Outdoing the COMPAS Model

An Exploration of In-Processing to Build a Fairer Recidivism Prediction Model

Tanusri Balla | Nadia Goldman | Natalie Wiegand | Grace Wu

CIS 399: The Science of Data Ethics

April 6, 2020

Table of Contents

Introduction

Claims of unfairness in the United States' criminal justice system have long involved issues of bias and discrimination. In attempt to counter prejudiced judges, witnesses, and police officers, many companies have worked to systematize processes of setting bail, parole, and sentencing through technology. To make these technologies as "fair" as possible, these companies work with experts in relevant fields to determine what data to input and how to assess future risk. And to evaluate the impact of these technologies, companies must define and understand fairness, as accuracy alone does not capture the legitimacy of a machine learning algorithm.

One company that specializes in this technology is Equivant, previously named Northpointe. Equivant claims to be "at the leading edge of the justice industry," as one of their most notable technologies is their Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) software.[1] This technology is widely used to determine potential recidivism risk in the United States, including in two of the country's most populous states, New York and California. In addition to being used to determine whether an individual may skip trial or commit another crime during legal proceedings, COMPAS scores have been used to determine the likelihood that an individual recidivates after release from prison. Further, these scores have been used to determine the likelihood that an individual recidivates by committing, specifically, a violent crime.

Though COMPAS scores are meant to counter personal biases that may permeate decisions made throughout the legal process, these scores may come with their own inherent and systemic biases. In 2016, the nonprofit ProPublica released a report that concluded COMPAS scores had

[1] "About Us." Equivant, 30 Mar. 2020, www.equivant.com/about-us/.

much higher false positive rates for Black individuals than their White counterparts.[2] While the tool was about 70 percent accurate overall, previous analyses done did not measure which populations the tool might be less accurate for, nor did they analyze any variations within components of accuracy, such as the difference between false positive rates and false negative rates.[3] Though Equivant did not release their tool or details about the underlying machine learning algorithm, ProPublica conducted a study on more than 10,000 defendants in a Florida county to compare COMPAS scores and actual recidivism rates. ProPublica found that of defendants who did not recidivate in a two-year period following their release, Black defendants were nearly twice as likely to be misclassified as higher risk than their White counterparts – 45 percent of Black defendants were misclassified as higher risk while only 23 percent of White defendants faced that same misclassification. For the reverse misclassification, 48 percent of White defendants who recidivated in the two years following their release were misclassified as low risk, while only 28 percent of Black defendants were misclassified as low risk. These findings, amongst others that demonstrated further inconsistent accuracies across racial groups, led ProPublica to conclude Equivant's COMPAS algorithm has statistically significant racial biases.[4]

A few months later, Equivant published a counter to ProPublica's report. Equivant's report found that ProPublica had not accounted for differences in recidivism base rates between racial groups. Had those base rates been considered, ProPublica's statistical analysis would not have

[2] Throughout this paper, we capitalize both White and Black in accordance with race theory recognition of both as labels that have systemic importance, socially and politically.
[3] Angwin, Julia, et al. "Machine Bias." *ProPublica*, 9 Mar. 2019, www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing.
[4] Larson, Jeff, et al. "How We Analyzed the COMPAS Recidivism Algorithm." ProPublica, 9 Mar. 2019, www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm.

found statistically significant differences in recidivism predictions across racial groups.[5] Though both Equivant and ProPublica's interpretations are mathematically correct, it is important to consider whether it is fair to use recidivism base rates from data that might represent an already unfair system – data ridden with possibly racially biased policing, policies, and legal precedents.

Despite this, Equivant boasts a COMPAS prediction success rate of about 70 percent, which when examined alone may seem impressive. But a random sample of people with little to no experience in the criminal justice field were able to predict recidivism rates with higher accuracy than COMPAS after only being provided a few facts about the defendant, according to study conducted by Dartmouth.[6] And perhaps more importantly, accuracy alone is an insufficient measure of success.

Our goal is to try to paint a more complete picture of recidivism – how might we design a better algorithm that is not only accurate, but also fair? In this project, we won't achieve complete fairness, but hope to improve it, while at least maintaining the same accuracy of Equivant's COMPAS algorithm. Using error rate as one possible metric for fairness and using COMPAS training and testing data, we were able to predict whether an individual would recidivate with about 75 percent accuracy.[7]

Though we find ProPublica's definition of fairness as equality of false positive rates across racial groups to make more sense contextually, we wanted to see if we were able to create a

---

[5] Equivant, "Response to ProPublica: Demonstrating Accuracy Equity and Predictive Parity." *Equivant*, 21 Jan. 2020, www.equivant.com/response-to-propublica-demonstrating-accuracy-equity-and-predictive-parity/.
[6] Northpointe, "COMPAS Rick & Need Assessment System." Equivant, 2012, http://www.northpointeinc.com/files/downloads/FAQ_Document.pdf.
[7] Dressel, Julia, and Hany Farid. "The Accuracy, Fairness, and Limits of Predicting Recidivism." *Science Advances*, American Association for the Advancement of Science, 1 Jan. 2018, advances.sciencemag.org/content/4/1/eaao5580.

model that would be fairer even under one of Equivant's own definitions of fairness. As the ProPublica/Equivant debate only analyzes Black and White defendants, we also chose to only analyze those two groups. Working with Microsoft researchers' Fairlearn package and using pre-processing as a proxy for in-processing, we were able to create a model that was both more fair and more accurate than COMPAS.[8]

Overview of Reductions Approach to Fair Learning Algorithms

To make a fairer model, we needed to correct the inadvertent discrimination seen in these machine learning algorithms. As Microsoft researchers Agarwal et al. explain in "A Reductions Approach to Fair Classification," there are two commonly taken approaches to do so. The first approach is to include a specific quantitative definition of fairness in the machine learning algorithm.[9] The goal of this is to ensure that correlation is not found between the protected attribute and the result. For this algorithm, the goal would be to find no correlation between any given race and recidivism scores. While this may seem like a good approach, resulting fairness guarantees typically only hold under "strong distributional assumptions," and the approaches are tied to specific families of classifiers.[10] The second approach relies on pre-processing the data or post-processing the results and conceptualizing the machine learning classification method as a black box. We elected to use the second approach and buttress the algorithm's weaknesses through reduction to a series of cost-sensitive classification problems.[11] This can be done by assigning a cost to type of classification. For example, assigning 'will recidivate' when the

[8] We also recognize that racial categorization of defendants by police staff and correctional officers is very likely erroneous and incomplete.
[9] Agarwal, Alekh, et al., "A Reductions Approach to Fair Classification." Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, PLR 80, 2018, https://arxiv.org/pdf/1803.02453.pdf.
[10] Ibid.
[11] Ibid.

individual actually did recidivate will have a low cost (e.g., 0) and assigning 'will recidivate' when the individual actually did not recidivate will have a high cost (e.g.,1). From there, the decision is whether to factor in costs for negative predictions, i.e., 'will not recidivate.' If negative predictions are deemed sufficiently harmful, their costs should also be included so that assigning 'will not recidivate' when the individual did not recidivate will have a low cost and assigning 'will not recidivate' when the individual did recidivate will have a high cost.

The next step to creating a fairer model is to decide on a definition of fairness for our research purposes. Commonly used fairness definitions include that of statistical parity, bounded group loss, and equalized odds. Statistical parity, also known as demographic parity, defines fairness as having no correlation between the protected attribute and the result. Bounded group loss ensures that the prediction error of any protected group remains below some predetermined level. Equalized odds defines fairness similarly to statistical parity but with the exception that there can be correlation on the protected attribute to the result but only through other given attributes.[12]

By taking the second approach to making a fairer model, we were able to analyze the use of many of these different fairness definitions and how they impacted our results. Ultimately, we only used bounded group loss after finding that other fairness definitions were less appropriate for our goals.

Introduction to Fairlearn and the Use of Reductions

In order to implement the ideas that were discussed in the Microsoft paper, we decided to use

[12] Hardt, Moritz, et al. "Equality of Opportunity in Supervised Learning." *University of Chicago,* 2016. https://ttic.uchicago.edu/~nati/Publications/HardtPriceSrebro2016.pdf.

Fairlearn, a package also created by Microsoft. The Fairlearn package has two components: first, a dashboard for evaluating which groups are negatively impacted by a model and for comparing multiple models in terms of various fairness and accuracy metrics, and second, algorithms for mitigating unfairness. We mainly used the second part of the package, as we coded our own ways of evaluating the negative impacts of unfair models.

Before diving into details of the Fairlearn package and how we used it, it's necessary to clarify what fairness means in this context. In Fairlearn specifically, unfairness is defined by its impact on people. Thus, the focus is on harms, and specifically allocation harms and quality-of-service harms. Allocation harms occur when a system extends or withholds opportunities, resources or information. Real-life examples of this include the handling of loans and hiring. Quality-of-service harms describe whether a system works just as well for one group as it does for another, regardless if opportunities, resources, or information are extended or withheld. It's also notable that fairlearn uses a group fairness approach, which focuses on the groups of individuals that are at risk, as opposed to the individuals themselves.

Of the fairness algorithms implemented in Fairlearn, two of them were those described in the accompanying paper. The other two algorithms include a GridSearch approach for evaluating fair regressions and a post-processing algorithm that transforms an existing classifier's prediction to enforce a specified parity constraint. Thus we chose our approach from the two other options: a black-box ExponentiatedGradient approach and a black-box GridSearch approach. After choosing the latter algorithm because we felt more comfortable implementing this option, we had to choose a fairness definition to apply to our model. Options provided in Fairlearn included bounded group loss, conditional selection rate, and error rate. We believed error rate was an overly simplistic view of fairness that does not work particularly well in

criminal justice contexts. We also decided not to use conditional selection rate because we found few resources that were helpful in understanding and implementing the metric. Bounded group loss, on the other hand, seemed more appropriate for our goals, as it specifically pertains to constraining the prediction error for a protected group. In our case, we focused on protecting Black individuals, who experienced relatively greater harms in Equivant's model. Thus, our approach to creating a fairer model than COMPAS utilizes a GridSearch algorithm that measures fairness with bounded group loss.

Applying Fairlearn to the ProPublica Dataset

Objective

As described earlier, our goal with this project was to produce a fairer model than the Equivant model by implementing in-processing techniques. Because the reductions approach uses a combination of data pre-processing and black-box algorithms to achieve the same effects as in-processing techniques, it offers a simpler path to achieving our goal than modifying the actual code behind pre-defined Python learning packages (e.g. scikit-learn). Since the Fairlearn package provides an implementation of this approach, we set out to produce a fairer model than the Equivant model by applying reductions through the package.

Description of Approach

Our approach breaks down into 3 steps: 1) assessing the fairness of the Equivant model using COMPAS scores in the ProPublica dataset, 2) attempting to reproduce the Equivant model to select a class of models, and 3) applying Fairlearn to the chosen class of models to produce a fairer model. For the Python code, please refer to the Code section.

1) Fairness of the Equivant Model

The goal of this step was to set a standard for assessing fairness throughout our project by ensuring we could accurately measure fairness in the Equivant model using an approach similar to that of ProPublica. We used two columns of the dataset to assess fairness: "score_text" (to represent Equivant's predictions) and "is_recid" (to represent the true values for recidivism).

The "score_text" column has three possible values — "Low," "Medium," and "High" risk levels of recidivism — while the "is_recid" column has two possible values — 1 and 0 to represent whether a defendant did or did not recidivate. In order to reconcile the differences in categories between the predictions and true values, we transformed "score_text" to map "Low" to 0 and "Medium"/"High" to 1. This transformation aligns with ProPublica's transformations of data in their analysis.

With our transformed predictions and true values, we created confusion matrices for the overall results, the results for Black defendants, and the results for White defendants:

Overall:

	1 (True)	0 (True)	Error Rate = 34.22% False Positive Rate = 29.40% False Negative Rate = 39.35%
1 (Predicted)	1700	878	
0 (Predicted)	1103	2108	

Black Defendants:

	1 (True)	0 (True)	Error Rate = 34.95%

1 (Predicted)	1163	548	False Positive Rate = 41.33% False Negative Rate = 29.86%
0 (Predicted)	495	778	

White Defendants:

	1 (True)	0 (True)	Error Rate = 33.85% False Positive Rate = 21.63% False Negative Rate = 50.72%
1 (Predicted)	408	247	
0 (Predicted)	420	895	

The error rate aligns with that calculated by Equivant itself, and the error and false positive rates align with those calculated by ProPublica, confirming that our setup is similar to both groups'.

2) Reproducing the Equivant Model

The next step in our approach was to try to reproduce the Equivant model using different classes of models. The motivation behind this step was to identify the class of models that could contain the Equivant model, and then work to find a fairer model within this class in the last step of our approach.

To recreate the Equivant model, we included the available and relevant predictors from the dataset and trained models to predict the Equivant classification of an individual as "Low" (0) or "Medium"/"High" risk (1). The predictors we included were: "sex", "age", "race",

"juv_fel_count" (assumed to count the individual's number of juvenile felonies), "juv_misd_count" (assumed to count the individual's number of juvenile misdemeaners), "juv_other_count" (assumed to count the number of other juvenile charges against the individual), "priors_count" (assumed to count the individual's prior convictions), "c_jail_in" (assumed to mark the date when the individual was put in jail), "c_charge_degree" (assumed to mark the degree of severity of the most severe charge), "c_charge_desc" (assumed to describe the most severe charge).

We then compared our model's predictions on test data to Equivant's actual predictions (the true values in this comparison) to check the model's error rate. Doing this for a logistic regression, which was trained on 70 percent of the ProPublica dataset and tested on the remaining 30 percent, we found the following:

	1 (True)	0 (True)	Error Rate = 24.88%
1 (Predicted)	504	184	
0 (Predicted)	186	613	

We deemed this error rate to be a satisfactory indication that the Equivant model could be a logistic regression (and that further tuning of the model would have reduced the error rate even further). We therefore chose to move forward with the class of logistic regression models.

3) Analyzing Fairlearn

The last step of our approach was to apply reductions from the Fairlearn package to the ProPublica dataset and then identify the best model possible within the class of logistic regression models. Due to our unfamiliarity with Fairlearn prior to this project, this step mainly became an exploration of the tools the package provides, along with a continued effort to achieve our original goal to create a fairer model than Equivant.

We began by choosing to use Fairlearn's GridSearch algorithm for mitigating unfairness. This algorithm is an implementation of the black-box, reductions-based approach described in the Microsoft report, and can be used for binary classification or regression. The algorithm requires the following inputs: a class of models, a fairness constraint, and grid size.

For the class of models, we chose logistic regression based on the above reasoning. For the fairness constraint, we had a few options available through Fairlearn. The package offers three types of constraints through its Moments class: demographic parity, equalized odds, and bounded group loss. Demographic parity and equalized odds both seemed inappropriate for the use case of predicting recidivism as our goal was not to ensure that Black and White defendants are predicted to recidivate at equal or nearly equal rates. Bounded group loss seemed slightly more appropriate, because restricting error rates on Black and White defendants to be below a certain level seemed to be more likely to ensure that one race was not much more harmed by the model than the other race. Using this reasoning, we chose bounded group loss as our fairness constraint (see Choice of Fairness Constraint section for an in-depth explanation). Lastly, for grid size, we chose 100 so that the GridSearch algorithm would explore 100 different logistic regression models in its attempt to mitigate unfairness.

After setting up how we were going to use Fairlearn, we aimed to run the GridSearch algorithm with increasing error levels for bounded group loss. By doing so, we expected to find the minimum error level with which bounded group loss was a feasible fairness constraint for a model. We also expected to find that as we increased our chosen error levels, the algorithm would attempt to fix at least one race's error rate to the chosen level (in order to make the chosen error level useful in the fairness constraint). With our results from each run of GridSearch, we planned to compare the optimal models found at each chosen error level by 1) the error rates for Black defendants, 2) the error rates for White defendants, and 3) the difference in error rates for the two races.

Choice of Fairness Constraint

As noted earlier, we discussed the costs and benefits to using a variety of fairness constraints before deciding on bounded group loss. Although we entertained demographic parity, equalizing positive outcomes (i.e. the prediction that an individual recidivates) does not make sense in this context. Though some definitions of fairness may consider a focus on equalizing positive outcomes to be more fair to those who were harmed by defendants, we determined that taking a 'color-blind' approach to fairness did not make sense after considering relevant systematic racial disparities. As non-White individuals experience systematic disadvantages from everyday discrimination, higher levels of policing and visibility, and a dearth of education, health, and social resources, we decided that it did not make sense to consider a predictive model with the goal of 'color-blindness.'

Because error rate is a relatively simplistic view of fairness, we spent a little time playing with this definition in the code before moving onto bounded group loss. With bounded group loss, we were drawn to the guarantee of a maximum prediction error for a protected group. We

wanted to control the overall error rate, while taking into consideration that our world is not actually color-blind and that more may need to be done in order to protect at-risk groups. However, there were some difficulties with applying bounded group loss in our GridSearch algorithm. Because documentation and writing about proper implementation is relatively sparse, we had trouble determining why some features of our model were not behaving as expected. Further, we had trouble mathematically defining violations to bounded group loss, because violating a maximum error can be codified in several different ways. We ended up setting a maximum difference in error for racial groups, as we saw this as a simple yet effective way to compare prediction error while protecting Black inmates from unfair risk predictions.

Findings about Fairlearn

Unfortunately, the GridSearch algorithm for bounded group loss did not work as expected for us. When we attempted to run the algorithm for increasing error levels, we found no change in the models produced. This could be seen when we plotted our graphs for the results of the algorithm using maximum error levels of 0.05, 0.25, 0.5, and 1 (see Figures A-C for graphs). All the graphs were identical, indicating that there was no change in the results. There was no point at which our chosen error level was too small to produce results, nor was there a change in the resulting models when we changed the type of loss used for bounded group loss (i.e. square loss vs. absolute loss). This behavior of the algorithm was unexpected, and may have been caused by our use of bounded group loss for binary classification (when it is intended for regressions).

Effectiveness of Final Model

Despite the complications with our use of Fairlearn, the GridSearch algorithm did provide 100 different models to consider in our attempt to find a fairer model than the Equivant model. These models can still be evaluated using our planned comparisons of their error rates for Black

defendants, error rates for White defendants, and difference in error rates for both races.

We used Figure A for evaluating the models, labeled from 0 to 99, against the measure of error rates for Black defendants. When plotting this error rate on the x-axis and the overall error rate of each model on the y-axis, we found a very short Pareto curve consisting of models 29 and 80. Model 29 had an error rate for Black defendants of 24.78 percent and an overall error rate of 24.68 percent, while Model 80 had an error rate for Black defendants of 24.89 percent and an overall error rate of 24.61 percent.

We used Figure B for evaluating the same models against the measure of error rates for White defendants. When plotting this error rate on the x-axis and the overall error rate of each model on the y-axis, we found a longer Pareto curve, once again consisting of two models: 50 and 80. Model 50 had an error rate for White defendants of 23.52 percent and an overall error rate of 24.68 percent, while Model 80 had an error rate for White defendants of 24.19 percent and an overall error rate of 24.61 percent.

Finally, we used Figure C for evaluating the models against the measure of difference in error rates for both races. Here, our Pareto curve consisted of three models: 95, 86, and 80. Model 95 had a difference in error rates of 0.01 percent and an overall error rate of 25.22 percent, Model 86 had a difference in error rates of 0.24 percent and an overall error rate of 24.68 percent, and Model 80 had a difference in error rates of 0.69 percent and an overall error rate of 24.61 percent. Because Model 80 appeared on the Pareto curve for all three graphs, we moved forward with considering it as our final model. When comparing the performance of Model 80 against the performance of the Equivant model (outlined in the Description of Approach section), we see that Model 80 offers a lower overall error rate (24.61 percent

compared to 34.22 percent), lower error rates for both Black and White defendants (24.89 percent compared to 34.95 percent and 24.19 percent compared to 33.85 percent), and a lower difference in error rates for both races (0.69 percent compared to 1.10 percent). This means that, in terms of the fairness definition used by Equivant that we discussed in class, we did successfully find a fairer model than the Equivant model.

Going further, we evaluated Model 80 through the lens of false positive rates, which we believe to be a more applicable definition of fairness than group error rates in the context of predicting recidivism. Model 80 had an overall false positive rate of 26.92 percent, a false positive rate for

Black defendants of 24.69 percent, and a false positive rate for White defendants of 35.97 percent. Comparing these values to the false positive rates of the Equivant model, we see that Model 80 performed better in overall false positive rate (26.92 percent compared to 29.40 percent) and the false positive rate for Black defendants (24.69 percent compared to 41.33 percent), but worse in the false positive rate for White defendants (35.97 percent compared to 21.63 percent). Looking at the difference between rates, we see that Model 80's false positive disparity was only 11.28 percent compared to the Equivant model's false positive disparity of 19.70 percent. This implies that although we were optimizing for a different definition of fairness, our final model was actually also more fair using a more relevant definition of fairness (false positive disparity).

Conclusion

As we were able to create a model that was both less erroneous than the Equivant model and more equitable in false positive rates, we can conclude that Equivant's model is not as fair or as accurate as it could be. Because of the life-long implications that sentencing can have on an

individual and their community, we hope that judges, government officials, and other stakeholders will take criticisms such as ours, along with those of ProPublica and the Dartmouth researchers, into account when weighing COMPAS scores and the larger risks and benefits of using Equivant's technology.

Code

In [ ]:

```
#CIS 399
```

In [ ]:

```
#Project 1: COMPAS Dataset Analysis
#Team: Tanusri Balla, Nadia Goldman, Natalie Wiegand, Grace Wu
```

In [ ]:

```
#Imports

import csv
import pandas as pd
import numpy as np
from sklearn.linear_model import SGDClassifier
from sklearn.linear_model import RidgeClassifier
from sklearn.linear_model import LogisticRegressionCV
from sklearn.model_selection import train_test_split

from fairlearn.reductions import GridSearch
from fairlearn.reductions import DemographicParity
from fairlearn.reductions import AbsoluteLoss
from fairlearn.reductions import ZeroOneLoss
from fairlearn.reductions import GroupLossMoment

import matplotlib.pyplot as plt

from sklearn.metrics import confusion_matrix
```

In [ ]:

```
#Import csv file into dataframe

compas_df = pd.read_csv("propublica_full.csv")
compas_df = compas_df.drop(columns="Unnamed: 0")
compas_df = compas_df.astype('category')
compas_df = compas_df.apply(lambda x: x.cat.codes)
compas_df = compas_df[compas_df.is_recid != -1]
compas_df.head(10)
```

In [ ]:

```
#Evaluate racial fairness of given COMPAS model
#We use the score_text column, and classify individuals as "Low" risk or "High"
(aka "Medium","High") risk (as ProPublica did)
#We define fairness using difference in subgroup error rates

is_recid_true = compas_df['is_recid']
is_recid_pred = compas_df['score_text']
#Low == 1, Medium == 2, High == 0 in the data
is_recid_pred = pd.np.where(is_recid_pred == 1, 0, 1)

reduced_compas_df = pd.DataFrame({'race': compas_df['race'], 'is_recid_true': is
_recid_true, 'is_recid_pred': is_recid_pred})
cm_all = confusion_matrix(reduced_compas_df['is_recid_true'], reduced_compas_df[
'is_recid_pred'])
tn, fp, fn, tp = cm_all.ravel()

#White is encoded as 2 in the dataframe
reduced_compas_df_white = reduced_compas_df[reduced_compas_df['race']==2]
cm_white = confusion_matrix(reduced_compas_df_white['is_recid_true'], reduced_co
mpas_df_white['is_recid_pred'])
tn_white, fp_white, fn_white, tp_white = cm_white.ravel()

#Black is encoded as 0 in the dataframe
reduced_compas_df_black = reduced_compas_df[reduced_compas_df['race']==0]
cm_black = confusion_matrix(reduced_compas_df_black['is_recid_true'], reduced_co
mpas_df_black['is_recid_pred'])
tn_black, fp_black, fn_black, tp_black = cm_black.ravel()

error_rate = (fp + fn) / (tn + fp + fn + tp)

error_rate_white = (fp_white + fn_white) / (tn_white + fp_white + tp_white + fn_
white)
error_rate_black = (fp_black + fn_black) / (tn_black + fp_black + tp_black + fn_
black)
violation_of_fairness_constraint = abs(error_rate_white - error_rate_black)

print("Confusion Matrix Overall")
print(cm_all)

print("Confusion Matrix White")
print(cm_white)
print("Error Rate White = ", error_rate_white)

print("Confusion Matrix Black")
print(cm_black)
print("Error Rate Black = ", error_rate_black)

print("Error Rate Disparity = ", violation_of_fairness_constraint)
print("Error Rate = ", error_rate)
```

In [ ]:

```
#Create a model that predicts decile scores as COMPAS model would (reproduce the
COMPAS model)

#Filter down to just white and black defendants
filtered_compas_df = compas_df
filtered_compas_df['race'] = pd.np.where(filtered_compas_df['race'] == 2, -2, fi
ltered_compas_df['race'])
filtered_compas_df = filtered_compas_df[filtered_compas_df['race'] < 1]
filtered_compas_df['race'] = pd.np.where(filtered_compas_df['race'] == -2, 2, fi
ltered_compas_df['race'])

#Variables we want to include in the model
X = filtered_compas_df[['sex', 'age', 'race', 'juv_fel_count',
                'juv_misd_count', 'juv_other_count',
                'priors_count', 'c_jail_in',
                'c_charge_degree', 'c_charge_desc']]

#We use the score_text column, and classify individuals as "Low" risk or "High"
(aka "Medium","High") risk (as ProPublica did)
#Low == 0, High == 1
Y = filtered_compas_df[['score_text']]
Y = pd.np.where(Y['score_text'] == 1, 0, 1)

#Split data into train and test
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_
state=101)

compas_model = LogisticRegression()
compas_model.fit(X_train, Y_train)
predictions = compas_model.predict(X_test)
```

In [ ]:

```
cm_compas_model = confusion_matrix(Y_test, predictions)
tn, fp, fn, tp = cm_compas_model.ravel()

#How good is our model at assigning risk levels the same way COMPAS would?
print("Error Rate = ", ((fp + fn) / (tn + fp + fn + tp)))
print(tn, fp, fn, tp)
```

In [ ]:

```
#Evaluate racial fairness of our recreation of COMPAS model
#We use the score_text column, and classify individuals as "Low" risk or "High"
(aka "Medium","High") risk (as ProPublica did)
#We define fairness using FP disparity

is_recid_true = Y_test
is_recid_pred = predictions

reduced_compas_df = pd.DataFrame({'race': X_test['race'], 'is_recid_true': is_re
cid_true, 'is_recid_pred': is_recid_pred})
cm_all = confusion_matrix(reduced_compas_df['is_recid_true'], reduced_compas_df[
'is_recid_pred'])
tn, fp, fn, tp = cm_all.ravel()

#White is encoded as 2 in the dataframe
reduced_compas_df_white = reduced_compas_df[reduced_compas_df['race']==2]
cm_white = confusion_matrix(reduced_compas_df_white['is_recid_true'], reduced_co
mpas_df_white['is_recid_pred'])
tn_white, fp_white, fn_white, tp_white = cm_white.ravel()

#Black is encoded as 0 in the dataframe
reduced_compas_df_black = reduced_compas_df[reduced_compas_df['race']==0]
cm_black = confusion_matrix(reduced_compas_df_black['is_recid_true'], reduced_co
mpas_df_black['is_recid_pred'])
tn_black, fp_black, fn_black, tp_black = cm_black.ravel()

error_rate = (fp + fn) / (tn + fp + fn + tp)

fp_rate_white = fp_white / (tn_white + fp_white)
fp_rate_black = fp_black / (tn_black + fp_black)
violation_of_fairness_constraint = abs(fp_rate_black - fp_rate_white)

print("Confusion Matrix Overall")
print(cm_all)

print("Confusion Matrix White")
print(cm_white)
print("FP Rate White = ", fp_rate_white)

print("Confusion Matrix Black")
print(cm_black)
print("FP Rate Black = ", fp_rate_black)

print("FP Rate Disparity = ", violation_of_fairness_constraint)
print("Error Rate = ", error_rate)
```

In [ ]:

```
#Fair Model Section

#Our goal here is to take our recreated COMPAS model and see how preprocessing t
he data for our fairness
#constraint affects the fairness of this specific model
```

In [ ]:

```
#Set up model using fairlearn

bgl_absolute_loss = GroupLossMoment(AbsoluteLoss(0, 1))
first_sweep = GridSearch(LogisticRegression(),
                         constraints=bgl_absolute_loss,
                         grid_size=100
                        )

#Fit model to data
first_sweep.fit(X_train, Y_train, sensitive_features=X_train['race'])
```

In [ ]:

```
def compare_models(Xs, Ys, grid_search_results):

    violation_of_fairness_white = np.array([])
    violation_of_fairness_black = np.array([])
    violation_of_fairness_diff = np.array([])
    error_rates = np.array([])

    for x in grid_search_results:
        Y_preds = x.predictor.predict(Xs)

        data_all = Xs.copy()
        data_all['is_recid_true'] = Ys
        data_all['is_recid_pred'] = Y_preds
        tn, fp, fn, tp = confusion_matrix(data_all['is_recid_true'], data_all['i
s_recid_pred']).ravel()

        data_white = data_all[data_all['race']==2]
        tn_white, fp_white, fn_white, tp_white = confusion_matrix(data_white['is
_recid_true'], data_white['is_recid_pred']).ravel()

        data_black = data_all[data_all['race']==0]
        tn_black, fp_black, fn_black, tp_black = confusion_matrix(data_black['is
_recid_true'], data_black['is_recid_pred']).ravel()

        error_rate_white = (fp_white + fn_white) / (tn_white + fp_white + fn_whi
te + tp_white)
        error_rate_black = (fp_black + fn_black) / (tn_black + fp_black + fn_bla
ck + tp_black)

        #Fairness definition = bounded group loss
```

```
        #Defining violation of fairness constraint as the error rate for black p
eople
        violation_of_fairness_constraint_black = error_rate_black

        #Defining violation of fairness constraint as the error rate for white p
eople
        violation_of_fairness_constraint_white = error_rate_white

        #Defining violation of fairness constraint as the difference in error ra
tes for black and white people
        violation_of_fairness_constraint_difference = abs(error_rate_white - err
or_rate_black)

        error_rate = (fp + fn) / (tn + fp + fn + tp)

        violation_of_fairness_black = np.append(violation_of_fairness_black, [vi
olation_of_fairness_constraint_black])
        violation_of_fairness_white = np.append(violation_of_fairness_white, [vi
olation_of_fairness_constraint_white])
        violation_of_fairness_diff = np.append(violation_of_fairness_diff, [viol
ation_of_fairness_constraint_difference])

        error_rates = np.append(error_rates, [error_rate])



    return error_rates, violation_of_fairness_black, violation_of_fairness_white
, violation_of_fairness_diff
```

In [ ]:

```
error_rates, violation_of_fairness_black, violation_of_fairness_white, violation
_of_fairness_diff = compare_models(X_test, Y_test, first_sweep.all_results)
```

In [ ]:

```
#Plot error rates for black defendants against overall error rates

fig = plt.figure(figsize=(20,20))

for i in range(len(error_rates)):
    x = violation_of_fairness_black[i]
    y = error_rates[i]
    plt.scatter(x, y)
    plt.text(x, y, i, fontsize=15)

plt.xlabel("Fairness")
plt.ylabel("Error Rate")
plt.title("Fairness = Error Rates for Black Defendants")

plt.show()
```

In [ ]:

```
#Plot error rates for white defendants against overall error rates

fig = plt.figure(figsize=(20,20))

for i in range(len(error_rates)):
    x = violation_of_fairness_white[i]
    y = error_rates[i]
    plt.scatter(x, y)
    plt.text(x, y, i, fontsize=15)

plt.xlabel("Fairness")
plt.ylabel("Error Rate")
plt.title("Fairness = Error Rates for White Defendants")

plt.show()
```

In [ ]:

```
#Plot difference in error rates between races against overall error rates

fig = plt.figure(figsize=(20,20))

for i in range(len(error_rates)):
    x = violation_of_fairness_diff[i]
    y = error_rates[i]
    plt.scatter(x, y)
    plt.text(x, y, i, fontsize=15)

plt.xlabel("Fairness")
plt.ylabel("Error Rate")
plt.title("Fairness = Difference in Error Rates")

plt.show()
```

Appendix

Figure A

Fairness = Error Rate for Black Defendants

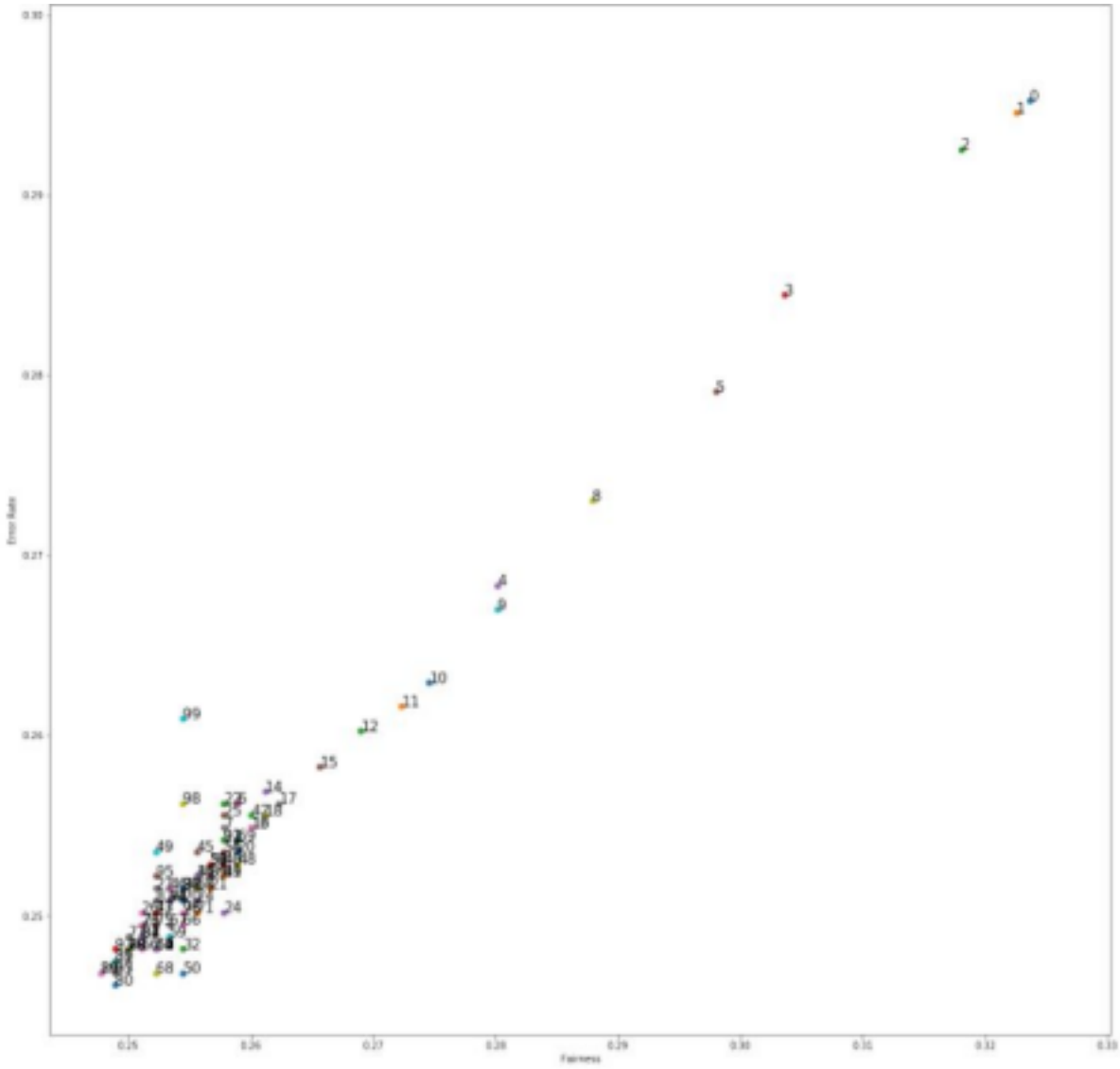

Figure B

Fairness = Error Rate for White Defendants

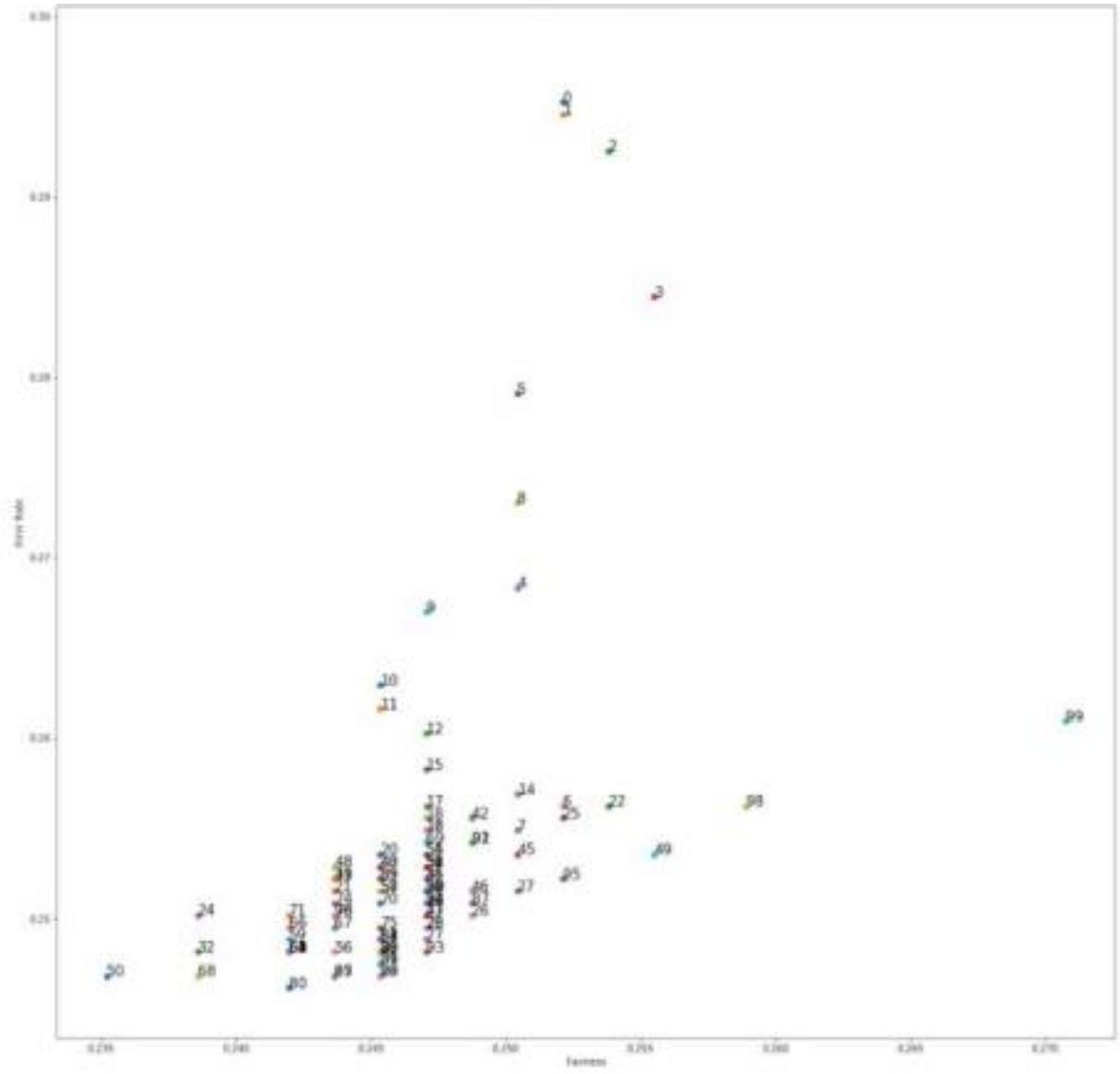

Figure C

Fairness = abs(Difference in Error Rates for Black and White Defendants)