Import Modules

Name:**Ajitkumar Vishwakarma Sharma**
Roll No:-**B212048**
Mini project

In [ ]:
```
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
```

Load the Dataset

In [ ]:
```
BASE_DIR = '/kaggle/input/utkface-new/UTKFace/'
```

In [ ]:
```
# 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]
```

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

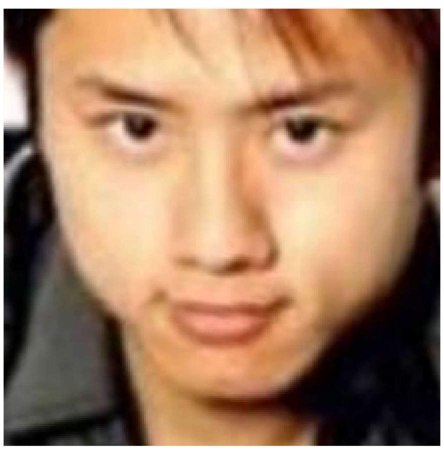
Out[4]:

	image	age	gender
0	/kaggle/input/utkface-new/UTKFace/26_0_2_20170...	26	0
1	/kaggle/input/utkface-new/UTKFace/22_1_1_20170...	22	1
2	/kaggle/input/utkface-new/UTKFace/21_1_3_20170...	21	1
3	/kaggle/input/utkface-new/UTKFace/28_0_0_20170...	28	0
4	/kaggle/input/utkface-new/UTKFace/17_1_4_20170...	17	1

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

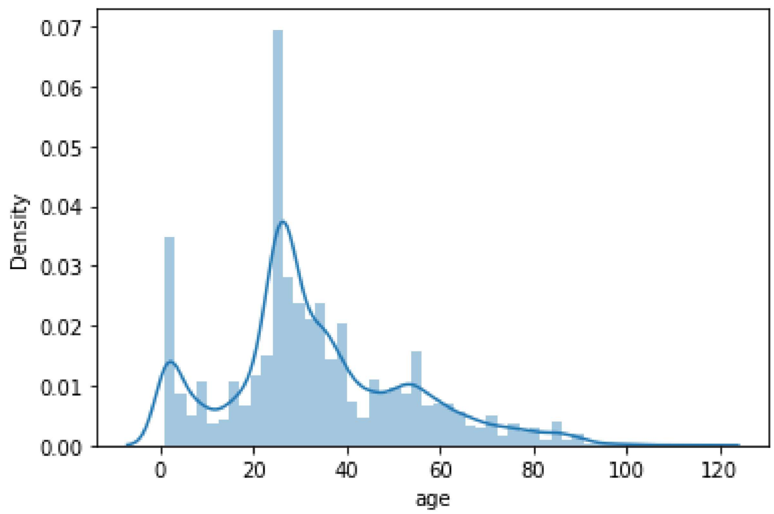
Exploratory Data Analysis

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



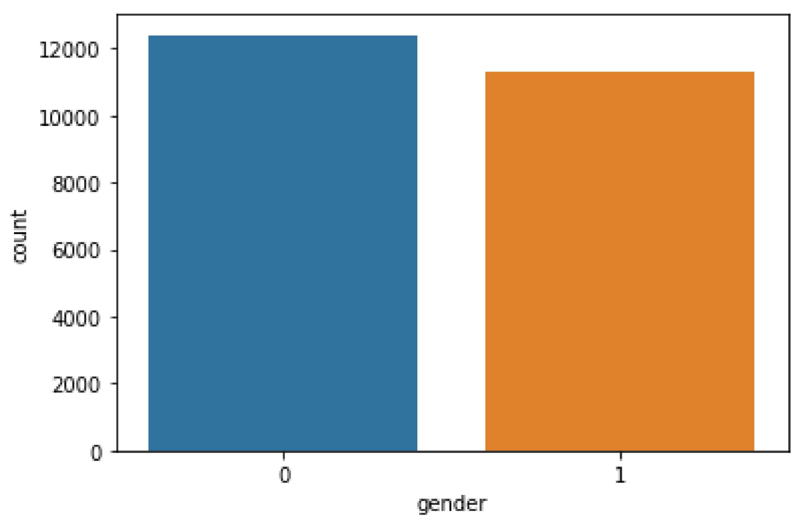
In [ ]:
```python
sns.distplot(df['age'])
```

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

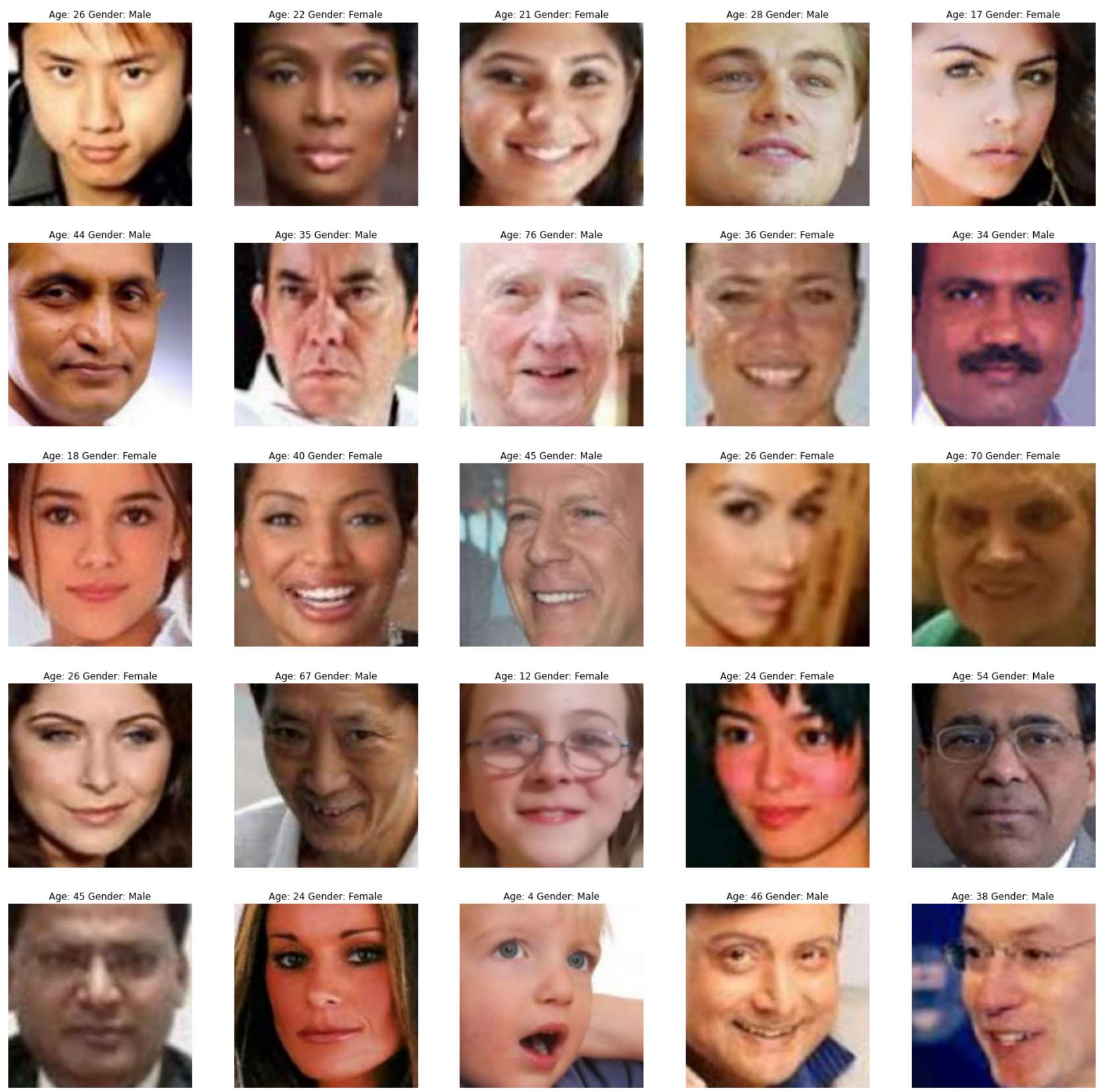


In [ ]:
```python
sns.countplot(df['gender'])
```

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



In [ ]:
```python
# to display grid of images
plt.figure(figsize=(25, 25))
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')
```



Feature Extraction

In [ ]:
```
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
```

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

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

In [ ]:
```
X.shape
```

Out[12]:
```
(23708, 128, 128, 1)
```

In [ ]:
```
X = X/255.0
```

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

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

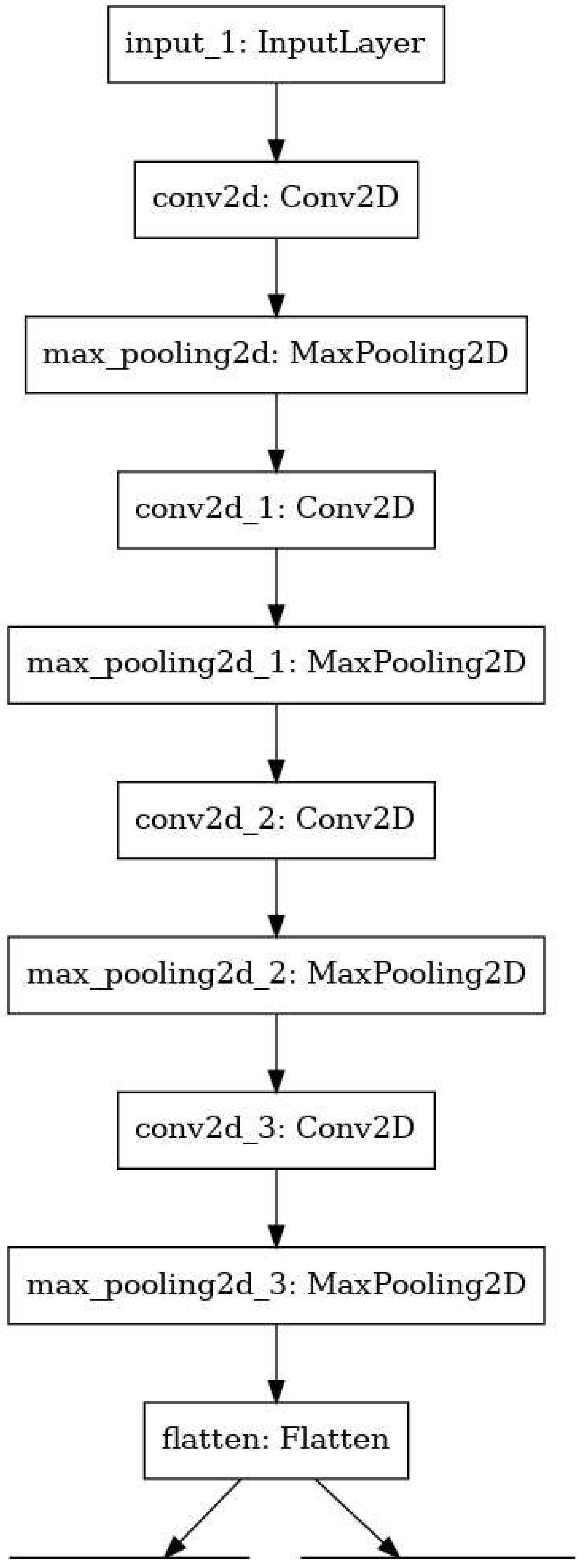
Model Creation

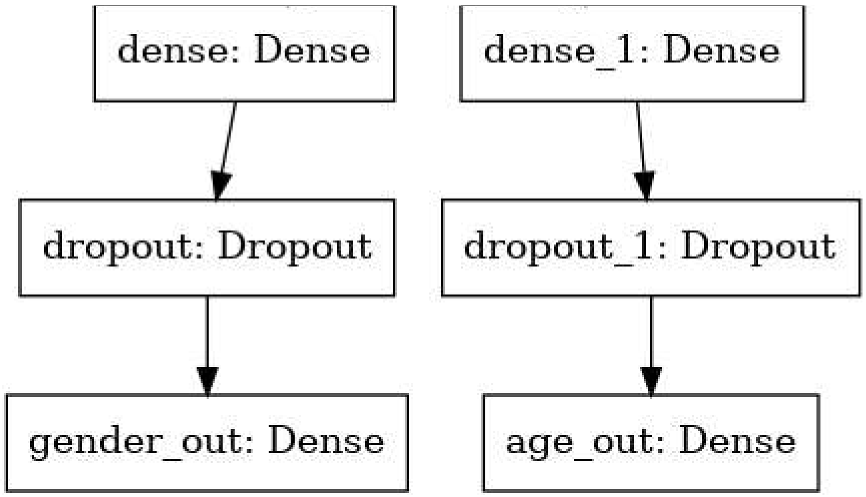
In [ ]:

```
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
```

In [ ]:

```
from tensorflow.keras.utils import plot_model
plot_model(model)
```


input_1: InputLayer
conv2d: Conv2D
max_pooling2d: MaxPooling2D
conv2d_1: Conv2D
max_pooling2d_1: MaxPooling2D
conv2d_2: Conv2D
max_pooling2d_2: MaxPooling2D
conv2d_3: Conv2D
max_pooling2d_3: MaxPooling2D
flatten: Flatten


dense: Dense
dense_1: Dense
dropout: Dropout
dropout_1: Dropout
gender_out: Dense
age_out: Dense

In [ ]:

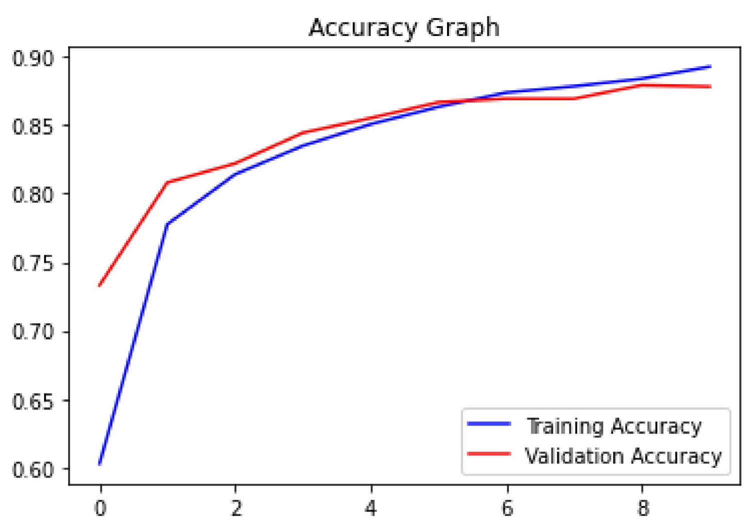
```
#train
history = model.fit(x=X , y=[y_gender, y_age], batch_size=32, epochs=10, valid
```

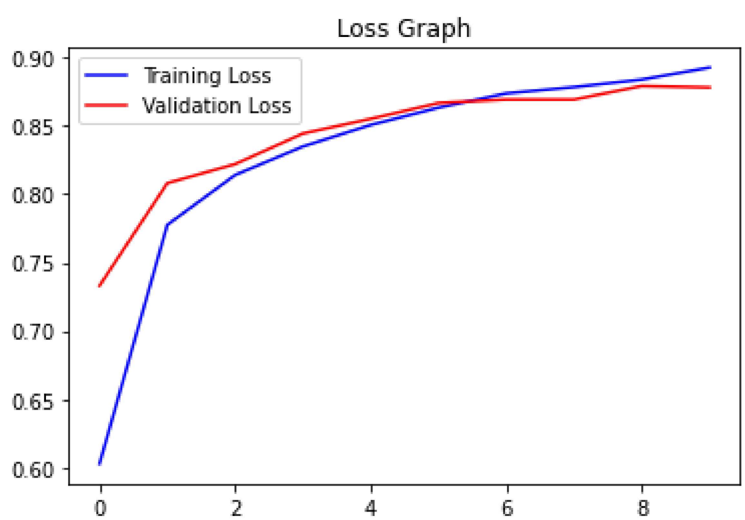
```
Epoch 1/10
593/593 [==============================] - 254s 426ms/step - loss: 15.2942 -
gender_out_loss: 0.6528 - age_out_loss: 14.6413 - gender_out_accuracy: 0.6028
- age_out_accuracy: 0.0462 - val_loss: 11.6634 - val_gender_out_loss: 0.5235
- val_age_out_loss: 11.1399 - val_gender_out_accuracy: 0.7330 - val_age_out_a
ccuracy: 0.0337
Epoch 2/10
593/593 [==============================] - 248s 419ms/step - loss: 11.1682 -
gender_out_loss: 0.4746 - age_out_loss: 10.6936 - gender_out_accuracy: 0.7775
- age_out_accuracy: 0.0252 - val_loss: 13.0804 - val_gender_out_loss: 0.4199
- val_age_out_loss: 12.6605 - val_gender_out_accuracy: 0.8081 - val_age_out_a
ccuracy: 0.0287
Epoch 3/10
593/593 [==============================] - 251s 423ms/step - loss: 9.6986 - g
ender_out_loss: 0.4069 - age_out_loss: 9.2917 - gender_out_accuracy: 0.8140 -
age_out_accuracy: 0.0143 - val_loss: 8.8840 - val_gender_out_loss: 0.3777 - v
al_age_out_loss: 8.5063 - val_gender_out_accuracy: 0.8220 - val_age_out_accur
acy: 0.0103
Epoch 4/10
593/593 [==============================] - 250s 422ms/step - loss: 8.6222 - g
ender_out_loss: 0.3601 - age_out_loss: 8.2622 - gender_out_accuracy: 0.8349 -
age_out_accuracy: 0.0111 - val_loss: 9.6180 - val_gender_out_loss: 0.3481 - v
al_age_out_loss: 9.2699 - val_gender_out_accuracy: 0.8444 - val_age_out_accur
acy: 0.0051
Epoch 5/10
593/593 [==============================] - 249s 420ms/step - loss: 8.1149 - g
ender_out_loss: 0.3300 - age_out_loss: 7.7849 - gender_out_accuracy: 0.8506 -
age_out_accuracy: 0.0110 - val_loss: 8.2791 - val_gender_out_loss: 0.3188 - v
al_age_out_loss: 7.9603 - val_gender_out_accuracy: 0.8551 - val_age_out_accur
acy: 0.0055
Epoch 6/10
593/593 [==============================] - 248s 418ms/step - loss: 7.6692 - g
ender_out_loss: 0.3035 - age_out_loss: 7.3657 - gender_out_accuracy: 0.8633 -
age_out_accuracy: 0.0106 - val_loss: 7.4283 - val_gender_out_loss: 0.3007 - v
al_age_out_loss: 7.1276 - val_gender_out_accuracy: 0.8667 - val_age_out_accur
acy: 0.0063
Epoch 7/10
593/593 [==============================] - 246s 415ms/step - loss: 7.2979 - g
ender_out_loss: 0.2845 - age_out_loss: 7.0134 - gender_out_accuracy: 0.8738 -
age_out_accuracy: 0.0105 - val_loss: 8.4701 - val_gender_out_loss: 0.3086 - v
al_age_out_loss: 8.1615 - val_gender_out_accuracy: 0.8693 - val_age_out_accur
acy: 0.0049
Epoch 8/10
593/593 [==============================] - 245s 414ms/step - loss: 7.0293 - g
ender_out_loss: 0.2719 - age_out_loss: 6.7574 - gender_out_accuracy: 0.8784 -
age_out_accuracy: 0.0105 - val_loss: 8.5517 - val_gender_out_loss: 0.2939 - v
al_age_out_loss: 8.2578 - val_gender_out_accuracy: 0.8693 - val_age_out_accur
acy: 0.0038
Epoch 9/10
593/593 [==============================] - 246s 414ms/step - loss: 6.8039 - g
ender_out_loss: 0.2583 - age_out_loss: 6.5456 - gender_out_accuracy: 0.8839 -
age_out_accuracy: 0.0095 - val_loss: 7.1854 - val_gender_out_loss: 0.2793 - v
al_age_out_loss: 6.9061 - val_gender_out_accuracy: 0.8792 - val_age_out_accur
acy: 0.0051
Epoch 10/10
593/593 [==============================] - 247s 417ms/step - loss: 6.3909 - g
ender_out_loss: 0.2481 - age_out_loss: 6.1428 - gender_out_accuracy: 0.8927 -
```

```
age_out_accuracy: 0.0094 - val_loss: 7.1050 - val_gender_out_loss: 0.2750 - v
al_age_out_loss: 6.8300 - val_gender_out_accuracy: 0.8781 - val_age_out_accur
acy: 0.0065
```

Plot the Results

In [ ]:
```
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_accuracy']
val_acc = history.history['val_gender_out_accuracy']


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



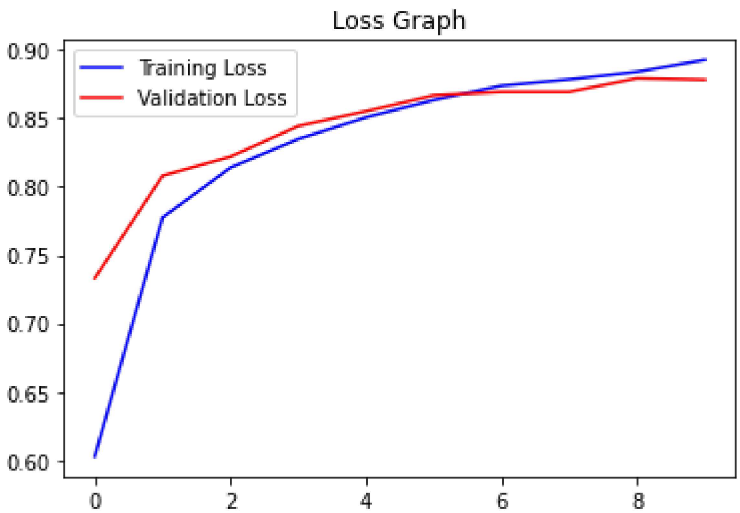


In [ ]:

```
# plot results for age
loss = history.history['age_out_loss']
val_loss = history.history['val_age_out_loss']
epochs = range(len(acc))

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

Out[20]:

```
<Figure size 432x288 with 0 Axes>
```




```
<Figure size 432x288 with 0 Axes>
```

Prediction with Test Data

In [ ]:
```
image_index = 1502
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",

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: 30
Predicted Gender: Male Predicted Age: 26
```

In [ ]:

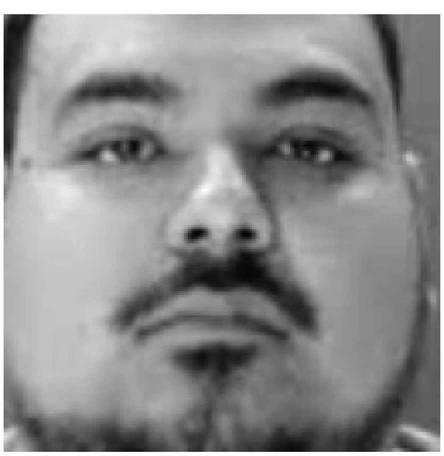
```
image_index = 2942
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",

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: 26
Predicted Gender: Female Predicted Age: 28
```

In [ ]:

```
image_index = 9967
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",

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: 25
```



In [ ]:

```
image_index = 4523
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",

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: 30
Predicted Gender: Female Predicted Age: 29
```

In [ ]:

```
image_index = 23500
print("Original Gender:", gender_dict[y_gender[image_index]], "Original Age:",

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: 23
Predicted Gender: Male Predicted Age: 26
```