gender-and-age-prediction-cnn

January 31, 2024

0.1 Import Modules

[1]:
```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from tqdm.notebook import tqdm
warnings.filterwarnings('ignore')
%matplotlib inline

import tensorflow as tf
from keras.preprocessing.image import load_img
from keras.models import Sequential, Model
from keras.layers import Dense, Conv2D, Dropout, Flatten, MaxPooling2D, Input
```

0.2 Load the Dataset

[2]:
```
BASE_DIR = '../input/utkface-new/UTKFace/'
```

[3]:
```
# labels - age, gender, ethnicity
image_paths = []
age_labels = []
gender_labels = []

for filename in tqdm(os.listdir(BASE_DIR)):
    image_path = os.path.join(BASE_DIR, filename)
    temp = filename.split('_')
    age = int(temp[0])
    gender = int(temp[1])
    image_paths.append(image_path)
    age_labels.append(age)
    gender_labels.append(gender)
```

```
  0%|          | 0/23708 [00:00<?, ?it/s]
```

[4]:
```
# convert to dataframe
df = pd.DataFrame()
df['image'], df['age'], df['gender'] = image_paths, age_labels, gender_labels
df.head()
```

[4]:

	image	age	gender
0	../input/utkface-new/UTKFace/26_0_2_2017010402…	26	0
1	../input/utkface-new/UTKFace/22_1_1_2017011223…	22	1
2	../input/utkface-new/UTKFace/21_1_3_2017010500…	21	1
3	../input/utkface-new/UTKFace/28_0_0_2017011718…	28	0
4	../input/utkface-new/UTKFace/17_1_4_2017010322…	17	1

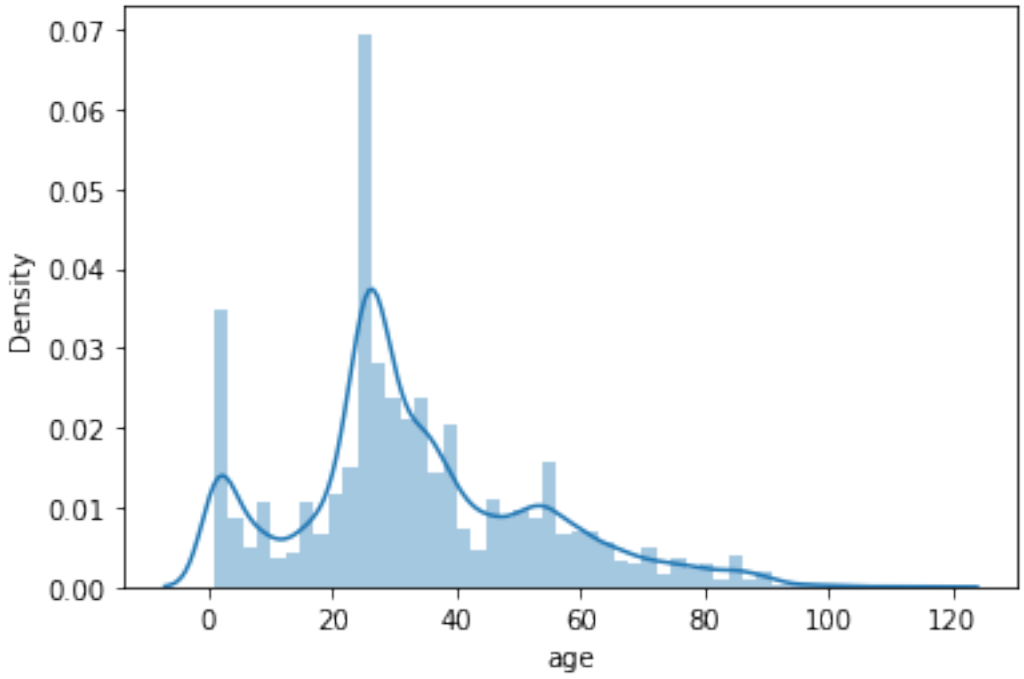
[5]:
```
# map labels for gender
gender_dict = {0:'Male', 1:'Female'}
```

0.3 Exploratory Data Analysis

[6]:
```
from PIL import Image
img = Image.open(df['image'][0])
plt.axis('off')
plt.imshow(img);
```

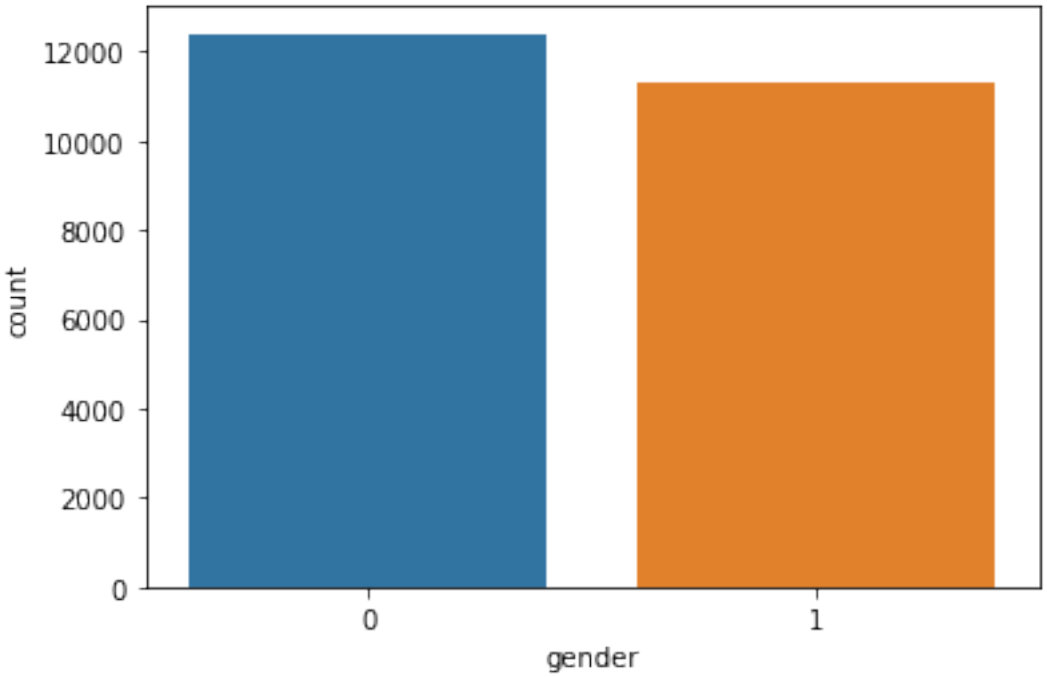
[7]:
```
sns.distplot(df['age'])
```

[7]:
```
<AxesSubplot:xlabel='age', ylabel='Density'>
```

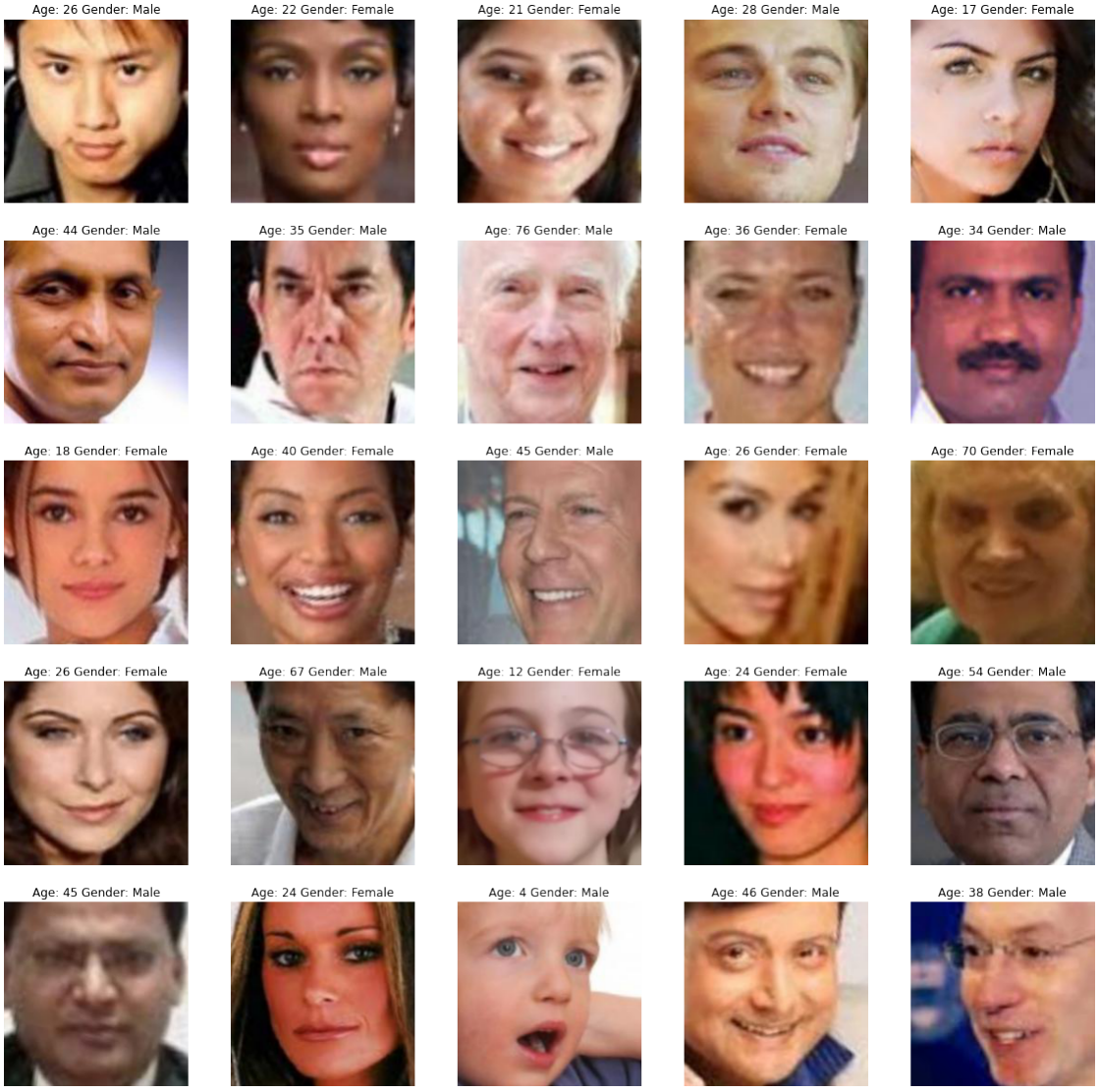


[8]:
```
sns.countplot(df['gender'])
```

[8]: <AxesSubplot:xlabel='gender', ylabel='count'>



[10]:
```
# to display grid of images
plt.figure(figsize=(20, 20))
files = df.iloc[0:25]

for index, file, age, gender in files.itertuples():
    plt.subplot(5, 5, index+1)
    img = load_img(file)
    img = np.array(img)
    plt.imshow(img)
    plt.title(f"Age: {age} Gender: {gender_dict[gender]}")
    plt.axis('off')
```



0.4 Feature Extraction

[11]:
```
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
```

[13]:
```
X = extract_features(df['image'])
```

```
  0%|          | 0/23708 [00:00<?, ?it/s]
```

[15]:
```
X.shape
```

[15]:
```
(23708, 128, 128, 1)
```

[16]:
```
# normalize the images
X = X/255.0
```

[17]:
```
y_gender = np.array(df['gender'])
y_age = np.array(df['age'])
```

[18]:
```
input_shape = (128, 128, 1)
```

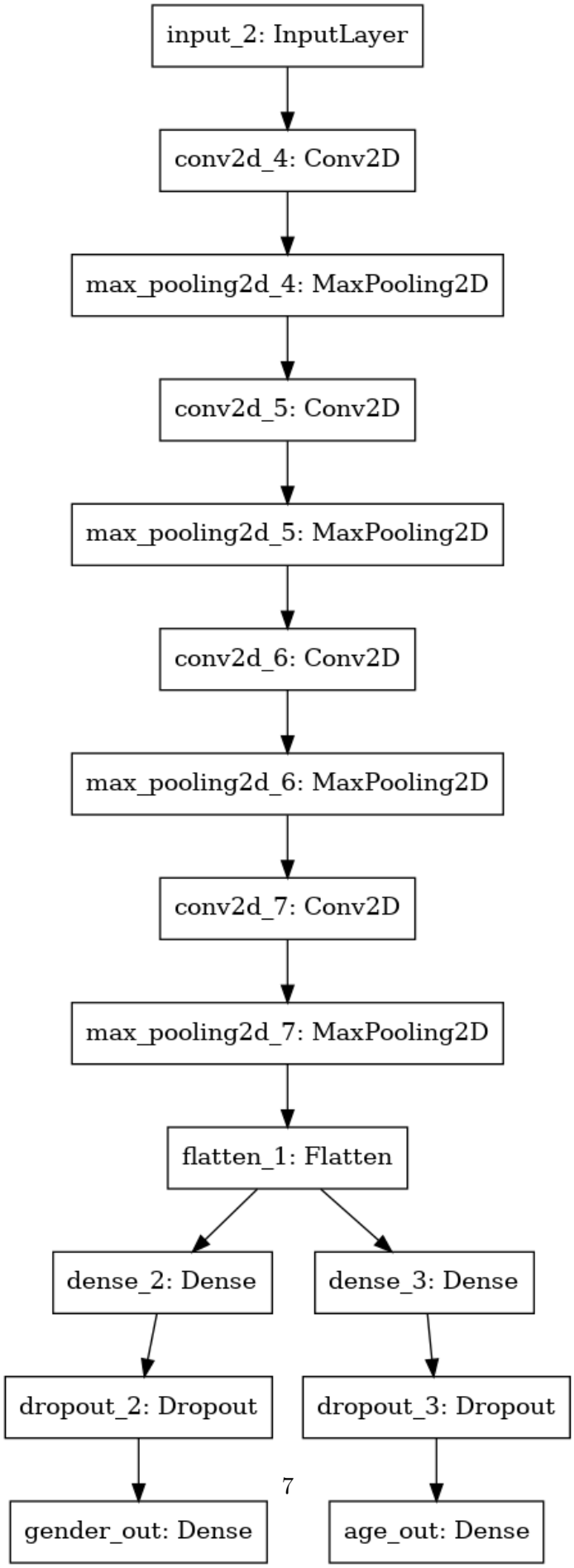
0.5 Model Creation

[21]:
```
inputs = Input((input_shape))
# convolutional layers
conv_1 = Conv2D(32, kernel_size=(3, 3), activation='relu') (inputs)
maxp_1 = MaxPooling2D(pool_size=(2, 2)) (conv_1)
conv_2 = Conv2D(64, kernel_size=(3, 3), activation='relu') (maxp_1)
maxp_2 = MaxPooling2D(pool_size=(2, 2)) (conv_2)
conv_3 = Conv2D(128, kernel_size=(3, 3), activation='relu') (maxp_2)
maxp_3 = MaxPooling2D(pool_size=(2, 2)) (conv_3)
conv_4 = Conv2D(256, kernel_size=(3, 3), activation='relu') (maxp_3)
maxp_4 = MaxPooling2D(pool_size=(2, 2)) (conv_4)

flatten = Flatten() (maxp_4)

# fully connected layers
```

```
dense_1 = Dense(256, activation='relu') (flatten)
dense_2 = Dense(256, activation='relu') (flatten)

dropout_1 = Dropout(0.3) (dense_1)
dropout_2 = Dropout(0.3) (dense_2)

output_1 = Dense(1, activation='sigmoid', name='gender_out') (dropout_1)
output_2 = Dense(1, activation='relu', name='age_out') (dropout_2)

model = Model(inputs=[inputs], outputs=[output_1, output_2])

model.compile(loss=['binary_crossentropy', 'mae'], optimizer='adam',␣
 ↪metrics=['accuracy'])
```

[25]:
```
# plot the model
from tensorflow.keras.utils import plot_model
plot_model(model)
```

[25]:


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

[26]:
```
# train model
history = model.fit(x=X, y=[y_gender, y_age], batch_size=32, epochs=30,␣
↪validation_split=0.2)
```

```
2022-03-20 12:29:48.141054: W
tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of 1242955776
exceeds 10% of free system memory.
2022-03-20 12:29:49.757808: W
tensorflow/core/framework/cpu_allocator_impl.cc:80] Allocation of 1242955776
exceeds 10% of free system memory.
2022-03-20 12:29:50.753984: I
tensorflow/compiler/mlir/mlir_graph_optimization_pass.cc:185] None of the MLIR
Optimization Passes are enabled (registered 2)

Epoch 1/30

2022-03-20 12:29:52.907433: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369]
Loaded cuDNN version 8005

593/593 [==============================] - 18s 17ms/step - loss: 16.1346 -
gender_out_loss: 0.6821 - age_out_loss: 15.4525 - gender_out_accuracy: 0.5472 -
age_out_accuracy: 0.0476 - val_loss: 12.7578 - val_gender_out_loss: 0.5521 -
val_age_out_loss: 12.2057 - val_gender_out_accuracy: 0.7269 -
val_age_out_accuracy: 0.0460
Epoch 2/30
593/593 [==============================] - 9s 16ms/step - loss: 11.2216 -
gender_out_loss: 0.4761 - age_out_loss: 10.7455 - gender_out_accuracy: 0.7741 -
age_out_accuracy: 0.0285 - val_loss: 11.5279 - val_gender_out_loss: 0.4163 -
val_age_out_loss: 11.1116 - val_gender_out_accuracy: 0.8064 -
val_age_out_accuracy: 0.0255
Epoch 3/30
593/593 [==============================] - 10s 16ms/step - loss: 9.3465 -
gender_out_loss: 0.3925 - age_out_loss: 8.9540 - gender_out_accuracy: 0.8214 -
age_out_accuracy: 0.0157 - val_loss: 8.4260 - val_gender_out_loss: 0.3558 -
val_age_out_loss: 8.0702 - val_gender_out_accuracy: 0.8361 -
val_age_out_accuracy: 0.0074
Epoch 4/30
593/593 [==============================] - 9s 16ms/step - loss: 8.5809 -
gender_out_loss: 0.3446 - age_out_loss: 8.2363 - gender_out_accuracy: 0.8416 -
age_out_accuracy: 0.0119 - val_loss: 8.5080 - val_gender_out_loss: 0.3214 -
val_age_out_loss: 8.1866 - val_gender_out_accuracy: 0.8541 -
val_age_out_accuracy: 0.0078
Epoch 5/30
593/593 [==============================] - 9s 16ms/step - loss: 8.0615 -
gender_out_loss: 0.3149 - age_out_loss: 7.7466 - gender_out_accuracy: 0.8602 -
age_out_accuracy: 0.0109 - val_loss: 7.5080 - val_gender_out_loss: 0.3134 -
```

```
val_age_out_loss: 7.1946 - val_gender_out_accuracy: 0.8551 -
val_age_out_accuracy: 0.0076
Epoch 6/30
593/593 [==============================] - 10s 16ms/step - loss: 7.6047 -
gender_out_loss: 0.2935 - age_out_loss: 7.3112 - gender_out_accuracy: 0.8672 -
age_out_accuracy: 0.0096 - val_loss: 7.5676 - val_gender_out_loss: 0.2822 -
val_age_out_loss: 7.2854 - val_gender_out_accuracy: 0.8747 -
val_age_out_accuracy: 0.0063
Epoch 7/30
593/593 [==============================] - 9s 15ms/step - loss: 7.2906 -
gender_out_loss: 0.2782 - age_out_loss: 7.0124 - gender_out_accuracy: 0.8743 -
age_out_accuracy: 0.0080 - val_loss: 7.1280 - val_gender_out_loss: 0.2800 -
val_age_out_loss: 6.8480 - val_gender_out_accuracy: 0.8739 -
val_age_out_accuracy: 0.0049
Epoch 8/30
593/593 [==============================] - 9s 16ms/step - loss: 6.9194 -
gender_out_loss: 0.2654 - age_out_loss: 6.6540 - gender_out_accuracy: 0.8818 -
age_out_accuracy: 0.0072 - val_loss: 8.0823 - val_gender_out_loss: 0.2770 -
val_age_out_loss: 7.8053 - val_gender_out_accuracy: 0.8766 -
val_age_out_accuracy: 0.0049
Epoch 9/30
593/593 [==============================] - 9s 15ms/step - loss: 6.6902 -
gender_out_loss: 0.2507 - age_out_loss: 6.4395 - gender_out_accuracy: 0.8903 -
age_out_accuracy: 0.0064 - val_loss: 7.1591 - val_gender_out_loss: 0.2882 -
val_age_out_loss: 6.8709 - val_gender_out_accuracy: 0.8707 -
val_age_out_accuracy: 0.0032
Epoch 10/30
593/593 [==============================] - 10s 16ms/step - loss: 6.4238 -
gender_out_loss: 0.2404 - age_out_loss: 6.1834 - gender_out_accuracy: 0.8941 -
age_out_accuracy: 0.0063 - val_loss: 7.0038 - val_gender_out_loss: 0.2649 -
val_age_out_loss: 6.7389 - val_gender_out_accuracy: 0.8842 -
val_age_out_accuracy: 0.0051
Epoch 11/30
593/593 [==============================] - 10s 16ms/step - loss: 6.2591 -
gender_out_loss: 0.2276 - age_out_loss: 6.0316 - gender_out_accuracy: 0.9011 -
age_out_accuracy: 0.0063 - val_loss: 6.8535 - val_gender_out_loss: 0.2642 -
val_age_out_loss: 6.5894 - val_gender_out_accuracy: 0.8876 -
val_age_out_accuracy: 0.0027
Epoch 12/30
593/593 [==============================] - 10s 16ms/step - loss: 5.9888 -
gender_out_loss: 0.2179 - age_out_loss: 5.7709 - gender_out_accuracy: 0.9047 -
age_out_accuracy: 0.0072 - val_loss: 6.8253 - val_gender_out_loss: 0.2690 -
val_age_out_loss: 6.5562 - val_gender_out_accuracy: 0.8851 -
val_age_out_accuracy: 0.0049
Epoch 13/30
593/593 [==============================] - 10s 16ms/step - loss: 5.7775 -
gender_out_loss: 0.2075 - age_out_loss: 5.5700 - gender_out_accuracy: 0.9118 -
age_out_accuracy: 0.0059 - val_loss: 7.1583 - val_gender_out_loss: 0.2630 -
```

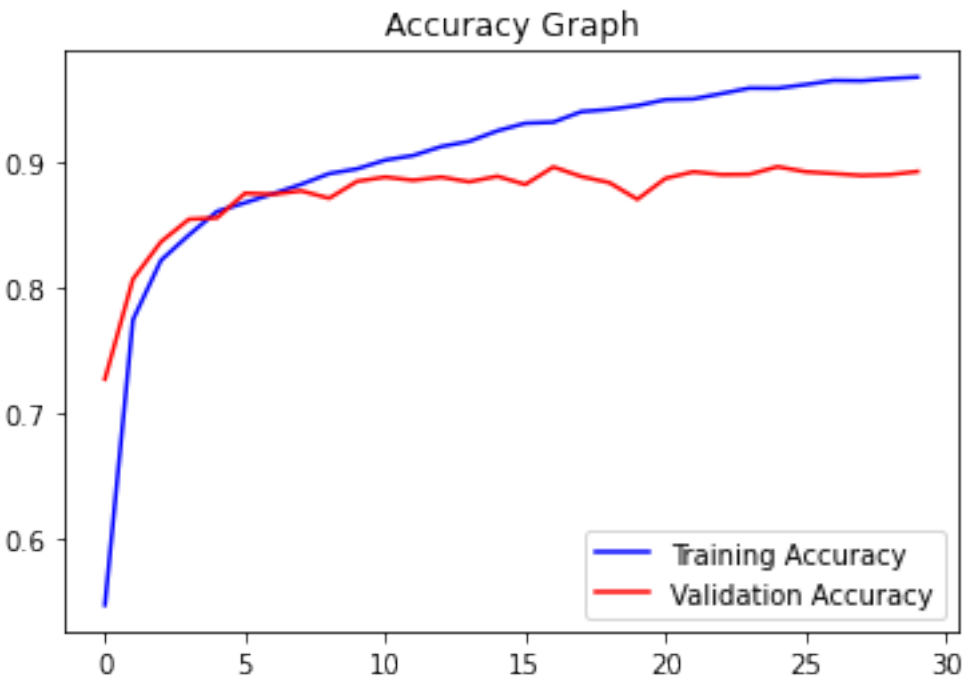
```
val_age_out_loss: 6.8953 - val_gender_out_accuracy: 0.8876 -
val_age_out_accuracy: 0.0036
Epoch 14/30
593/593 [==============================] - 9s 15ms/step - loss: 5.4795 -
gender_out_loss: 0.1951 - age_out_loss: 5.2844 - gender_out_accuracy: 0.9160 -
age_out_accuracy: 0.0054 - val_loss: 6.8055 - val_gender_out_loss: 0.2790 -
val_age_out_loss: 6.5264 - val_gender_out_accuracy: 0.8838 -
val_age_out_accuracy: 0.0034
Epoch 15/30
593/593 [==============================] - 9s 16ms/step - loss: 5.4528 -
gender_out_loss: 0.1831 - age_out_loss: 5.2697 - gender_out_accuracy: 0.9243 -
age_out_accuracy: 0.0057 - val_loss: 6.9825 - val_gender_out_loss: 0.2813 -
val_age_out_loss: 6.7012 - val_gender_out_accuracy: 0.8882 -
val_age_out_accuracy: 0.0034
Epoch 16/30
593/593 [==============================] - 9s 15ms/step - loss: 5.1696 -
gender_out_loss: 0.1693 - age_out_loss: 5.0004 - gender_out_accuracy: 0.9304 -
age_out_accuracy: 0.0051 - val_loss: 6.9286 - val_gender_out_loss: 0.2636 -
val_age_out_loss: 6.6650 - val_gender_out_accuracy: 0.8817 -
val_age_out_accuracy: 0.0044
Epoch 17/30
593/593 [==============================] - 10s 16ms/step - loss: 5.0638 -
gender_out_loss: 0.1638 - age_out_loss: 4.9000 - gender_out_accuracy: 0.9312 -
age_out_accuracy: 0.0054 - val_loss: 6.9296 - val_gender_out_loss: 0.2813 -
val_age_out_loss: 6.6483 - val_gender_out_accuracy: 0.8956 -
val_age_out_accuracy: 0.0042
Epoch 18/30
593/593 [==============================] - 9s 15ms/step - loss: 4.8813 -
gender_out_loss: 0.1494 - age_out_loss: 4.7319 - gender_out_accuracy: 0.9396 -
age_out_accuracy: 0.0053 - val_loss: 6.9294 - val_gender_out_loss: 0.2971 -
val_age_out_loss: 6.6323 - val_gender_out_accuracy: 0.8880 -
val_age_out_accuracy: 0.0040
Epoch 19/30
593/593 [==============================] - 9s 15ms/step - loss: 4.8204 -
gender_out_loss: 0.1428 - age_out_loss: 4.6776 - gender_out_accuracy: 0.9414 -
age_out_accuracy: 0.0057 - val_loss: 6.9242 - val_gender_out_loss: 0.3056 -
val_age_out_loss: 6.6185 - val_gender_out_accuracy: 0.8832 -
val_age_out_accuracy: 0.0042
Epoch 20/30
593/593 [==============================] - 9s 16ms/step - loss: 4.6624 -
gender_out_loss: 0.1350 - age_out_loss: 4.5274 - gender_out_accuracy: 0.9444 -
age_out_accuracy: 0.0057 - val_loss: 7.0920 - val_gender_out_loss: 0.3745 -
val_age_out_loss: 6.7175 - val_gender_out_accuracy: 0.8699 -
val_age_out_accuracy: 0.0070
Epoch 21/30
593/593 [==============================] - 9s 16ms/step - loss: 4.5481 -
gender_out_loss: 0.1267 - age_out_loss: 4.4214 - gender_out_accuracy: 0.9491 -
age_out_accuracy: 0.0068 - val_loss: 6.9295 - val_gender_out_loss: 0.3286 -
```

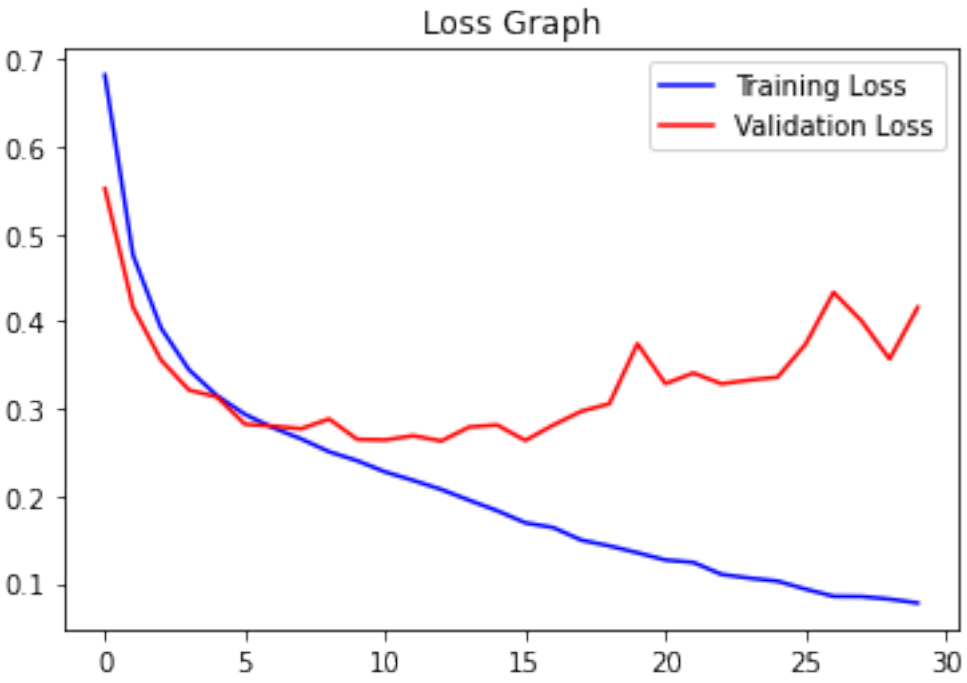
```
val_age_out_loss: 6.6009 - val_gender_out_accuracy: 0.8865 -
val_age_out_accuracy: 0.0032
Epoch 22/30
593/593 [==============================] - 9s 16ms/step - loss: 4.4753 -
gender_out_loss: 0.1239 - age_out_loss: 4.3514 - gender_out_accuracy: 0.9497 -
age_out_accuracy: 0.0054 - val_loss: 7.0483 - val_gender_out_loss: 0.3409 -
val_age_out_loss: 6.7075 - val_gender_out_accuracy: 0.8918 -
val_age_out_accuracy: 0.0070
Epoch 23/30
593/593 [==============================] - 9s 16ms/step - loss: 4.4120 -
gender_out_loss: 0.1102 - age_out_loss: 4.3018 - gender_out_accuracy: 0.9540 -
age_out_accuracy: 0.0090 - val_loss: 6.9948 - val_gender_out_loss: 0.3285 -
val_age_out_loss: 6.6663 - val_gender_out_accuracy: 0.8895 -
val_age_out_accuracy: 0.0105
Epoch 24/30
593/593 [==============================] - 10s 16ms/step - loss: 4.2673 -
gender_out_loss: 0.1059 - age_out_loss: 4.1614 - gender_out_accuracy: 0.9583 -
age_out_accuracy: 0.0193 - val_loss: 7.0131 - val_gender_out_loss: 0.3328 -
val_age_out_loss: 6.6803 - val_gender_out_accuracy: 0.8897 -
val_age_out_accuracy: 0.0243
Epoch 25/30
593/593 [==============================] - 9s 15ms/step - loss: 4.1578 -
gender_out_loss: 0.1024 - age_out_loss: 4.0553 - gender_out_accuracy: 0.9582 -
age_out_accuracy: 0.0264 - val_loss: 6.8706 - val_gender_out_loss: 0.3361 -
val_age_out_loss: 6.5345 - val_gender_out_accuracy: 0.8958 -
val_age_out_accuracy: 0.0287
Epoch 26/30
593/593 [==============================] - 9s 15ms/step - loss: 4.0662 -
gender_out_loss: 0.0933 - age_out_loss: 3.9730 - gender_out_accuracy: 0.9611 -
age_out_accuracy: 0.0299 - val_loss: 7.2064 - val_gender_out_loss: 0.3738 -
val_age_out_loss: 6.8326 - val_gender_out_accuracy: 0.8918 -
val_age_out_accuracy: 0.0266
Epoch 27/30
593/593 [==============================] - 9s 15ms/step - loss: 4.0040 -
gender_out_loss: 0.0851 - age_out_loss: 3.9189 - gender_out_accuracy: 0.9644 -
age_out_accuracy: 0.0311 - val_loss: 7.1397 - val_gender_out_loss: 0.4333 -
val_age_out_loss: 6.7064 - val_gender_out_accuracy: 0.8903 -
val_age_out_accuracy: 0.0331
Epoch 28/30
593/593 [==============================] - 10s 16ms/step - loss: 3.9340 -
gender_out_loss: 0.0848 - age_out_loss: 3.8492 - gender_out_accuracy: 0.9641 -
age_out_accuracy: 0.0344 - val_loss: 7.0291 - val_gender_out_loss: 0.4004 -
val_age_out_loss: 6.6287 - val_gender_out_accuracy: 0.8889 -
val_age_out_accuracy: 0.0251
Epoch 29/30
593/593 [==============================] - 9s 16ms/step - loss: 3.9378 -
gender_out_loss: 0.0819 - age_out_loss: 3.8559 - gender_out_accuracy: 0.9659 -
age_out_accuracy: 0.0329 - val_loss: 6.9958 - val_gender_out_loss: 0.3569 -
```

```
val_age_out_loss: 6.6389 - val_gender_out_accuracy: 0.8895 -
val_age_out_accuracy: 0.0346
Epoch 30/30
593/593 [==============================] - 9s 15ms/step - loss: 3.8026 -
gender_out_loss: 0.0774 - age_out_loss: 3.7252 - gender_out_accuracy: 0.9671 -
age_out_accuracy: 0.0341 - val_loss: 7.0322 - val_gender_out_loss: 0.4161 -
val_age_out_loss: 6.6161 - val_gender_out_accuracy: 0.8920 -
val_age_out_accuracy: 0.0259
```

0.6 Plot the Results

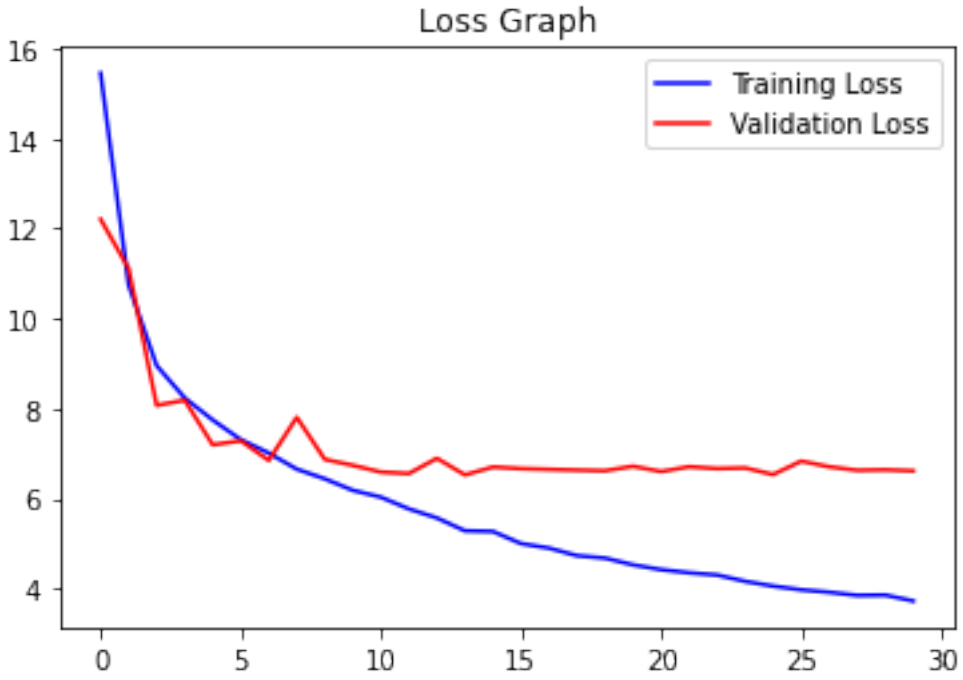
[27]:
```
# plot results for gender
acc = history.history['gender_out_accuracy']
val_acc = history.history['val_gender_out_accuracy']
epochs = range(len(acc))

plt.plot(epochs, acc, 'b', label='Training Accuracy')
plt.plot(epochs, val_acc, 'r', label='Validation Accuracy')
plt.title('Accuracy Graph')
plt.legend()
plt.figure()

loss = history.history['gender_out_loss']
val_loss = history.history['val_gender_out_loss']

plt.plot(epochs, loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Loss Graph')
plt.legend()
plt.show()
```





[28]:
```
# plot results for age
loss = history.history['age_out_loss']
```

```
val_loss = history.history['val_age_out_loss']
epochs = range(len(loss))

plt.plot(epochs, loss, 'b', label='Training Loss')
plt.plot(epochs, val_loss, 'r', label='Validation Loss')
plt.title('Loss Graph')
plt.legend()
plt.show()
```



1 Prediction with Test Data

[32]:
```
image_index = 100
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",␣
  ↪y_age[image_index])
# predict from model
pred = model.predict(X[image_index].reshape(1, 128, 128, 1))
pred_gender = gender_dict[round(pred[0][0][0])]
pred_age = round(pred[1][0][0])
print("Predicted Gender:", pred_gender, "Predicted Age:", pred_age)
plt.axis('off')
plt.imshow(X[image_index].reshape(128, 128), cmap='gray');
```

```
Original Gender: Female Original Age: 3
Predicted Gender: Female Predicted Age: 1
```

[33]:
```
image_index = 3000
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",␣
 ↪y_age[image_index])
# predict from model
pred = model.predict(X[image_index].reshape(1, 128, 128, 1))
pred_gender = gender_dict[round(pred[0][0][0])]
pred_age = round(pred[1][0][0])
print("Predicted Gender:", pred_gender, "Predicted Age:", pred_age)
plt.axis('off')
plt.imshow(X[image_index].reshape(128, 128), cmap='gray');
```

```
Original Gender: Male Original Age: 28
Predicted Gender: Male Predicted Age: 32
```

[34]:
```
image_index = 10000
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",␣
 ↪y_age[image_index])
# predict from model
pred = model.predict(X[image_index].reshape(1, 128, 128, 1))
pred_gender = gender_dict[round(pred[0][0][0])]
pred_age = round(pred[1][0][0])
print("Predicted Gender:", pred_gender, "Predicted Age:", pred_age)
plt.axis('off')
plt.imshow(X[image_index].reshape(128, 128), cmap='gray');
```

```
Original Gender: Male Original Age: 42
Predicted Gender: Male Predicted Age: 38
```