1.1 Introduction

The Gender and Age Prediction is a classification project where we have to analyze an image and predict the gender and age. This is a deep learning project where we use image classification and regression models to obtain the results. We will use Convolutional Neural Network (CNN) for image feature extraction and visualize the results with plot graphs. We will create an image classification model for the gender prediction and a regression model for the age prediction

```
Original Gender: Female Original Age: 7
Predicted Gender: Male Predicted Age: 8
```

.

Convolutional Neural Network (CNN) -These are a special kind of multi-layer neural networks, designed to recognize visual patterns directly from pixel images with minimal pre-processing. It is a special architecture of artificial neural networks. Convolutional neural network uses some of its features of visual cortex and have therefore achieved state of the art results in computer vision tasks. Convolutional neural networks are comprised of two very simple elements, namely convolutional layers and pooling layers. The reason why convolutional neural network is hugely popular is because of their architecture as there is no need of feature extraction. The system learns to do feature extraction and the core concept, it uses convolution of image and filters to generate invariant features which are passed on to the next layer. The features in next layer are convoluted with different filters to generate more invariant and abstract features and the process continues till it gets final feature/output which is invariant to occlusions. The most commonly used architectures of convolutional neural network are LeNet, AlexNet, ZFNet, GoogLeNet.

Use: Predictions of age and gender made with AI can be applied to many areas such as intelligent human- machine interface development ,security , cosmetics , electronic commerce.

. Security and video surveillance, electronic customer relationship management, biometrics, electronic vending machines, human-computer interface, entertainment, cosmetology, and forensic art are just a few of the real-world applications where age and gender classification might come in handy.

1 Dataset Information-

UTKFace dataset is a large-scale face dataset with long age span (0-116 years old). The dataset consists of over 23j

K face images with annotations of age, gender and ethnicity. 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 ,landmark, localization etc.

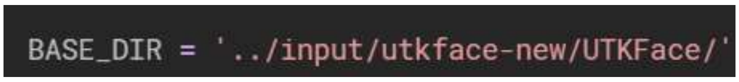
Import Modules

First we import modules in our Kaggle Environment

- **pandas** - used to perform data manipulation and analysis
- **numpy** - used to perform a wide variety of mathematical operations on arrays
- **matplotlib** - used for data visualization and graphical plotting
- **seaborn** - built on top of matplotlib with similar functionalities
- **os -** used to handle files using system commands
- **tqdm** - progress bar decorator for iterators
- **warnings** - to manipulate warnings details, filterwarnings('ignore') is to ignore the warnings thrown by the modules (gives clean results)
- **load_img** - used for loading the image as numpy array
- **tensorflow** - backend module for the use of Keras
- **Dense** - single dimension linear layer
- **Dropout** - used to add regularization to the data, avoiding over fitting & dropping out a fraction of the data
- **Activation** - layer for the use of certain threshold
- **Flatten -** convert a 2D array into a 1D array
- **Conv2D** - convolutional layer in 2 dimension
- **MaxPooling2D** - function to get the maximum pixel value to the next layer

Load the Dataset

Now we load the dataset for processing using directory where we stored the dataset.

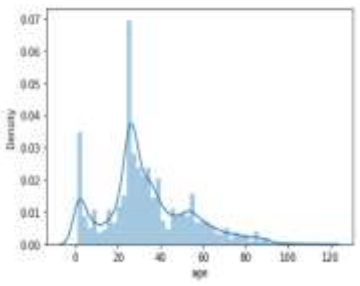


```
BASE_DIR = '../input/utkface-new/UTKFace/'
```

Exploratory Data Analysis

3.5 - Data Analysis

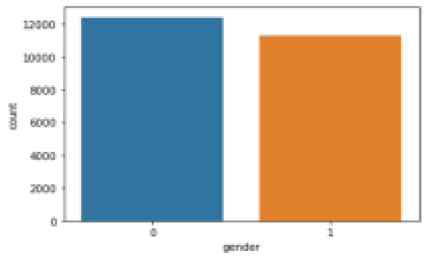
- Display of the first image in the dataset
- You may resize the image to a uniform width and height for easier processing
- In this project we will resize all images to 128 x 128 due to limited resources



Graph- Age & Density

Distplot of the age attribute

- The majority are in between ages 25 to 30 years old.
- You may convert this distribution into a scaled format using Standard Scalar (or) Min Max Normalization

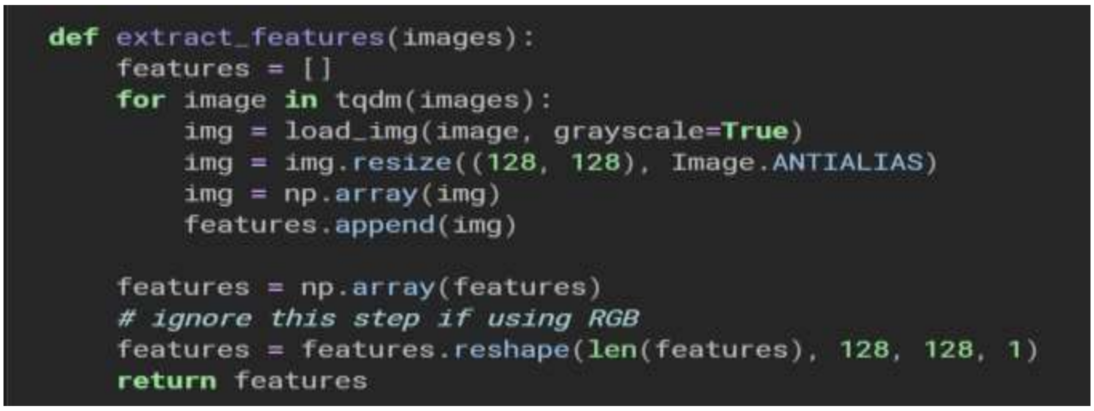


Graph - Gender & Count

Visualization of the gender attribute and it's in uniform distribution

3.6 Feature Extraction

Now we define the feature extraction function
Image reshaped is defined and in grayscale for quicker processing

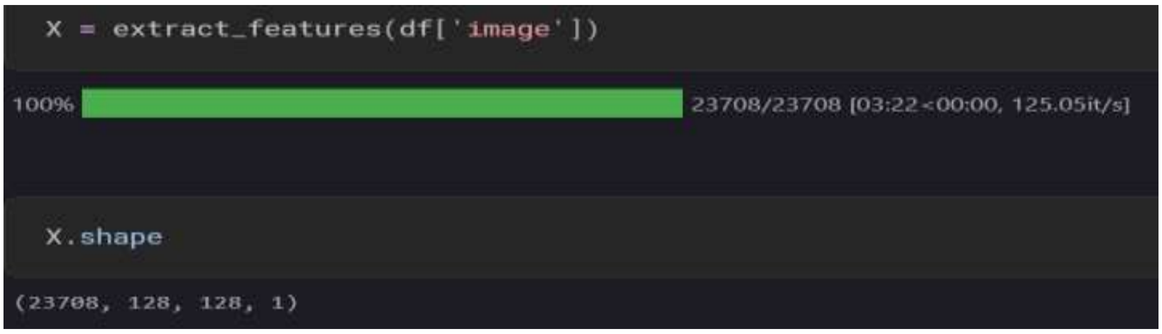


```
def extract_features(images):
    features = []
    for image in tqdm(images):
        img = load_img(image, grayscale=True)
        img = img.resize((128, 128), Image.ANTIALIAS)
        img = np.array(img)
        features.append(img)

    features = np.array(features)
    # ignore this step if using RGB
    features = features.reshape(len(features), 128, 128, 1)
    return features
```

Feature Extraction

Testing the Feature Extraction



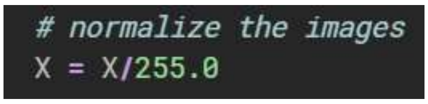
```
X = extract_features(df['image'])
```

```
100%  23708/23708 [03:22<00:00, 125.05it/s]
```

```
X.shape
```

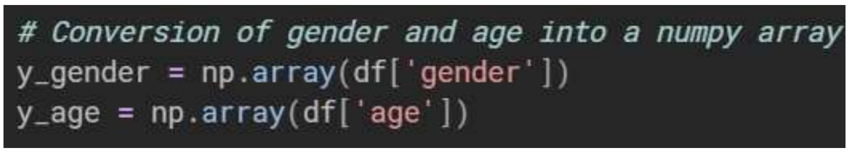
```
(23708, 128, 128, 1)
```

Feature Extracted



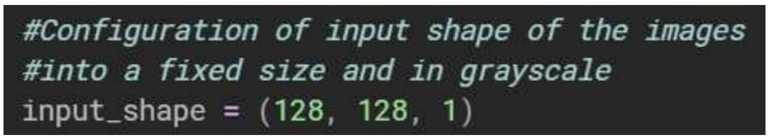
```
# normalize the images
X = X/255.0
```

Normalized Images from Range 1-255 into 0 to 1



```
# Conversion of gender and age into a numpy array
y_gender = np.array(df['gender'])
y_age = np.array(df['age'])
```

Conversion

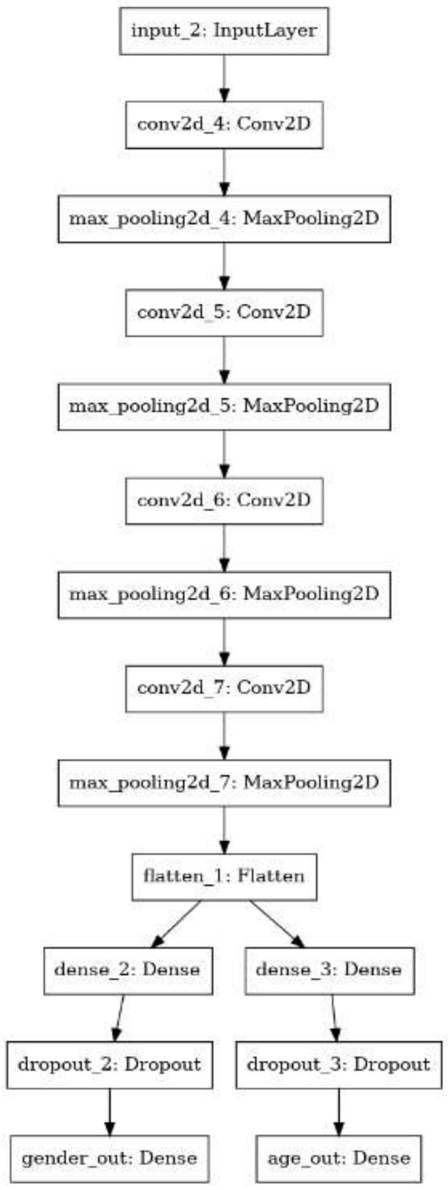


```
#Configuration of input shape of the images
#into a fixed size and in grayscale
input_shape = (128, 128, 1)
```

Configuration

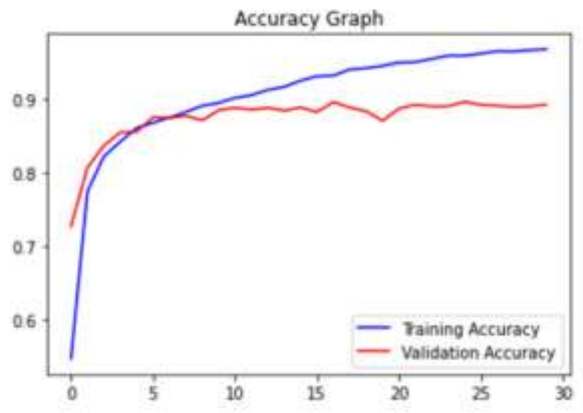
Model Creation

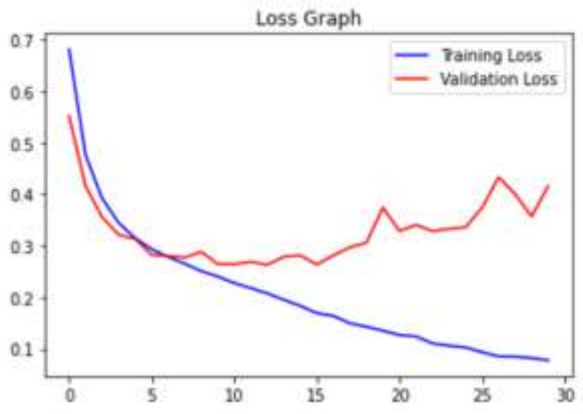
- **Dropout()** - used to add regularization to the data, avoiding over fitting & dropping out a fraction of the data from the layers
- **activation='sigmoid'** - used for binary classification
- **optimizer='adam'** - automatically adjust the learning rate for the model over the no. of epochs
- **loss='binary_crossentropy'** - loss function for binary outputs
- **Model plot** shows the image processing layers and split into 2 dense layers for classification and regression outputs
- **Train the model**

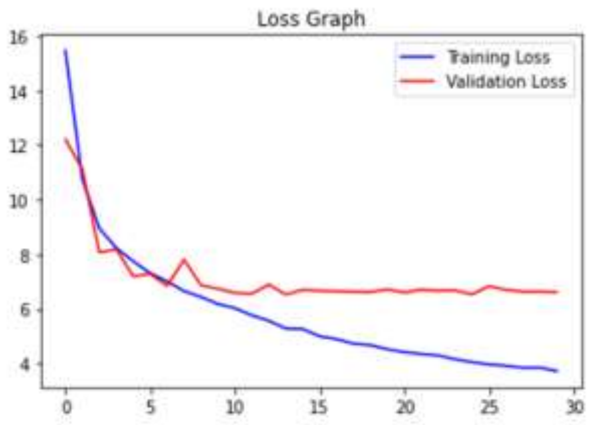

input_2: InputLayer
conv2d_4: Conv2D
max_pooling2d_4: MaxPooling2D
conv2d_5: Conv2D
max_pooling2d_5: MaxPooling2D
conv2d_6: Conv2D
max_pooling2d_6: MaxPooling2D
conv2d_7: Conv2D
max_pooling2d_7: MaxPooling2D
flatten_1: Flatten
dense_2: Dense
dense_3: Dense
dropout_2: Dropout
dropout_3: Dropout
gender_out: Dense
age_out: Dense

Model Plot

Plot the Results

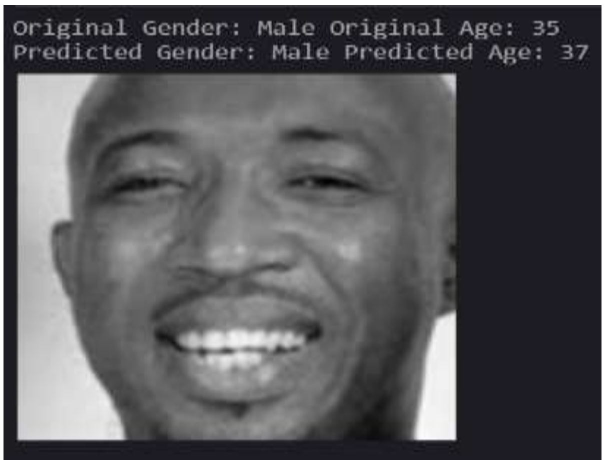




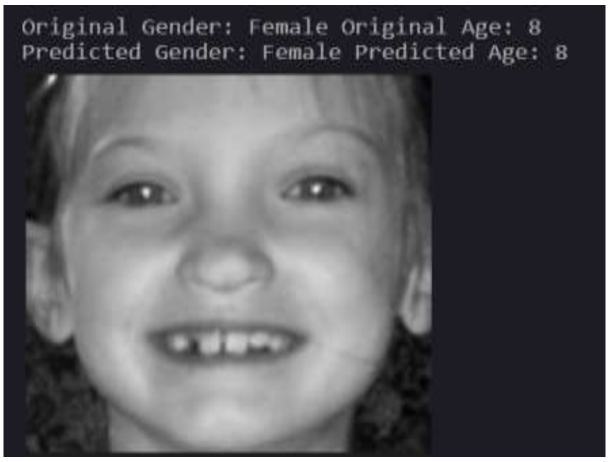


Prediction with Test Data

Some of the results:



```
Original Gender: Male Original Age: 35
Predicted Gender: Male Predicted Age: 37
```



```
Original Gender: Female Original Age: 8
Predicted Gender: Female Predicted Age: 8
```

Final Thoughts

- Training the model by increasing the no. of epochs can give better and more accurate results.
- Processing large amount of data can take a lot of time and system resources.

Code's functionality:

We used two o/p labels from the dataset.

Age is considered as a regression problem.

Gender is considered as a classification problem.

1-female 0-male

We used Kaggle environment as we can directly load the dataset from here.

We enabled GPU as we are going to deal with the neural networks to speed up the process.

Load the dataset-------
Make base directory to load the dataset

Load images alongwith the labels:
Labels-age,gender,ethnicity
We make 3 lists:image_paths,age_labels,gender_labels
Temp to split image path by "_" as age_gender_ethnicity_imageid

os.listdir() method in python is used to get the list of all files and directories in the specified directory. *The return type of this method is list.*

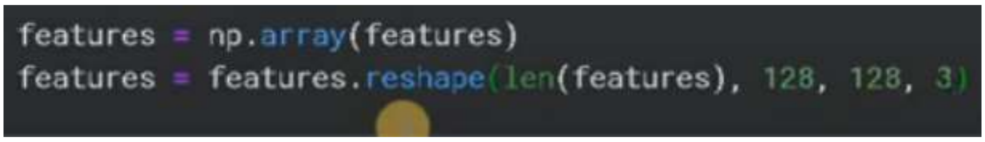
DataFrame is a 2 dimensional data structure, like a 2 dimensional array, or a table with rows and columns.

We use gender dictionary to display the corresponding details for interpretation.

Exploratory data analysis:
Displaying only 1 image
Module: import Image
Plt.axis('off')-removes the axis
Plt.imshow(img)-removes the additional text associated with image

Using grayscale(1 dimension) for better processing as ram is not sufficient . rgb(3 dimensions)

Feature Extraction:
To avoid distortions we use ANTIALIAS
NumPy array is **a powerful N-dimensional array object**

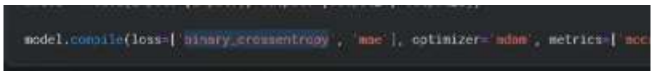


```
features = np.array(features)
features = features.reshape(len(features), 128, 128, 3)
```

2 represents rgb…..which is converted to grayscale I.e.,1

X.shape-to check the shape of data
Normalize the data:x=x/255(pixel value=255)

32 filters
Relu activation:improves model performance
Flatten converts matrix into 1d structure



```
model.compile(loss=['binary_crossentropy', 'mae'], optimizer='adam', metrics=['acc
```

Binary classification   Mean absolute error as matrix for regression

epoch is the number of passes a training dataset takes around an algorithm

we have used adam optimizer
regression: forecasts the likelihood of a target variable.
Classification: Classification is defined as the process of recognition, understanding, and grouping of objects and ideas into preset categories