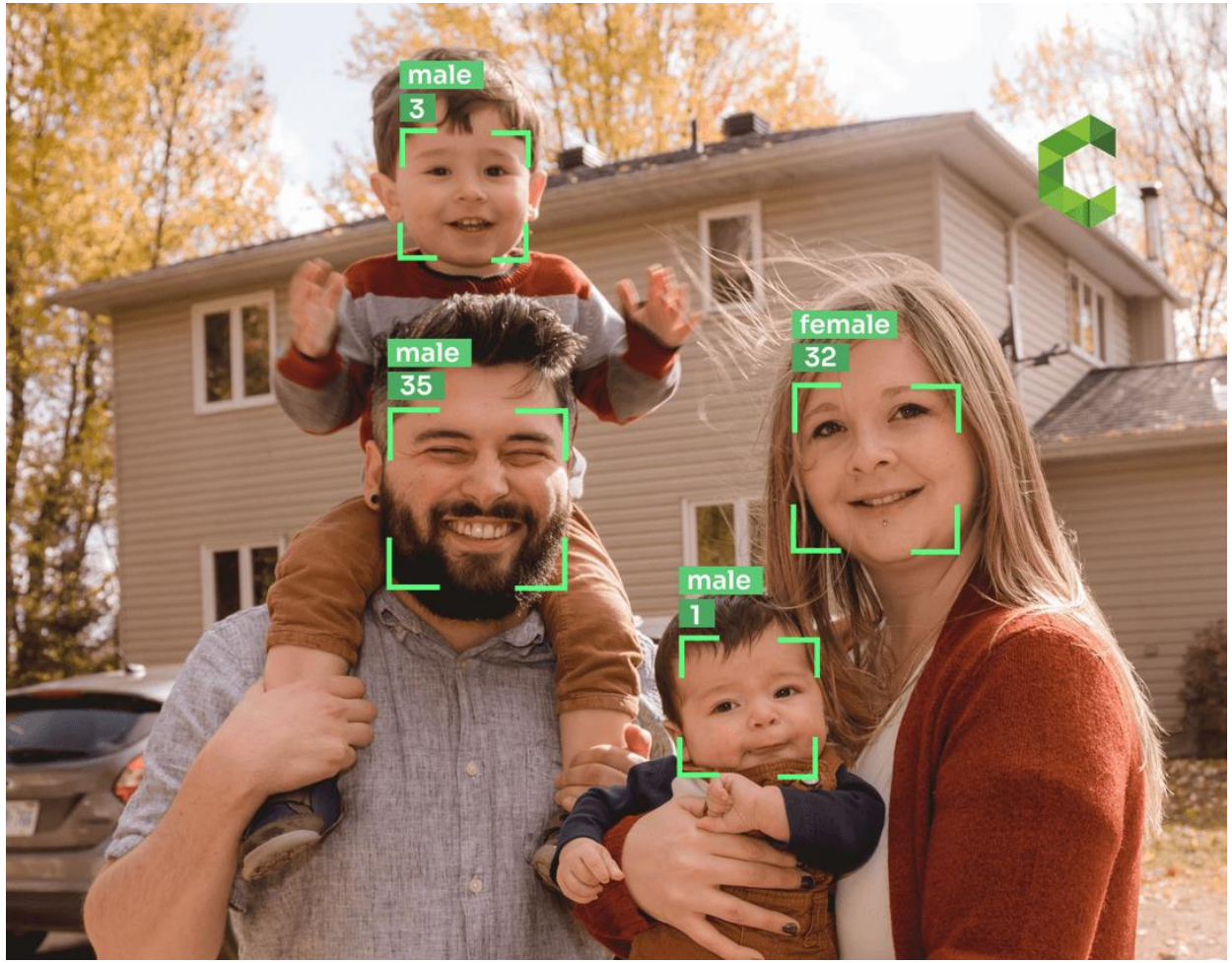


Report

Gender and Age Prediction - CNN

Liang Gong

May 2024

https://github.com/gongl1/age_gender_prediction_cnn_gong

Problem Statement

In dating app settings, discrepancies in self-reported gender/age and actual gender/age can occur. I aim to address this issue by developing a predictive model that utilizes verified profile pictures to identify such discrepancies. By employing **machine learning techniques for gender and age prediction based on verified profile images**, I aim to **flag instances where reported information diverges from predictions**. This effort aims to enhance profile accuracy and user trust within the platform

I will create an Age and Gender Prediction model using Keras Functional API, which will perform both Regression to predict the Age of the person and Classification to predict the Gender from face of the person.

Age and Gender Prediction
Keras Functional API offers a flexible way to make models of higher complexity. I unravel a scenario to create a model that can perform **Regression** and **Classification** predictions.

Dataset
I will use the Age, Gender (Face Data) CSV dataset for this purpose and to achieve this I will use a Convolutional Neural Network (CNN). CNN is a powerful tool in Deep Learning that helps the user classify an Image and has the most usage in Computer Vision. It enables the Machine to visualize and interpret Images and Image data.

One limitation is that dataset only has 23705 rows.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 23705 entries, 0 to 23704
Data columns (total 5 columns):
 #   Column     Non-Null Count  Dtype 
---  ------     --------------  ----- 
 0   age        23705 non-null  int64 
 1   ethnicity  23705 non-null  int64 
 2   gender     23705 non-null  int64 
 3   img_name   23705 non-null  object
 4   pixels     23705 non-null  object
dtypes: int64(3), object(2)
memory usage: 926.1+ KB
```

Assumptions/Hypotheses about data and model

1.Data Representation:
1. Assumption: The dataset represents diverse demographics in terms of age, ethnicity, and gender.
2. Hypothesis: Pixel values contain relevant information for age and gender prediction.

2.Model Complexity:
1. Assumption: The model complexity is sufficient to capture intricate relationships in the data.
2. Hypothesis: Deep learning models, like CNNs, are effective for learning complex patterns in images.

3.Generalization:
1. Assumption: The trained model generalizes well to unseen data.
2. Hypothesis: Regularization techniques prevent overfitting and enhance generalization.

4.Evaluation Metrics:
1. Assumption: Evaluation metrics align with task goals and dataset characteristics.
2. Hypothesis: Metrics like accuracy, MSE are suitable for assessing model performance.

5.Ethnicity Consideration:
1. Assumption: Ethnicity may influence facial features, impacting prediction accuracy.
2. Hypothesis: Certain ethnic groups exhibit distinct facial characteristics affecting model performance.

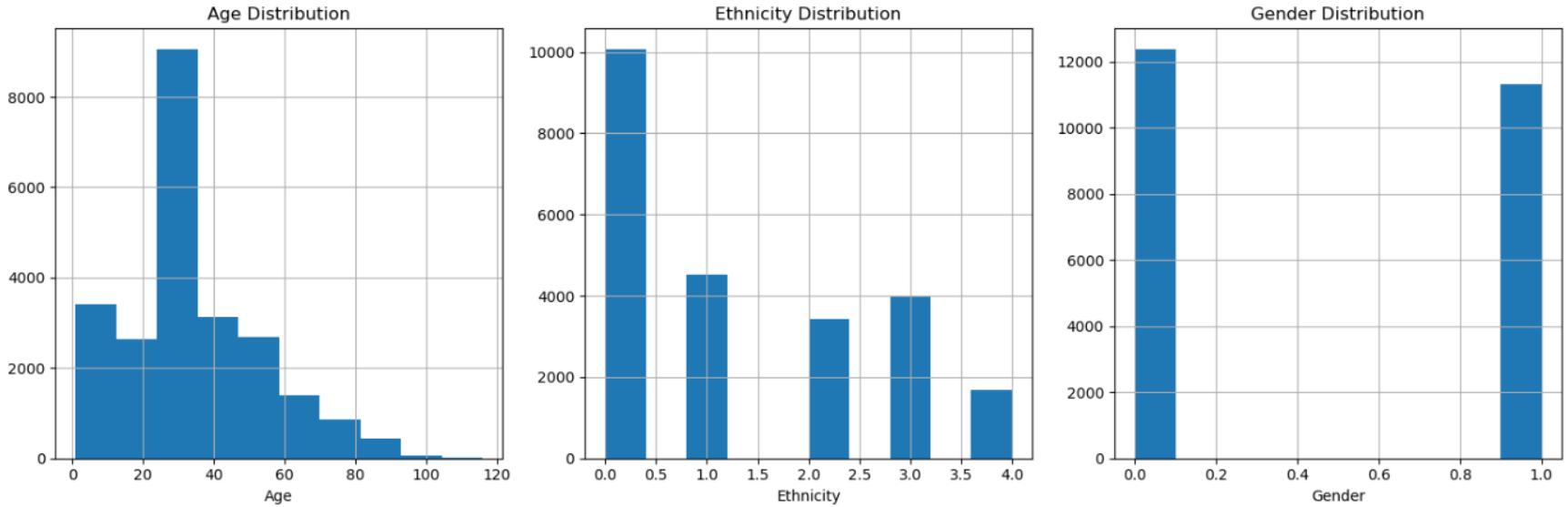
6.Data Preprocessing:
1. Assumption: Proper preprocessing ensures dataset quality and consistency.
2. Hypothesis: Techniques such as resizing, normalization, and augmentation improve model robustness.

1.Independence and Identically Distributed (i.i.d.) Data:
1. Assumption: The samples in the dataset are independent and drawn from the same probability distribution.
2. Hypothesis: Each image and its corresponding label are independent of other images and labels, and they are all drawn from the same underlying distribution.

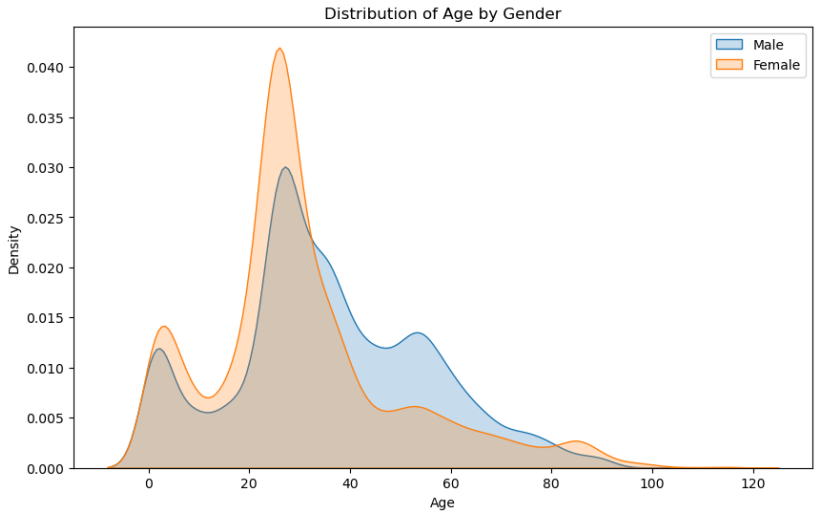
2.Same Distribution Between Train and Test Data:
1. Assumption: The distribution of the training data is representative of the distribution of the test data.
2. Hypothesis: The images and labels in the test set come from the same distribution as those in the training set.
   This ensures that the model trained on the training data generalizes well to unseen data.

Exploratory Data Analysis

Visualize distributions for Age, Ethnicity and Gender



Visualize distributions of Age by Gender



Feature Engineering & Transformations

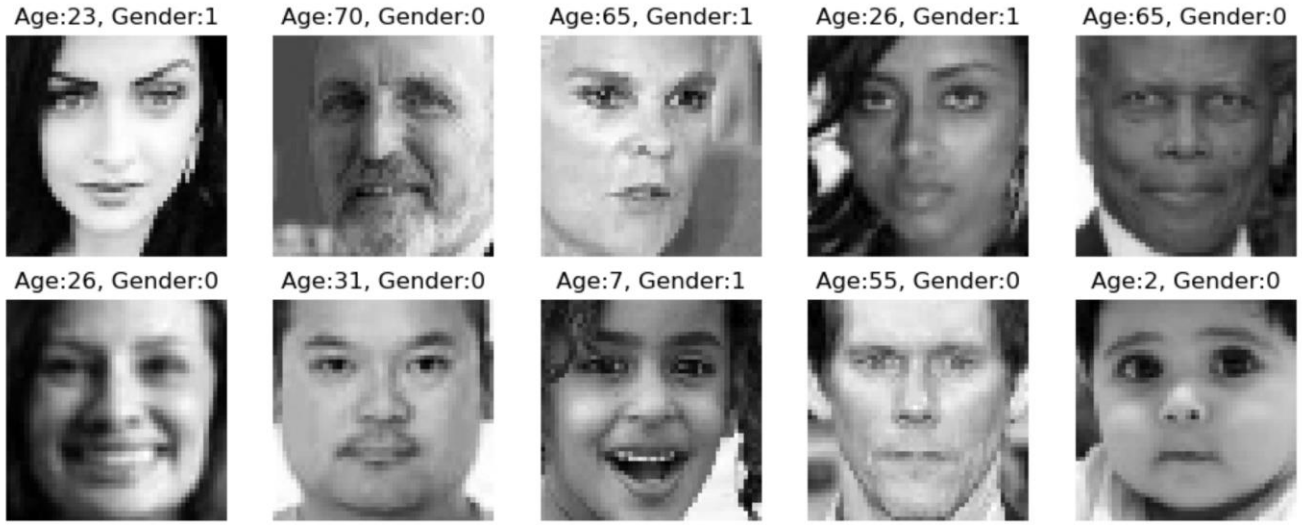
Based on the “pixels” column (object type), create a new column “new_pixels” , each list contains the pixels of that image.

	age	ethnicity	gender	img_name	pixels	new_pixels
0	1	2	0	20161219203650636.jpg.chip.jpg	129 128 128 126 127 130 133 135 139 142 145 14...	[129, 128, 128, 126, 127, 130, 133, 135, 139, ...
1	1	2	0	20161219222752047.jpg.chip.jpg	164 74 111 168 169 171 175 182 184 188 193 199...	[164, 74, 111, 168, 169, 171, 175, 182, 184, 1...
2	1	2	0	20161219222832191.jpg.chip.jpg	67 70 71 70 69 67 70 79 90 103 116 132 145 155...	[67, 70, 71, 70, 69, 67, 70, 79, 90, 103, 116,...
3	1	2	0	20161220144911423.jpg.chip.jpg	193 197 198 200 199 200 202 203 204 205 208 21...	[193, 197, 198, 200, 199, 200, 202, 203, 204, ...
4	1	2	0	20161220144914327.jpg.chip.jpg	202 205 209 210 209 209 210 211 212 214 218 21...	[202, 205, 209, 210, 209, 209, 210, 211, 212, ...

df.shape

(23705, 6)

Generate a list of random indices value from the dataset and using that to plot a subplot of images. To plot the image, first reshape the data into a (48,48,1) shape.



Check the input Face images

Proposed Approaches (Model) with checks for overfitting/underfitting

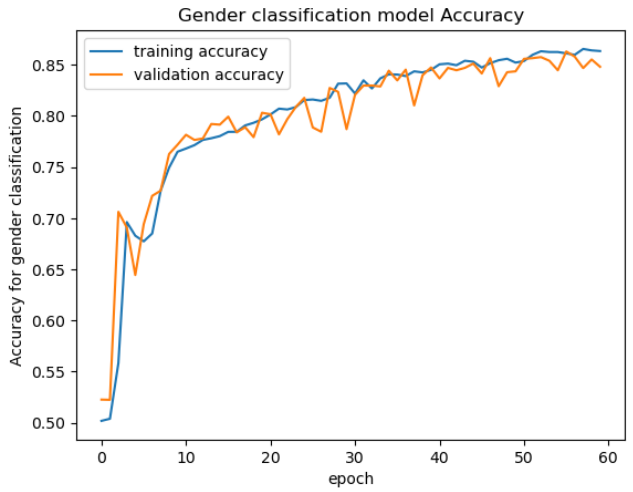
Layer (type)	Output Shape	Param #	Connected to
Input image (InputLayer)	(None, 48, 48, 1)	0	-
conv2d (Conv2D)	(None, 46, 46, 16)	160	Input image[0][0]
conv2d_1 (Conv2D)	(None, 44, 44, 32)	4,640	conv2d[0][0]
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0	conv2d_1[0][0]
conv2d_2 (Conv2D)	(None, 12, 12, 64)	18,496	max_pooling2d[0][0]
conv2d_3 (Conv2D)	(None, 10, 10, 64)	36,928	conv2d_2[0][0]
flatten (Flatten)	(None, 6400)	0	conv2d_3[0][0]
dense (Dense)	(None, 128)	819,328	flatten[0][0]
dense_1 (Dense)	(None, 32)	4,128	dense[0][0]
g_clf (Dense)	(None, 1)	33	dense_1[0][0]
a_reg (Dense)	(None, 1)	33	dense_1[0][0]

Total params: 883,746 (3.37 MB)
Trainable params: 883,746 (3.37 MB)
Non-trainable params: 0 (0.00 B)

- Proposed final model shown on left chart
- Adam optimizer with learning_rate=0.003
- EarlyStopping with patience=25
- model.evaluate shown below

```
model.evaluate(X_val, [y_clf_val, y_reg_val])
```

```
149/149 ━━━━━━━━━━━━━━━━━━━━ 1s 4ms/step - a_reg_mse: 88.7488 - g_clf_accuracy: 0.8420 - loss: 89.0890
[87.8670654296875, 87.53805541992188, 0.8479223847389221]
```



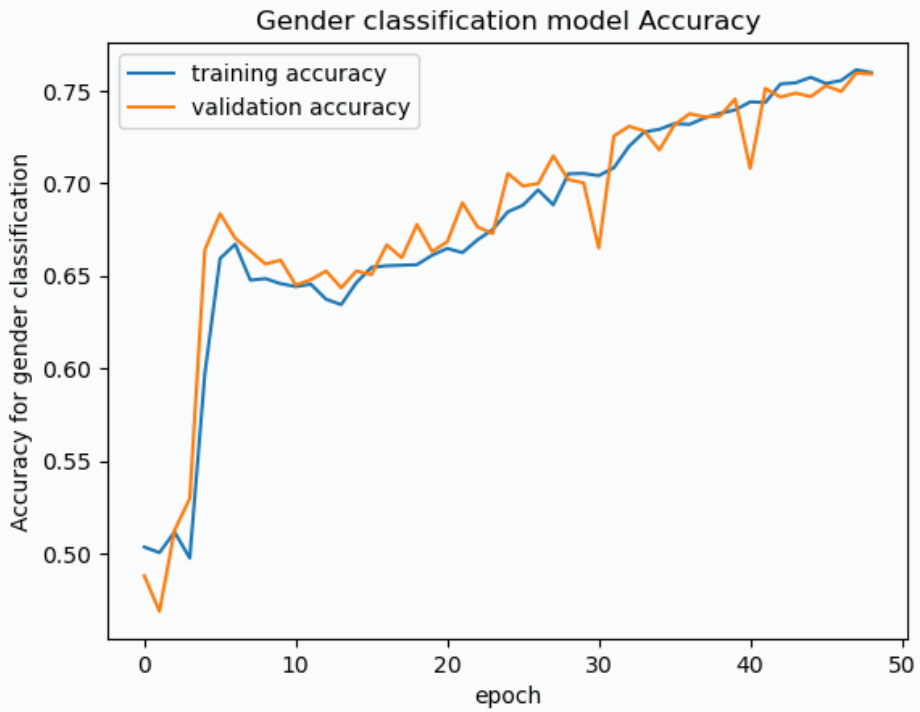
- Model performs well on gender classification

- Model shows overfitting on age regression
- Could stop around epoch 20

Proposed Solution (Model Selection) with regularization, if needed

With regularization: **model does not improve much**

x = layers.Dense(128, activation='relu', kernel_regularizer=tf.keras.regularizers.l2())(x)



Proposed Solution (Model Selection) with regularization, if needed

With RMSprop optimizer rather than Adam optimizer: **higher loss on validation set**
tf.keras.optimizers.RMSprop(learning_rate=0.003)

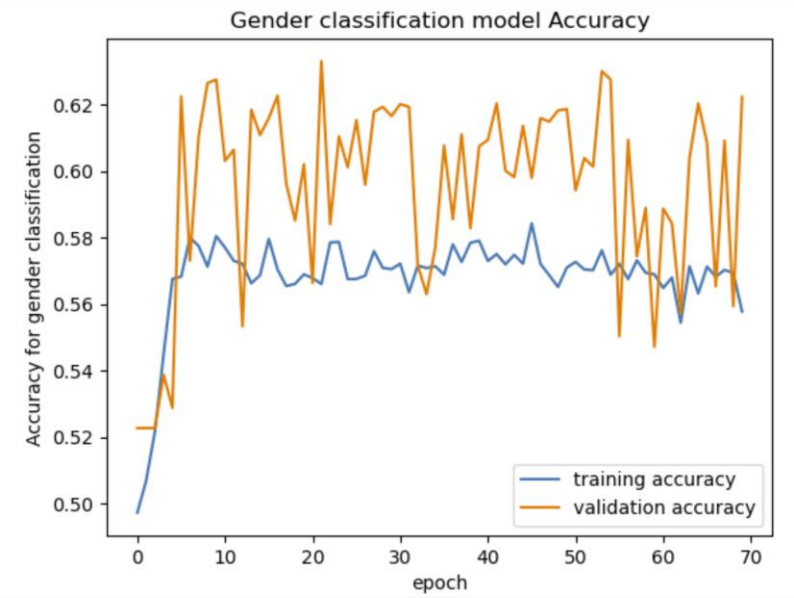
```
model.evaluate(X_val, [y_clf_val, y_reg_val])
```

```
149/149 ━━━━━━━━━━━━━━━━━━━━ 4s 25ms/step - a_reg_mse: 99.1097 - g_clf_accuracy: 0.7709 - loss: 99.6468
```

With added dropout layer: **overfitting improved much better for age regression but lower accuracy on gender classification**
x = Dropout(0.5)(x)

```
model.evaluate(X_val, [y_clf_val, y_reg_val])
```

```
149/149 ━━━━━━━━━━━━━━━━━━━━ 1s 7ms/step - a_reg_mse: 100.6960 - g_clf_accuracy: 0.6161 - loss: 101.3684
```



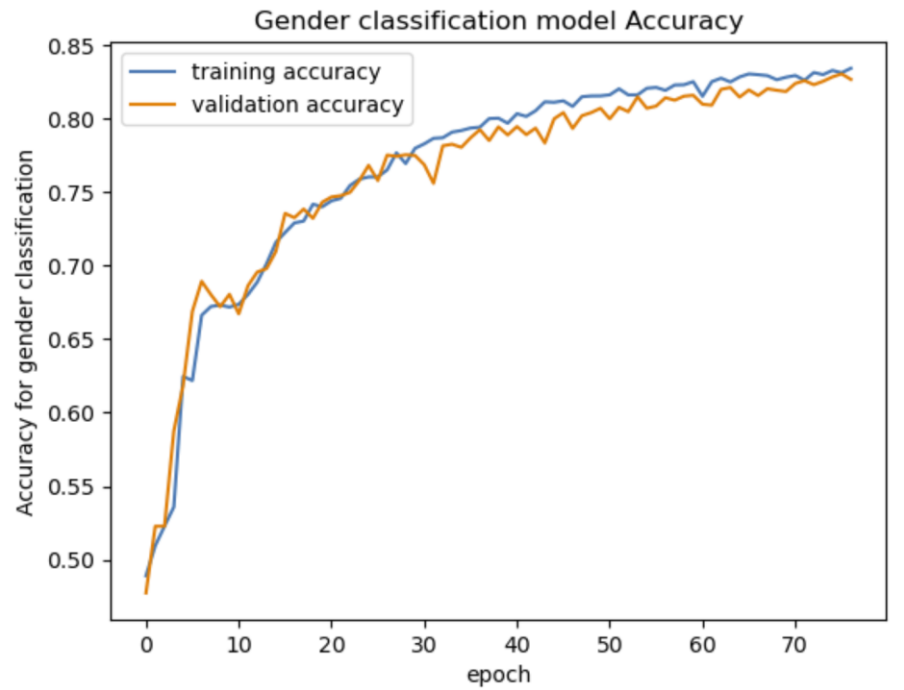
Proposed Solution (Model Selection) with regularization, if needed

With reducing the number of parameters (decreasing the number of filters in the convolutional layers and reducing the number of units in the dense layers) : **worse overfitting on age regression, but can consider stop around 20 epochs**

x = layers.Conv2D(8, 3, activation="relu")(input_layer)

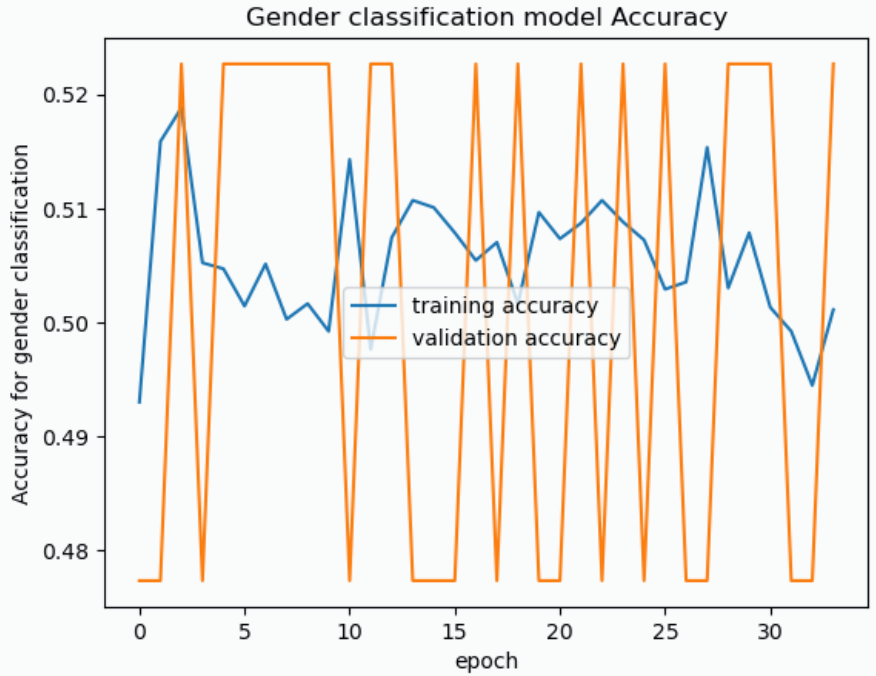
```
model.evaluate(X_val, [y_clf_val, y_reg_val])
```

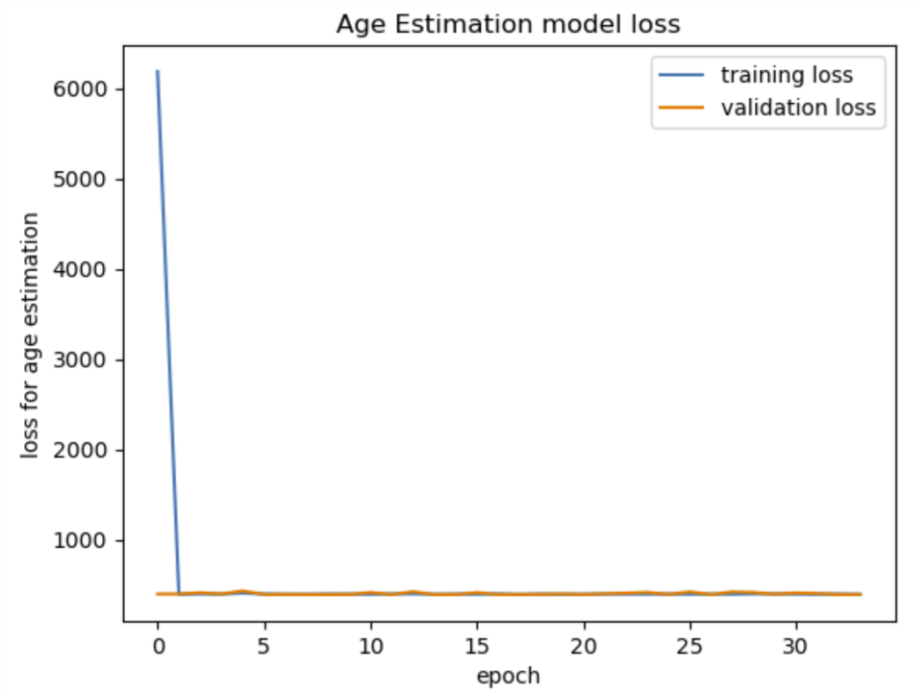
```
149/149 ━━━━━━━━━━━━━━━━━━━━ 0s 3ms/step - a_reg_mse: 117.2380 - g_clf_accuracy: 0.8220 - loss: 117.6223
```



Proposed Solution (Model Selection) with regularization, if needed

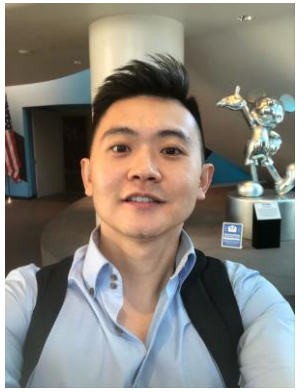
With higher learning rate: **model can not converge properly on gender classification**
optimizer = tf.keras.optimizers.Adam(learning_rate=0.015)





Results (Accuracy) and Learnings from the methodology

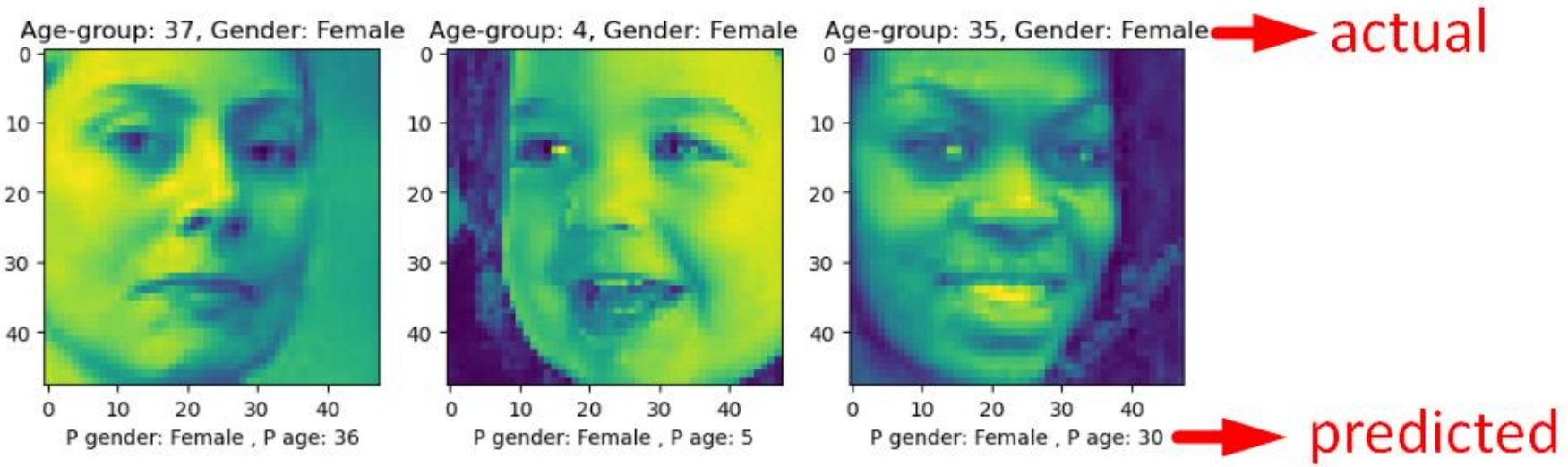
- Adam optimizer with small learning_rate=0.003
- Stop at Epoch 20
- Started with very simple model as baseline and tried adding/adjusting different hyperparameters
- Would be easier to do hyperparameter tuning when setting up 2 separate models for gender classification and age regression
- For the 2 external images, first crop to only face area like the training images and then import to the model for prediction



Predicted Gender: Male
Predicted Age: 17

Predicted Gender: Male
Predicted Age: 26

g_clf_accuracy: 0.84
loss: 83



Future Work

Race factor: Incorporate race as a in the model to predict age.

Data Augmentation: Explore advanced data augmentation techniques such as rotation, translation, scaling, and flipping to increase the diversity of the training dataset and enhance model generalization.

Retraining the Model: Consider using techniques such as transfer learning, where initializing the model with weights from the previous training and fine-tune it on the new dataset.

Real-time Prediction: Develop the capability for real-time age and gender prediction directly within the dating app, enabling immediate feedback to users during profile creation or verification processes.

Privacy and Ethical Considerations: Investigate privacy-preserving techniques to ensure that user data, especially facial images, is handled securely and ethically. This includes anonymization, data encryption, and adherence to privacy regulations.

Deployment and Scalability: Deploy the model into a production environment, ensuring scalability and reliability to handle large volumes of user requests efficiently.

Thank You