Gender Classification of Image Data

Godwin Anguzu | Haoliang Jiang | Emma Mavis | Nansu Wang

Machine Learning, Spring 2022

1. Abstract

We present an analysis of gender classification on image data that is imbalanced across race, gender, and age. The purpose is to emphasize how the imbalance greatly impacts model performance, despite the reality that image classification applications still operate on imbalanced data. We ran experiments across extremely skewed gender and racially imbalanced groups of image data. We first generated baseline models and narrowed down the choice of a neural network on the data set with no experimentation. Then we applied the best performing neural network model to various imbalanced data and compared the performance on the different subgroups. We found that model performance differs greatly across different subgroups, implying that equitable representation in data is of the utmost importance for model generalizability across subgroups.

2. Introduction

The world of image classification is filled with many experiments and justifications of how important facial recognition technology is in a progressing world. The specific classifications of gender, age, and race aid in this branch of machine learning applications, opening new frontiers of research and data ethics.

The use of facial recognition techniques is almost useless without the use of facial attribute recognition – a very similar sounding, but different aspect of research. They are sometimes used interchangeably, but while facial recognition is usually the field that is in the news for controversial stories, facial estimation is a much less developed field of research.[i]

Because of this lack of development, facial estimation is very sensitive to bias, especially bias that goes undetected. There have been many additional papers addressing this bias, such as *Uncovering the Bias in Facial Expressions*[ii], *Effects of the Facial and Racial Features on Gender Classification*[iii], and *Ethical Considerations of Facial Classification: Reducing Racial Bias in AI*[iv]. We are therefore motivated by exploring this bias on our own and have developed an experiment structure to highlight this phenomenon.

3. Background

This field began as early as the 1960s when researchers began using computers to recognize the human face. It was not until the 1980s when linear algebra applications were put to this problem, allowing great advancements to occur moving into the new century[v].

It may not come as a surprise that one of the key turning points in the history of facial recognition created a controversy: in 2001 law enforcement used this technology on crowds at the Superbowl that year, invoking critics to call attention to citizens' Fourth Amendment rights[vi]. There are endless examples of police departments, technology giants, and advertisers using this technology over the last two decades that call into question the ethics of image classification[vii]. Along with privacy violations, very many of the classification models have proven to be biased, which we will be exploring in this paper.

4. Data

The dataset we worked with consists of 200MB of face images[viii]. Each row of the data is a face with pre-labelled gender, ethnicity, and age. The last column of the dataset is already-vectorized pixel data of each image. This means we avoid the processes of pooling, compression, and vectorization of images.

One limitation starting out is that this data only accounts for four ethnicity labels: White, Black, Asian, Indian, and Others. Therefore, we are not able to distinguish ethnicities such as Hispanic, Middle Eastern, Pacific Islander, etc. with just this data. Additionally, we chose to combine Asian and Indian images into the same "Asian" group.

4.1 *Exploratory Data Analysis*

Of the three specified races, more than half (54%) of the images were labelled White, 31% were labelled Asian, and only 4% were labelled Black. Additionally, females made up most of the images, 56%, and the remaining 44% were labelled male. We binned the age label into four groups: children, 0-13 years old; adolescents, 14-25 years old; adults, 26-45; and mature adults, 46 and above. Age group was composed of 33% children, 18% adolescents, 22% adults, and 27% mature adults.

Figures 1 – 3, show an under representation of the black racial group, the male gender, and the adolescent age group. If this imbalance is not taken care of, it will affect the proper detection of these less represented groups. To demonstrate how drastic this performance capability is, we intentionally perform modelling with these imbalances and show you how attempting to classify these groups, even with the best models, leads to bias in classification of the less represented groups.

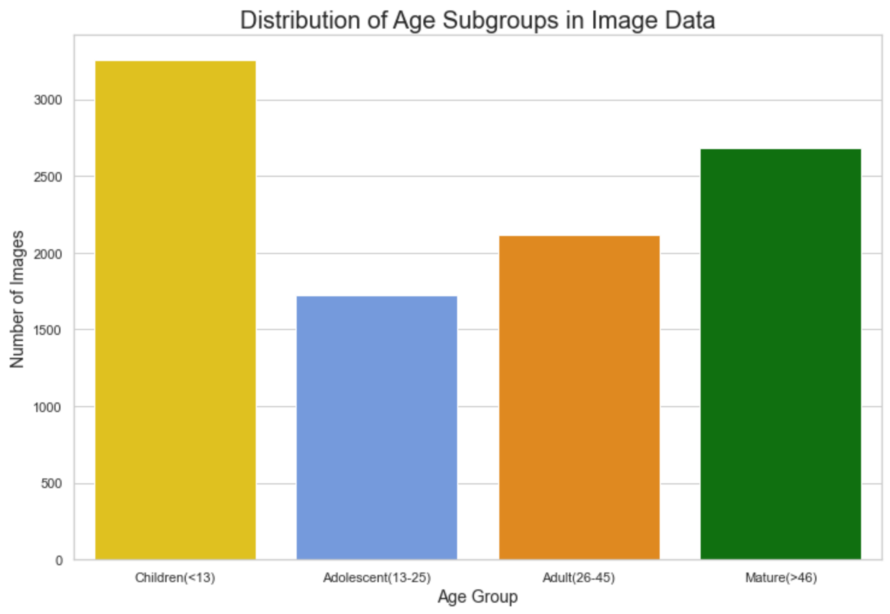

Figure 1: Distribution of age groups within image data

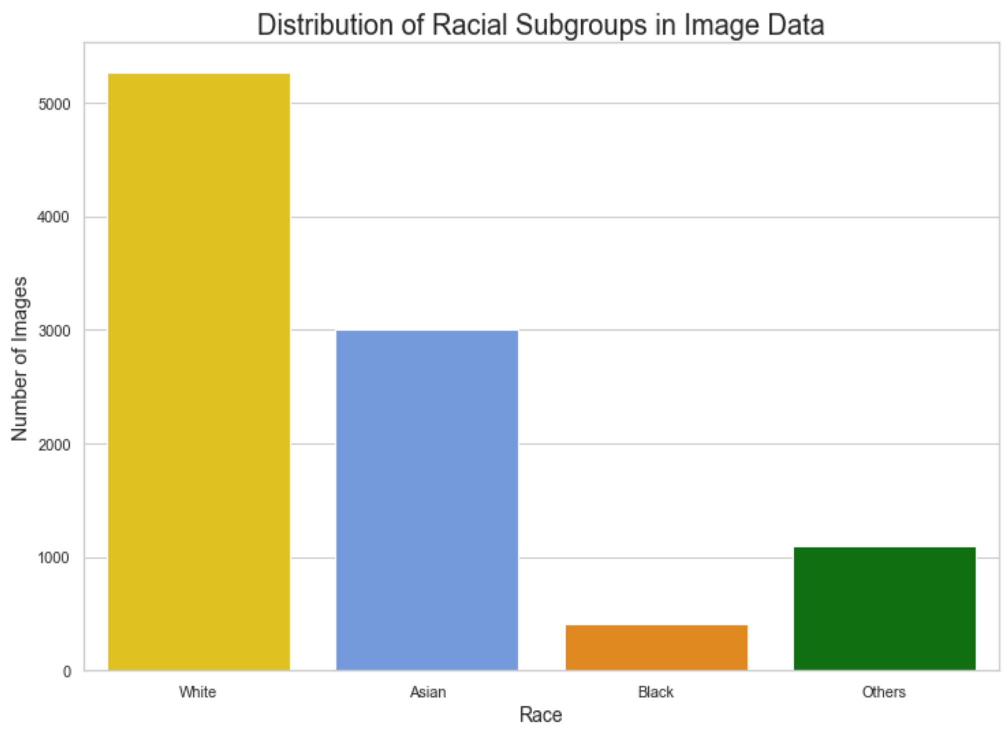

Figure 2: Distribution of racial groups within image data

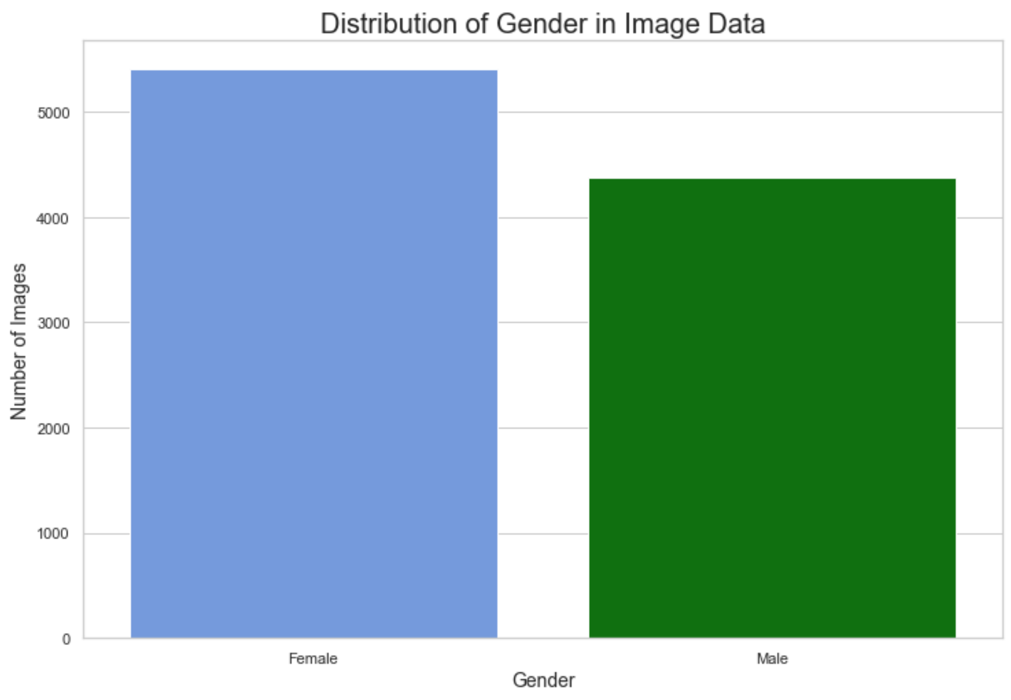

Figure 3: Distribution of gender within image data

5. Methods

All our modelling was trained on the image data to predict whether an image is either female (1) or male (0). Below is an outline of our first three baseline models, and then the final experimentation with the neural network.

5.1 *Baseline Model 1: Logistic Regression*

We first implemented a logistic regression model to set up a baseline for the metrics. We chose logistic regression because it is not computationally expensive to train, and it is a traditional tool used for binary classification. We sampled 1,000 images from the overall dataset for our training and testing purposes due to the size of the dataset. It was impossible with our resources to train on the entire dataset using only Google Colab Pro.

Our data pre-processing consisted of splitting the 1,000-sample set into train (80%), validation (10%), and test (10%) sets. We flattened the X image data since each image was originally represented by a 200 by 200 matrix. We then normalized the X matrix, which marked the completion of data pre-processing.

As for the hyperparameter tuning of the logistic regression model, we used LASSO instead of ridge regression because of the sparsity of the data. We then chose the corresponding lib-linear solver for the model. So, the only hyperparameter that we are interested in tuning is C (the inverse of the regularization strength). We picked the optimal C according to the highest AUC score of the prediction on the validation set. The optimal AUC was found to be 0.84 and the average precision was 0.86, as shown in figure 4. The metrics look decent, which means that the simple logistic regression has basic binary classification

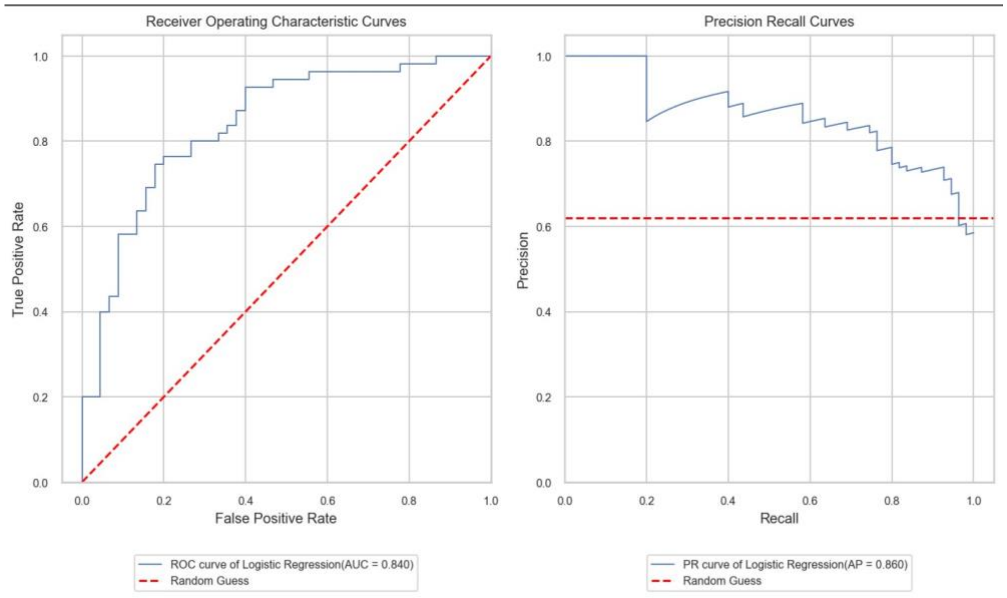

Figure 4: ROC and PR curves of logistic regression classifier

ability on gender and can serve well as the baseline model for us to later compare with the other models.

During this elementary phase of the project, we also tried to explore dimension reduction methods to see if the performance will be negatively affected. We kept only the grayscale channel and removed the RGB channels of these images. But it turned out that dimension reduction decreased model performance and we chose not to utilize grayscale images for our study.

5.2 *Baseline Model 2: Random Forest*

Because the overall data set is very large, we mostly relied on the power of Google Colab's Tesla GPU for building the second baseline model. The random forest model is built via utilizing the GPU-powered cuML module instead of the CPU powered Sci-kit learn module to increase the training speed by a factor of 45[ix]. The data processing steps are the same as what we did in the logistic regression notebook, except that the datasets were transformed to be GPU readable at the end of data processing.

We first ran a random forest model with no specified hyperparameters, and the validation accuracy turned out to be only 62%. This led us to think about that the random forest model is overfitted by the training data and thorough hyperparameter tuning and decision tree trimming should be performed. We adopted the randomized search cross validation method to narrow down the long list of hyperparameter candidates.

Next was grid search cross validation to find the best combination of hyperparameters. The grid search trimmed each decision tree in the forest by lowering the depth, but the accuracy still did not change by much. So, we performed an additional search that optimized on the AUC score instead of accuracy, just to boost the generalization performance. Even after this third round of tuning, the performance of the model still did not meet our expectation. Both the AUC and the AP scores were relatively low, as seen in figure 5.

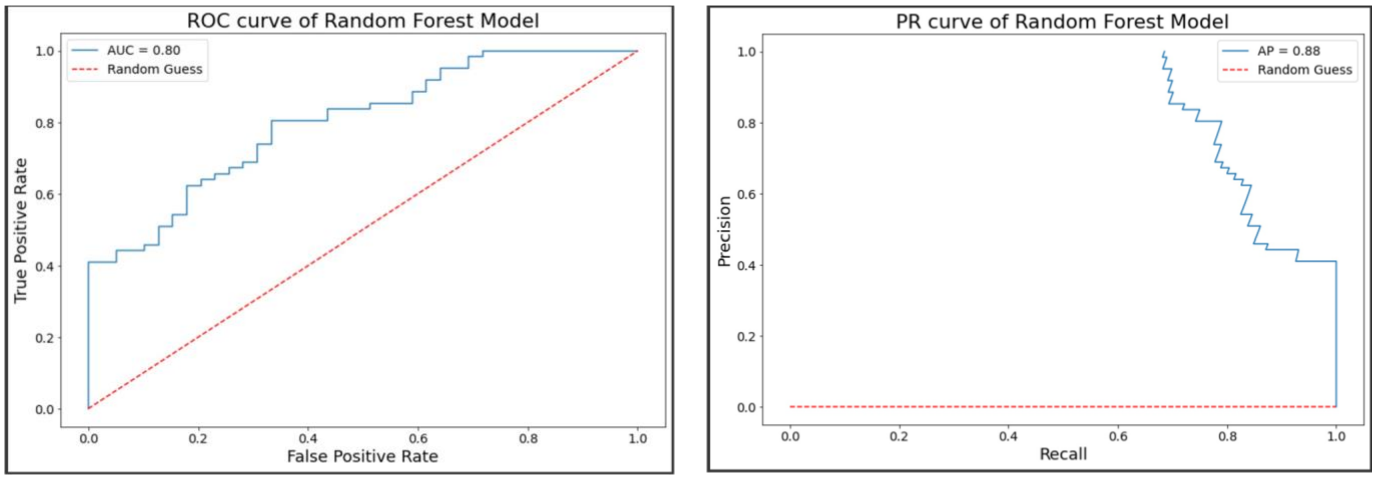

Figure 5: ROC and PR curves of random forest classifier

Overall, we believe that the random forest learned too many details on each image and thus led to very poor generalization performance. We therefore planned to treat this as our second baseline model and continued to explore further options.

5.3 *Baseline Model 3: Neural Network*

Neural networks were the third model we explored for this problem. Since there are many well-defined neural network structures for computer vision tasks, we decided to use a transfer learning workflow to take advantage of those pre-trained models. The transfer learning was learned using the Keras library[x].

The learning process contains two parts: preprocessing then training and fine-tuning. In preprocessing, all the X image values were rescaled to be in the range [-1, 1]. Then, prefetching and caching methods were adopted for faster speed. Each input image was randomly augmented using random rotation and flips.

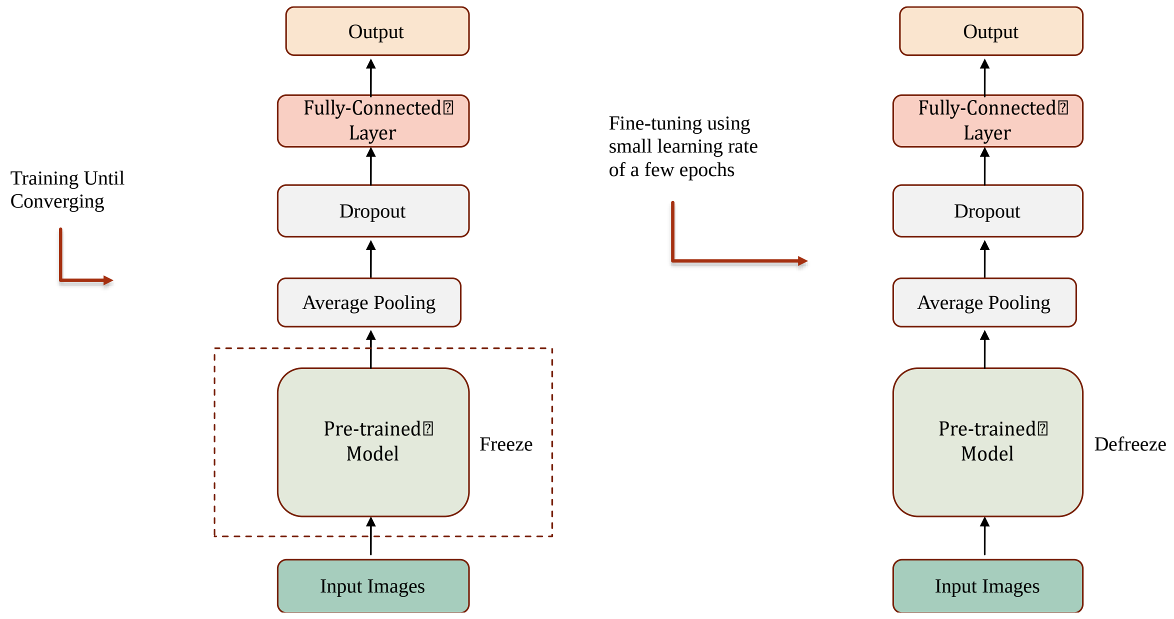

Figure 6: Transfer learning workflow that was adopted for this application

In training and fine-tuning, as shown in figure 6, our model was built around the pre-trained model to fit our input and output. Specifically, the top layer of the pre-trained model was excluded to fit our input size. After the pre-trained model, an average pooling layer, a dropout layer for regularization, and a fully connected layer to output were added.

When training, the pre-trained weights were frozen first and the model was trained until it converged. We prevented the large gradients from changing the pre-trained weights too much by first freezing them. Next, we fine-tuned the model for better performance. The pre-trained weights were defrosted, and the model was trained using a small learning rate during a few epochs.

Eight pre-trained models were trained to be the base model. As shown in figure 7, ResNet101V2[xi] was the best both in AUC and AP. More parameters do not always mean better

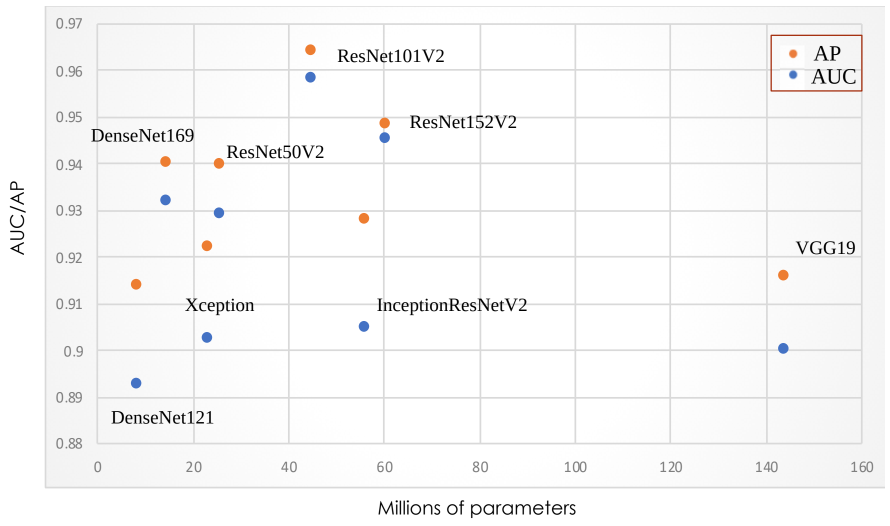

Figure 7: Performance comparison of eight, pre-trained neural networks

performance. Bias and variance tradeoff was considered here.

For hyper-parameters, dropout rates were tuned manually using a subset of random samples due to the computation cost. In the final model, the 0.1 dropout rate was selected. It

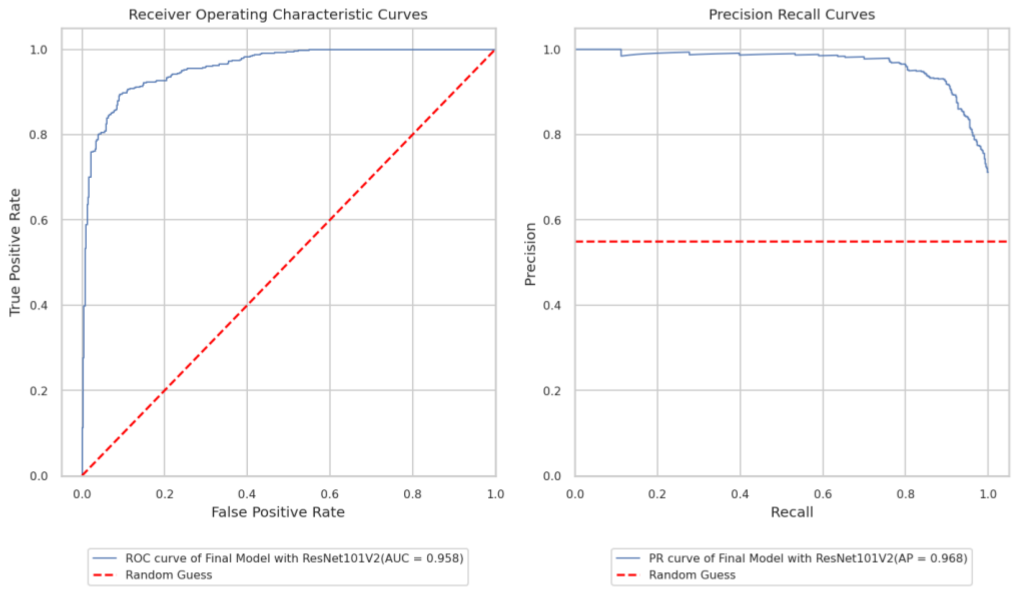

Figure 8: ROC and PR curves for the neural network classifier

usually took 20 epochs for the model to converge. When fine-tuning, the learning rate for fine-tuning was set to be 10^{-5}, and the optimizer was Adam[xii]. In the final model, ten epochs of fine-tuning led to a 0.0778 improvement in the case of binary accuracy in the validation set (from 0.7883 to 0.8661).

The final model achieved 0.958 AUC in the ROC curve and 0.968 AP in the PR curve, as seen in figure 8. It greatly outperformed the other two models that were explored in this project, and therefore we will use the neural network for the experimentation portion of this analysis.

5.4 Experimentation: Neural network applied to imbalanced data sets

As explained in our introduction, we chose to create synthetically imbalanced training data to evaluate the performance on several different subpopulations. We applied our best-performing baseline model to this training data. The purpose of this is to illustrate how even a very well-performing classification model can falter when presented with biased data.

To explore this phenomenon within racial subgroups, a held-out method was adopted. We would leave out one race subgroup, e.g., White, train the model on that subsetted data, then test it on each of the subgroups and evaluate its performance.

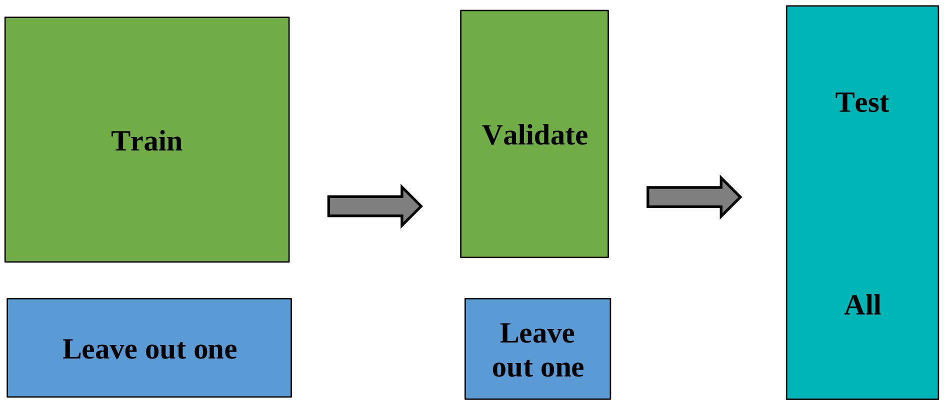

Figure 9: Flowchart depicting the held-out subgroup experimentation method

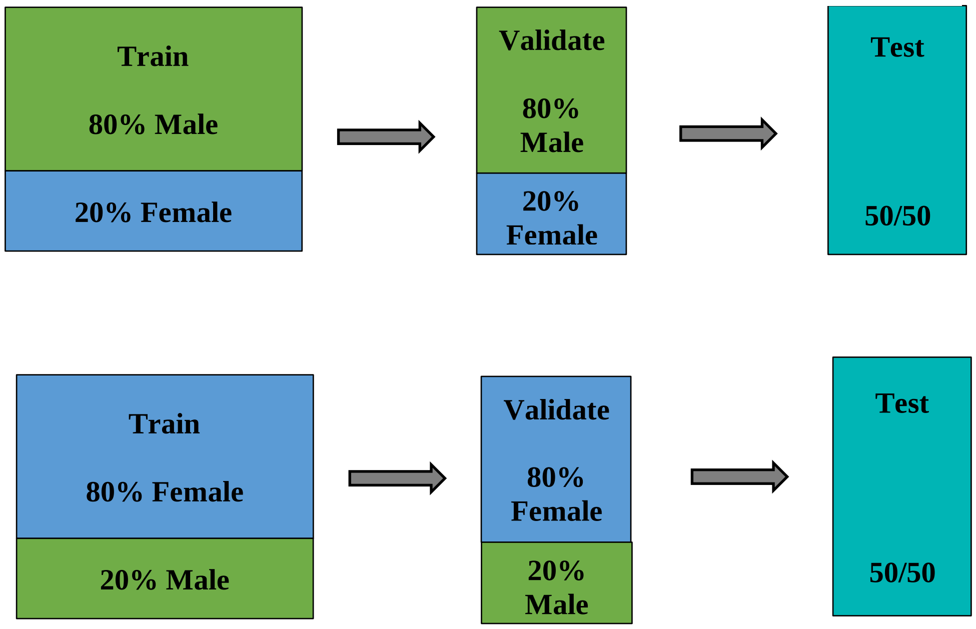

Figure 10: Flowchart depicting the skewed gender experimentation method

For exploration of gender imbalance, we created skewed data sets. We would constrain the data to be 80% female and 20% male, and vice versa, train the model, and then evaluate the performance of each one on the test data.

6. Results

Figure 11 shows the results of each held-out racial subgroup applied on each of the separate racial subgroups. The rows indicate which race was held out, and the columns indicate the race that the model was evaluated on. We can see that the performance values very closely mirror the relative density distribution of the racial subgroups.

Subgroup Evaluated On

AUC/AP

		White	Black	Asian	Other
Subgroup Held Out	White	0.94, 0.95	0.77, 0.74	0.93, 0.95	0.94, 0.96
	Black	0.98, 0.99	0.86, 0.84	0.93, 0.95	0.95, 0.96
	Asian	0.97, 0.98	0.87, 0.85	0.91, 0.94	0.94, 0.95
	Other	0.98, 0.98	0.87, 0.85	0.93, 0.95	0.94, 0.93

Figure 11: AUC and AP values from racial subgroup experiment

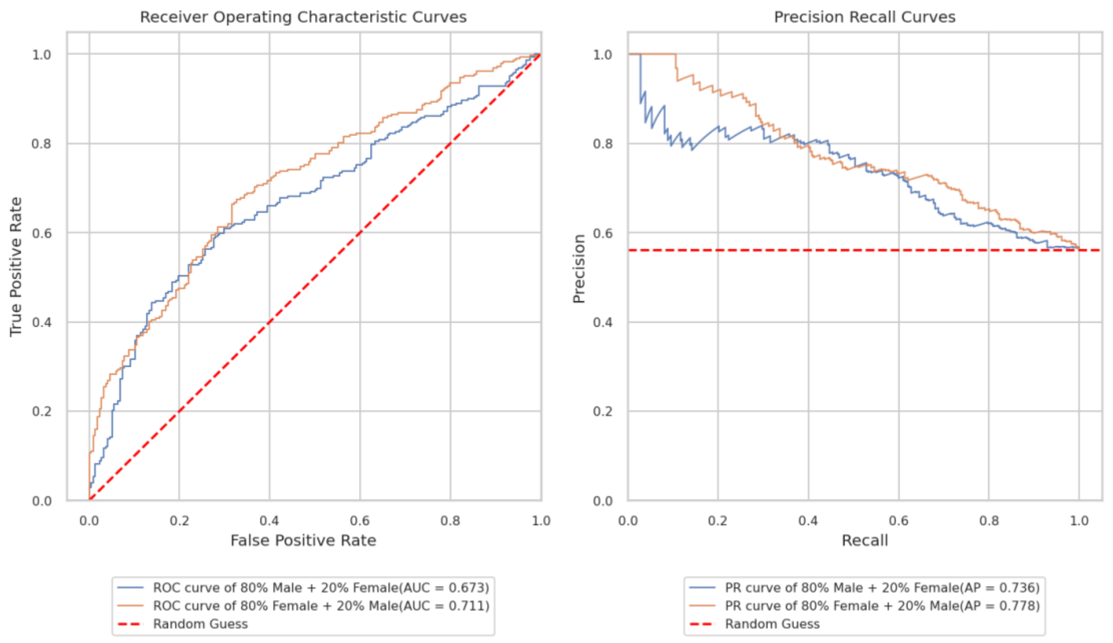

Figure 12: ROC and PR curves from the skewed gender experimentation

The most important thing to note from figure 11 is that the performance values differ across each of the racial groups, especially for the Black subgroup. This is a strong indication that our initial hypothesizing about model performance on biased data does change, even when the model has performed well initially.

Figure 12 shows that the model that was trained on the heavily skewed female data performed better than the model that was trained on the heavily skewed male data. Again, these performance values follow the relative density distribution of gender, like race. This makes sense because the data was originally imbalanced towards females, showing once more that even a well-performing model is extremely sensitive to the data it is fed.

7. Conclusions

This experiment has clearly shown that even the best-performing models are still quite sensitive to the data they are trained on, especially if that data is biased. We have seen drastic performance differences when applying a model trained on a subset of racial groups, and most notably seen that the performance on the Black subgroup was low each time. It appears that the classification of gender is not as robust as one might think – to perform well on all groups of people, it needs to train equally on all groups of people.

We can confidently state that poor performance across subgroups by neural network classifiers can be attributed to imbalance in those groups, specifically race in gender in this case. Specifically balancing racial groups before facial estimation models is vital to achieve better classification results. But of course reality hardly lends itself to provide balanced data, so methods for mitigating bias need to be at the forefront of any machine learning pipeline.

8. Roles

Administrative:

- Team Coordinator (Godwin):
  - Schedule team meetings
  - Initiate dialogue with professor for questions
  - Check in with team members on progress
  - Update timeline as needed
  - Touch base with members on meeting minutes when they are absent
- Media Coordinator (Haoliang):
  - Gather materials for report
  - Gather reference materials
  - Create presentations
  - Record videos if needed
- Code Overseer (Nansu):
  - Organize Github repo
  - Create intuitive file structure
  - Help fix bugs
  - Act as task delineator if needed for further technical needs
- Writer/Project Tracker (Emma):
  - Keep notes on all project processes for final report

    - Make sure all project processes stay in a coherent timeline
    - Support team coordinator
    - Act as overseer on final report

Technical:

- Everyone must complete EDA on their own to become familiar with data
- Everyone must implement the logistic regression model and random guess model on their own
- Further model implementation will be divided amongst team members
    - Complex and/or time-consuming models can be divided to two members
- Division of remaining technical duties TBD (ask professor and TA's for suggestions)

9. References

[i] Eidenger, Eran. “Age and Gender Estimation of Unfiltered Faces.” *IEEE Xplore*, https://ieeexplore.ieee.org/document/6906255.

[ii] Deuschel, Jessica. “Uncovering the Bias in Facial Expressions - Arxiv.org.” *Arvix.org*, https://arxiv.org/pdf/2011.11311.pdf.

[iii] Ozbudak, Ozlem. “Effects of the Facial and Racial Features on Gender Classification.” *IEEE Xplore*, https://ieeexplore.ieee.org/document/5476346.

[iv] Angileri, John. “Ethical Considerations of Facial Classification: Reducing Racial Bias in AI.” *Researchgate*, Dec. 2019, https://www.researchgate.net/publication/338225415.

[v] “A Brief History of Facial Recognition - NEC New Zealand.” *NEC*, 14 May 2021, https://www.nec.co.nz/market-leadership/publications-media/a-brief-history-of-facial-recognition/.

[vi] “Facial Recognition Is Everywhere. Here's What We Can Do about It.” *The New York Times*, The New York Times, 15 July 2020, https://www.nytimes.com/wirecutter/blog/how-facial-recognition-works/.

[vii] Lindholm, Andreas. “Machine Learning - a First Course for Engineers and Scientists.” *Sml*, Cambridge University Press, 30 Apr. 2021, https://smlbook.org/.

[viii] *Utkface*, https://susanqq.github.io/UTKFace/.

[ix] “Accelerating Random Forests up to 45x Using Cuml.” *NVIDIA Technical Blog*, 14 July 2021, https://developer.nvidia.com/blog/accelerating-random-forests-up-to-45x-using-cuml/.

[x] fchollet, "Transfer learning & fine-tuning." Keras. Updated May 12, 2020 [Website]. Available: https://keras.io/guides/transfer_learning/, Accessed on: April 12, 2022.

[xi] K. He, X. Zhang, S. Ren, J. Sun, "Identity mappings in deep residual networks". In: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, vol. 9908, pp. 630-645, 2016. https://doi.org/10.1007/978-3-319-46493-0_38

[xii] "Keras Applications." Keras. [Website]. Available: https://keras.io/api/applications/, Accessed on: April 12, 2022.