Investigating discrimination bias in predictive modelling

Jonathan Rittmo Sara Thiringer

2020-10-27

Introduction

As more data and consequently more data-driven decisions have entered the world, the problem with algorithms reinforcing discriminatory structures have received an increasing amount of attention. Several cases showcase how algorithms that were designed to be neutral decision-makers have made discriminatory predictions. Examples include facial recognition being less accurate for people with darker skin (Buolamwini & Gebru, 2018), ads for higher paid jobs being shown more frequently to men (Datta et al., 2015), healthcare predictions underestimating the illness of black people (Obermeyer et al., 2019) as well as individual tech company scandals such as the Apple card seemingly granting men a higher credit limit than women and Amazon's automated recruitment tool unrighteously favoring men. However unintentional, these examples show the need for an awareness of discrimination and fairness when collecting data, training models and using predictions for decision-making.

In this project we have looked at methods for dealing with potential discrimination in models. First, we present x ways of measuring fairness in order to be able to evaluate it. In this part we also discuss different methods of dealing with bias. Secondly, we train several statistical models on a dataset were we know that discriminatory bias is present. We then use the R package `fairmodels` to evaluate how the model performed both in terms of accuracy and fairness. Lastly, we use a pre-processing bias mitigation method called impact disparate remover to reduce the presence of bias in the models. We evaluate the success of this tool and discuss it in relation to the different models.

For simplicity, this project focuses solely on classification methods and the case of a binary protected variable.

Understanding Fairness in Statistical Models

In order to build models that are less discriminatory than previously mentioned examples have proven to be, we first need a way to measure fairness. In recent years, a lot of work have been put into defining fairness in a quantitative way for machine learning purposes. These different definitions, listed and explained below, implicate somewhat different views on what fairness really is. This ongoing discussion draws on previous social and legal research on equality and fairness. There are several measures available for such evaluation, but for simplicity we will focus on two of the most common.

Demographic parity

This criterion states that the target variable should be independent of "protected" characteristics (such as race and gender). This means that if Y is our decision and X the protected variable in question:

$$P(Y = 1|X = 1) = P(Y = 1|X = 0)$$

This rule will enhance rules which classify the same fraction in each group as positive. This might appear unappealing, as it has a way of "forcing" equality. However, as Goel et al. (2018) point out, many classifiers impact decisions which create future labels in future training data, and as such a lack of demographic parity risks causing ever-looping cycles of discrimination. Another negative with this measure, however, is that it makes sure we cannot use X as a predictor which we in some circumstances want.

Equalised odds

This criterion proposes that the predictor we use and the protected variable should be independent conditional on the true outcome, i.e. if G is the predictor we want to use:

$$P(G = 1|X = 0, Y = 1) = P(G = 1|X = 1, Y = 1)$$

This measure ensures that the true positive rates of the classifier are the same for both groups of X. It can be seen as a measure of "equal opportunity" and hence doesn't keep models from unequal predictions (in terms of outcomes). As such, it is sometimes seen as a "weaker" measure on fairness.

Optimising for Fairness

As we already know, statistical learning may have other goals than just maximizing accuracy. For example, different costs for different types of misclassification may lead us to choose a different model than the one that has the highest overall accuracy rate (James et al., 2013). Tuning for a higher rate of fairness introduces yet another way to both pre-process data, train models and make predictions. Since discriminatory bias can be introduced in several different ways and at different stages of model training, there are several methods to optimize for fairness.

Pre-Processing

Since discrimination may have been present in both data collection and the systems (institutions, products, etc) that generated data, we might want to deal with the issue already in the pre-processing stage. Attempts to do this includes resampling and synthesizing data. For example, one approach with several suggested algorithms proposes mapping the datapoints to a new feature space were as much relevant information as possible is kept while information about the protected variable is lost (Creager et al., 2019; Feng et al., 2019; Zemel et al., 2013). Resampling, a technique commonly used to deal with undersampled groups or otherwise skewed data, have also proven to be an alternative for moderating discrimination in statistical learning. Kamiran & Calders (2012) suggests reweighing and resampling observations so as to enhance the presence of an initially undersampled groups (for example high income women). Here we will deal with three preprocessing techniques:

- Disparate impact removal

  - Introduced in Feldman et al. (2015). This is a geometric method reshaping distributions on numerical variables conditioned on the protected. It is generally an apprioriate method when there are non-protected variables that are correlation with the protected variable, which means using them for prediction introduces disparate treatment by the model.

- Two resampling methods introduced in Kamiran & Calders (2012). In a scenario with a binary protected variable and a binary classification problem the four sample sizes relating to the joint distribution of classes and protected features that would make the dataset discrimination-free are calculated. Stratified sampling from these two groups are then performed. Observations are then resampled with one out of two techniques.
  - Uniform resampling, where each observation has equal porbability to be duplicated or skipped to increase or decrease the size of a group.
  - Preferential resampling, where observations close to the decision boundary (of a classifier) gets higher priority for being duplicated or skipped.

Training

Naturally, many methods for mitigating discrimination by algorithms focus on the training phase of statistical learning. From avoiding overfitting, we know how to constrain the learning by penalizing learned parameters and thus restraining them from having an unproportionally large impact in a classifier. Among classical apporaches ridge and lasso regression can be mentioned (Hastie et al., 2009). Bias have been shown to sometimes remain in models despite the removal of the protected attribute, due to their correlation to other non-sensitive attributes (källa). In this sense, penalizing variables that are highly correlated to protected features looks like a promising solution.

A similar approach, also focusing on the learning phase, is the *weighted sum of logs technique* propsed by (Goel et al., 2018). This algorithm models both the historical bias in the data and empirical bias imposed by the original algorithm in order to establish a "poportionally fair" classifier by restricting the optimization space of a logistic regression model.

It has even been suggested that one solution could be to build a multiple of models - i.e. one for each social group. Such an approach have proved to be succesful in some cases (Calders & Verwer, 2010), but has a clear disatvantage when it comes to dealing with multiple (intertwined) protected variables.

Dut to time constraints, we are not going to investigate any of these methods in practice.

Prediction

Finally, another way to remove discriminatory outcomes is by post-processing the predictions. For example, we normally calculate the best possible cut-off value with regards to accuracy in classificaiotn models (Hastie et al. (2009)). Instead of only optimising for accuracy, we could include a measure of fairness and have it play a role in the decision of the threshold value. Another similar technique is to modify the cut-off value differently for each group, as proposed by Hardt et al. (2016), although they do point out that a better option is to first "invest in better features and more data". A benefit to this approach is that it is much less complex and hence has a more straight-forward implementation than many of the previously mentioned techinques. We will look at how changing cut-off values with regards to minimisng demographic parity and equalised odds affect model accuracy.

Analysis

Classification on the COMPAS Data

The case we have selected to use for building statistical learning models, evaluating their discrimintion bias and furthermore mitigating these effects, is the compas[1] data made available by Florida Department of Corrections and ProPublica in 2016. The dataset sourced from the `fairmodels` package, slightly modified from the original COMPAS dataset, is a collection of $n = 6172$ observations with the variables: `Two_yr_Recidivism, Number_of_Priors, Age_Above_FourtyFive, Age_Below_TwentyFive, Misdemeanor, Ethnicity, Sex`. `Two_yr_Recidivism` is the variable we want to predict. It indicates whether an inmate have made another offense within two years of release. As such, the dataset is being used to decide on the penalty of criminals. The compas data is an excellent case for studying discriminatory algorithms, as it both exhibits bias and is being used for actual classification for a highly sensitive cause. Because of this, many researchers who explore fairness in machine learning have chosen to study it as an example. For the present purpose we filtered the data on `Ethnicity`, for `Caucasian` and `African_American` to create a binary protected variable. Rendering n to a total of 5278 with no missing values. A summary of the dataset can be seen below. Data Frame Summary
COMPAS
Dimensions: 5278 x 7
Duplicates: 4706

No	Variable	Stats / Values	Freqs (% of Valid)	Missing
1	Two_yr_Recidivism [factor]	1. 0 2. 1	2795 (53.0%) 2483 (47.0%)	0 (0%)
2	Number_of_Priors [integer]	Mean (sd) : 3.5 (4.9) min < med < max: 0 < 2 < 38 IQR (CV) : 5 (1.4)	36 distinct values	0 (0%)
3	Age_Above_FourtyFive [factor]	1. 0 2. 1	4182 (79.2%) 1096 (20.8%)	0 (0%)
4	Age_Below_TwentyFive [factor]	1. 0 2. 1	4122 (78.1%) 1156 (21.9%)	0 (0%)
5	Misdemeanor [factor]	1. 0 2. 1	3440 (65.2%) 1838 (34.8%)	0 (0%)
6	Ethnicity [factor]	1. African_American 2. Caucasian	3175 (60.2%) 2103 (39.8%)	0 (0%)
7	Sex [factor]	1. Female 2. Male	1031 (19.5%) 4247 (80.5%)	0 (0%)

[1] Correctional Offender Management Profiling for Alternative Sanctions

Pre-Processing Methods for Bias Mitigation

Resampling

The two resampling methods aims to even out inequalities between the depraved and privileged groups having positive and negative outcome attributes respectively. That is, in our case, the methods try to make the joint distribution of `Ethnicity` and `Two_yr_Recidivism` uniform by duplicating some observations and removing others. In effect undersampling some groups and oversampling others. If we look at the original joint distribution of `Ethnicity` and `Two_yr_Recidivism` from the training data in table 2.

Table 2: Joint distribution of Ethnicity and Recidivism

	African_American	Caucasian
0	1202	1035
1	1317	670

Note: 1 = Recidivism

In uniform resampling all observations have equal probability of being duplicated or removed. However, in preferential resampling we fit a logisitc regression model with `Two_yr_Recidivism` as outcome variable and all the other variables as predictors, then we use the resulting probabilities as sampling probabilities, where borderline cases (i.e. cases that is on the border of being classified as either 1 or 0 in `Two_yr_Recidivism`) are duplicated or removed more frequently than "certain" cases. In table 3 the new joint distribution when using uniform resampling can be seen and in table 4 preferential sampling has been used.

Table 3: Joint distribution of Ethnicity and Recidivism, uniform resampling

	African_American	Caucasian
0	1334	903
1	1185	802

Note: 1 = Recidivism

Table 4: Joint distribution of Ethnicity and Recidivism, preferential resampling

	African_American	Caucasian
0	1334	903
1	1185	802

Note: 1 = Recidivism

As can be seen, the distributions are similar even though different observations have been resampled.

Disparate Impact Remover

Disparate impact remover, as proposed by Feldman et al. (2015), is a method to adjust distributions. The algorithm for this has been implemented in `fairmodels` and can be used on continous variables. This is unfurtunate since only one of our precidtors, `Number_of_Priors`, is non-categorical. Strictly speaking, as it is a measure of counting, it is not even truly continous. Therefore, we can expect it to have a limited effect.

In figure 1, we can see the distribution of `Number_of_Priors` amongst the two groups. We note two things. First, the groups had a difference in their respective distributions, but it wasn't very large. Secondly, applying Disparate Impact Remover seems to have introduced some kind of grouping along the range of values, leading to the vary curvy nature of the density function.

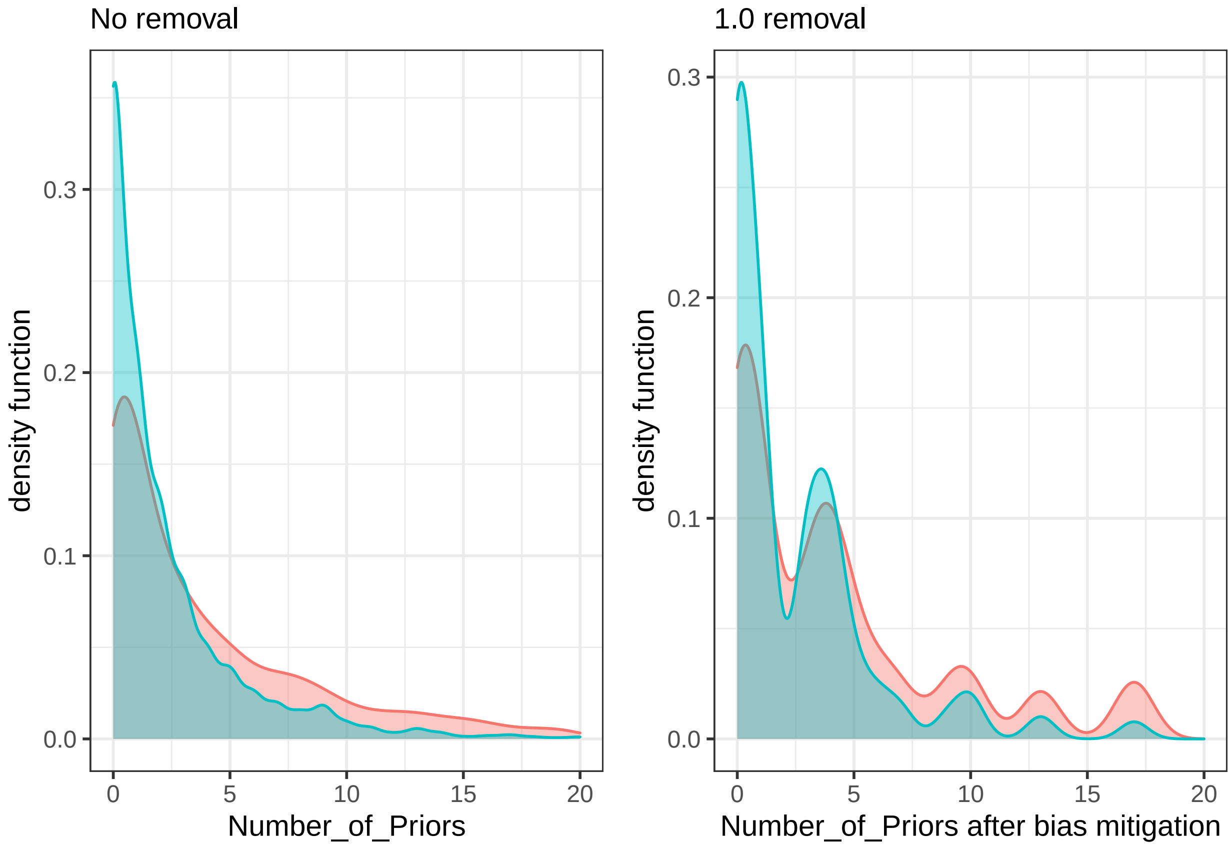

Figure 1: The distribution of `Number_of_Priors` for the two ethnicity groups. To the left, the original distirbutions. To the right, the distributions after removal of disparate impact.

Models

We trained five different models on the COMPAS data tuned for various parameters. Each of these models was then fit again but on data having undergone one of the three preprocessing mitigation methods. We fit the models using all predictors in the dataset including the protected variable. The K-nearest neighbour model turned out to perform worse than chance when fit on mitigated datasets and was therefore left out of the analysis. The dataset was split for training and testing with 20% of the observations reserved for testing.

Model	Tuning	Abbreviation
Random Forest	Number of predictors sampled at each split	RF
Artificial neural net	Number of hidden nodes	ANN
Logistic ridge regression	Penalisation	LR

Model	Tuning	Abbreviation
K-nearest neighbour (left out)	Number of neighbours	KNN
AdaBoost	Number of predictors sampled at each split	None

Model evaluation

When evaluating bias in machine learning models one often utilise the ratio of the terms on either side of the equality in e.g. demographic parity or equalised odds described previously. I.e. if

$$\frac{P(Y=1|X=0)}{P(Y=1|X=1)} = 1 \quad \text{and/or} \quad \frac{P(G=1|X=0,Y=1)}{P(G=1|X=1,Y=1)} = 1$$

we can say that our model is completely fair. In 1979 the US Equal Employment Opportunity Commission deemed that if this ratio falls below 0.8 disparate impact is present, known as the 80% rule (Feldman et al., 2015). Hence, we want a model that satisfies the above criteria as closely as possible but still yield a high accuracy. The plot below in figure 2 shows the metrics on our fairness criterias.

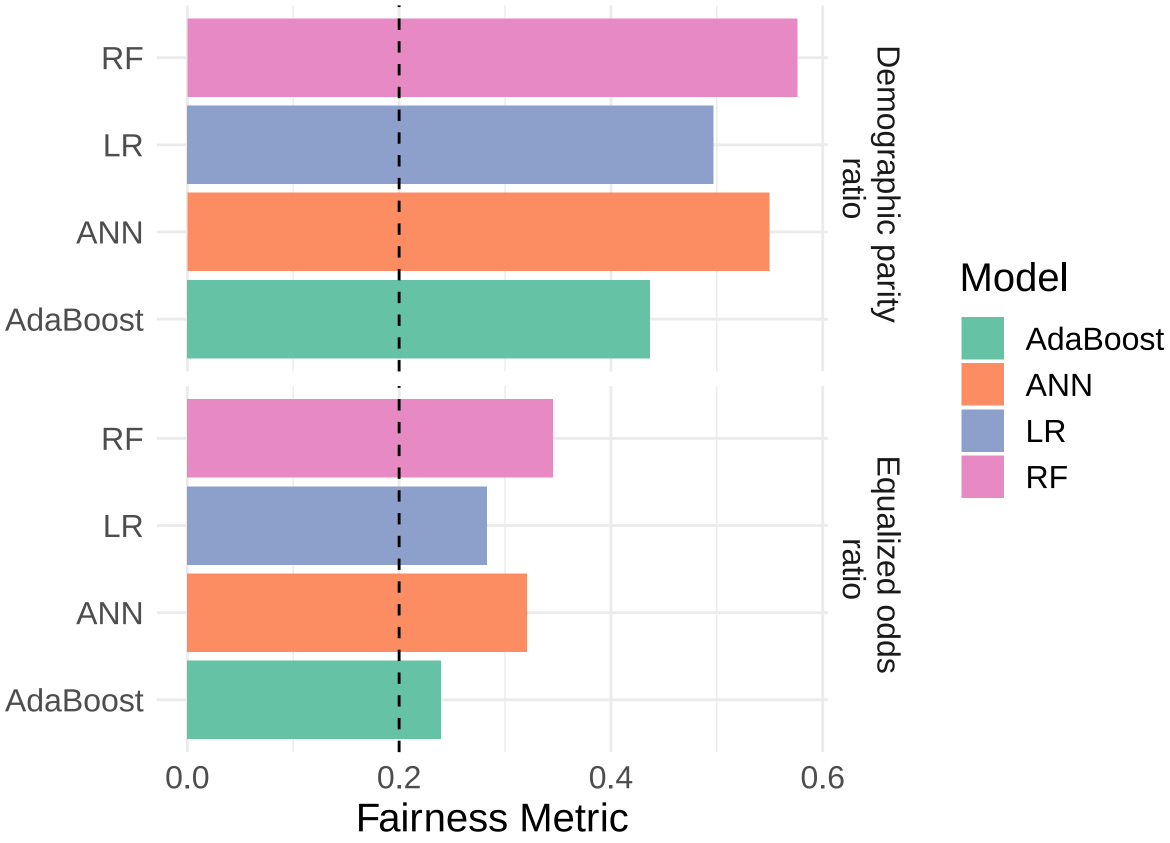

Figure 2: All models evaluated against our two measures on fairness. The 0.2 line indicates the kind of unequality we would be willing to accept.

Let us for simplicity solely consider demographic parity (this also often referred to as statistical parity) as a measure from now on. We can then compare the acquired accuracy of each model together with parity loss evaluated by this measure (note the acquired accuracy is measured on training data) as seen in figure 3.

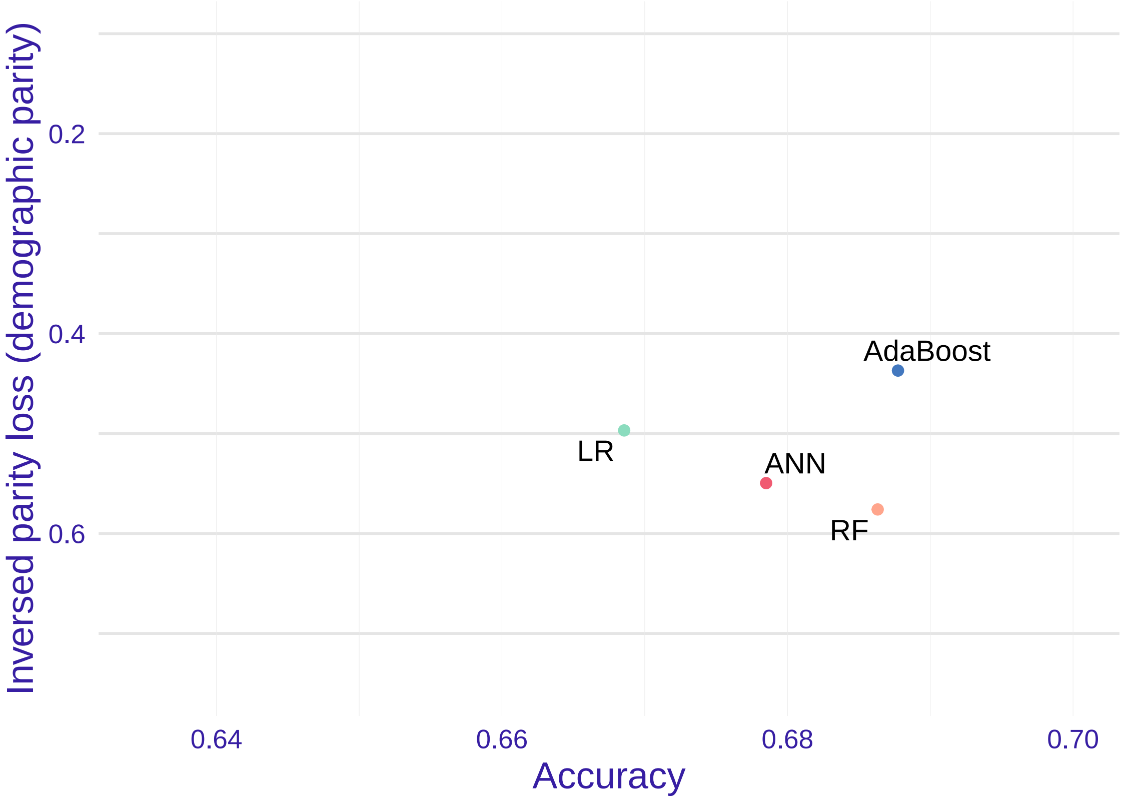

Figure 3: Accuracy and fairness comparison for training data on the fitted models. Note that the y-axis is inverted. This is not the ratio explained previously but the "parity loss" which should be interpreted as the difference between the parity ratio and 1 [IS THIS TRUE??]. Hence, a measure that we want to minimise.

Figure 3 tells us that using demographic parity as a fairness measure none of our models satisfy the 80% rule and thus disparate impact is present. Using the preprocessing techniques previously described we now fit the models on these processed datasets and compare which model and processing technique yields best results both in terms of fairness and accuracy (upper right corner of 3). The results can be seen in 4.

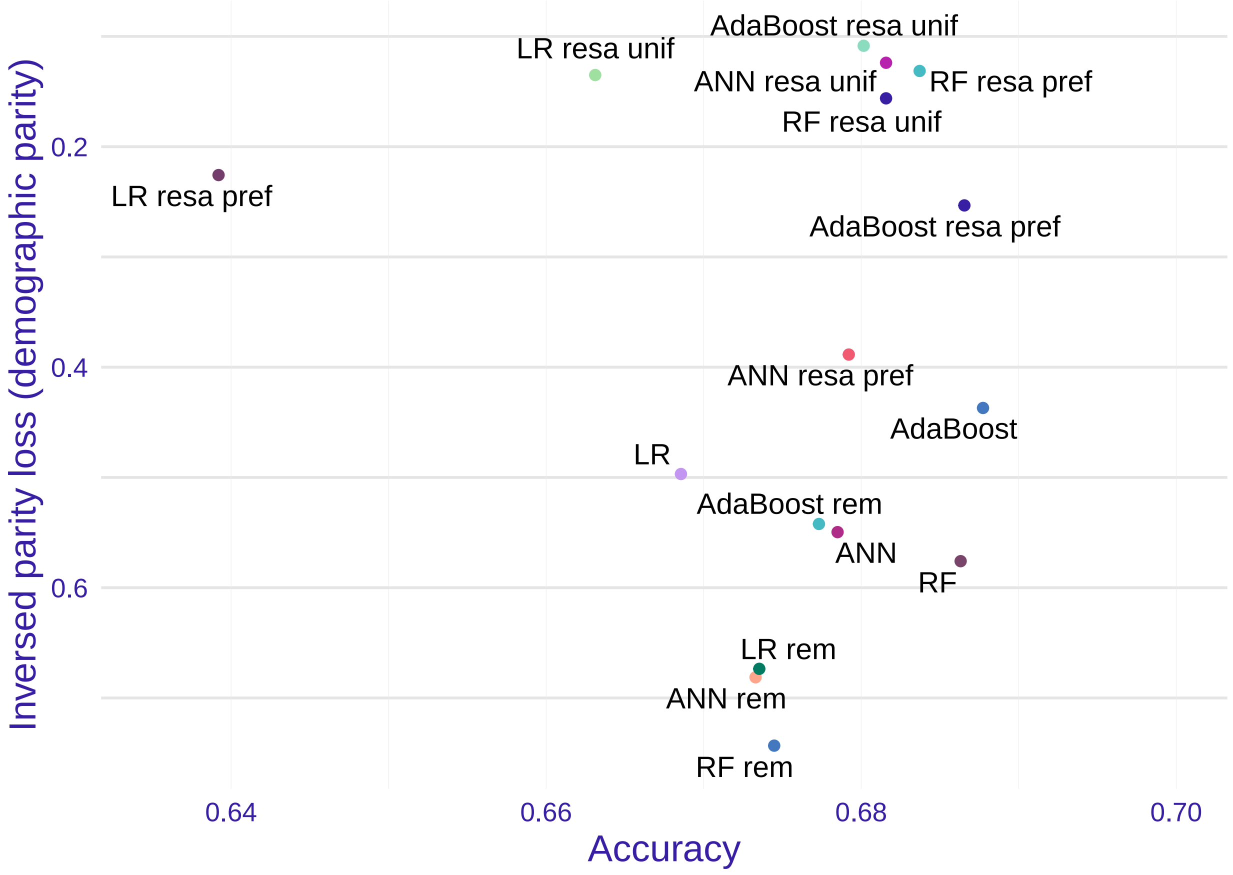

Figure 4: Accuracy and fairness comparison for preprocessed training data on the fitted models. The three methods can be seen in the suffixes to the models: "rem" = disparate impact removal, "resa pref" = preferential resampling and "resa unif" = uniform resampling. Note that the y-axis is inverted.

The resampling methods seems to have best effect on minimising parity loss and the disparate impact removal technique even increase parity loss for some models. This is probably because of the reasons stated earlier about this method. However, most of the resampled models have a parity loss of less than 0.2 (i.e. they satisfy the 80% rule). Choosing the four most accurate of these we can investigate whether applying a different probability cutoff than the conventional 0.5 would mitigate parity loss even further. In figure 5 it can be seen how the parity measures are affected by varying the cutoffs and the values that would minimise demographic parity *and* equalised odds.

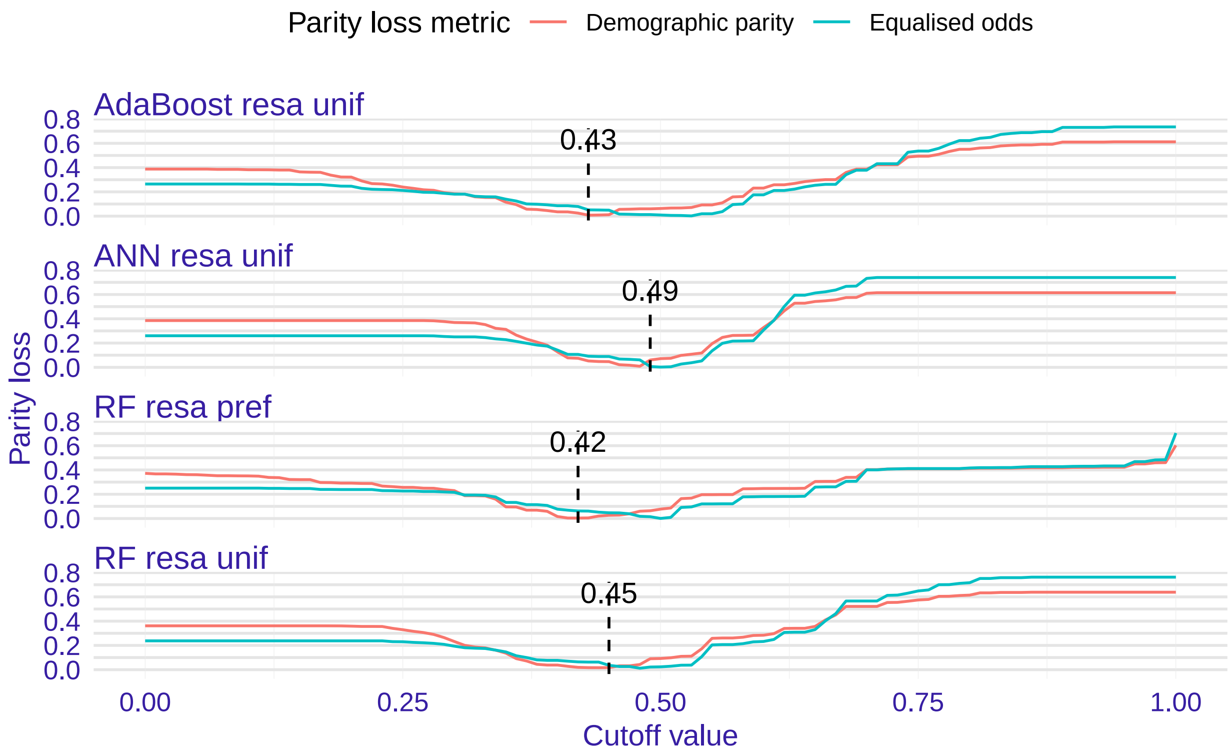

Figure 5: The four best performing models and their possible cut-off value in relation to two measures on fairness. The mark indicates the cut-off value which gives the lowest parity loss on both measures.

Applying the cutoff values from figure 5 and comparing them to models using 0.5 we can evaluate whether this drop in parity loss would be worth any potential drop in accuracy. This comparison can be seen in figure 5. Adjusting cutoffs can have quite large effect, however, since all models already satisfies the 80% rule we want to argue that the drop in accuracy is not worth it. Hence, if the goal of the analysis is to find a single model we would choose Random Forest trained on uniformly resampled data. As a coincidence it turns out that this model produces the smallet parity loss as measured by equlised odds, further increasing our confidence.

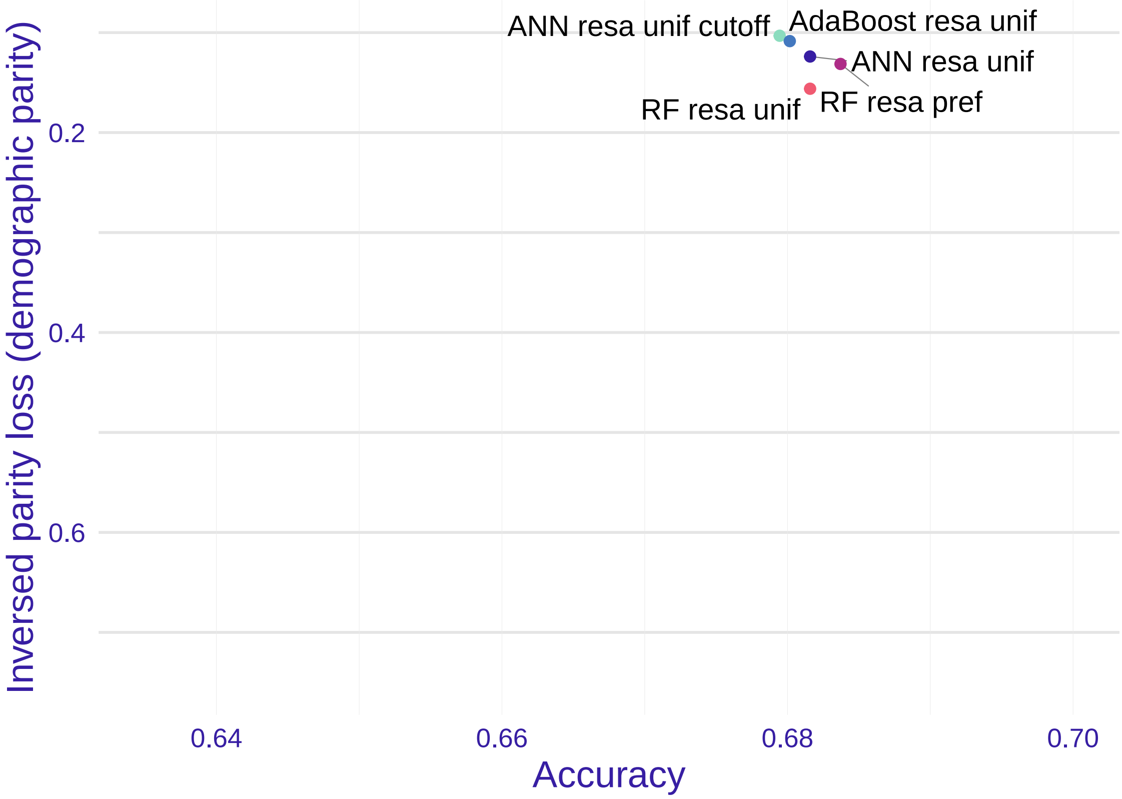

Figure 6: Plot of the four best model with and without optimal cut-off value implemented. Generally, a higher parity loss results in lower accuracy, illustrating the accuracy-fairness trade-off that can be present in statistical models.

Final Model Evaluation

Since we now have a chosen model we want to see how it performs on out of sample data. Due to the nature of our analysis it might also be interesting to see how well it performs in comparison to the other models fitted. Accuracy and parity loss as measured by demographic parity for all models using unseen testing data can be seen in figure 7 and our chosen model appears to have a satisfactory performance on test data as well, however, it just barely meets the 80% criterion and it is no longer the model with best accuracy among the ones with a demographic parity below 0.2.

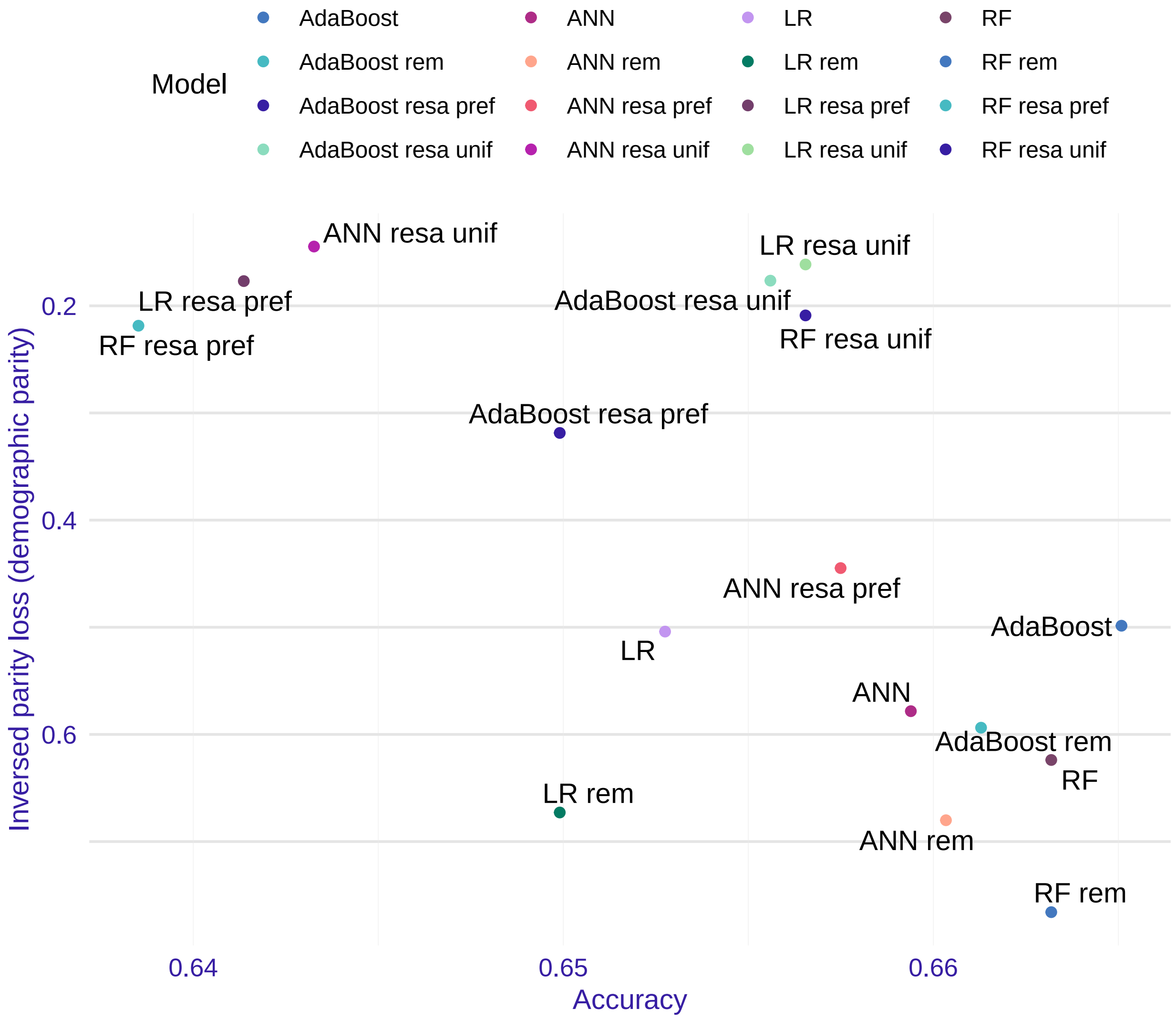

Figure 7: All fitted models and their respective bias-mitigated version, evaluated on previously unseen test data. As aspected, accuracy is slightly lower than for the training data. We can also note that only four of the 16 models has a parity loss of 0.2 or higher.

Discussion

We have in this report investigated methods to mitigate discriminatory bias in machine learning. Specifically we have analysed the COMPAS dataset to predict whether an offender will commit another crime within two years after release. This is obviously useful information when judging whether to release inmates or not. However, to do this we want models that are accurate but do not take certain sensitive variables into account when making predictions. Several measures are available to estimate fairness in models each with slightly different interpretation. We have only addressed two of these measures in this report and even further limited our analysis by mainly taking demographic parity into account when evaluating models. Our results show that it is possible to limit the effect of unfair models to a great extent without losing too much accuracy. However, caution is advised for using this data to inform decisions as sensitive as whether to release an inmate or not due to the accuracy (of any model) not even exceeding 70%.

Appendix: Code

The code is presented in the following way: First, we present the initial pre-processing steps we took. Then we present the code for building all five models, including training and tuning. Finally, we present the three ways that we pre-process data a-new to mitigate bias. The models were fit on this new data, but in the same way as when we initially built them, so to save some space we will not show that code once again. Finally, the models were fit on the testing data from the initial split.

Initial Pre-Processing

```r
knitr::opts_chunk$set(eval = FALSE)

set.seed(123)
compas <- fairmodels::compas
compas <- filter(compas, Ethnicity == "African_American" | Ethnicity == "Caucasian")
compas$Two_yr_Recidivism <- as.factor(ifelse(compas$Two_yr_Recidivism == ’1’, ’0’, ’1’))
compas$Ethnicity <- droplevels(compas$Ethnicity)

split <- initial_split(compas, prop = 0.8, strata = "Two_yr_Recidivism")
compas_train <- training(split)
compas_test <- testing(split)

y_numeric <- as.numeric(compas_train$Two_yr_Recidivism)-1
y_numeric_test <- as.numeric(compas_test$Two_yr_Recidivism)-1

resamples <- vfold_cv(compas_train, 5)
```

Models

```r
knitr::opts_chunk$set(eval = FALSE)

###########################
#### RANDOM FOREST ########
###########################

rf_mod <-
  rand_forest() %>%
  set_args(mtry = tune()) %>%
  set_engine("ranger", importance = "impurity") %>%
  set_mode("classification")

rf_rec <-
  recipe(Two_yr_Recidivism ~., data = compas_train)  %>%
  step_nzv(everything(), -all_outcomes())

rf_wf <-
  workflow() %>%
  add_model(rf_mod) %>%
  add_recipe(rf_rec)

rf_res <-
  rf_wf %>%
  tune_grid(resamples = resamples,
            metrics = metric_set(roc_auc, accuracy),
            control = control_grid(save_pred = TRUE))

rf_best <-
  rf_res %>%
  select_best(metric = "accuracy")

final_wf <-
  rf_wf  %>%
  finalize_workflow(rf_best)

rf_fit <-
  final_wf %>%
  fit(data = compas_train)


rf_explainer <- explain_tidymodels(rf_fit, data = compas_train[,-1],
                                   y = y_numeric,
                                   label = "RF")
rf_test_exp <- update_data(rf_explainer, data = compas_test[,-1],
                                   y = y_numeric_test, verbose = FALSE)


###########################
##### NEURAL NET ##########
###########################
```

```
nn_mod <-
  mlp(hidden_units = tune()) %>%
  set_engine("nnet") %>%
  set_mode("classification")

nn_rec <-
  recipe(Two_yr_Recidivism ~ ., data = compas_train) %>%
  step_dummy(Sex, Ethnicity, Age_Above_FourtyFive,
             Age_Below_TwentyFive, Misdemeanor) %>%
  step_nzv(everything(), -all_outcomes()) %>%
  step_normalize(everything(), -all_outcomes())

nn_wf <-
  workflow() %>%
  add_model(nn_mod) %>%
  add_recipe(nn_rec)

nn_res <-
  nn_wf %>%
  tune_grid(resamples = resamples,
            metrics = metric_set(roc_auc, accuracy),
            control = control_grid(save_pred = TRUE))

nn_best <-
  nn_res %>%
  select_best(metric = "accuracy")

final_nn_wf <-
  nn_wf  %>%
  finalize_workflow(nn_best)

nn_fit <-
  final_nn_wf %>%
  fit(data = compas_train)

nn_explainer <- explain_tidymodels(nn_fit, data = compas_train[,-1],
                                   y = y_numeric,
                                   label = "ANN")
nn_test_exp <- update_data(nn_explainer, data = compas_test[,-1],
                           y = y_numeric_test, verbose = FALSE)


#######################
```

```r
#### LOG REG ##########
#######################

lr_mod <-
  logistic_reg(penalty = tune()) %>%
  set_engine("glmnet") %>%
  set_mode("classification")

lr_rec <-
  recipe(Two_yr_Recidivism ~ ., data = compas_train) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_nzv(everything(), -all_outcomes()) %>%
  step_normalize(everything(), -all_outcomes())

lr_wf <-
  workflow() %>%
  add_model(lr_mod) %>%
  add_recipe(lr_rec)

lr_res <-
  lr_wf %>%
  tune_grid(resamples = resamples,
            metrics = metric_set(roc_auc, accuracy),
            control = control_grid(save_pred = T))

lr_best <-
  lr_res %>%
  select_best(metric = "accuracy")

final_lr_wf <-
  lr_wf  %>%
  finalize_workflow(lr_best)

lr_fit <-
  final_lr_wf %>%
  fit(data = compas_train)

lr_explainer <- explain_tidymodels(lr_fit,
                                   data = compas_train[,-1],
                                   y = y_numeric,
                                   label = "LR")
lr_test_exp <- update_data(lr_explainer, data = compas_test[,-1],
                           y = y_numeric_test, verbose = FALSE)
```

```r
#######################
#### KNN ##############
#######################


knn_mod <-
  nearest_neighbor(neighbors = tune()) %>%
  set_engine(’kknn’) %>%
  set_mode("classification")

knn_rec <-
  recipe(Two_yr_Recidivism ~ ., data = compas_train) %>%
  step_bagimpute(everything()) %>%
  step_scale(all_numeric(), -all_outcomes()) %>%
  step_dummy(Sex)

knn_wf <-
  workflow() %>%
  add_model(knn_mod) %>%
  add_recipe(knn_rec)

knn_res <-
  knn_wf %>%
  tune_grid(resamples = resamples,
            metrics = metric_set(roc_auc, accuracy),
            control = control_grid(save_pred = TRUE))

knn_best <-
  knn_res %>%
  select_best(metric = "accuracy")

final_knn_wf <-
  knn_wf  %>%
  finalize_workflow(knn_best)

knn_fit <-
  final_knn_wf %>%
  fit(data = compas_train)

knn_explainer <- explain_tidymodels(knn_fit,
                                    data = compas_train[,-1],
                                    y = y_numeric,
                                   label = "KNN")
rm(knn_mod, knn_rec, knn_wf, knn_res, final_knn_wf, knn_best)

#######################
#### BOOOSTING ########
#######################

ab_mod <-
  boost_tree(mtry = tune()) %>%
  set_engine("xgboost") %>%
  set_mode("classification")

ab_rec <-
  recipe(Two_yr_Recidivism ~ ., data = compas_train) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
  step_nzv(everything(), -all_outcomes()) %>%
  step_normalize(everything(), -all_outcomes())

ab_wf <-
  workflow() %>%
  add_model(ab_mod) %>%
  add_recipe(ab_rec)

ab_res <-
  ab_wf %>%
  tune_grid(resamples = resamples,
            metrics = metric_set(roc_auc, accuracy),
            control = control_grid(save_pred = T))

ab_best <-
  ab_res %>%
  select_best(metric = "accuracy")

final_ab_wf <-
  ab_wf  %>%
  finalize_workflow(ab_best)

ab_fit <-
  final_ab_wf %>%
  fit(data = compas_train)

boost_explainer <- DALEX::explain(ab_fit, data = compas_train[,-1],
                                  y = y_numeric,
```

```r
                                  label = "AdaBoost")
boost_test_exp <- update_data(boost_explainer, data = compas_test[,-1],
                           y = y_numeric_test, verbose = FALSE)
```

Bias Mitigation Pre-Processing

```r
knitr::opts_chunk$set(eval = FALSE)

# Disparate impact remover
compas_train_mit <- compas_train %>%
  mutate(Number_of_Priors = as.numeric(Number_of_Priors)) %>%
  disparate_impact_remover(protected = compas_train$Ethnicity,
                           features_to_transform = c("Number_of_Priors"))

# Uniform resampling
uniform_indexes <- resample(protected = compas_train$Ethnicity,
                            y = y_numeric)

# Preferential resampling
probs <- glm(Two_yr_Recidivism ~., data = compas_train, family = binomial())$fitted.values
pref_ind <- resample(protected = compas_train$Ethnicity,
                     y = y_numeric,
                     type = "preferential",
                     probs = probs)
```

References

Buolamwini, J., & Gebru, T. (2018). *Gender Shades: Intersectional Accuracy Disparities in Commercial Gender Classification* (S. A. Friedler & C. Wilson, Eds.; Vol. 81, pp. 77–91). PMLR. http://proceedings.mlr.press/v81/buolamwini18a.html

Calders, T., & Verwer, S. (2010). Three naive bayes approaches for discrimination-free classification. *Data Min. Knowl. Discov.*, *21*, 277–292. https://doi.org/10.1007/s10618-010-0190-x

Creager, E., Madras, D., Jacobsen, J.-H., Weis, M. A., Swersky, K., Pitassi, T., & Zemel, R. (2019). *Flexibly fair representation learning by disentanglement.* http://ludwig.lub.lu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=edsarx&AN=edsarx.1906.02589&site=eds-live&scope=site

Datta, A., Tschantz, M. C., & Datta, A. (2015). Automated Experiments on Ad Privacy Settings. *Proceedings on Privacy Enhancing Technologies*, *2015*(1), 92–112. https://doi.org/https://doi.org/10.1515/popets-2015-0007

Feldman, M., Friedler, S. A., Moeller, J., Scheidegger, C., & Venkatasubramanian, S. (2015). Certifying and Removing Disparate Impact. *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15*, 259–268. https://doi.org/10.1145/2783258.2783311

Feng, R., Yang, Y., Lyu, Y., Tan, C., Sun, Y., & Wang, C. (2019). Learning Fair Representations via an Adversarial Framework. *arXiv:1904.13341 [Cs, Stat]*. http://arxiv.org/abs/1904.13341

Goel, N., Yaghini, M., & Faltings, B. (2018). Non-Discriminatory Machine Learning through Convex Fairness Criteria. *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, 116–116. https://doi.org/10.1145/3278721.3278722

Hardt, M., Price, E., & Srebro, N. (2016). Equality of opportunity in supervised learning. *CoRR*, *abs/1610.02413*. http://arxiv.org/abs/1610.02413

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. Springer New York. https://books.google.co.uk/books?id=tVIjmNS3Ob8C

James, G., Tibshirani, R., SpringerLink (Online, s., Hastie, T., & Witten, D. (2013). *An introduction to statistical learning. [Elektronisk resurs] with applications in r*. Springer New York. http://ludwig.lub.lu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=cat07147a&AN=lub.5846582&site=eds-live&scope=site

Kamiran, F., & Calders, T. (2012). Data preprocessing techniques for classification without discrimination. *Knowledge and Information Systems*, *33*(1), 1–33. https://doi.org/10.1007/s10115-011-0463-8

Obermeyer, Z., Powers, B., Vogeli, C., & Mullainathan, S. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, *366*(6464), 447. https://doi.org/10.1126/science.aax2342

Zemel, R., Wu, Y., Swersky, K., Pitassi, T., & Dwork, C. (2013). *Learning fair representations* (S. Dasgupta & D. McAllester, Eds.; Vol. 28, pp. 325–333). PMLR. http://proceedings.mlr.press/v28/zemel13.html