Project Documentation: Face-Based Age Estimation Using Regression

GR. 5

1. Project Overview

This project implements a deep learning solution to predict the exact age of a person based on facial images. It treats the age estimation task as a regression problem, predicting a continuous age value using a Convolutional Neural Network (CNN), specifically the ResNet-18 architecture. The solution employs Mean Squared Error (MSE) loss during training and evaluates performance using Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). The dataset used is UTKFace, it contains 23 707 images of faces with annotated ages.

Some example images with age (after preprocessing):

2. Project Structure and File Descriptions

The project is structured into clearly defined files:

Directories:

- data/raw: Contains original, unprocessed images.
- data/processed: Contains processed (resized and grayscale) images used for training.
- outputs/models: Stores the best-performing model checkpoints.
- plots: Stores generated plots for visual analysis.

Main Scripts:

- **train.py**:
  - Trains the ResNet regression model.
  - Implements data loading, model training loop, validation checks, and saves the best model based on validation loss.
  - Produces training and validation loss plots.
- **evaluate.py**:
  - Evaluates the trained model's performance on the validation dataset.
  - Generates detailed performance metrics (MAE, RMSE).
  - Produces multiple visualizations to analyze model predictions, residuals, and errors.

Supporting Modules:

- **data_loader.py**:
  - Defines AgeRegressionDataset class for loading and transforming images.
  - Provides a utility function get_dataloaders() to return PyTorch DataLoader objects for training and validation.
- **models/resnet.py**:
  - Implements a customized ResNet-18 and ResNet-50 CNN model tailored for regression.
  - Modifies the final fully connected layer for single continuous output.
- **utils.py**:
  - Includes various utility functions for setting random seed, plotting visualizations (e.g., predictions vs. actual, error distributions, loss curves), and analysis.
- **config.py**:

    - Stores configuration parameters such as directory paths, hyperparameters (e.g., epochs, batch size, learning rate), and random seed.

3. Detailed Instructions for Running the Code

Step 1: Data Preparation

- Ensure raw images are placed in data/raw.
- Run the preprocessing script to convert and resize images:

```
python preprocess.py
```

Step 2: Train the Model

- Execute the training script:

```
python train.py
```

- Model checkpoints are automatically saved in outputs/models.
- Training loss plots are saved in the plots directory.

Step 3: Evaluate the Model

- After training completes, evaluate the model using:

```
python evaluate.py
```

- Evaluation script prints performance metrics (MAE and RMSE) and saves detailed analysis plots in the plots directory.
- Alternatively the train and evaluate scripts can be combined by calling main.py

4. Results and Analysis

We tested both the ResNet-18 and ResNet-50 models on a validation set of 4,740 face images. Below are the key results:

ResNet-18

- Mean Absolute Error (MAE): **6.09**
- Root Mean Squared Error (RMSE): **8.82**

ResNet-50

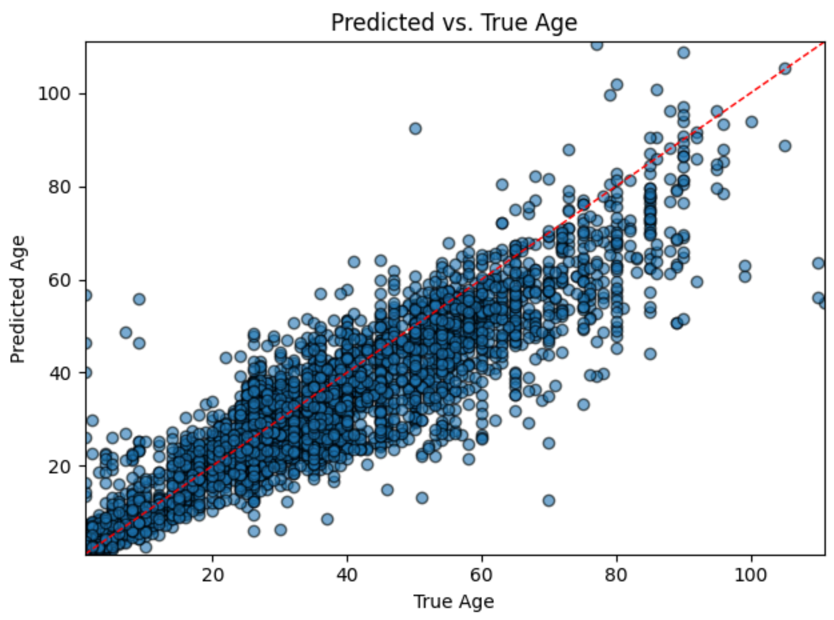
- Mean Absolute Error (MAE): **5.34**
- Root Mean Squared Error (RMSE): **7.62**

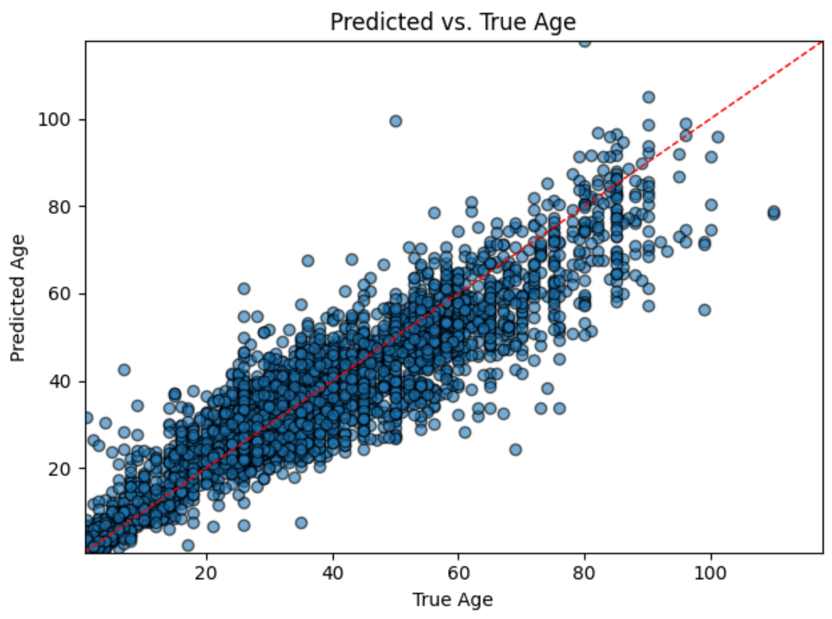
These numbers show that ResNet-50 outperformed ResNet-18, making more accurate predictions on average. While both models give reasonable results, ResNet-50’s deeper architecture seems better at picking up on visual features related to age.

Visualizations

- **Predicted vs. Actual Ages** (plots/pred_vs_true.png):
    - Illustrates the model's prediction accuracy by comparing predicted ages against true ages.

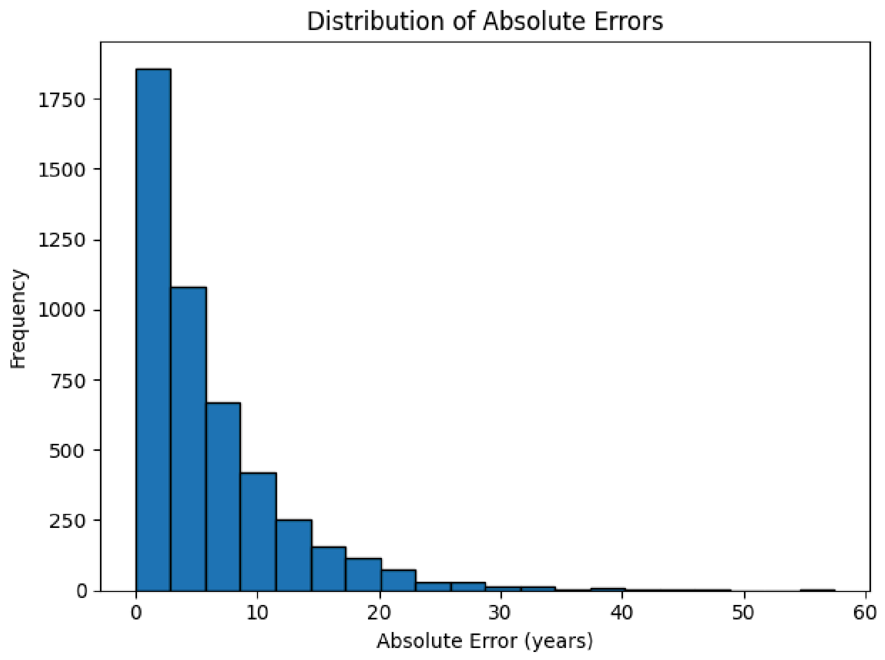
Resnet18 vs Resnet50:

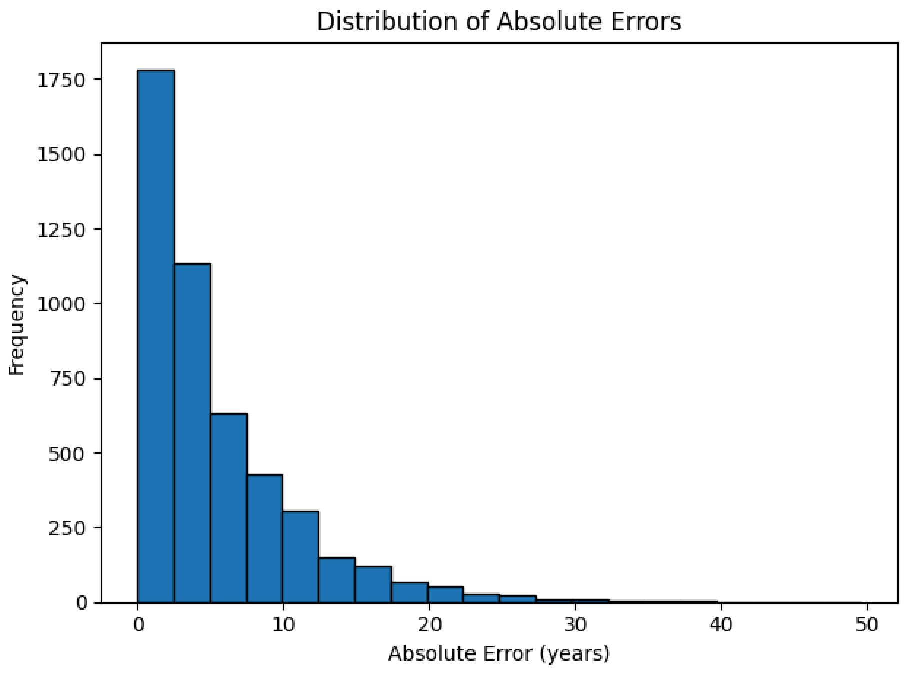




- **Error Distribution** (plots/error_dist.png):
    - Provides insight into the frequency of absolute prediction errors.
    - We can observe that most of the errors are within 0-3 years.

Resnet18 vs Resnet50:

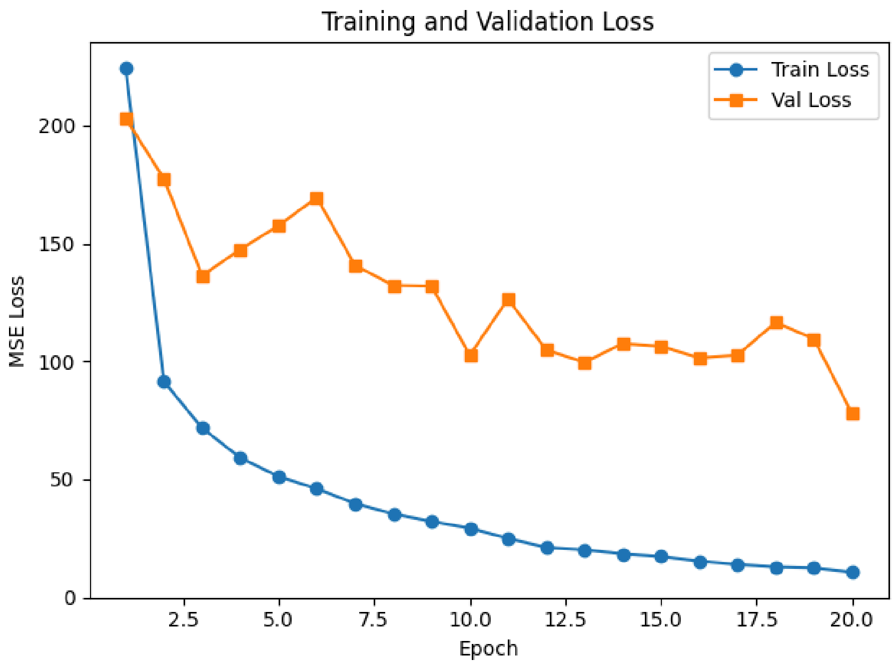


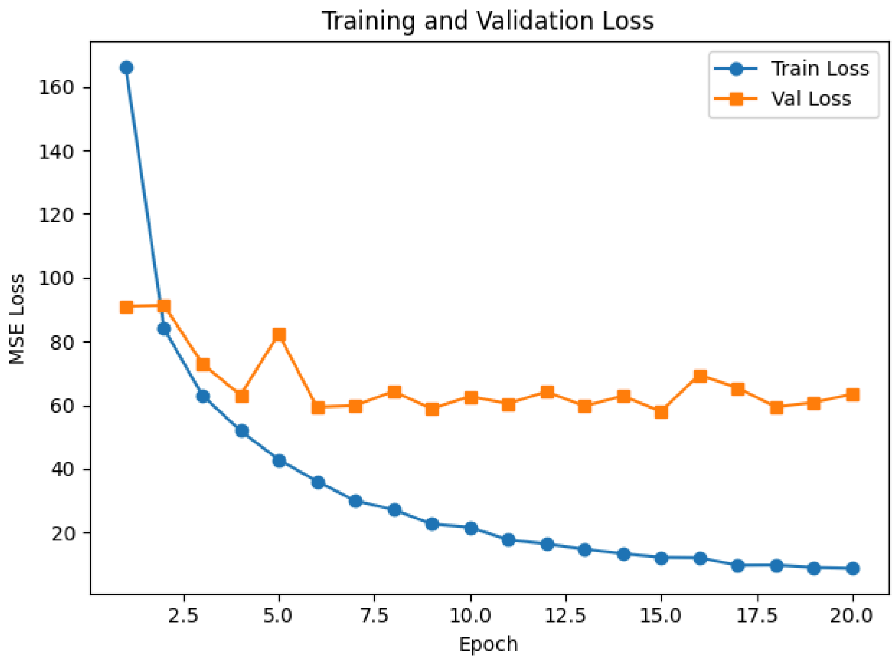


- **Training and Validation Loss Curve** (plots/loss_curve.png):
  - Shows the progression of the model's learning and potential overfitting patterns
  - We can observe on the Val loss that the model steadily learns, although comparing 2 plots the difference between train and val loss may suggest overfitting

- ○ It can be easily observed that the ResNet 50 converges much quicker due to more hidden layers.

Resnet18 vs Resnet50:

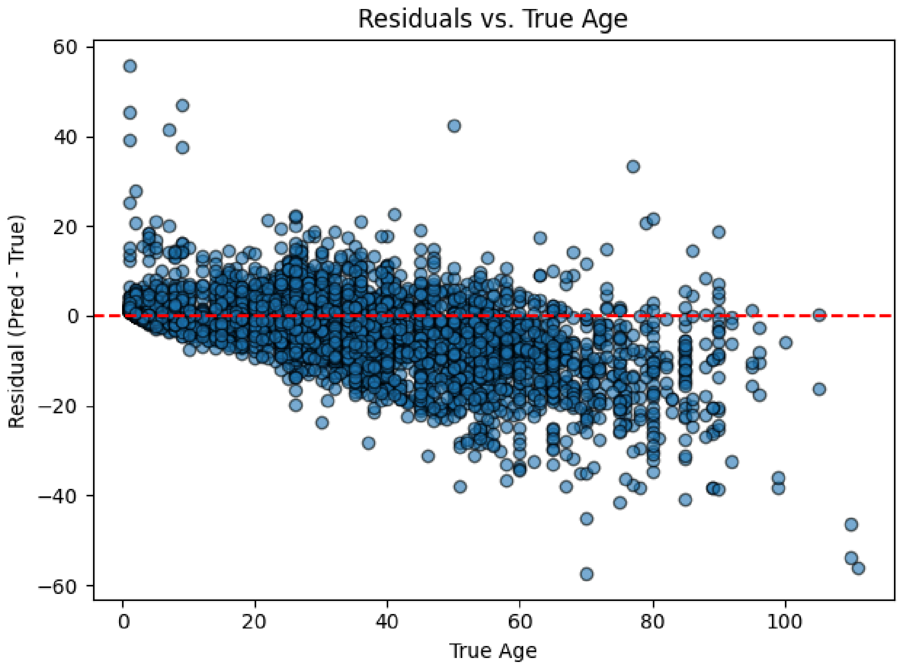


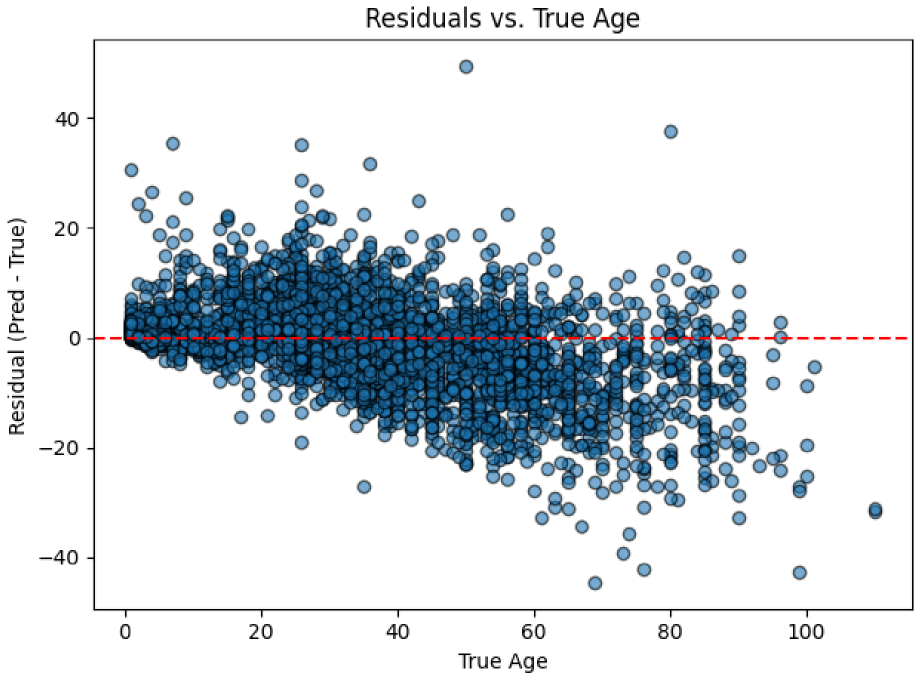


- **Residuals Analysis** (plots/residuals_vs_true.png):

- Examines prediction residuals (errors) against true age values to identify biases or trends.
- It can be observed that the higher the true age, the predictions stay low, it may be because of the unbalanced dataset

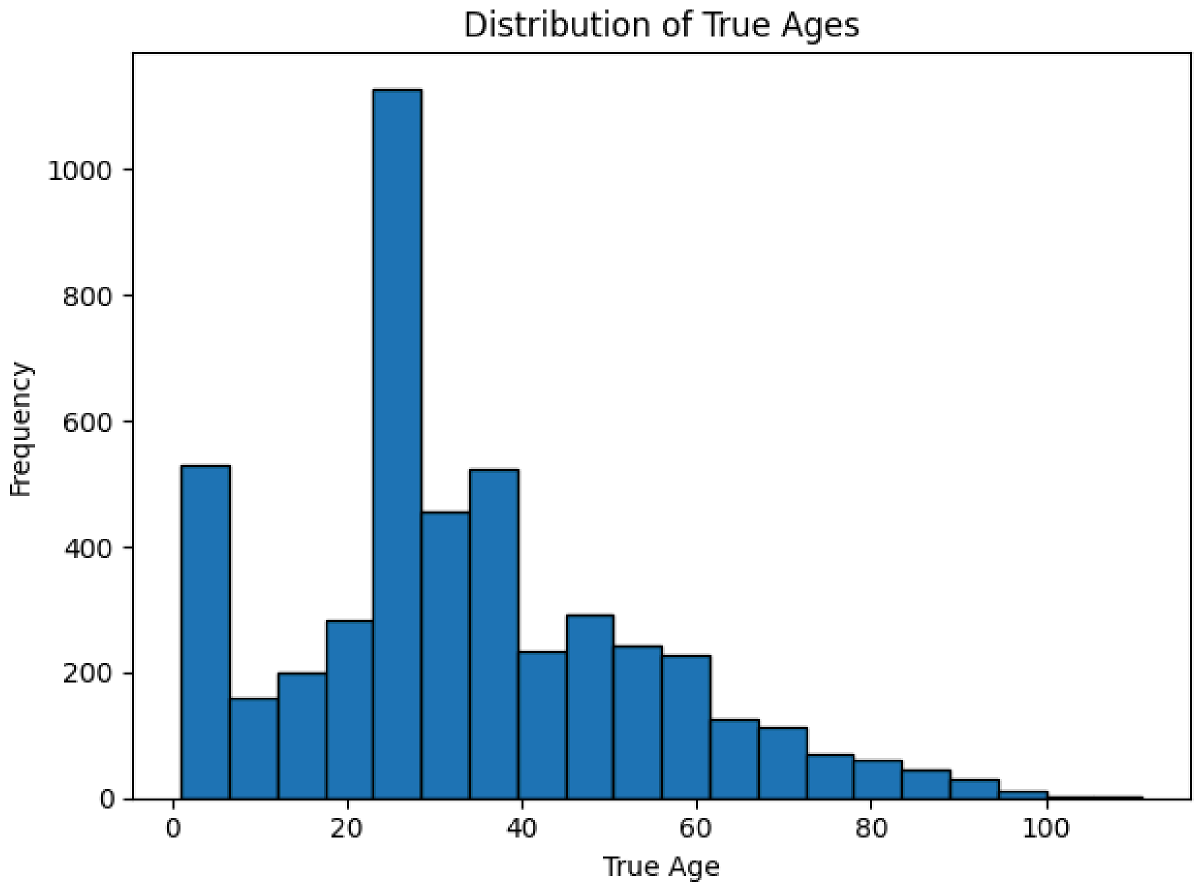
Resnet18 vs Resnet50:



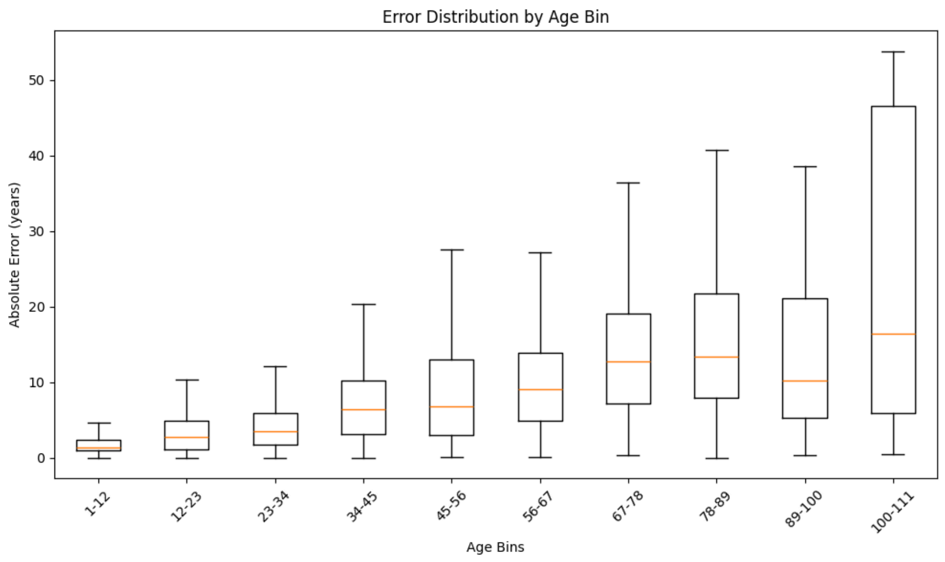


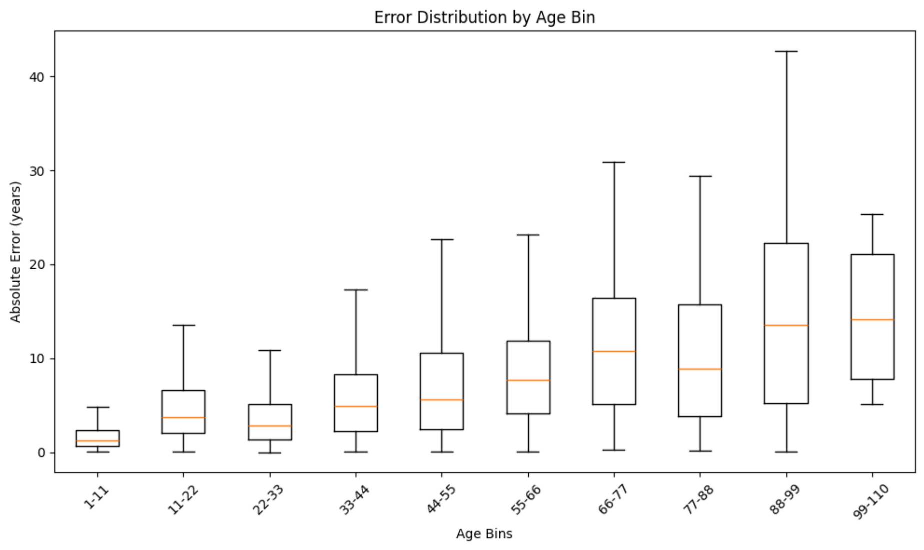
- **True Age Distribution** (plots/true_age_dist.png):
    - Highlights age distribution within the validation set.
    - On this plot we can see the distribution which suggest very unbalanced dataset in terms of ages

    - The same for both models (same seed)



- **Error by Age Bins** (plots/error_by_age_bin.png):
    - Analyzes prediction accuracy across different age groups, highlighting specific areas of strength or weakness.
    - Easy to observe that the higher this plot has a similar but reversed pattern to the previous plot, the less samples we get in training, the larger the error is.
    - It can be observed in ResNet 50 the error in higher ages is not that large, even though the number of samples from this range did not change. It may be because due to more layers, it is able to find more complex patterns in images. It may also be random, because at that age (>90) there are just a few images.





5. Challenges and Solutions

- **Filename Parsing and Data Loading**:
    - Challenge: The annotations were inside the images names instead of csv, which was new to me
    - Solution: Implemented text splitting for extracting and assigning the age to each image

- **Finding an environment**
    - Challenge: Find an environment with enough computing power to run the training

    - Solution: Used google colab with bought credits for GPU usage

6. Conclusion

The age prediction models we built, especially ResNet-50, do a good job guessing a person's age just from a photo. The steps we followed—like cleaning the data, training the models, checking the results, and creating clear visualizations—make the whole process easy to understand, repeat, and improve.

ResNet-18 gave a Mean Absolute Error (MAE) of **6.09 years**, while ResNet-50 did better with an MAE of **5.34 years**. That's very close to how accurate humans are research shows people are typically off by about **5.15 years** when guessing someone's age from a face photo ([link to study]). So even though there's still room to improve, these models are already working at a level similar to human judgment, making this a strong foundation for future work.