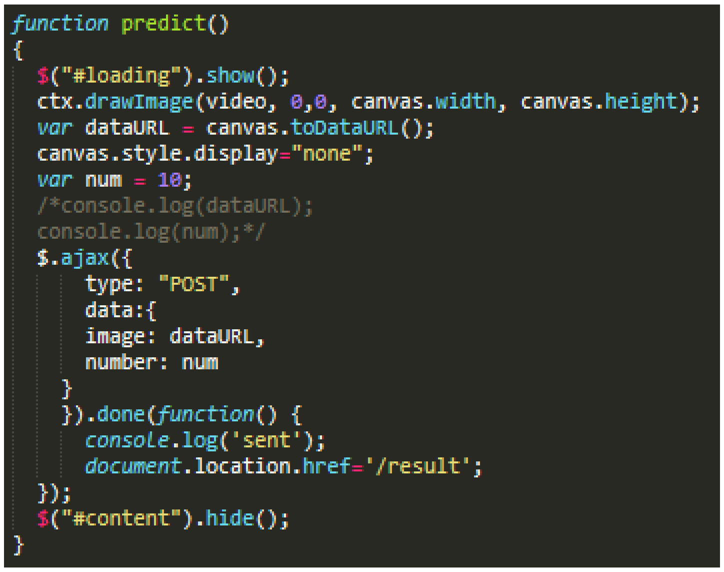
Flask is a web application framework written in python. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine.

JavaScript which is a lightweight, interpreted programming language, is used for creating network-centric applications. Javascript is very easy to implement because it is integrated with HTML. It is used to send the image from web page to flask backend.

Data stream is taken from the web page and all the bytes in the image is passed in base64 format to the Flask backend. No image data is saved, fully or partially whatsoever, and once the processing is done all the bytes of the image is deleted from the memory. This helps in insuring that the user can trust all this features of age prediction and ensure their privacy.

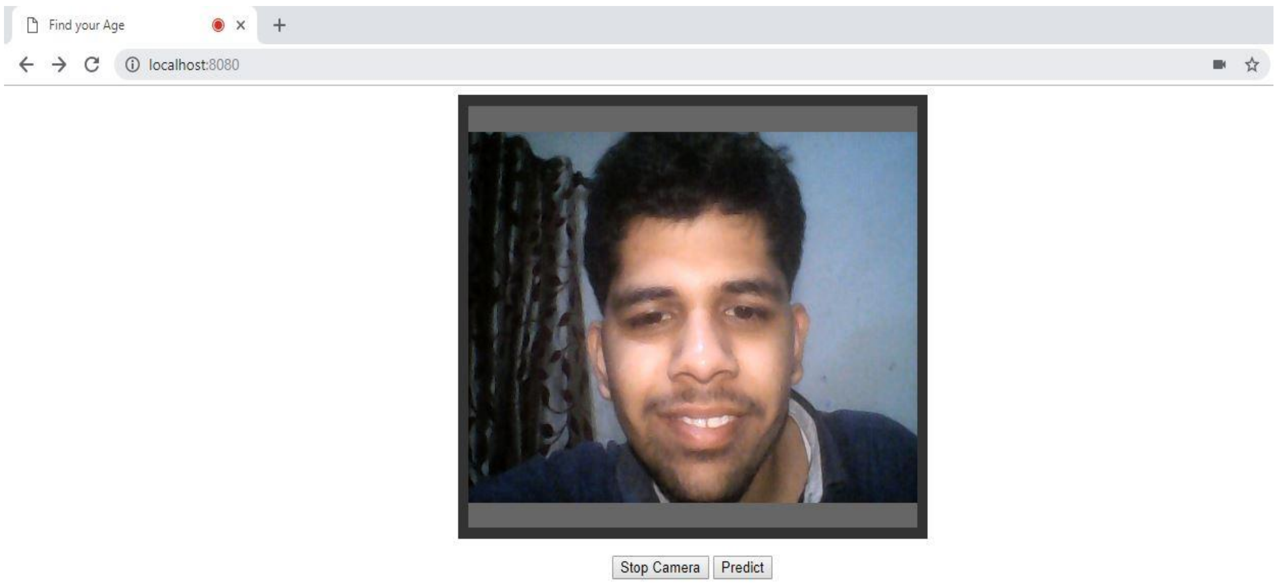
To make it look more presentable CSS styling is used. It is important that the processing must be fast hence the ajax call is used to send the data from the JavaScript while Jinja is used to receive the data from python backend to html. Version of Jinja used is Jinja2.

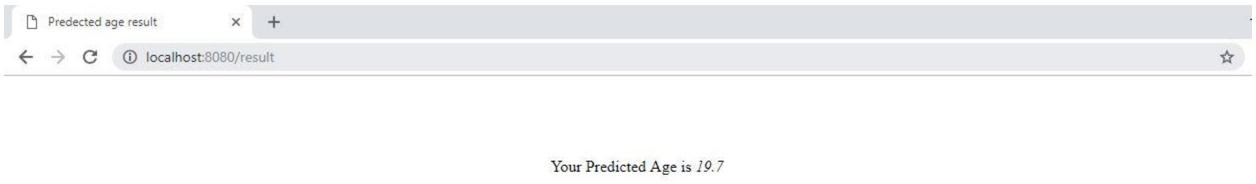
Conversion of base64 to image requires the use of base64 decoder as well as the PIL library. For the main page, i.e. /index.html, both the methods of 'get' and 'post' need to implemented. But for '/result.html' only 'get' method is necessary to load the content on the page.



```
function predict()
{
  $("#loading").show();
  ctx.drawImage(video, 0,0, canvas.width, canvas.height);
  var dataURL = canvas.toDataURL();
  canvas.style.display="none";
  var num = 10;
  /*console.log(dataURL);
  console.log(num);*/
  $.ajax({
      type: "POST",
      data:{
      image: dataURL,
      number: num
    }
    }).done(function() {
      console.log('sent');
      document.location.href='/result';
  });
  $("#content").hide();
}
```

Fig 4: JavaScript function to send the information to python backend

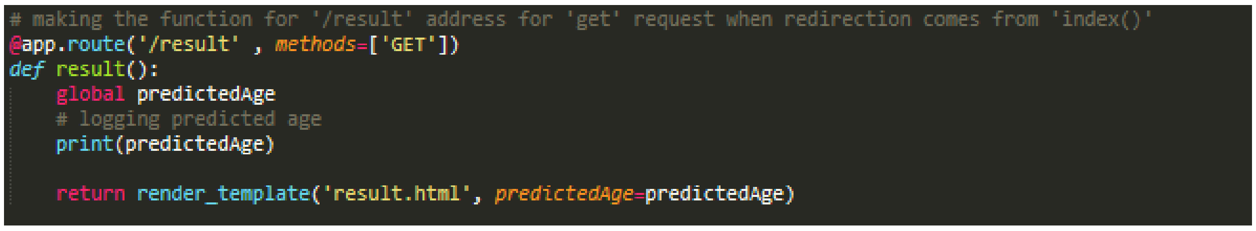




Your Predicted Age is *19.7*

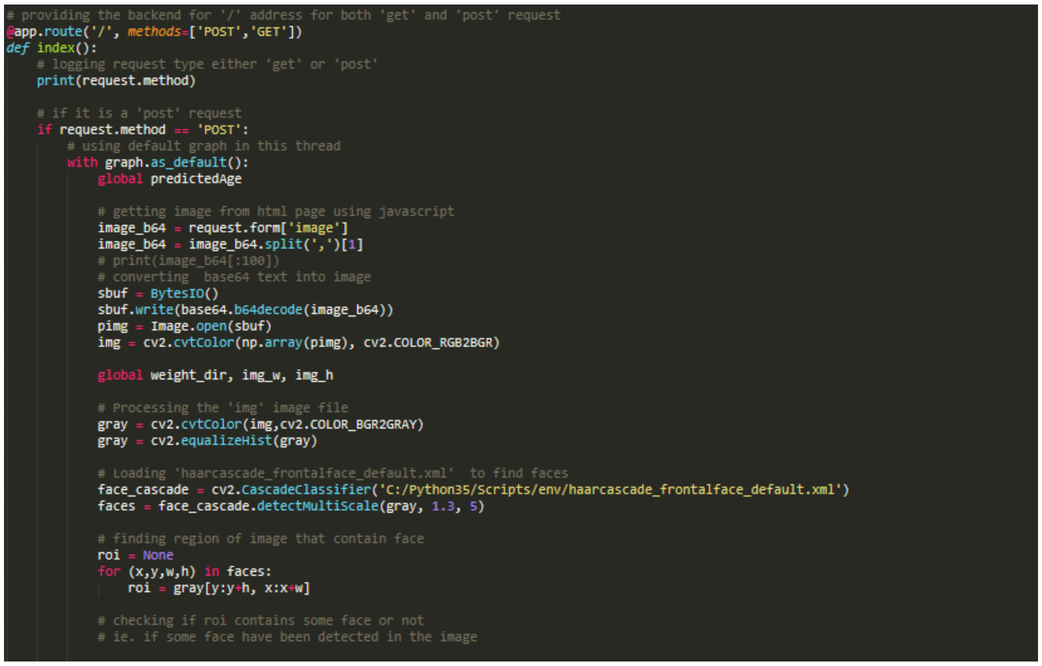
Fig 5: Webpages screen shot of both ‘/’ and ‘/result’ pages

Code for webpage backend (in flask) is –

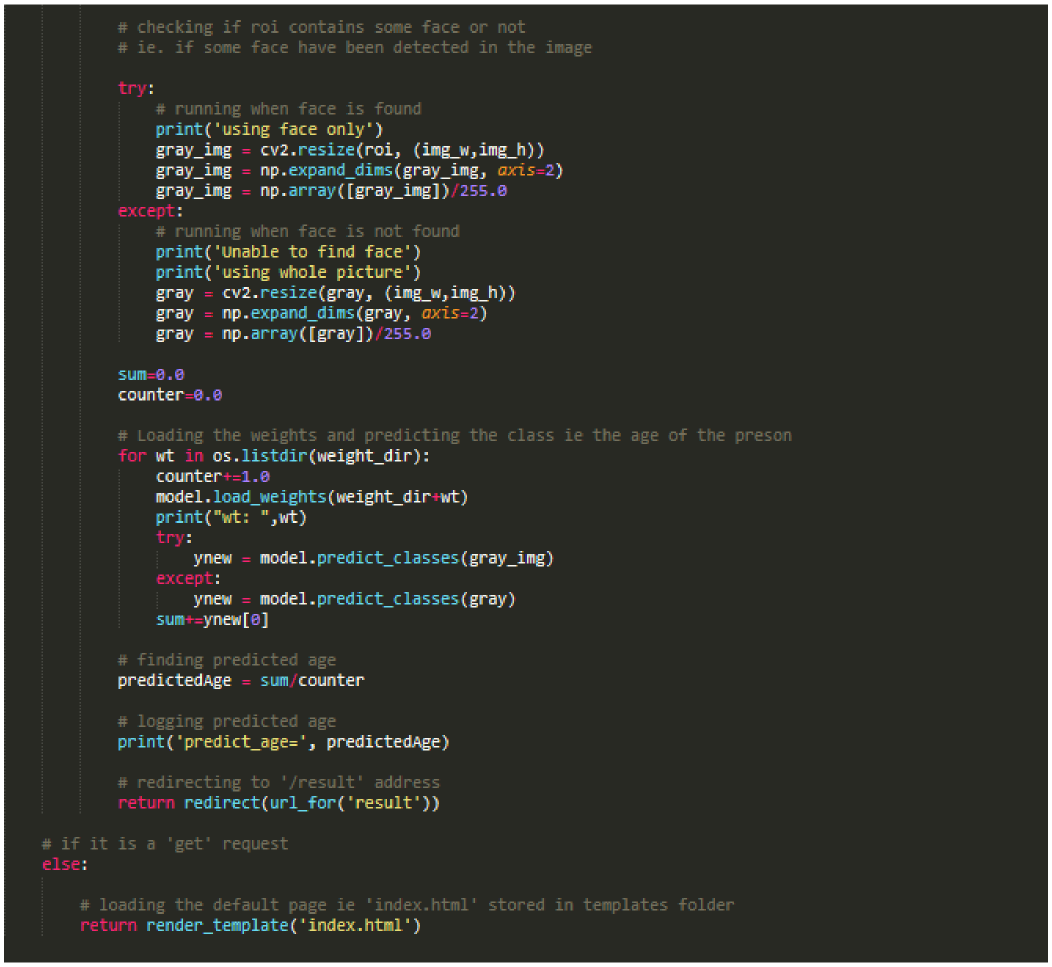


```
# making the function for '/result' address for 'get' request when redirection comes from 'index()'
@app.route('/result' , methods=['GET'])
def result():
    global predictedAge
    # logging predicted age
    print(predictedAge)

    return render_template('result.html', predictedAge=predictedAge)
```

Fig 6: Code of backend of ‘/result’ page in result()



```python
# providing the backend for '/' address for both 'get' and 'post' request
@app.route('/', methods=['POST','GET'])
def index():
    # logging request type either 'get' or 'post'
    print(request.method)

    # if it is a 'post' request
    if request.method == 'POST':
        # using default graph in this thread
        with graph.as_default():
            global predictedAge

            # getting image from html page using javascript
            image_b64 = request.form['image']
            image_b64 = image_b64.split(',')[1]
            # print(image_b64[:100])
            # converting  base64 text into image
            sbuf = BytesIO()
            sbuf.write(base64.b64decode(image_b64))
            pimg = Image.open(sbuf)
            img = cv2.cvtColor(np.array(pimg), cv2.COLOR_RGB2BGR)

            global weight_dir, img_w, img_h

            # Processing the 'img' image file
            gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
            gray = cv2.equalizeHist(gray)

            # Loading 'haarcascade_frontalface_default.xml'  to find faces
            face_cascade = cv2.CascadeClassifier('C:/Python35/Scripts/env/haarcascade_frontalface_default.xml')
            faces = face_cascade.detectMultiScale(gray, 1.3, 5)

            # finding region of image that contain face
            roi = None
            for (x,y,w,h) in faces:
                roi = gray[y:y+h, x:x+w]

            # checking if roi contains some face or not
            # ie. if some face have been detected in the image
```



```python
            # checking if roi contains some face or not
            # ie. if some face have been detected in the image

            try:
                # running when face is found
                print('using face only')
                gray_img = cv2.resize(roi, (img_w,img_h))
                gray_img = np.expand_dims(gray_img, axis=2)
                gray_img = np.array([gray_img])/255.0
            except:
                # running when face is not found
                print('Unable to find face')
                print('using whole picture')
                gray = cv2.resize(gray, (img_w,img_h))
                gray = np.expand_dims(gray, axis=2)
                gray = np.array([gray])/255.0

            sum=0.0
            counter=0.0

            # Loading the weights and predicting the class ie the age of the preson
            for wt in os.listdir(weight_dir):
                counter+=1.0
                model.load_weights(weight_dir+wt)
                print("wt: ",wt)
                try:
                    ynew = model.predict_classes(gray_img)
                except:
                    ynew = model.predict_classes(gray)
                sum+=ynew[0]

            # finding predicted age
            predictedAge = sum/counter

            # logging predicted age
            print('predict_age=', predictedAge)

            # redirecting to '/result' address
            return redirect(url_for('result'))

    # if it is a 'get' request
    else:

        # loading the default page ie 'index.html' stored in templates folder
        return render_template('index.html')
```

Fig 6: Code of backend of '/' page i.e. default page in index()

Chapter 3

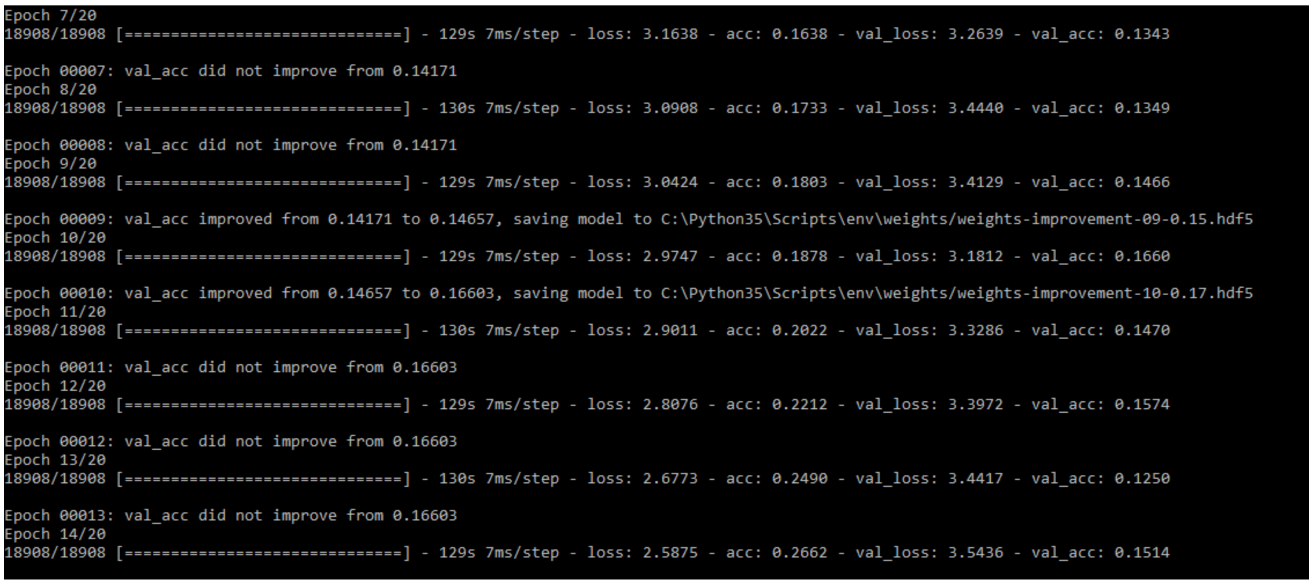
EXPERIMENTS AND RESULTS

After the model is trained, predicted age, absolute error along with true age is predicted for each image with its filename.

Thus, we can check the absolute error and find whether the model passes some set threshold, for example, mean absolute error must be less than 5 years.

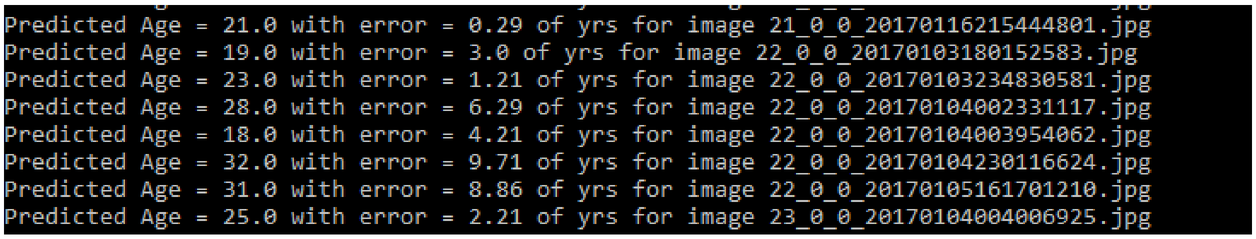
It is also important to state that the validation error is not a correct way of finding the accuracy of the model since only *exact* match will be counted as a match!

But in real life situation, inaccuracy of few years can be considered as some difference in inherent qualities of a person's facial features.

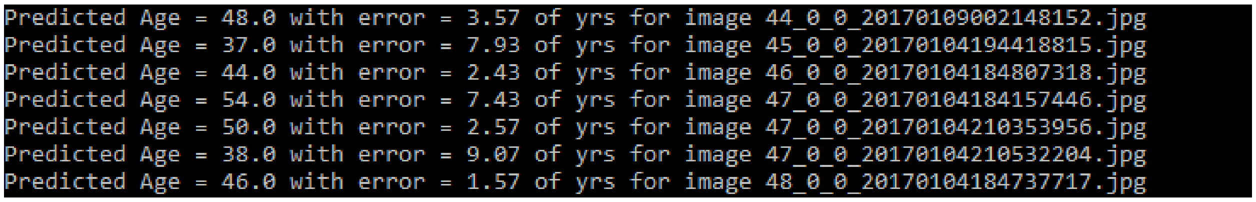


```
Epoch 7/20
18908/18908 [==============================] - 129s 7ms/step - loss: 3.1638 - acc: 0.1638 - val_loss: 3.2639 - val_acc: 0.1343

Epoch 00007: val_acc did not improve from 0.14171
Epoch 8/20
18908/18908 [==============================] - 130s 7ms/step - loss: 3.0908 - acc: 0.1733 - val_loss: 3.4440 - val_acc: 0.1349

Epoch 00008: val_acc did not improve from 0.14171
Epoch 9/20
18908/18908 [==============================] - 129s 7ms/step - loss: 3.0424 - acc: 0.1803 - val_loss: 3.4129 - val_acc: 0.1466

Epoch 00009: val_acc improved from 0.14171 to 0.14657, saving model to C:\Python35\Scripts\env\weights/weights-improvement-09-0.15.hdf5
Epoch 10/20
18908/18908 [==============================] - 129s 7ms/step - loss: 2.9747 - acc: 0.1878 - val_loss: 3.1812 - val_acc: 0.1660

Epoch 00010: val_acc improved from 0.14657 to 0.16603, saving model to C:\Python35\Scripts\env\weights/weights-improvement-10-0.17.hdf5
Epoch 11/20
18908/18908 [==============================] - 130s 7ms/step - loss: 2.9011 - acc: 0.2022 - val_loss: 3.3286 - val_acc: 0.1470

Epoch 00011: val_acc did not improve from 0.16603
Epoch 12/20
18908/18908 [==============================] - 129s 7ms/step - loss: 2.8076 - acc: 0.2212 - val_loss: 3.3972 - val_acc: 0.1574

Epoch 00012: val_acc did not improve from 0.16603
Epoch 13/20
18908/18908 [==============================] - 130s 7ms/step - loss: 2.6773 - acc: 0.2490 - val_loss: 3.4417 - val_acc: 0.1250

Epoch 00013: val_acc did not improve from 0.16603
Epoch 14/20
18908/18908 [==============================] - 129s 7ms/step - loss: 2.5875 - acc: 0.2662 - val_loss: 3.5436 - val_acc: 0.1514
```

Fig 7: Training Epoch Snapshot

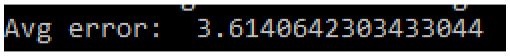


```
Predicted Age = 21.0 with error = 0.29 of yrs for image 21_0_0_20170116215444801.jpg
Predicted Age = 19.0 with error = 3.0 of yrs for image 22_0_0_20170103180152583.jpg
Predicted Age = 23.0 with error = 1.21 of yrs for image 22_0_0_20170103234830581.jpg
Predicted Age = 28.0 with error = 6.29 of yrs for image 22_0_0_20170104002331117.jpg
Predicted Age = 18.0 with error = 4.21 of yrs for image 22_0_0_20170104003954062.jpg
Predicted Age = 32.0 with error = 9.71 of yrs for image 22_0_0_20170104230116624.jpg
Predicted Age = 31.0 with error = 8.86 of yrs for image 22_0_0_20170105161701210.jpg
Predicted Age = 25.0 with error = 2.21 of yrs for image 23_0_0_20170104004006925.jpg
```



```
Predicted Age = 48.0 with error = 3.57 of yrs for image 44_0_0_20170109002148152.jpg
Predicted Age = 37.0 with error = 7.93 of yrs for image 45_0_0_20170104194418815.jpg
Predicted Age = 44.0 with error = 2.43 of yrs for image 46_0_0_20170104184807318.jpg
Predicted Age = 54.0 with error = 7.43 of yrs for image 47_0_0_20170104184157446.jpg
Predicted Age = 50.0 with error = 2.57 of yrs for image 47_0_0_20170104210353956.jpg
Predicted Age = 38.0 with error = 9.07 of yrs for image 47_0_0_20170104210532204.jpg
Predicted Age = 46.0 with error = 1.57 of yrs for image 48_0_0_20170104184737717.jpg
```

Fig 8: Output Snap Shots showing error



```
Avg error:  3.6140642303433044
```

Fig 9: Mean Absolute error

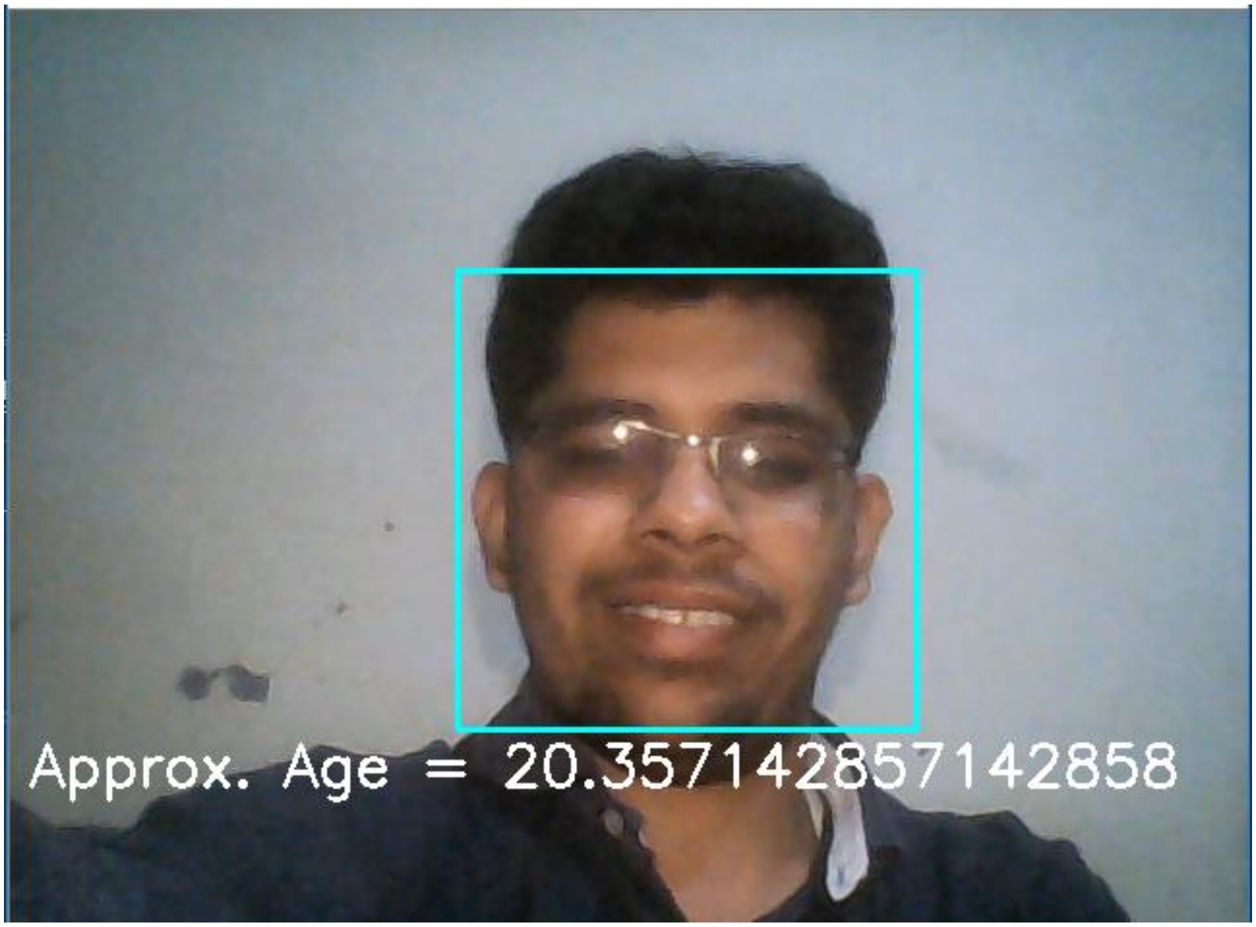
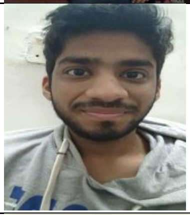
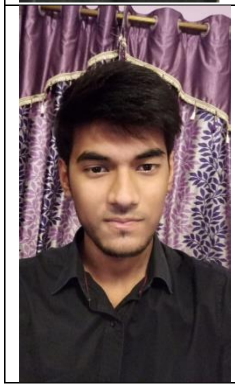


Fig 10: Age using Webcam image

Image	Actual Age	Predicted Age	Absolute Error
	20	20.25	0.75
	18	17.25	0.75
	21	25.0	4.0
	21	19.0	2.0
	21	20.25	0.75
	20	19.25	0.75

Tab 1: Predicted age with error

Chapter 4

CONCLUSION

In this project, we proposed an neural network based approach for prediction of age using biometric feature majorly consisting of facial expression. Various Convolutional Neural Network model Architecture were analyzed on dataset consisting of more than 23000 images.

We are predicting the age along with the absolute error for each image that have been processed.

Finally, the GUI is used to display this information in more presentable formats. A web application is made using Flask as the backend tool which helps in prediction using the image captured through a standard webcam.

Chapter 5

Future Scope

For future work we would like to extend this work to work for multiple faces in single image as well. We would also like to make it more accurate for not so clear images.

More CNN architectures and use of detection of faces pattern like the distance between different pre-specified points so that it could also recognize the person in the image as well.

REFERENCES

[1] Age Estimation from Face Image using Wrinkle Features : International Conference on Information and Communication Technologies (ICICT 2014), Ranjan Janaa, Debaleena Dattaa, Rituparna Sahaa.

[2] Facial Age Estimation With Age Difference : Zhenzhen Hu, Yonggang Wen, Senior Member, IEEE, Jianfeng Wang, Meng Wang, Member, IEEE, Richang Hong, Member, IEEE, and Shuicheng Yan, Fellow, IEEE

[3] Deep Convolutional Neural Network for Age Estimation based on VGG-Face Model: Zakariya Qawaqneh(1), Arafat Abu Mallouh(1), Buket D. Barkana(2) (1)Department of Computer Science and Engineering, University of Bridgeport, (2)Department of Electrical Engineering, University of Bridgeport, Technology Building, Bridgeport CT 06604 USA

[4] Convolutional Neural Network based Age Estimation from Facial Image and Depth Prediction from Single Image : Jiayan Qiu B. Eng. (Honours) Australian National University

[5] https://keras.io/

[6] https://susanqq.github.io/UTKFace

[7] http://flask.pocoo.org/docs/0.12/