# Face Age Prediction

Age group classification using a deep learning model

Sri Spoorthi Vattem

# Problem Statement

The objective of this project is to develop a deep learning-based classification model that can accurately predict the age group of an individual based on their facial image. By analyzing key facial features, the model will categorize each image into predefined age ranges. This involves training a Convolutional Neural Network (CNN) to effectively recognize and distinguish age-related patterns across different demographics.

# Application of Selecting the Project

This project was selected due to its practical relevance in areas such as age verification, personalized services, and demographic analysis. It provides a hands-on experience in image classification while offering real-world applications

# Type of Training Dataset Options(Initially)

The training dataset will consist of labeled facial images covering a wide range of age groups and demographic diversity. Datasets like IMDB-WIKI or Adience provide age-labeled facial data that will be used to train and validate the model. Preprocessing such as face alignment and augmentation will be applied to enhance the dataset quality

# About the project:

- The task is to classify images into age groups, where each age group spans 5 years (e.g., 0-4, 5-9, etc.).
- The model uses grayscale facial images as input and predicts the age group of the individual.

# Dataset:

- The dataset used in this project is UTKFace Dataset, which contains images labeled with age, gender, and other attributes.
- format: 1_0_0_20161219140623097.jpg.chip.jpg (age_gender_race...)

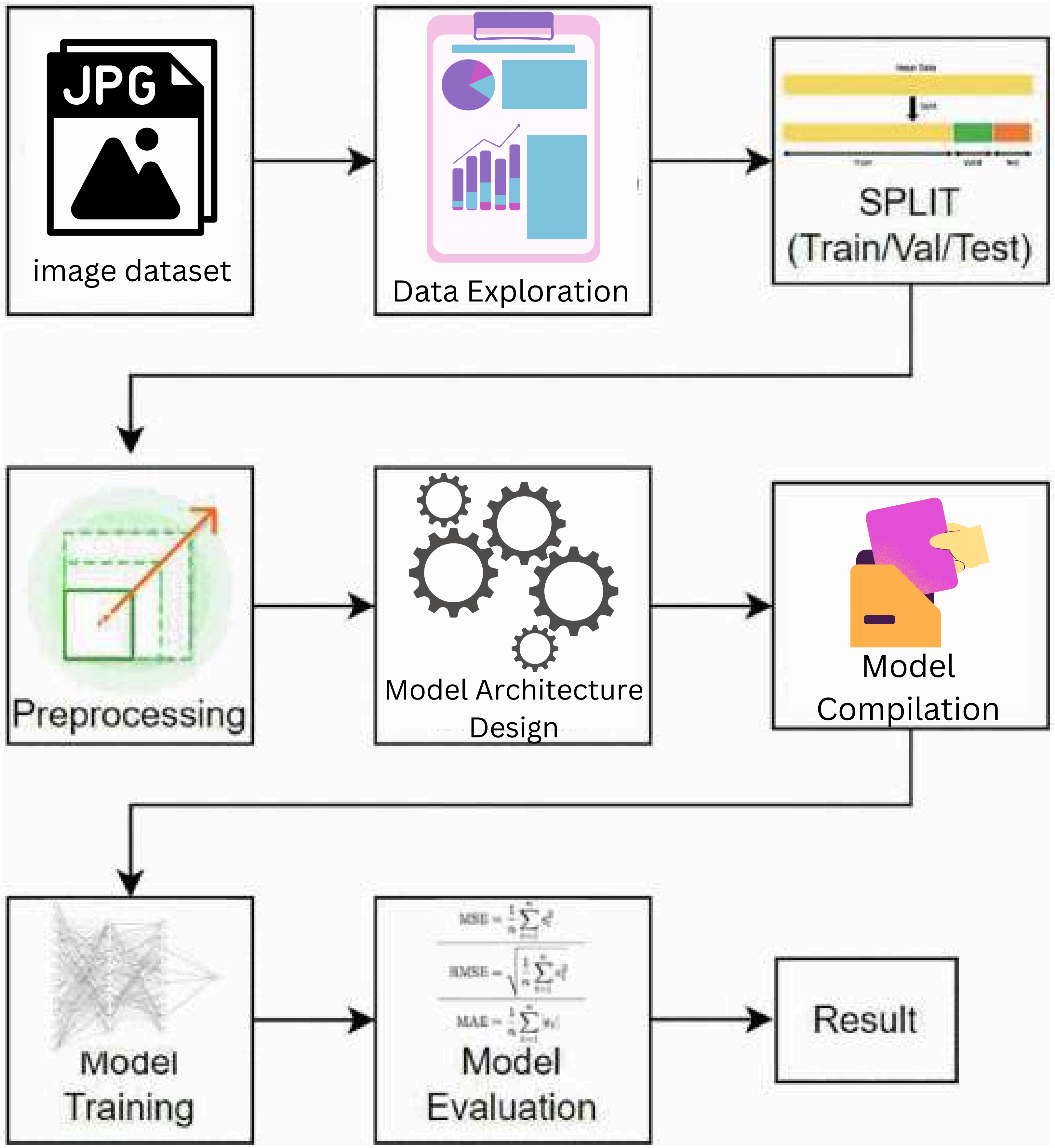
# Key preprocessing steps:

- Resizing images to 64x64.
- Converting images to grayscale.
- Normalizing pixel values to a range of 0-1.
- Splitting the dataset into training and testing subsets.

# Model Architecture:

- The model is a Convolutional Neural Network (CNN)
- Input Layer: Takes 64x64 grayscale images.
- Convolutional Layers: Extracts spatial features from the images.
- Pooling Layers: Reduces the dimensionality while retaining important features.
- Fully Connected Layers: Maps features to a single numeric output representing normalized age.
- Output Layer: A regression output that predicts the normalized age.

# Basic Outline of the model



# Libraries and Tools used:

Numpy :

- Used for numerical operations and array manipulation.

os:

- To interact with the file system.

OpenCV (cv2):

- Used for image processing.

Matplotlib (matplotlib.pyplot):

- Used for data visualization.

Scikit-learn (sklearn.model_selection):

- Used for splitting the dataset into training and testing sets.

Keras (keras):

- A deep learning library used for building, training, and evaluating neural network models.

Key Modules:

- keras.utils.to_categorical: For one-hot encoding the target labels.
- Layers: Input, Conv2D, MaxPooling2D, Flatten, Dense, Dropout(CNN Model Architecture)

# Hyperparameters Used:

Image Preprocessing:

- Input image dimensions: 64 x 64 pixels.
- Grayscale images with 1 channel.

Dataset Splitting:

- Train-test split ratio: 80:20.
- Number of Classes: 21 age groups (e.g., 0-4, 5-9, ..., 100+).

Model Architecture:

1. Convolutional layers:

- Filters: 32, 64, and 128.
- Kernel size: (3, 3).
- Activation function: ReLU.

2. Pooling layers: MaxPooling with a pool size of (2, 2).

Dropout:

- 0.25 after pooling layers.
- 0.5 after fully connected layers.

Dense layers:

- 128 and 64 units with ReLU activation.

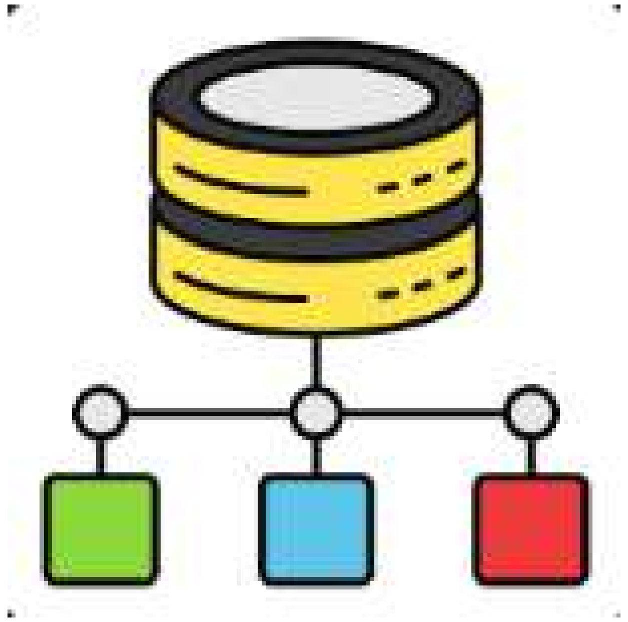
Training Hyperparameters:

- Optimizer: Adam.
- Loss function: Categorical Crossentropy.
- Metrics: Accuracy.
- Batch size: 128.
- Number of epochs: 25.

# Initial Approaches:

1. Broad Age Categorization:
- Initially, ages were grouped into 4 broad categories:
- 1: 0-17 years (children/teenagers); 2: 18-29 years (young adults); 3: 30-79 years (adults); 4: 80+ years (seniors)
- The target variable was normalized to a continuous range [0, 1], treating the problem as a regression task rather than classification.
2. Loss Function:
- The model used Mean Squared Error (MSE) as the loss function and Mean Absolute Error (MAE) as a metric. This regression-based approach was less effective at capturing distinct categories for age groups.
3. Simpler Model Architecture:
- The initial model had a similar convolutional architecture but was designed for regression with a single neuron output layer instead of a multi-class output.

1

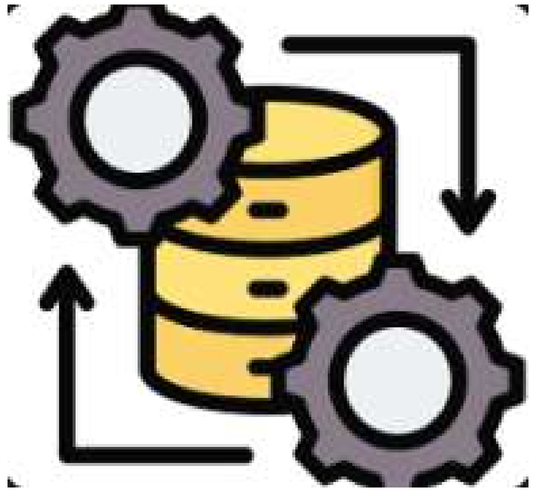


## Data Loading

- Read each image in grayscale.
- Resize the image to 64x64 pixels.
- Extract the age from the filename

OUTPUT:Lists images and ages

2



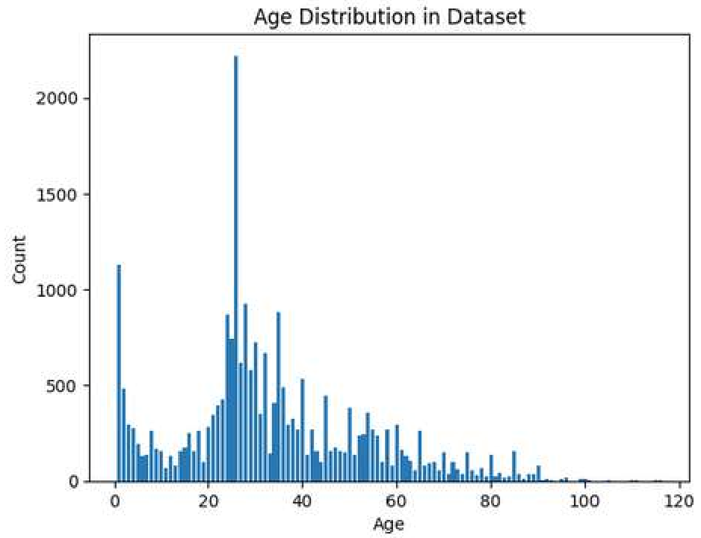
## Data Preprocessing

INPUT:Lists images and ages

- Normalize image pixel values (divide by 255).
- One-hot encode the age group (age_group function).

OUTPUT:Preprocessed image features and one-hot encoded labels target.

3




## Visualization

INPUT:Lists ages

- Create a list of unique age values (x_ages).
- Count the frequency of each age (y_ages).
- Plot a bar chart to visualize the age distribution.

OUTPUT: Age distribution plot.

4

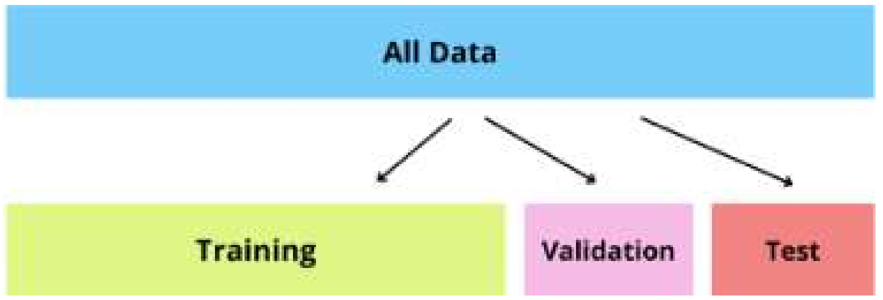
## Age Group Classification Function (age_group:)

INPUT:Age value.

- Divide the age by 5 to assign it to an age group (e.g, age 23 maps to group 4).If the group exceeds 20, cap it at 20 (for ages 100+).

OUTPUT: Age group index.

5



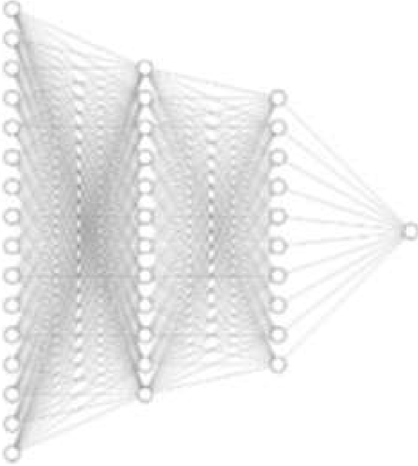

## Test-Train Split:

INPUT:Preprocessed features and target

- Split data into training and testing sets (80% training, 20% testing)

OUTPUT: x_train, x_test, y_train, y_test

6



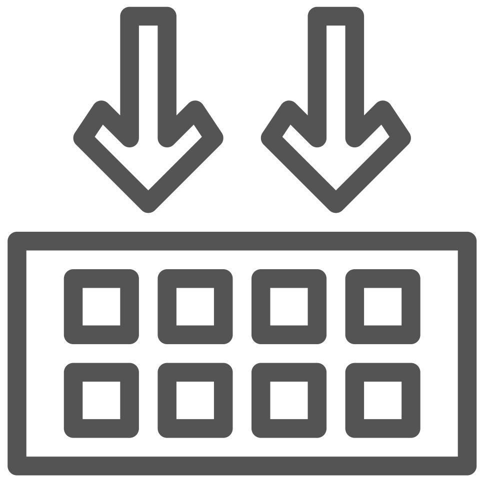
## Model Architecture:

INPUT: Input shape of (64, 64, 1) (grayscale image).

- Creating a Convolutional Neural Network (CNN) with 12 layers

OUTPUT: CNN model.

7



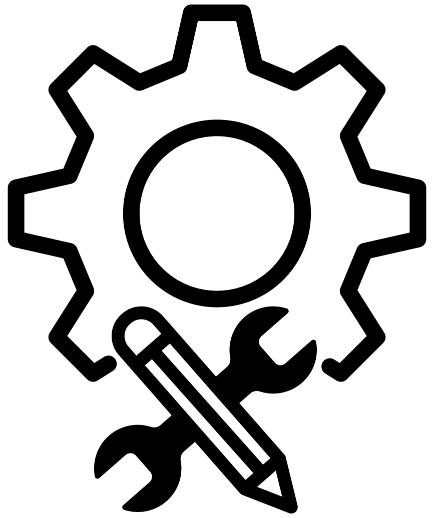
## Model Compilation:

INPUT: The defined model.

- Compile the model with Adam optimizer, categorical cross-entropy loss, and accuracy metrics.

OUTPUT: Compiled model.

8



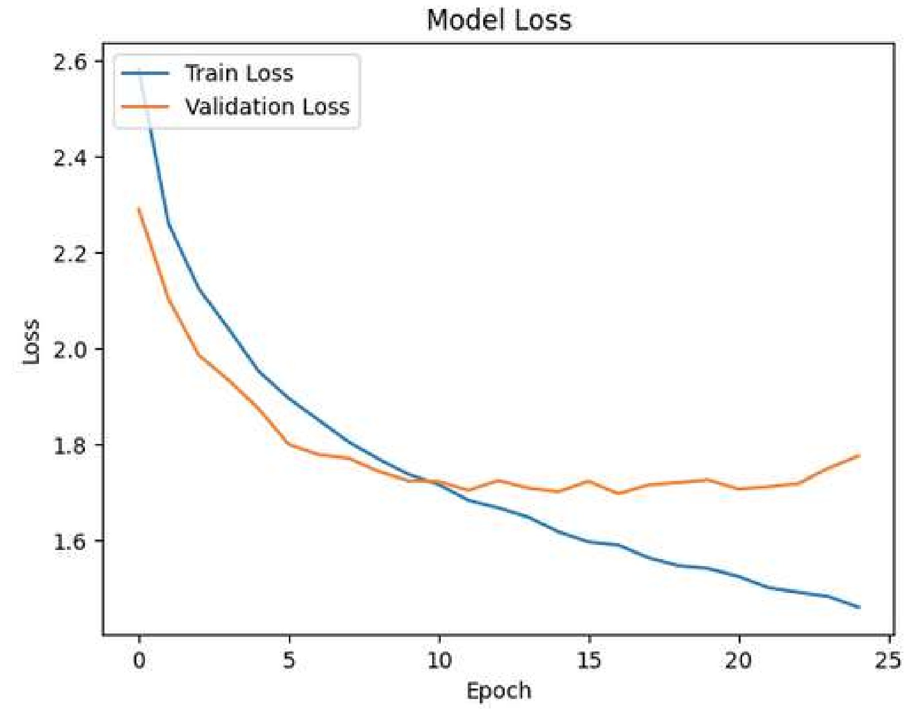
## Model Training:

INPUT: Training data (x_train, y_train), testing data (x_test, y_test).

- Train the model for 25 epochs with a batch size of 128.
- Use training and validation data for monitoring loss and accuracy.

OUTPUT: Compiled model.

9




## Training & Validation Loss Plot

10

## Age Range Prediction Function (get_age):

INPUT: Model prediction (probabilities).

- Convert the predicted class (index) to an age range.
- For the highest predicted class, map to a specific age range (e.g., group 0 maps to 0-4, group 20 maps to "100+").

OUTPUT: Predicted age range.

11

## Sample Prediction and Display:

INPUT: Sample image.

- Normalize the sample image.
- Predict the age group using the trained model.
- Map the prediction to an age range using get_age.
- Display the sample image and print both actual and predicted age range.

OUTPUT: Compiled model.

12

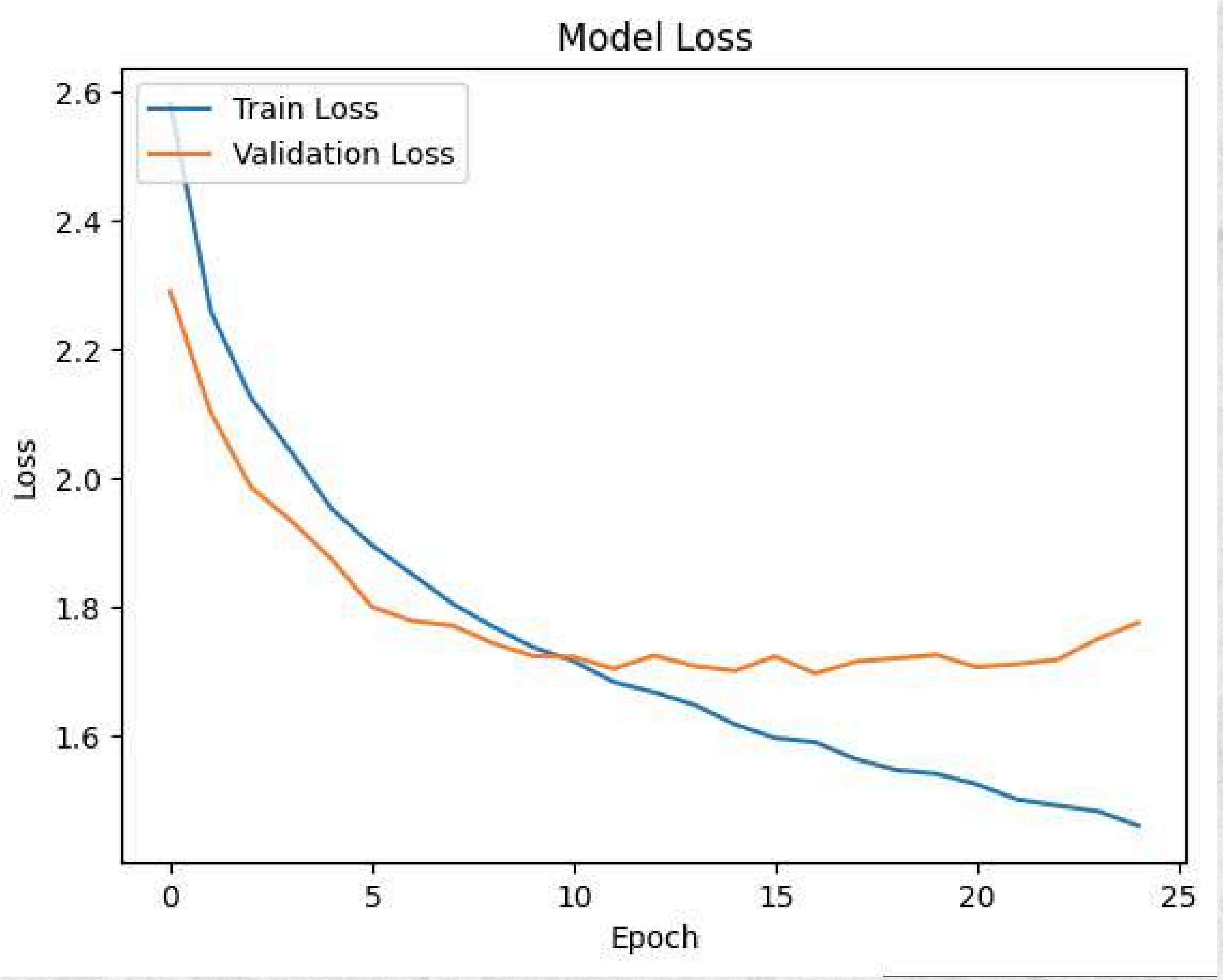
## Testing with Specific Samples:

INPUT: Input: List of sample indices.

- For each index, fetch the corresponding image and display it.
- Display the actual age and predicted age range for the sample.

OUTPUT: Compiled model.

# Training vs Validation loss:




Observation:

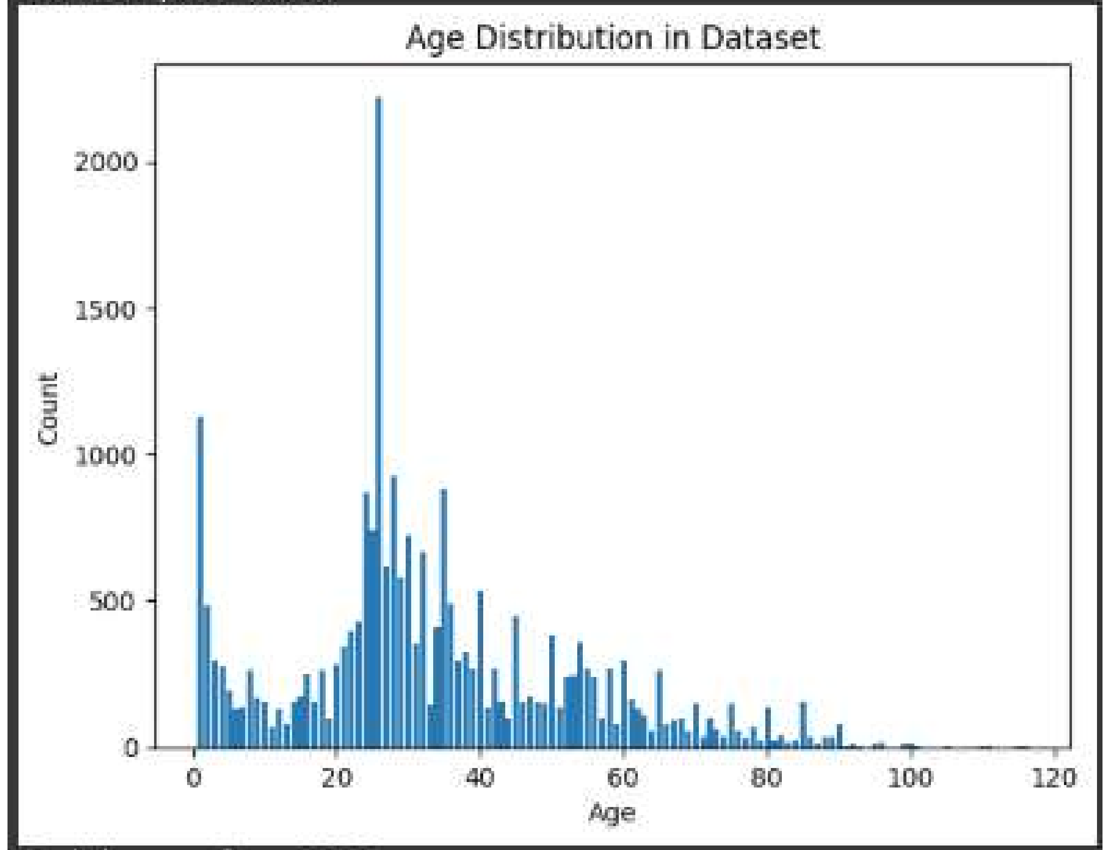
- The training loss is steadily decreasing, indicating that the model is learning well on the training data.
- The validation loss initially decreases but starts to plateau and slightly increase after ~15 epochs, which could suggest little overfitting after this point.

# Conclusion:

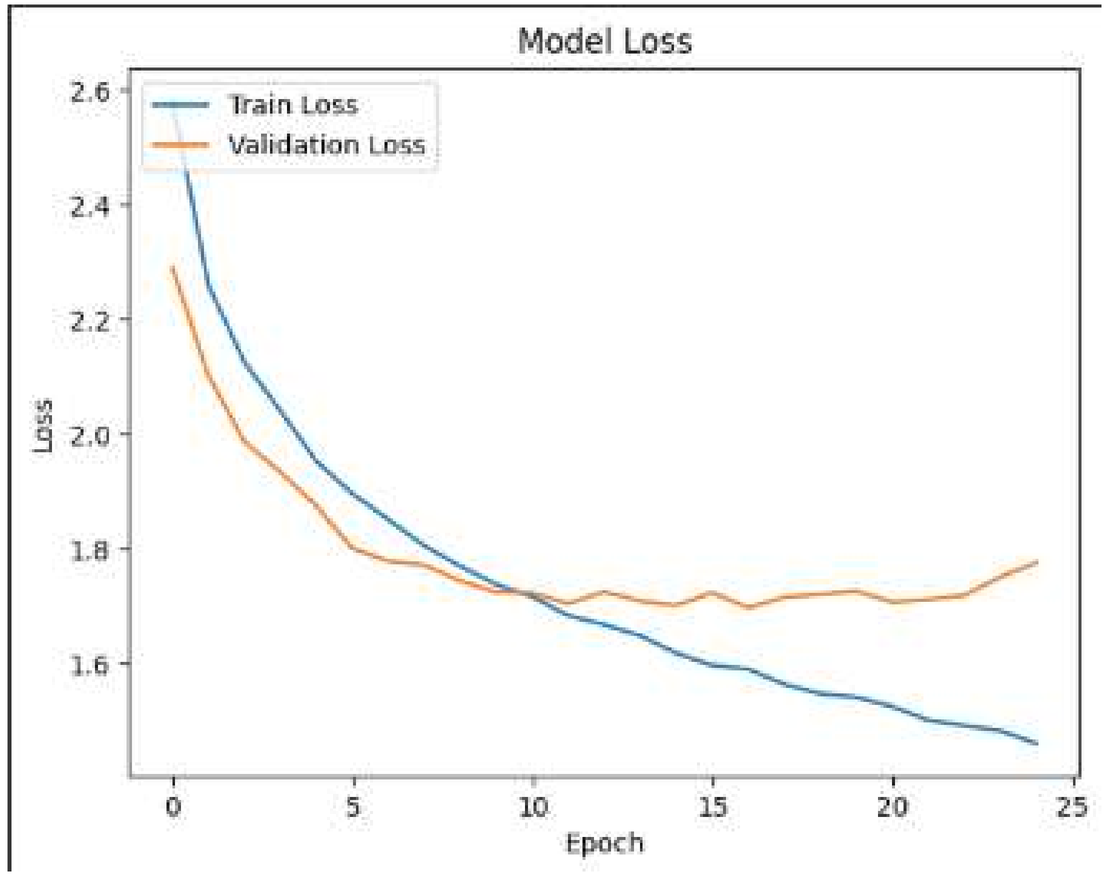
- Key Achievements:
  - Successfully implemented an age group classification model using the UTKFace dataset.
  - Leveraged a CNN architecture to classify images into 5-year age intervals with a cap at 100+ years.
- Challenges Faced:
  - Managing class imbalance in the dataset due to uneven age group distribution.
  - Preprocessing large image datasets efficiently while handling corrupt or invalid files.
- Model Performance:
  - Achieved reasonable accuracy on the test set, demonstrating the model's ability to generalize across age groups.
  - Visualized age predictions for individual samples, validating the approach qualitatively.
- Future Scope:
  - Explore techniques like data augmentation to address class imbalance further.
  - Improve accuracy by experimenting with deeper networks etc..

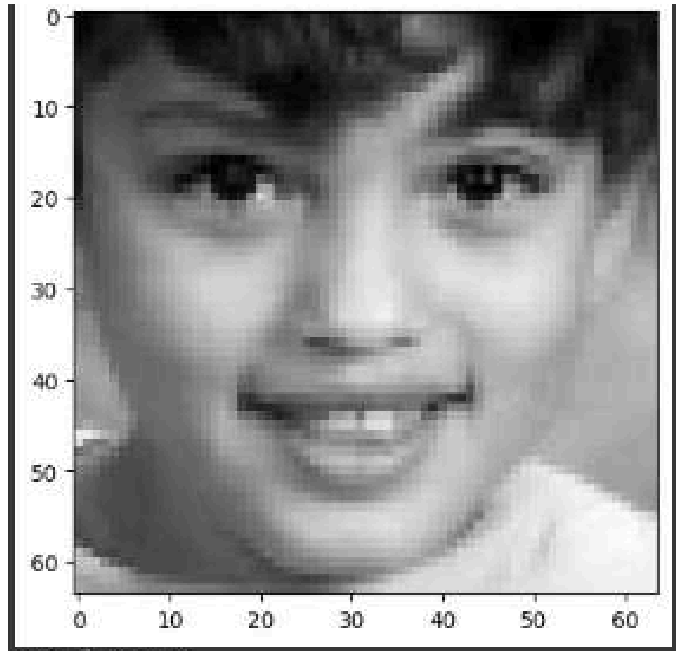
# Output

Total samples: 23878

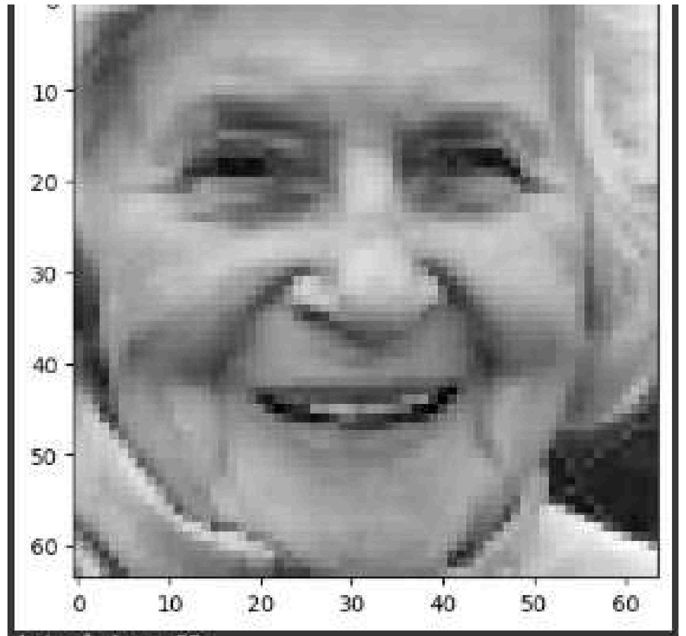



Training samples: 19102
Testing samples: 4776

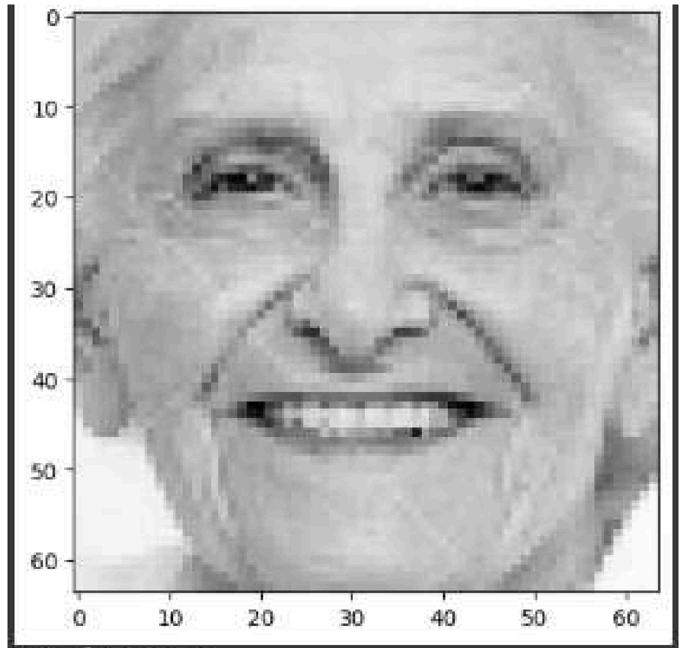


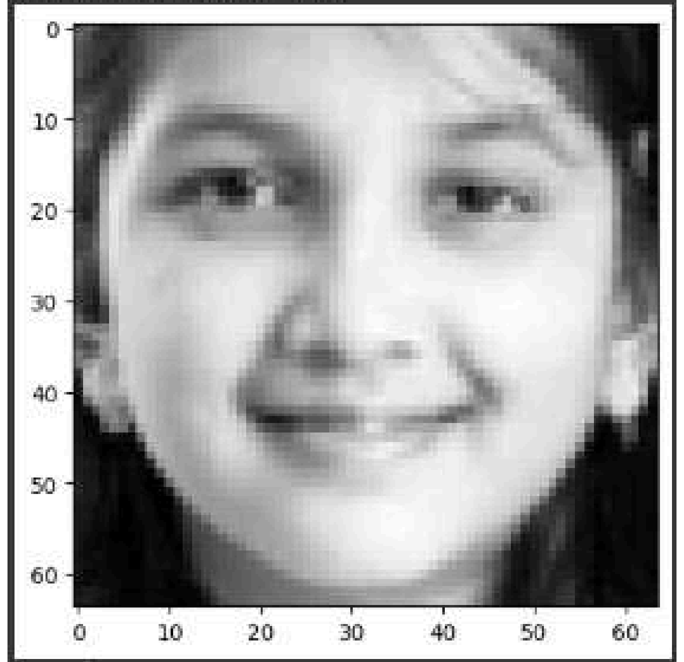
Actual Age: 8
Predicted Age Range: 5-9



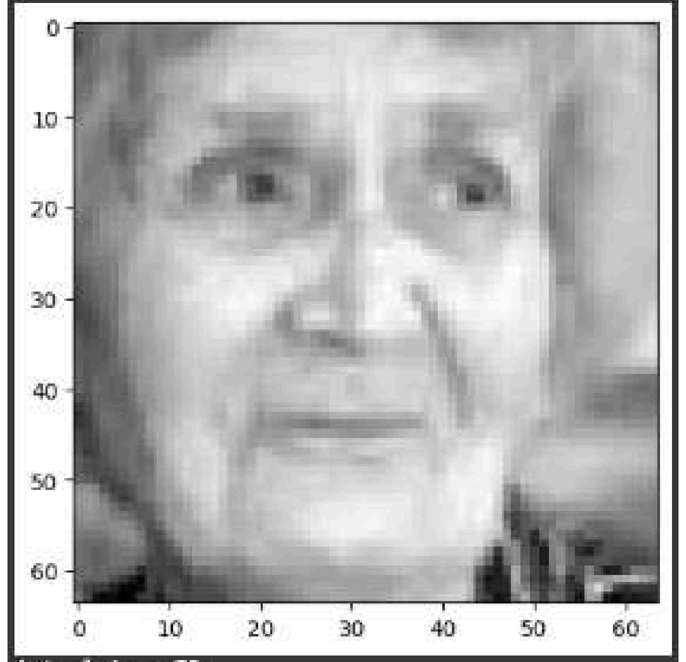
Actual Age: 85
Predicted Age Range: 85-89



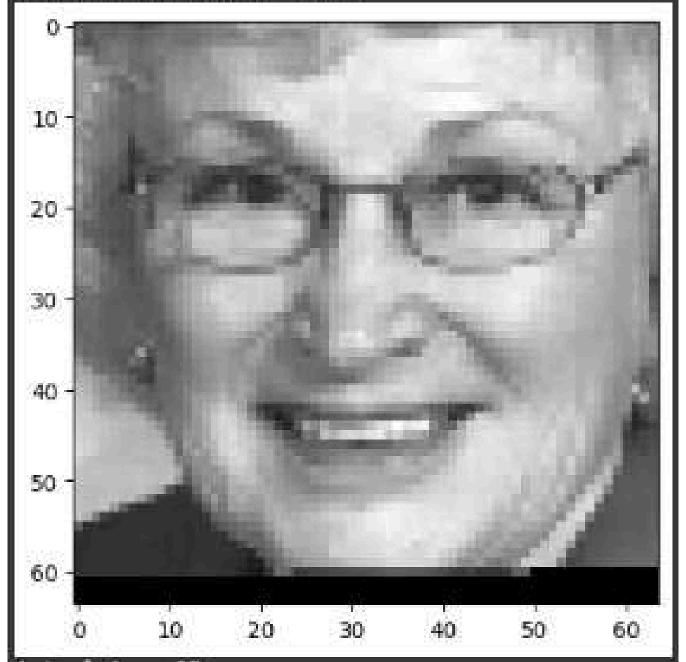
Actual Age: 85
Predicted Age Range: 85-89



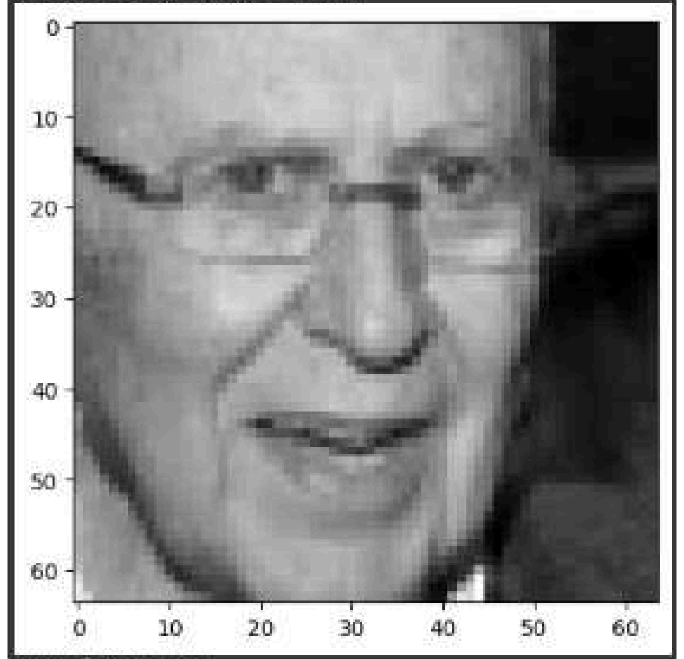
Actual Age: 8
Predicted Age Range: 10-14



Actual Age: 73
Predicted Age Range: 85-89

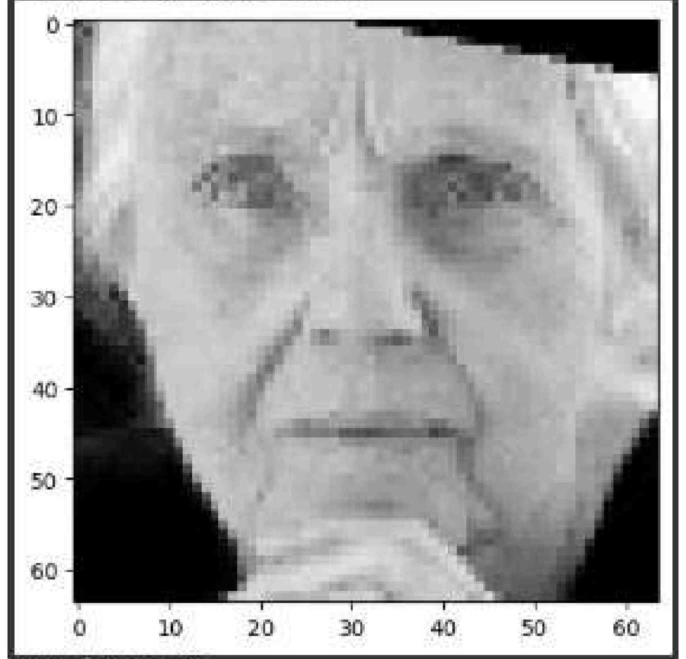


Actual Age: 85
Predicted Age Range: 85-89



Actual Age: 83
Predicted Age Range: 85-89

Actual Age: 86
Predicted Age Range: 85-89



Actual Age: 85
Predicted Age Range: 85-89