1. INTRODUCTION

1.1 About CNN

When it comes to artificial intelligence, Convolutional Neural Networks (CNNs) are the best option, particularly for computer vision applications. CNNs are designed specifically to process grid-like data, like images, more efficiently than traditional neural networks. Convolutional, pooling, and fully connected layers are some of the specialized layers that help achieve this.

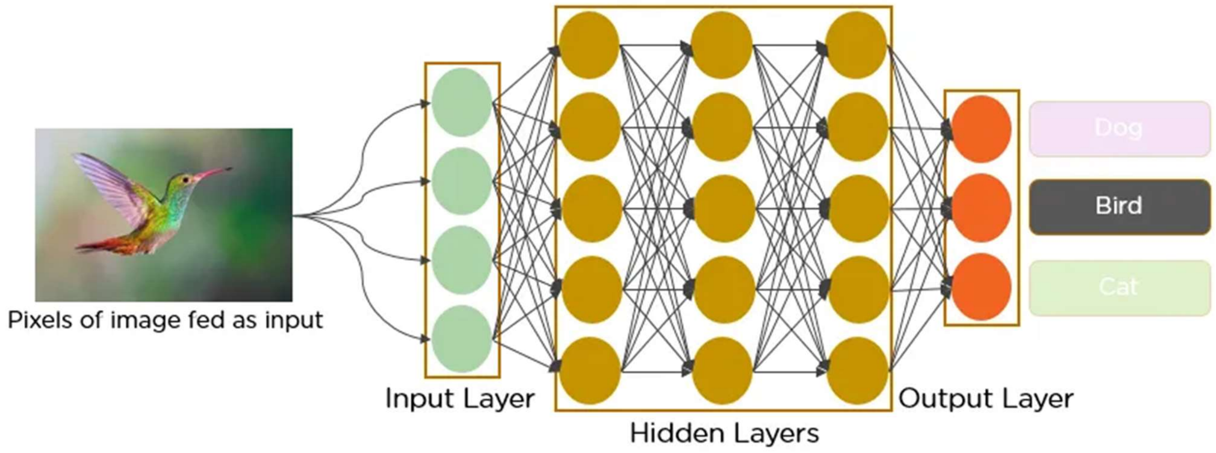


Fig. 1 CNN Architecture

Convolutional layers apply tiny filters to the input image to extract features, such as spatial hierarchies and local patterns. The extracted features are then downsampled by pooling layers, which lowers computational complexity without sacrificing important information. Ultimately, these high-level features are mapped to the intended output classes by fully connected layers. CNNs autonomously learn hierarchical representations of visual data through a training process in which the network modifies its parameters to minimize errors.
With this ability, CNNs can perform exceptionally well in tasks like object recognition, image classification, and semantic segmentation, which makes them essential tools for advancing artificial intelligence in a variety of fields.

1.2 About the dataset

This dataset contains over 1000 odd labeled facial images, each annotated with information such as age, gender, and ethnicity. The age range spans from 0 to 116 years, providing a broad spectrum of facial appearances across different stages of life. Additionally, the dataset includes variations in facial expressions, poses, and lighting conditions, making it suitable for robust training and evaluation of algorithms.

2. LITERATURE REVIEW

i. "ImageNet Classification with Deep Convolutional Neural Networks" by Krizhevsky et al. (2012): AlexNet, a deep convolutional neural network that produced ground-breaking results on the ImageNet dataset, was first introduced in this groundbreaking work. AlexNet was composed of eight layers: three fully-connected layers with max-pooling and ReLU activations, and five convolutional layers. With a top-5 error rate of 15.3% after being trained on more than 1.2 million images in 1,000 classes, AlexNet outperformed earlier techniques and showed that deep CNNs are useful for large-scale image classification.

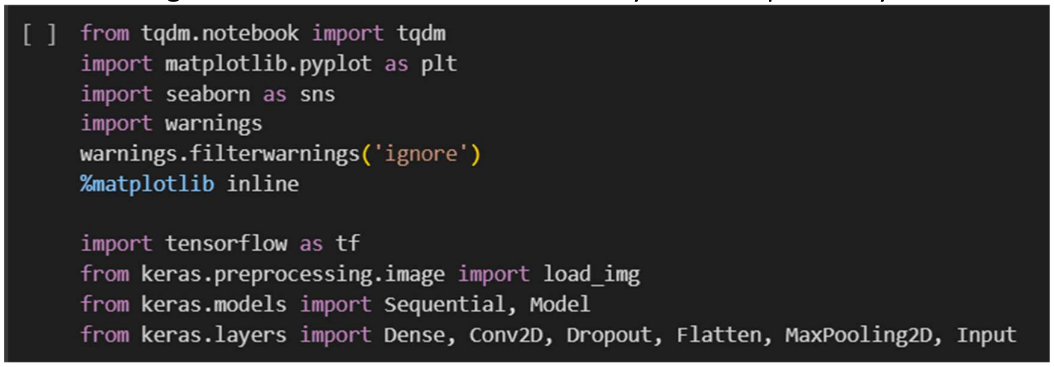
ii. "Age and Gender Classification using Convolutional Neural Networks" by Tal Hassner and Shai Harel et al. (2015): They presented a Convolutional Neural Network (CNN)-based method that uses a large dataset of more than 200,000 labeled facial images to classify people based on their age and gender. Their model used fully connected layers after a series of convolutional and pooling layers to automatically extract discriminative features from face images. The model was able to learn representations for tasks involving gender and age prediction thanks to this architecture. Test analyses proved the effectiveness of the suggested CNN-based method, obtaining cutting-edge results on benchmark datasets for gender and age categorization. The model demonstrated exceptional precision in predicting age and gender, highlighting the capacity of deep learning methodologies to extract significant features from facial photos.

iii. "NASNet: Neural Architecture Search Network" by Barret Zoph et al. (2018): In this paper, a specialized neural architecture search strategy and reinforcement learning were used to discover the CNN architecture known as NASNet. Recurrent neural networks were trained to create model architectures as part of the search process, and their performance on the ImageNet dataset was predicted by a different CNN. With a top-1 accuracy of 82.7% on ImageNet, NASNet outperformed manually constructed mobile models like MobileNet and intricately designed architectures like Inception-v4. This was state-of-the-art performance.

3. METHODOLOGY

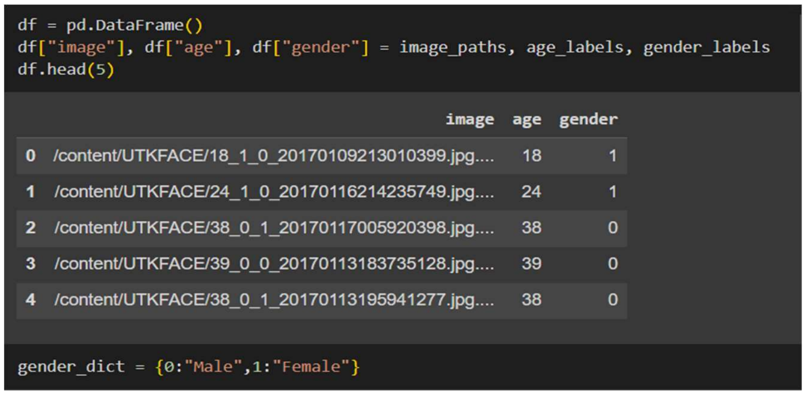
3.1 Data Collection and Preprocessing:

Cleanse and preprocess the collected data by handling missing values, removing outliers, and standardizing the features to ensure consistency and comparability across the dataset.



```
[ ]  from tqdm.notebook import tqdm
     import matplotlib.pyplot as plt
     import seaborn as sns
     import warnings
     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
```

Fig. 2 Importing Libraries



```
df = pd.DataFrame()
df["image"], df["age"], df["gender"] = image_paths, age_labels, gender_labels
df.head(5)
```

```
                                                image  age  gender
0  /content/UTKFACE/18_1_0_20170109213010399.jpg....   18       1
1  /content/UTKFACE/24_1_0_20170116214235749.jpg....   24       1
2  /content/UTKFACE/38_0_1_20170117005920398.jpg....   38       0
3  /content/UTKFACE/39_0_0_20170113183735128.jpg....   39       0
4  /content/UTKFACE/38_0_1_20170113195941277.jpg....   38       0
```

```
gender_dict = {0:"Male",1:"Female"}
```

Fig. 3 Sample

3.2 Feature Selection and Engineering:

Conduct thorough analysis and exploration of the collected data to identify relevantfeatures that may influence stock price movements.
Utilize domain knowledge and statistical techniques to engineer new features ortransform existing ones to enhance the predictive power of the model.

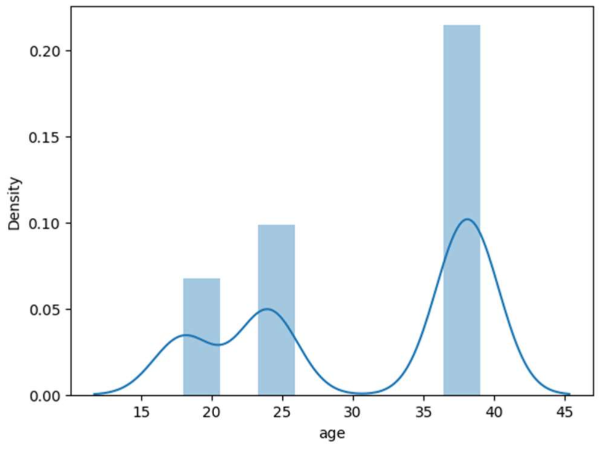


Fig. 4 Age Distribution

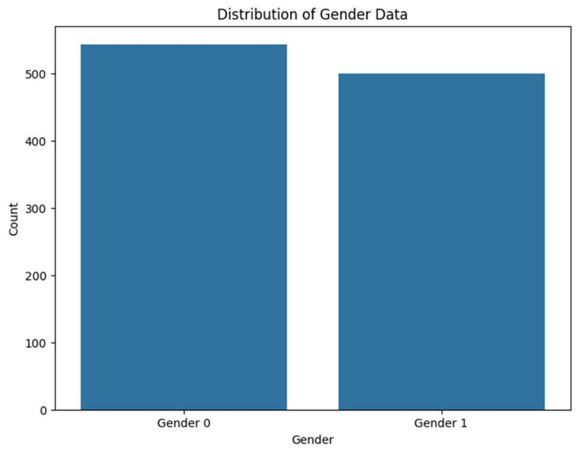
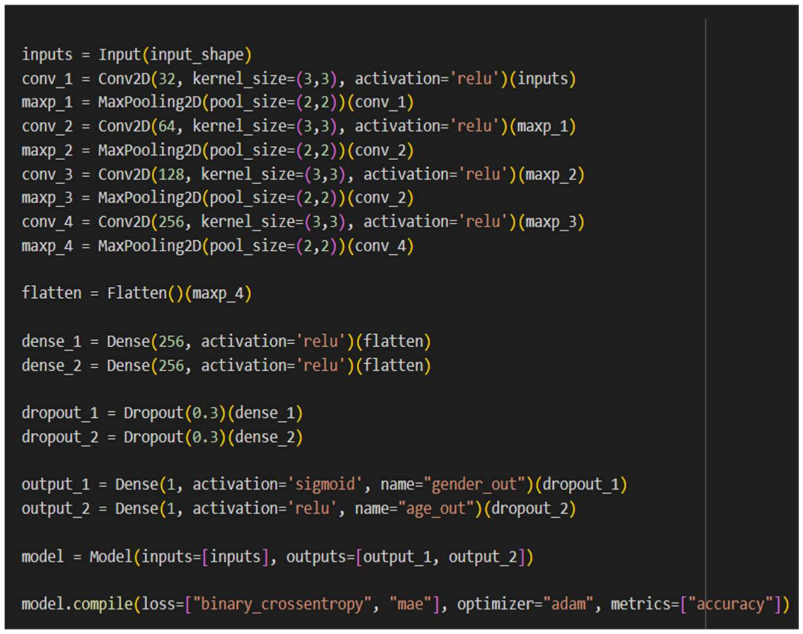


Fig. 5 Gender Distribution

3.3 Model Development:

We implemented a robust architecture comprising an input layer, convolutional layers, pooling layers, fully connected layers, and output layers. The input layer processes the facial images, which vary in terms of expressions, poses, and lighting conditions. Convolutional layers apply filters across the input images to extract features such as edges, textures, and facial patterns. Pooling layers downsample the extracted features, reducing computational complexity while retaining essential information. Fully connected layers then map these high-level features to the desired output classes, representing age groups and genders.



```
inputs = Input(input_shape)
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_2)
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)

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=["accuracy"])
```

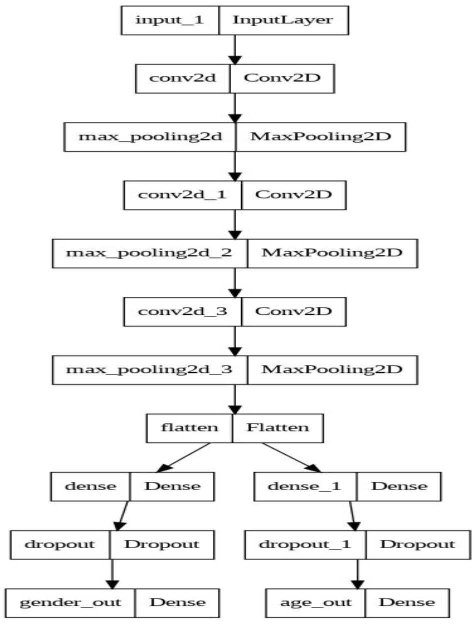
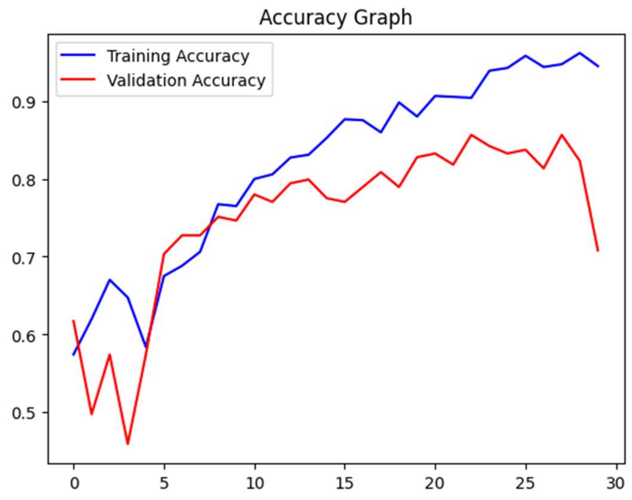


Fig. 6 CNN Model Code

Fig. 7 CNN Model Architecture



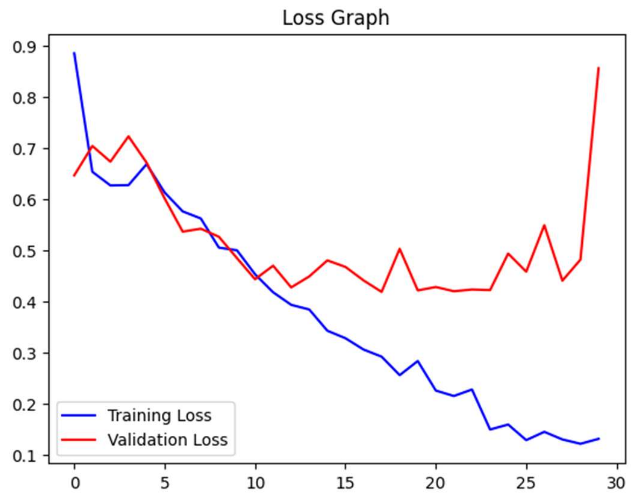
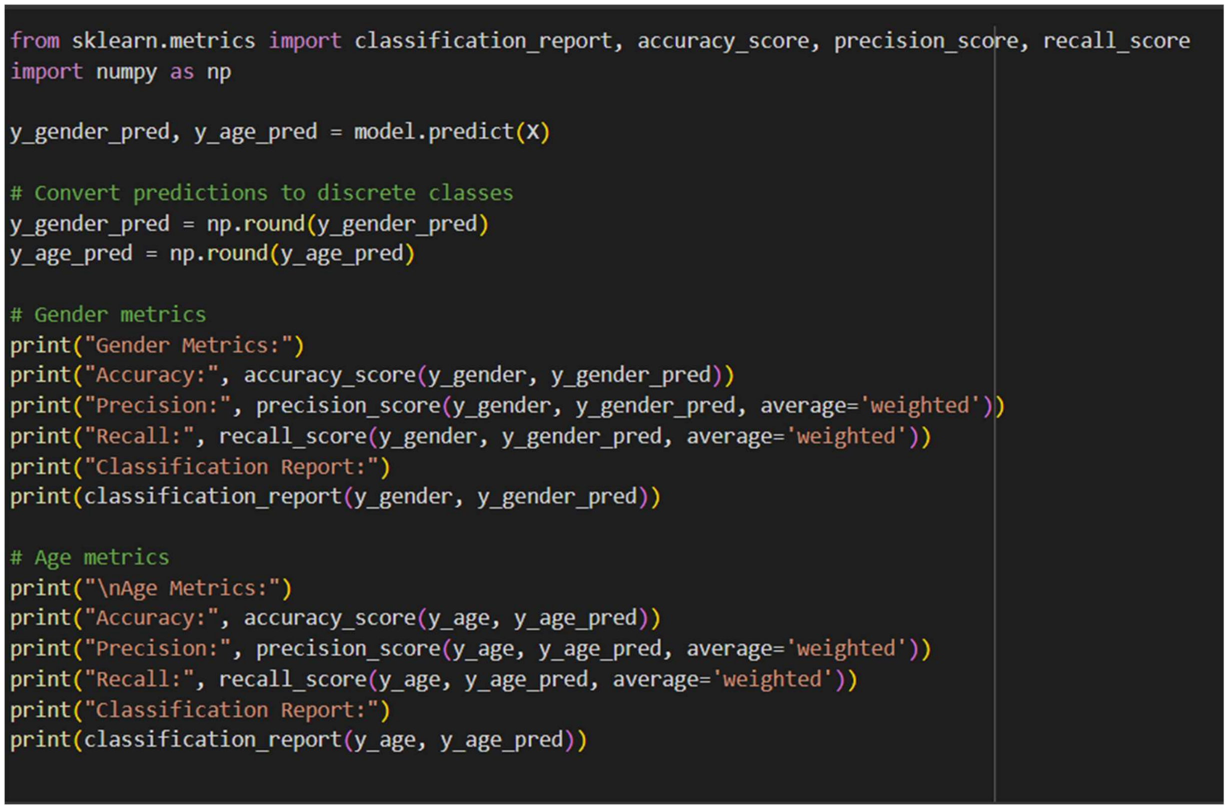


Fig. 8 Accuracy Graph **Fig. 9 Loss Graph**

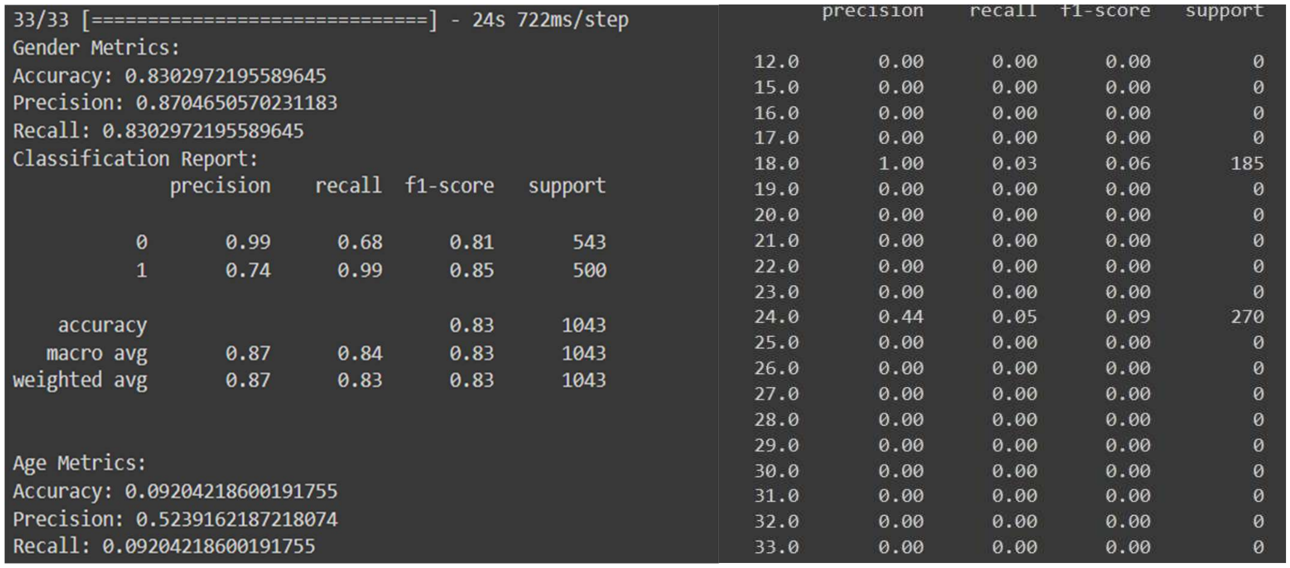
3.4 Training and Validation:

Splitting the preprocessed dataset into training, validation, and test sets, ensuring temporal consistency to reflect real-world scenarios.
Training the Convolutional neural network on the training set using backpropagation or other appropriate optimization algorithms, monitoring the validation set for early stopping to prevent overfitting.



```
from sklearn.metrics import classification_report, accuracy_score, precision_score, recall_score
import numpy as np

y_gender_pred, y_age_pred = model.predict(X)

# Convert predictions to discrete classes
y_gender_pred = np.round(y_gender_pred)
y_age_pred = np.round(y_age_pred)

# Gender metrics
print("Gender Metrics:")
print("Accuracy:", accuracy_score(y_gender, y_gender_pred))
print("Precision:", precision_score(y_gender, y_gender_pred, average='weighted'))
print("Recall:", recall_score(y_gender, y_gender_pred, average='weighted'))
print("Classification Report:")
print(classification_report(y_gender, y_gender_pred))

# Age metrics
print("\nAge Metrics:")
print("Accuracy:", accuracy_score(y_age, y_age_pred))
print("Precision:", precision_score(y_age, y_age_pred, average='weighted'))
print("Recall:", recall_score(y_age, y_age_pred, average='weighted'))
print("Classification Report:")
print(classification_report(y_age, y_age_pred))
```

Fig. 10 Evaluation

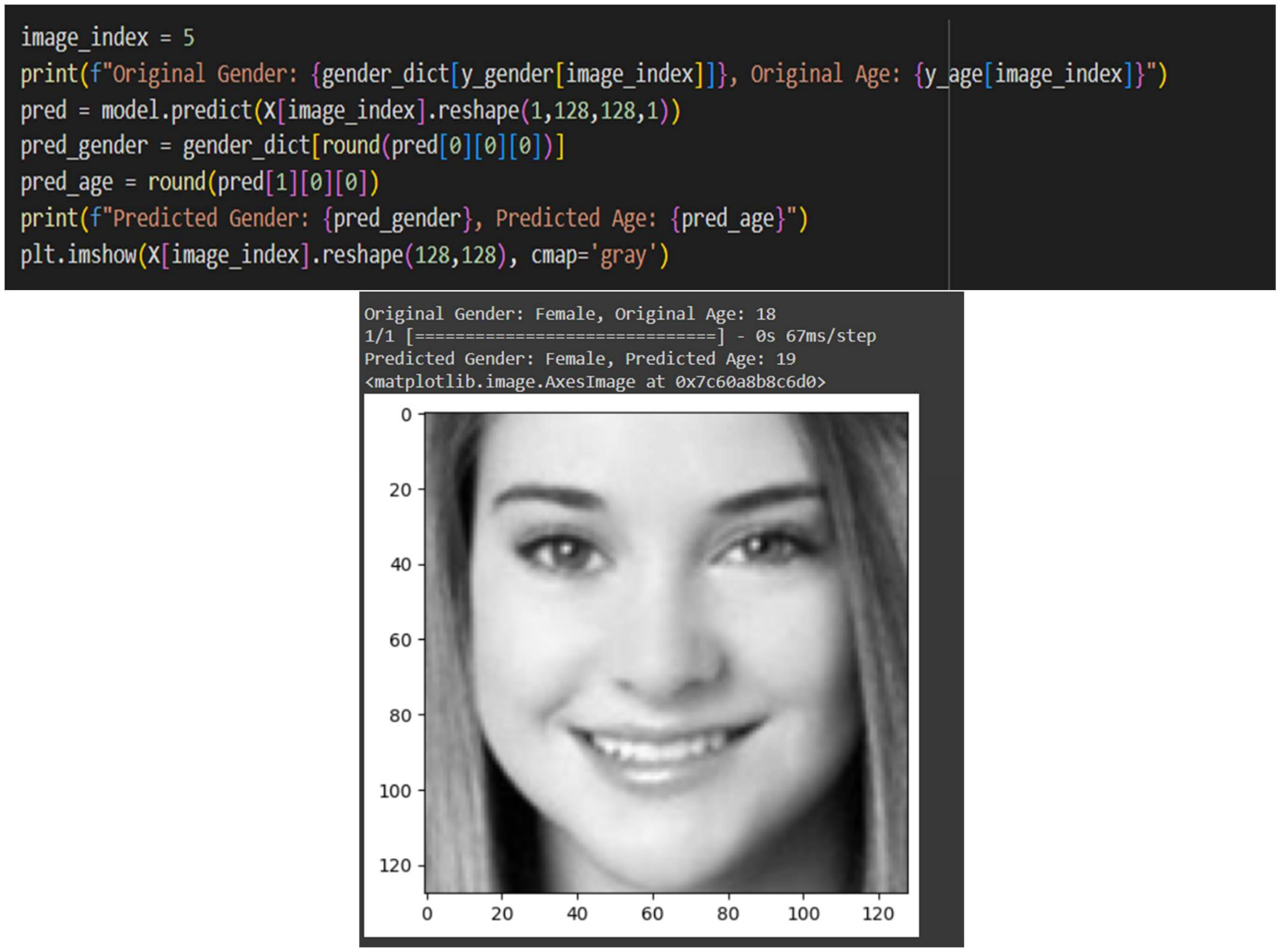


```
33/33 [==============================] - 24s 722ms/step
Gender Metrics:
Accuracy: 0.8302972195589645
Precision: 0.8704650570231183
Recall: 0.8302972195589645
Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.68      0.81       543
           1       0.74      0.99      0.85       500

    accuracy                           0.83      1043
   macro avg       0.87      0.84      0.83      1043
weighted avg       0.87      0.83      0.83      1043


Age Metrics:
Accuracy: 0.09204218600191755
Precision: 0.5239162187218074
Recall: 0.09204218600191755
```

```
              precision    recall  f1-score   support

        12.0       0.00      0.00      0.00         0
        15.0       0.00      0.00      0.00         0
        16.0       0.00      0.00      0.00         0
        17.0       0.00      0.00      0.00         0
        18.0       1.00      0.03      0.06       185
        19.0       0.00      0.00      0.00         0
        20.0       0.00      0.00      0.00         0
        21.0       0.00      0.00      0.00         0
        22.0       0.00      0.00      0.00         0
        23.0       0.00      0.00      0.00         0
        24.0       0.44      0.05      0.09       270
        25.0       0.00      0.00      0.00         0
        26.0       0.00      0.00      0.00         0
        27.0       0.00      0.00      0.00         0
        28.0       0.00      0.00      0.00         0
        29.0       0.00      0.00      0.00         0
        30.0       0.00      0.00      0.00         0
        31.0       0.00      0.00      0.00         0
        32.0       0.00      0.00      0.00         0
        33.0       0.00      0.00      0.00         0
```

Fig. 11 Evaluation Result

4. RESULTS



```python
image_index = 5
print(f"Original Gender: {gender_dict[y_gender[image_index]]}, Original Age: {y_age[image_index]}")
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(f"Predicted Gender: {pred_gender}, Predicted Age: {pred_age}")
plt.imshow(X[image_index].reshape(128,128), cmap='gray')
```

```
Original Gender: Female, Original Age: 18
1/1 [==============================] - 0s 67ms/step
Predicted Gender: Female, Predicted Age: 19
<matplotlib.image.AxesImage at 0x7c60a8b8c6d0>
```

Fig. 12 Prediction code and output

With regard to gender classification, the model's 83.03% accuracy is impressive. This means that roughly 83.03% of all the predictions made are accurate. Furthermore, the accuracy rate of 87.05% represents the percentage of accurately predicted gender classifications out of all cases assigned a particular gender. The accuracy score of 83.03% is closely matched by the recall score, which reflects the model's capacity to accurately identify instances of a specific gender. A closer look at the classification report shows that although the model performs well in terms of precision and recall for the gender label "1," which denotes male gender, it performs slightly worse in terms of correctly classifying gender "0," which denotes female gender, as shown by the precision of 99% and 74%, respectively. All things considered, the gender classification model exhibits an accuracy of 83.03%.

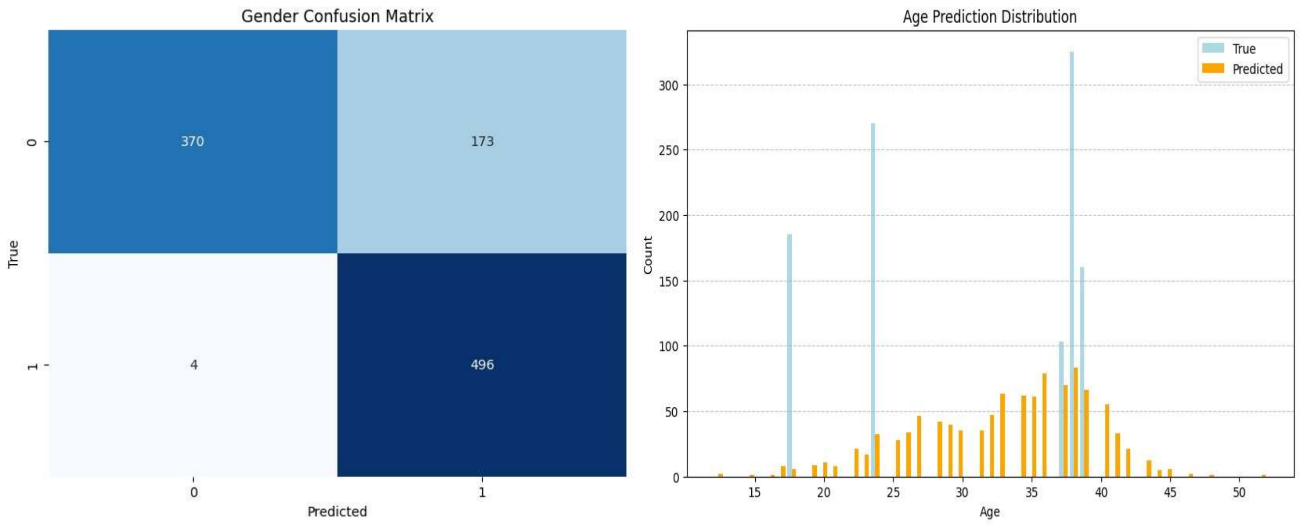


Fig 13. Gender Confusion Matrix and Age Prediction Distribution

The provided confusion matrix represents the evaluation of a Convolutional Neural Network (CNN) model for predicting both gender and age. The matrix is divided into four quadrants. On one axis, we have the true values, representing the actual gender and age classifications, while on the other axis, we have the predicted values, indicating the model's predictions.

In the top left quadrant, we have the true positive count for gender prediction, indicating that the model correctly predicted 370 instances of one gender category. Moving to the top right quadrant, we have the false negative count for gender prediction, suggesting that the model misclassified 173 instances of the other gender category.

In the bottom left quadrant, we see the false positive count for gender prediction, showing that the model incorrectly predicted 4 instances of one gender when they actually belonged to the other gender category. Finally, in the bottom right quadrant, we have the true negative count for gender prediction, indicating that the model correctly classified 496 instances of the other gender category.

5. CONCLUSION

To sum up, the Convolutional Neural Network (CNN) model created for classifying people based on their age and gender shows encouraging performance metrics, providing important information about demographic characteristics from facial images. The model demonstrates a strong ability to distinguish between male and female subjects, as evidenced by its 83.03% gender classification accuracy, high precision, and recall scores. In the future, the model's performance may be further improved by adjusting hyperparameters and making possible architectural changes. This could lead to improvements in demographic analysis, targeted marketing, and customized user experiences.

All things considered, the CNN-based method of classifying people based on their age and gender has a great deal of promise for a variety of practical uses. It will also spur innovation in computer vision and artificial intelligence while opening the door to more inclusive and precise demographic analysis.

6. REFERENCE

- Tal Hassner and Shai Harel et al. [2015] "Age and Gender Classification using Convolutional Neural Networks"
- Barret Zoph et al. [2018] "NASNet: Neural Architecture Search Network"
- Krizhevsky et al. [2012] “ImageNet Classification with Deep Convolutional Neural Networks"