Age & Gender Prediction CNN UTKFace

2024W-T3

Final Report

Group

Charmi Raghavani

Reebika Bhatta

Ruchita Soni

Shirish Thapaliya

Sweekrit Acharya

Abstract

In recent years, the proliferation of images has surged, fueled by the widespread use of cameras and the rise of selfies. This surge in facial images has sparked a growing interest in automatic age and gender prediction from facial data. This paper addresses this challenging problem by leveraging deep learning techniques, particularly convolutional neural networks (CNNs). We focus on developing a robust gender and age prediction system capable of accurately inferring demographic attributes from single facial images.

The significance of our project lies in its ability to address real-world challenges posed by the abundance of facial data. With applications ranging from security systems to personalized advertisements, accurate age and gender prediction from facial images have become increasingly important. Through this research, we aim to demonstrate the effectiveness of CNNs in tackling the complexities of facial age and gender estimation, offering insights into their potential applications in various domains.

Table of Contents

1. Introduction

In recent years, the proliferation of facial images due to widespread camera use and the rise of selfie culture has sparked interest in automatic age and gender prediction. This surge in data availability has driven advancements in facial recognition technology, with gender prediction emerging as a significant task within this domain.

The project delves into gender prediction using Convolutional Neural Networks (CNNs) applied to the UTKFace dataset. CNNs, well-known for their prowess in image processing, are leveraged to accurately predict the gender of individuals depicted in facial images.

One notable aspect of our project is the balanced nature of the UTKFace dataset, which contains a diverse collection of facial images with associated age and gender. This balance ensures that our CNN model is trained on representative data, contributing to its robustness and generalization capabilities.

Our primary goal in this project is to develop a robust gender prediction system capable of inferring demographic attributes from single facial images. Through the utilization of CNN architectures and deep learning techniques, we aim to achieve high accuracy rates in both gender classification and age estimation tasks.

Through rigorous experimentation and optimization, our CNN model has attained an impressive accuracy of 87%, highlighting its effectiveness in tackling this complex problem.

2. Methodology

2.1 Dataset

The UTKFace dataset is a valuable resource for researchers in facial recognition, age estimation, and gender classification tasks. With approximately 23,710 images of individuals ranging from 0 to 116 years of age, annotated with age, gender, and ethnicity, it offers a comprehensive dataset for training and evaluating machine learning models.

One of the notable aspects of the UTKFace dataset is its balanced distribution of gender, with 52.3% males and 47.7% females. This balanced gender distribution ensures that the dataset is suitable for gender classification tasks without bias towards any particular gender. However, for our specific project, which focuses on age estimation and gender distribution.

2.2 File Renaming

- The initial step involves renaming files with the ".chip" extension and the ".jpg.jpg" extension to ensure uniformity and remove unwanted extensions.
- Many datasets may contain filenames with non-standard extensions or duplicates. Removing these unwanted extensions ensures consistency in file naming conventions, making it easier to work with the dataset.

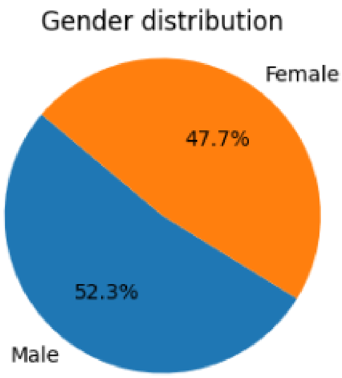
2.2 Data Extraction

- Extract gender and age information from filenames to create structured data.
- Filenames often contain metadata relevant to the data, such as gender and age in this case. Extracting this information allows for better organization and understanding of the dataset.

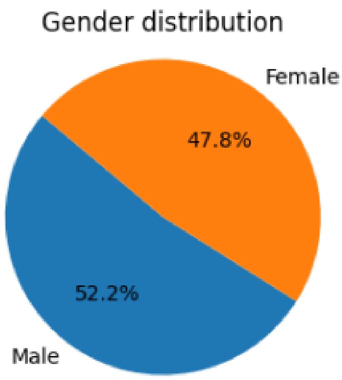
2.3 Dataset Splitting

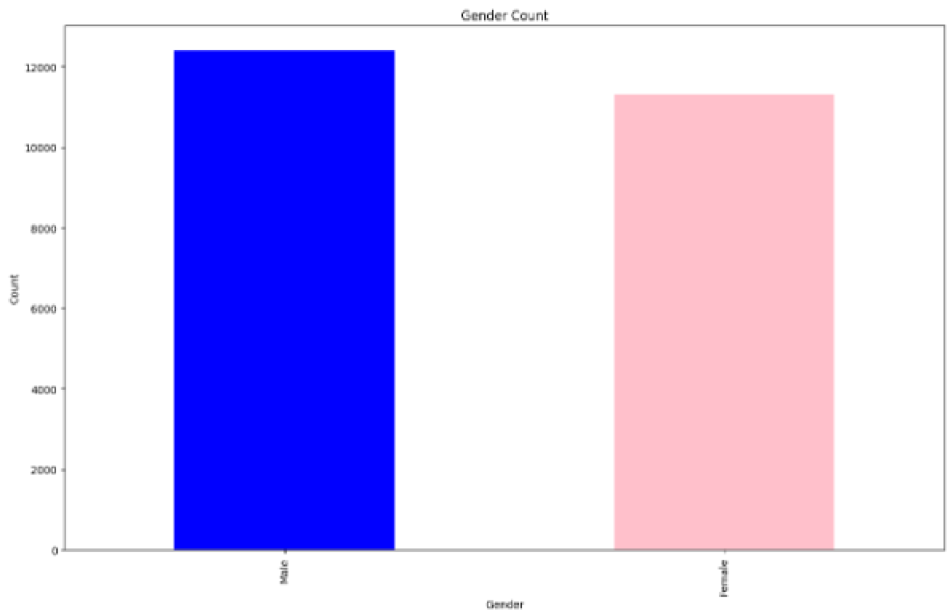
- Divide the dataset into training, validation, and testing sets to assess model performance.
- Splitting the dataset ensures that the model is trained on one subset, validated on another subset to tune hyperparameters, and tested on a separate subset to evaluate its generalization performance. This prevents overfitting and provides a more reliable assessment of the model's performance.

Training



Testing





2.4 Data Augmentation

- we employ data augmentation techniques to enhance the diversity of our training samples, crucial for robust model training.
- Leveraging the ImageDataGenerator from the Keras library, we apply various transformations to our images.
- Horizontal and vertical flips, rotation, and rescaling are among the transformations used. These techniques introduce variability in the training data, exposing the model to different image variations.
- By rescaling pixel values to the range [0, 1] and applying flips and rotations, we ensure that our model learns from a broader range of image variations, making it less prone to overfitting and more capable of generalizing to unseen data.
- This approach is particularly vital in tasks like facial recognition and age estimation, where the ability to handle diverse image inputs is essential for model effectiveness.

3. Model Architectures

3.1 Convolutional Neural Network (CNN)

3.1.1 Architecture: The CNN architecture is designed to sequentially apply convolutional layers for feature extraction, followed by pooling layers for spatial downsampling, and fully connected layers for classification.

3.1.2 Layers: The CNN architecture comprises approximately **6 layers**, including convolutional, max-pooling, dropout, and dense layers. The specific configuration of these layers may vary depending on the complexity of the task and the size of the dataset.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 222, 222, 128)	3,584
max_pooling2d (MaxPooling2D)	(None, 111, 111, 128)	0
dropout (Dropout)	(None, 111, 111, 128)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	73,792
max_pooling2d_1 (MaxPooling2D)	(None, 54, 54, 64)	0
dropout_1 (Dropout)	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 32)	18,464
max_pooling2d_2 (MaxPooling2D)	(None, 26, 26, 32)	0
conv2d_3 (Conv2D)	(None, 24, 24, 32)	9,248
flatten (Flatten)	(None, 18432)	0
dense (Dense)	(None, 64)	1,179,712
dense_1 (Dense)	(None, 1)	65

This model architecture consists of several layers, starting with a convolutional layer (conv2d) followed by max-pooling layers (max_pooling2d) and dropout layers (dropout) to prevent overfitting. The convolutional layers progressively reduce the spatial dimensions of the input while increasing the number of channels, effectively extracting features from the images. After multiple convolutional and pooling operations, the feature maps are flattened into a one-dimensional vector using the flatten layer. This vector is then passed through dense layers (dense) for further feature processing and classification. The final dense layer (dense_1) produces a single output representing the predicted gender. Overall, the model has a total of 1,284,865 parameters, with the majority concentrated in the dense layers, indicating its capacity to learn complex patterns from the input data.

3.1.3 Activation Function: **ReLU** (Rectified Linear Unit) activation function is typically used for the convolutional layers to introduce non-linearity and improve model performance. For **binary classification** tasks, such as gender classification, the output layer typically employs a **sigmoid** activation function to produce probabilistic outputs.

3.1.4 Optimizer: The **Adam** optimizer is utilized for optimization, which adapts the learning rate during training to converge faster and more efficiently. The **binary cross-entropy** loss function is commonly employed for binary classification tasks, such as gender classification, as it measures the dissimilarity between predicted probabilities and actual labels.

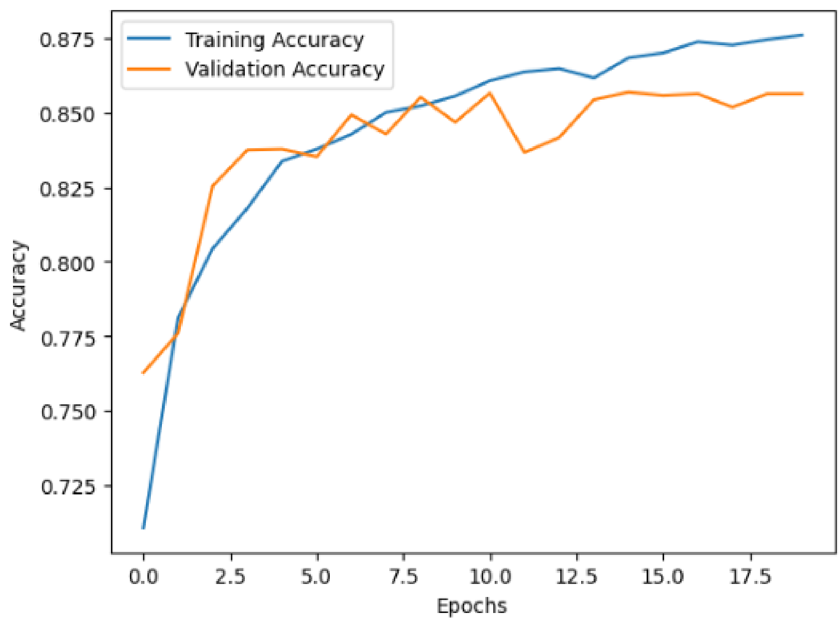
3.1.5 Overfitting and Underfitting: To mitigate overfitting and underfitting, various strategies are implemented:

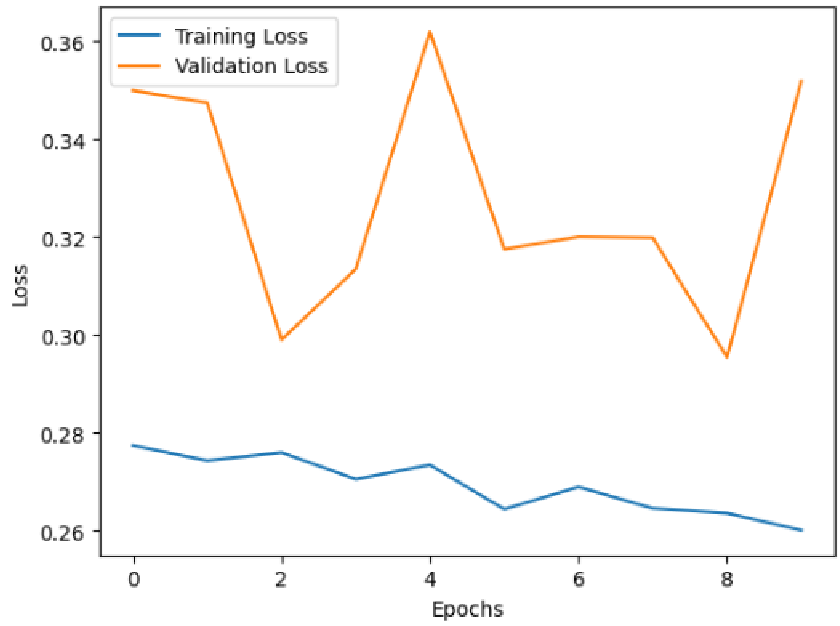
1. **Dropout Layers**: Dropout layers are inserted between dense layers to randomly deactivate a fraction of neurons during training, preventing the model from relying too heavily on specific features and improving generalization.

2. **Early Stopping Callbacks**: Early stopping callbacks are employed to monitor the model's performance on a validation dataset and halt training when the performance starts to degrade, thus preventing overfitting.

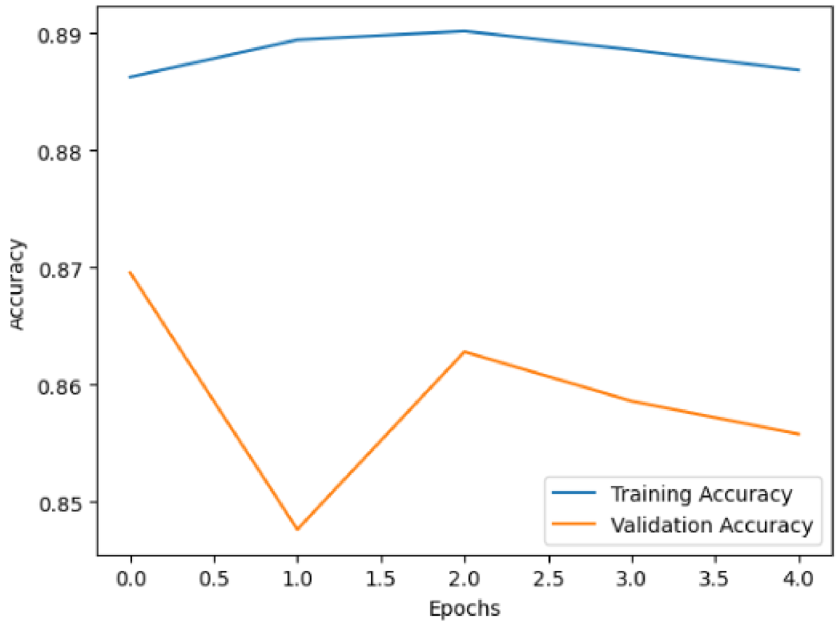
3. **Model Training**: The model is trained using the fit() function with training and validation data generators. The training process is monitored for 20 epochs, with a batch size of 32. ModelCheckpoint is used to save the best-performing model based on validation accuracy. After training for the specified number of epochs, the model achieves an impressive accuracy of 87%.

3.1.6 Training History Visualization: The training and validation accuracy and loss are plotted to visualize the model's performance over **20 epochs.**





- The training and validation accuracy of the model's performance over **5 epochs.**



3.1.7 Visualization of Test Results : Test results are visually inspected, the trained model is saved in both HDF5 and Keras formats, loaded from Keras format to ensure integrity, and used for predictions on the test dataset with subsequent visualization.

"Visual Analysis: Before and After Model Loading"

3.1.8 Model Evaluation and Performance Metrics:

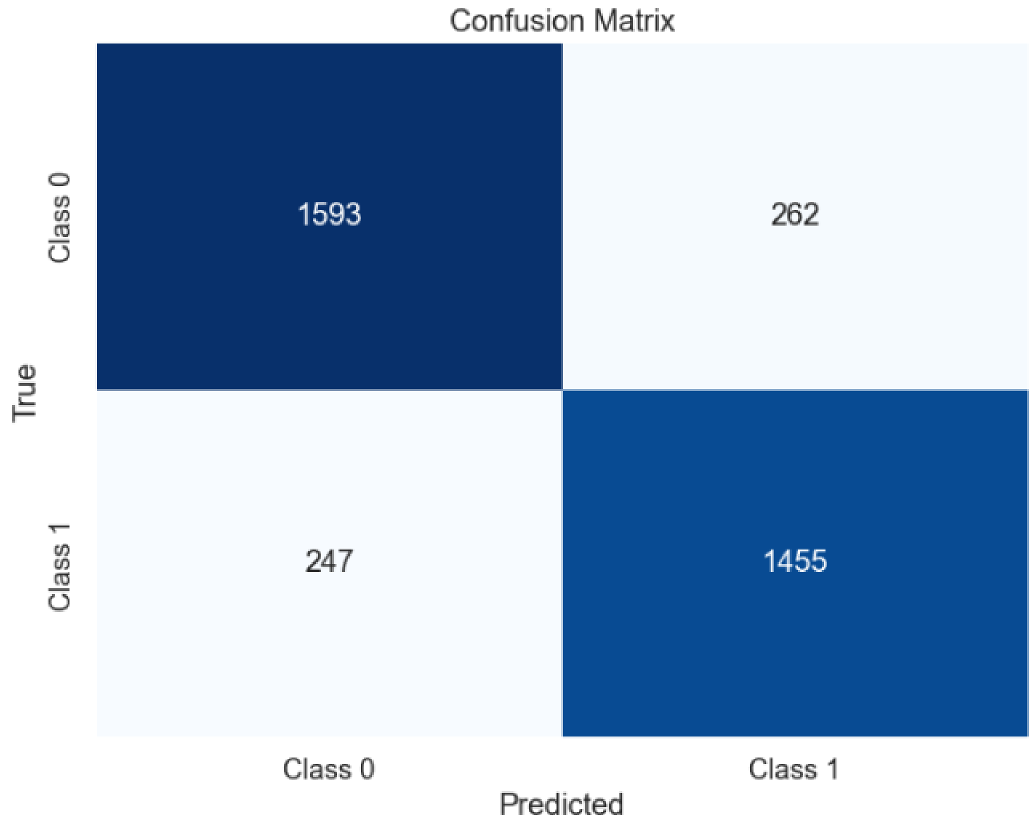
- **Test Loss and Accuracy**: The trained CNN model is evaluated using the evaluate() function, yielding a test loss of 0.326 and a test accuracy of 86.03%. These metrics provide insights into the model's performance on unseen data, indicating a satisfactory level of accuracy.

3.1.9 Classification Report:

- **Precision, Recall, and F1-Score**: The classification_report function is utilized to compute precision, recall, and F1-score for both male and female genders. The report reveals high precision and recall values for both classes, indicating a balanced performance of the model in classifying male and female genders.
- **Accuracy:** The overall accuracy of the model on the test dataset is reported to be 86%, demonstrating its ability to correctly classify gender with a high level of precision.

3.1.10 Confusion Matrix:

- **Visualization**: The confusion matrix is computed and visualized using seaborn, providing a clear overview of the model's performance in terms of true positive, false positive, true negative, and false negative predictions for each class (male and female).



3.2 Logistic Regression model

3.2.1 Model Architecture:

- The logistic regression model is chosen for its simplicity and interpretability, making it suitable for baseline gender classification tasks.
- The model architecture comprises a logistic regression classifier initialized with a random state of 42 and a maximum iteration limit of 1000.

3.2.2 Training Process:

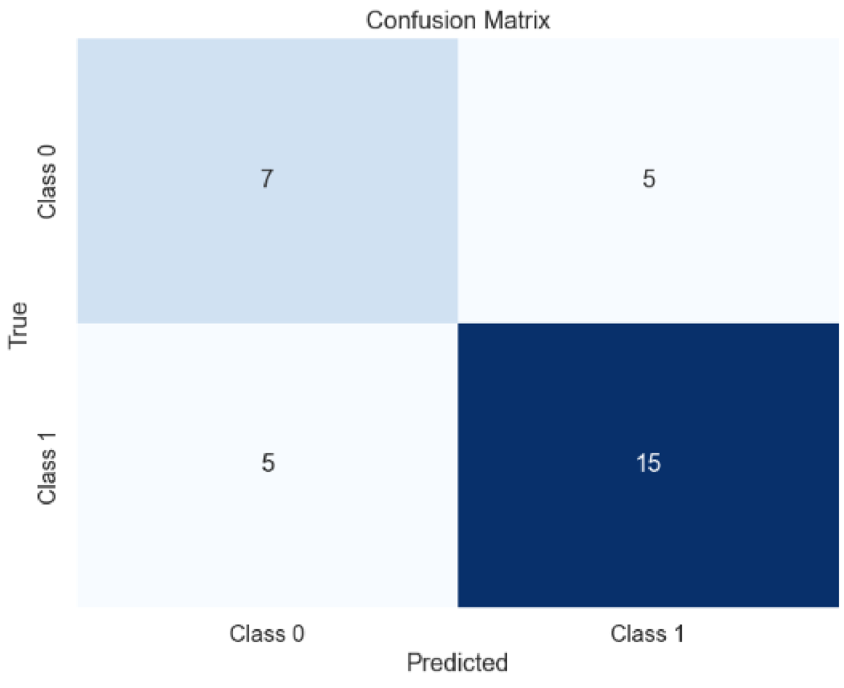
- Data preprocessing involves reshaping the image data to have two dimensions and splitting it into training, validation, and test sets using data generators.
- The logistic regression model is trained using the fit() function, with the training process monitored for 20 epochs. The ModelCheckpoint callback is utilized to save the best-performing model based on validation accuracy.

3.2.3 Performance Evaluation

- The logistic regression model achieves a remarkable training accuracy of 100%, indicating its ability to learn from the training data effectively.
- Classification reports are generated for both the validation and test datasets, revealing precision, recall, and F1-score metrics for each gender class (0 for male, 1 for female).
- On the validation dataset, the model achieves an accuracy of 75%, while on the test dataset, it achieves an accuracy of 69%. The classification reports provide insights into the model's performance and its ability to distinguish between male and female genders.

3.2.4 Confusion Matrix Analysis

- Confusion matrices are constructed to visually represent the model's performance on both the validation and test datasets.
- The confusion matrices aid in understanding the distribution of correct and incorrect predictions for each gender class, facilitating the identification of areas for improvement in the model.



3.3 Decision Tree Model

3.3.1 Model Architecture

- The decision tree classifier employed in this analysis serves as a hierarchical model for decision-making.
- It recursively partitions the feature space based on the attributes of the training data, constructing a tree-like structure where each node represents a decision based on a specific feature.

3.3.2 Training Process

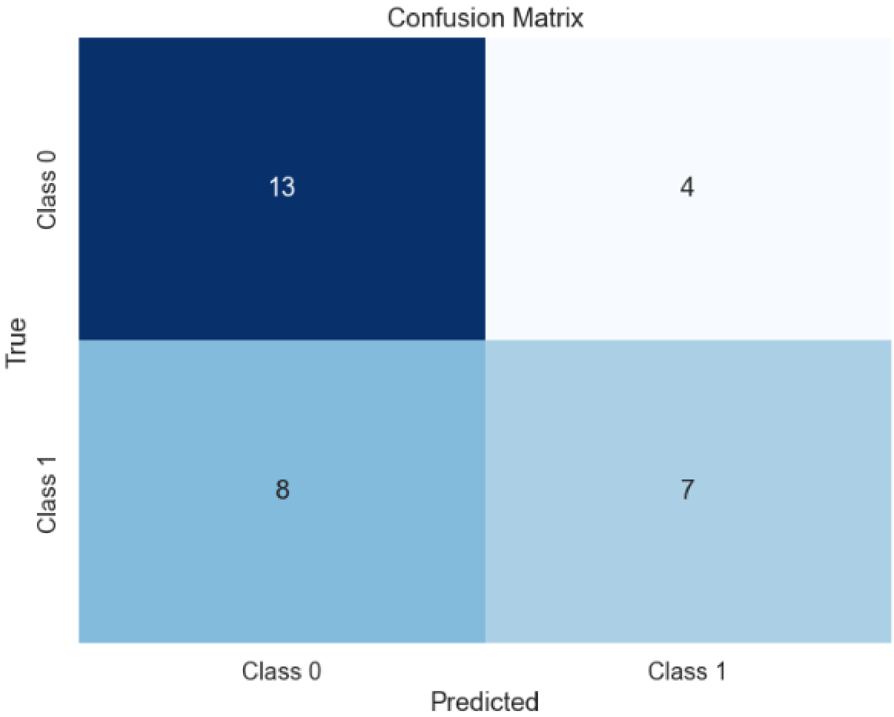
- During the training phase, the decision tree classifier learns to make decisions based on the features of the training data.
- It achieves a remarkable training accuracy of 100%, indicating that the model perfectly fits the training data. However, such high training accuracy may suggest potential overfitting, which needs to be further evaluated on unseen data.

3.3.3 Performance Evaluation

- The model's performance is evaluated on both the validation and test sets to assess its generalization ability.
- On the validation set, the classifier achieves an accuracy of 62.50%, suggesting moderate performance in generalizing to unseen data.
- The classification report provides additional insights into the model's performance, showing precision, recall, and F1-score for each class.

3.3.4 Confusion Matrix

- The confusion matrix visually represents the classifier's performance by showing the counts of true positive, false positive, true negative, and false negative predictions.
- From the confusion matrix, we can observe how well the model distinguishes between the two classes (Class 0 and Class 1) and identify any potential misclassifications.
- In this case, the confusion matrix helps us understand the classifier's performance and identify areas for improvement, such as addressing potential biases or adjusting model parameters to optimize performance.


Confusion Matrix
True
Class 0
Class 1
13
4
8
7
Class 0
Class 1
Predicted

4. Results

4.1 Comparison of Models

Model	**Training_accuracy**	**Testing_accuracy**	**Validation_accuracy**
CNN	88%	86%	85%
Logistic Regression	100%	69%	75%
Decision Tree	100%	62%	62%

5. Conclusion

The project successfully developed and evaluated gender prediction models using Convolutional Neural Networks (CNNs), logistic regression, and decision tree algorithms on the UTKFace dataset. CNNs exhibited superior performance with an accuracy of 87% on the test dataset, highlighting the effectiveness of deep learning techniques in facial image analysis. The balanced nature of the dataset contributed to robust model training, while logistic regression provided a simple yet competitive alternative. However, the decision tree model showed moderate performance, indicating potential overfitting. Overall, the project demonstrated the significance of dataset balance, model complexity, and deep learning techniques in achieving accurate gender predictions, paving the way for further advancements in facial recognition technology.

6. References

[1] Moritz Mitterdorfer, 'Age Estimation on the UTKFace Dataset,' Weights & Biases, Available: https://wandb.ai/moritzm00/UTKFace-Age-Regression/reports/Age-Estimation-on-the-UTKFace-Dataset--Vmlldzo0MzE3MjE3

[2] Eward96, 'Age and Gender Prediction on UTKFace,' Kaggle, Available: https://www.kaggle.com/code/eward96/age-and-gender-prediction-on-utkface

[1] [Authors], 'Age and Gender Estimation on UTKFace Dataset,' ArXiv, October 2021. Available: https://arxiv.org/ftp/arxiv/papers/2110/2110.12633.pdf