Final Year Project

Exploratory Fairness Analysis

Lorcan Rooney

Student ID: 16413092

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science

Supervisor: Dr. Simon Caton

UCD School of Computer Science

University College Dublin

May 21, 2020

Table of Contents

Abstract

What does it mean for a machine learning algorithm to be fair? From screening job applications to deciding if a candidate is eligible for a loan, machine learning algorithms are being used to make significant decisions that affect the lives of ordinary people, everyday. Since these decisions are made by a computer, we'd assume they should lack any human conscious or subconscious biases? Unfortunately this is not the case. These algorithms can be just as biased as their human counterparts, if not more biased in some cases. Numerous methods to detect and mitigate bias have been proposed recently. Which method provides the 'fairest results?' There is no definitive answer with each author claiming that their algorithm is 'the best', but there is a notable lack of proper comparison between popular models. Contrasting and comparing these methods in order to gain a better insight into their strengths and shortcomings is pivotal to understanding how these algorithms work, and when they should be applied and is the main goals in this project.

Chapter 1: Project Specification

The aim of this project is to analyze, compare and contrast fairness metrics and bias mitigation algorithms currently available. **50** combinations of bias mitigation algorithms and classifiers were applied to **5** popular datasets to evaluate their strengths and pitfalls, based on **5** fairness and **4** performance metrics. A fairness metric is a measure of fairness, while a bias mitigation algorithm aims to improve fairness in some way. Numerous implementations of both algorithms and metrics have been introduced over the last number of years, each boldly claiming superior results to the next, but with little to no actual proof to back it up. Understanding which algorithms and metrics work well together, which algorithms test well with the most metrics and how certain algorithms and metrics work with different data sets and classifiers is essential and will give insight into what metric/algorithm to apply in what situation. The algorithms will be evaluated individually, then in combination with one another. Finally their fairness results will be ranked to see what information can be gleamed. Potential problems are run time and having to introduce a parallel programming implementation to fix this.

Chapter 2: **Introduction**

There has been an explosion of data in the last decade that would have been previously unimaginable. Eric Schmidt, former CEO of Google claims that there were *'5 exabytes of information created between the dawn of civilization through 2003, but that much information is now created every 2 days*[1] *'.* There are 5 billion search engine queries daily, five new Facebook profiles created a second and 456,000 tweets a minute[2]. These numbers are incomprehensibly huge. Humans are generating data at a rate faster than ever before in our history, and frankly, we are drowning in it. This availability of data and, more specifically, the patterns unearthed from it, is profoundly changing the world we live in today. This has lead to various organisations taking a keen interest in this data. Everyone from governments to citizen groups are producing algorithmic tools to analyze this data and expose hidden patterns and trends to further their interests.

A machine learning algorithm is trained on a set of seen data so that it can make informed decisions on an unseen set. Every aspect of our society is being affected by these algorithms; from criminal justice where these these algorithms are used to predict re-offence rates of ex-convicts to banking where they are used to decide whether applicants are eligible for a loan. This leads to a new question- are these algorithms fair? Answering this question is not as straightforward as one may think. Narayanan, 2018[1] tells us that there are at least 21 mathematical definitions of fairness. Fairness is a complex, multi-faceted issue. What definition to use in which situation is highly case and context dependent, and this problem will not be solved over-night.

The issue of fairness in machine learning has exploded in popularity over the last number of years, with the number of papers on the topic growing almost exponentially year on year. Various measures to detect fairness and mitigate bias in machine learning algorithms have been proposed recently, and indeed many have been implemented into public libraries and tool-kits. So what bias mitigation algorithm should be used to satisfy what definition of fairness? What fairness metric should I use on what size and shape of data set? What bias mitigation algorithm is 'best?' I hope to gain a deeper understanding and answer these questions and more over the course of this project.

Unfortunately, 'there's no such thing as a free lunch', that is to say, a perfect, one-size fits all solution doesn't exist. No one algorithm satisfies all the definitions of fairness. However, gaining a deeper insight into how these algorithms work is key in understanding what aspect of fairness they address, their advantages and disadvantages, and in turn in what context they should be used in. I hope to shed light on this in this project. Fairness is deeply dependent on context but currently, there is a distinct lack of context in the field of fairness in machine learning.

[1] DisruptorDaily: `https://www.disruptordaily.com/7-facts-know-big-data/-` last accessed 13/11/2019

[2] Forbes: `https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-#2bc6de6160ba-lastaccessed11/11/2019-` last accessed 13/11/2019

Chapter 3: Related Work and Ideas

3.1 Examples of Algorithmic Discrimination

If computer scientists are aware of the potential pitfalls, and remain vigilant during the design of machine learning algorithms, they will surely prove to be fair? One would assume computer algorithms should lack any human prejudices? Surprisingly, this isn't the case. Lepri et al [2] describes worryingly that *'even with the best intentions, data-driven algorithmic decision-making can lead to discriminatory practices and outcomes'*. Algorithms can replicate biases already found in data sets, replicate historical biases, or reflect biases found in society today, Lepri et al state[2]. There has been a number of high profiles examples of algorithmic discrimination uncovered recently. Googles sentiment analyzer [1] was designed to determine whether sentences expressed negative or positive sentiment, on a scale of -1 to 1. The algorithm classified statements where users expressed belonging to certain ethnic minorities as negative. For example, it labelled being a homosexual or 'a Jew' as negative. This is a case of the training data being biased, as it was found that 'Jew' was used more frequently by anti-Semitic websites, whereas established sources were more likely to use the term 'Jewish'.

Amazon had to abandon a job recruiting service[2] it had been working on, as the algorithm was found to be discriminating against women. The system was trained using resumes submitted to the company over the past ten years. Most of these applicants were male as the technology sector has been traditionally male dominated. The algorithm, in effect, *'taught itself that male candidates were preferable.'*, penalising resumes with the word 'woman' in them, including female clubs and schools. Another infamous example is predictive policing software[3] that has resulted in more officers being sent to certain disadvantaged areas, because of historic biases. Machine Learning algorithms being used in the public sector is a relatively new concept, yet there has already been extremely concerning examples of bias in these algorithms, which has reflected poorly on the field as a whole.

3.2 Measuring and Rectifying Discrimination

Numerous different measures have been put forward recently to calculate and indeed, even rectify these biases , Binns [3] explains. He elaborates further, *'A common theme is comparing differences in treatment between [privileged] and non-[privileged] groups'*. In this context an unprivileged group is people separated by some protected attribute in which they should have parity ie. race, gender, with the privileged group having had an historical advantage. So why not simply train the algorithm to ignore these protected attributes while making it's decisions? Hardt et al[4] explains

[1] Vive: 'https://www.vice.com/en_us/article/j5jmj8/google-artificial-intelligence-bias'- last accessed on 14/11/2019

[2] Reuters: 'https://www.reuters.com/article/us-amazon-com-jobs-automation-insight/amazon-scraps-secret-ai-recruiting-tool-that-showed-bias-against-women-idUSKCN1MK08G'- last accessed 15/11/2019

[3] NewScientists: https://www.newscientist.com/article/mg23631464-300-biased-policing-is-made-worse-by-errors- last accessed on 15/11/2019

that *'this idea of "fairness through unawareness" is ineffective due to the existence of redundant encodings, that is, ways of predicting protected attributes from other features'*. Ignoring the protected attributes will very often cripple the utility of the algorithm as well, which is detrimental to the whole exercise.

We should consider a system that meets all the measures fairness between privileged and non-privileged groups, Binns [3] concludes, but there is still a problem. Some measures prove to be mathematically impossible to satisfy simultaneously. This raises many questions. *'Which measures of fairness are most appropriate in a given context?'*, and *'Should fairness consists of ensuring everyone has an equal probability of obtaining some benefit, or should we aim instead to minimise the harms to the least advantaged?'* Binns ponders. *'Despite the demand, a vetted methodology for avoiding discrimination against protected attributes in machine learning is lacking'* M Hardt [4] et al illustrates. While bias mitigation and fairness in machine learning is an active and growing field, it is still unclear what metrics and mitigation strategies are most effective or appropriate in a given context.

3.3 Bias Rectification Techniques

Two of the most popular bias rectifying techniques proposed over the last couple of years are *'Certifying and Removing Disparate Impact'* by Feldman et al [5] and *'Reweighing'* [6] by F. Kamiran et al. Disparate Impact occurs when a *'selection process has widely different outcomes for different groups, even as it appears to be neutral,'* M Feldman et al explains. This definition is heavily linked to the definition of discrimination itself. Feldman et al's goals are to quantify and subsequently remove this disparate impact.

The US Supreme court has not recognized a formula to formally define disparate impact, so Feldman et al [5] use *'the 80 percent rule'* which has been advocated by the 'US Equal Employment Opportunity Commission'. The idea is that the prediction rates for any two groups should not differ by more than 80 percent. Feldman et al calculates the disparate impact using the formula and then removes disparate impact by modifying attributes so that the marginal distribution based on the subset of that attribute are equal. They call this *repaired data* that any algorithm can thus be trained on .

This sounds effective in theory but has a number of drawbacks. In cases where different groups actually have different statistical trends, accurate classification is greatly hindered. Disparate Impact Removal does not satisfy both individual and group fairness either. Removing disparate impact ensures a high group fairness that is, groups defined by protected attributes receiving similar treatment, but doesn't guarantee high individual fairness, which is defined as similar individuals receiving similar treatment. Another possible disadvantage is that as it is a preprocessing algorithm, it recognises the training data as the cause of discrimination behind the learning algorithm. This leads to the learning algorithm itself being treated as a 'black box' and can lead to further losses in accurate classification. The algorithm can only be used with numerical non-protected attributes, which is potentially limiting in certain scenarios.

However, Feldman et al's disparate impact remover can be used with any classifier and can work with multiple protected attributes which are two extremely useful features in this space. Being independent from the learning model can also be an advantage as it is isolated from the model. This means that the model does not have to adjust which can add complications.

Kamiran and Calders (2012) [6] proposes another preprocessing technique called reweighing. Reweighing revolves around assigning weights to the tuples in the training data set. *'By care-*

fully choosing the weights, the training data set can be made discrimination-free' Kamiran et al claims. This algorithm only works with one protected attribute, unlike the disparate impact remover. It can be trained with any margin based classifier which is useful. As it is a preprocessing technique it suffers from the same disadvantages as disparate impact remover.

3.4 Tool-kits

What tool-kits are available right now? Industry leaders such as Facebook and Google claim to have been developing bias mitigation algorithms for the past few years but nothing of note has been made available yet. There are however several open source projects currently available.

Fairness Measures[4] is one such toolkit. It provides several *'Fairness Classification Algorithms'* including mean difference which is the *'difference in target score mean values of the privileged vs. unprivileged group'* and impact ratio which is the ratio of positive outcomes for the un-privileged group over the privileged group. It also has several data sets and a link to their code repository on GitHub[5]. Unfortunately, this toolkit only offers means to detect algorithmic bias and lacks any bias mitigation functionality.

Most tool-kits and libraries available lack any methods to mitigate bias. One that does, and one of the more impressive ones at that is Fairness Comparison [6]. It provides several fairness metrics including a measure of disparate impact, and four bias mitigation methods, including Feldman's disparate impact remover among others. It is open source and allows for more metrics and algorithms to be added. Five real-world data sets are also at hand and all their code is also available on GitHub. There was however one toolkit that was even more impressive. That was IBM's AI Fairness 360.

3.5 AIF360

The most comprehensive and robust toolkit available is IBM's AIF360 [7]. It is an open source library which *'aims to unify [the other libraries] and bring [them] together in one open source toolkit a comprehensive set of bias metrics, bias mitigation algorithms, bias metric explanations, and industrial usability.'* Around seventy fairness metrics and eight bias mitigation techniques are provided, as well as seven popular data sets. This is the most by far in all three categories. AIF360 has a website [7] which gives the user a taste of the library.

The website allows the user to select one of three data sets, which an unnamed classifier is trained on behind the scenes. Five fairness metrics are run and graphed for the user, who can then choose one of four bias mitigation techniques including a preprocessing, in-processing or Post-Processing technique. These different techniques affect different parts of the model. Prepossessing modifies the data-set which we have seen already. In-processing modifies the model , while Post-Processing algorithms modify the results. The classifier is then trained on the data set again with the bias

[4]Fairness Meaures Website: 'https://www.fairness-measures.org/' - last accessed 13/11/2019

[5]Fairness Measures Github: \https://github.com/megantosh/fairness_measures_code/tree/master- last accessed 13/11/2019

[6]Fairness Comparison Github: 'https://github.com/algofairness/fairness-comparison' - last accessed 13/11/2019

[7]AIF360 Website: 'https://aif360.mybluemix.net'-lastaccessed13/11/2019- last accessed 13/11/2019

mitigation algorithm selected now in place. The fairness metrics are recalculated and relayed back to the user, comparing results from before. This is a nice introduction to AIF360s functionality, but an introduction only, as most of the heavy lifting is being done behind the scenes.

There are two tutorial notebooks provided, however, which provide the user with a deeper introduction to the code itself. The first notebook is relatively straightforward while the second is much more in-depth, with numerous mitigation techniques on display. IBM also provide an API which greatly elaborates, and explains the tool-kits classes, metrics, and algorithms. Table 3.1 below gives a limited sample of what AIF360 has to offer.

Metrics	Disparate impact Statistical parity difference Average odds difference Equal opportunity difference
Classifiers	Logistic regression (LR), Random forest classifier (RF), Neural Network (NN)
Pre-processing Algorithms	Re-weighing (Kamiran & Calders, 2012) Optimized pre-processing (Calmon et al., 2017) Learning fair representations (Zemel et al., 2013) Disparate impact remover (Feldman et al., 2015)
In-processing Algorithms	Adversarial debiasing (Zhang et al., 2018) Prejudice remover (Kamishima et al., 2012)
Post-processing Algorithms	Equalized odds post-processing (Hardt et al., 2016) Calibrated eq. odds postprocessing (Pleiss et al., 2017) Reject option classification (Kamiran et al., 2012)

Table 3.1: Brief Overview of AIF360s functionality as seen in [7]

3.6 AIF360 vs Aequitas

Another leading toolkit in the fairness aware machine learning field is Aequitas, which is designed to audit machine learning models for discrimination and bias, and to make informed and equitable decisions around developing and deploying predictive tools. Like AIF360, it is open source, is implemented in Python and has a website providing a run through of its functionality. Two datasets are available to choose between on this website, *Adult* and *Compas*, or the user can upload their own custom dataset. A choice of protected attribute, privileged group and six fairness metrics is given to the user, including *disparate impact*. A fairness report is then generated, detailing the results of the test, assigning a 'Pass' or a 'Fail' for each of the criteria specified.

Aequitas does supply the user with a 'fairness tree' to help them decide which metric to abide by for their particular situation. This is a welcome addition and something that is lacking in AIF360. Unlike AIF360 however, Aequitas has no bias mitigation functionality, justifying the former choice for this project. The datasets and metrics available are also encompassed in AIF360.

Chapter 4: Data Considerations

In order to be considered for this project, a data-set must be publicly available and contain a protected attribute (i.e sex, race), which partitions the population into privileged and unprivileged groups. Only binary datasets are considered, meaning the target variable can only be one of two possible instances, known as favourable and unfavourable labels. A favourable label is a desired outcome (eg received loan, achieved promotion), with the unfavourable label being undesired. AIF360 can only be utilised with numerical datasets, although converting data sets with categorical data efficiently is trivial in Python. Datasets considered for this project include:

Dataset	Protected Attribute A	Privileged Group	Unprivileged Group	Protected Attribute B	Privleged Group	Unprivileged Group	Favourable Label	Un-Favourable Label
Adult	Sex	Male	Female	Race	White	Non-White	>50K	<=50K
Bank	Age	>= 25	< 25	-	-	-	Loan	Rejected
Compas	Sex	Female	Male	Race	Caucasian	Non-White	Didn't Recide	Did Recide
German	Sex	Male	Female	Age	>= 25	< 25	Good Credit	Bad Credit
Ricci	Race	White	Non-White	-	-	-	Promotion	Rejected

Table 4.1: Brief Overview of Datasets

Having a range of datasets to effectively analyze the algorithms present was a key consideration in their choosing. *Adult* is the largest dataset with 45,222 rows while *Ricci* is the smallest with 236. A variety of protected attributes and labels were among the criteria. *Adult*, *Bank*, *Compas* and *German* are all integrated in AIF360 as they are some of the most popular datasets in the field of fairness in machine learning, with *Adult* utilised in [5], [8], [6], [9], [10], [11] and [12], *German* in [5], [8] and [10], *Compas* in [10], [10] and [12] and *Ricci* in [5].

Chapter 5: **Fairness in ML**

Fairness in machine learning is all-encompassing term that includes fairness metrics and bias mitigation algorithms. Fairness metrics detect bias in some way, usually comparing classification results between privileged and unprivileged groups. Bias mitigation algorithms are applied to a specific stage of the machine learning pipeline in order to rectify bias.

5.1 Fairness Metrics

Five fairness metrics have been utilised in this project and they include:

Disparate Impact: Ratio of the probability of the favourable label being achieved by the unprivileged group compared to the privileged group.

Mean Difference: Difference of the rate of favourable outcomes for unprivileged group compared to privileged group.

Equal Opportunity Difference: Difference of true positive rates between the unprivileged and privileged group. True positive rate is the number of correctly predicted 'true' outcomes, divided by total 'true' outcomes.

Average Odds Difference: Average difference of false positive and true positive rates, between privileged and unprivileged groups. False positive rate is the number of correctly predicted 'false' outcomes divided by the total.

Theil Index: Generalized entropy measure of all individuals in the dataset, with alpha = 1.

5.2 Categories of Bias Mitigation Algorithms

Pre-processing algorithms work on the data at the start of the machine learning pipeline, i.e before a model has been trained. The training data is inputted, modified and outputted, ready to be fed to a classifier. The fairness aware preprocessing algorithm operates under the assumption that a model trained on biased data will produce biased results. *'Like the fruit of a poisoned tree, an AI model trained on problematic or missing data will likely produce problematic outcomes'*[13]. A myriad of factors could be causing this including an under representation of the unprivileged class or that historical discrimination is present in the training data. Pre-processing does not require the classifier itself to be amended, which can bring about further complications.

In-processing amends the machine learning pipeline during the learning of the model. These type of algorithms often add extra constraints to create a fairness-aware model. In-processing algorithms are the only group of bias mitigation algorithms that do not treat the the classifier as a 'black box', i.e delve into its inner workings. It can be difficult to gauge exactly whats happening without understanding the classification process.

Post-processing algorithms are applied at the end of the machine learning pipeline. They are fitted on the final, classified dataset and the original test dataset, i.e the predicted label dataset and the true label dataset. Post-processing algorithms have the advantage of not being constrained to a limited number of classifiers like pre-processing and in-processing algorithms. This is important as different classifiers perform optimally in different contexts.

5.3 Algorithms

Pre-processing algorithms: *Disparate Impact Remover* [5] repairs data with the aim of minimising the fairness utility trade-off, that is, amending the data could lead to an increase in fairness, but a loss of overall classification accuracy. Firstly, each attribute in the dataset is ranked, followed by the smallest unprivileged group being found. This number is used to determine how many buckets to create. Other unprivileged groups are then divided by the same number, placed into buckets and the median value found. This value is used to represent each bucket. Depending on repair size original data is adjusted relative to the median values. *Reweighing*[6] modifies data by generating weights for each group. This is much less intrusive than Disparate Impact Remover as the labels themselves are not changed. However, unlike Disparate Impact Remover, Reweighing requires the model 'to be able to work with weighted tuples'. Certain classifiers are unable to do so.

In-processing Algorithms: Two variations of *Meta Fair Classifier*[10] are implemented, which is a classifier optimized with regard to a specified fairness metric. The two implementations differ in the fairness metric they have been optimized with, with false discovery rate(fdr) and statistical rate (sr), currently supported. False Discovery rate is a way of conceptualising false positive rate. It is computed by dividing the number of type 1 errors or false discoveries, divided by the total number of rejected hypothesis. Statistical rate is another name for disparate impact, which occurs when a privileged group receives favourable treatment, compared to an unprivileged group. *Prejudice Remover*[11] aims to minimize prejudice in the dataset. Prejudice is described as a statistical dependence between protected features and other information. Two regularized terms are added to a trained logistic regression model. The first regularizer is used to avoid overfitting, the second to enforce fair classification.

Post-processing Algorithims: *Calibrated Equal Odds Post-processing*[12] and *Equalized Odds Post-Processing*[12][4] strive to correct the equal odds difference between a privileged and unprivileged group. *Reject Option Classification*[14] invokes the reject option and labels instances belonging to [un-privileged] and [privileged] groups in a manner that reduces discrimination" Instances in the critical region are values which are neither close to the positive (1) or negative (0) classes, (i.e. 0.5). These values labels are removed and marked as true if they belong to the unprivileged group or false otherwise.

Chapter 6: **Data Context**

Adult Census:[1] Information on 48,842 individuals is represented and whether their income exceeded 50,000/yr, with the favourable label being 'High Income' and 'Low Income' being unfavourable. A favourable label is an advantageous outcome, while an unfavourable label is the opposite. Two protected attributes are contained within the dataset, *sex* and *race*. *Male* and *Female* are the privileged and unprivileged attributes for sex, while *White* and *Non-white* are the privileged and unprivileged attributes for race, respectively. .

Bank:[2] Portuguese banking data revolving around a direct marketing campaign. The target variable is whether the client will subscribe to a deposit. The protected variable is *age*, with participants over twenty five making up the privileged group.

Compas: ProPublica recidivism:[3] Containing data collected from the risk assessment tool COMPAS that was utilised in Broward County, Florida, it includes information such as criminal history and prison time for 6,167 individuals. The algorithms goal is to predict whether an individual will re-offend within two years with *Did not recidivate* being the favourable label and *Did recidivate* unfavourable. This data set also contains two protected attributes, *race* with *Caucasian* being privileged and *Not Caucasian* unprivileged and *sex* with 'Female' being privileged this time and 'Male' unprivileged.

German Credit:[4] This data set has 1000 rows, each representing one individual and 20 columns. Trained on this data set, a classifiers goal is to predict an individual's credit risk with *Good credit* and *Bad credit* being the favourable and unfavourable labels respectively. The data set contains two protected attributes, *sex* and *age*. *Male* is privileged and *Female* unprivileged for *sex*, while *Young* is privileged and *Old* unprivileged for *age*.

Ricci:[5] The *Ricci* dataset contains information on 118 firefighters, based on the Ricci vs DeStefano case, a high profile legal battle based on discrimination in promotion. Five columns include ranks, either Captain or Lieutenant, written exam score, oral exam score and a weighted combination of both. The protected attribute is *race*, containing three values, *white*, *black* and *hispanic*. For simplicity, *black* and *hispanic* were grouped together an *non-white*. In reality promotion was based on not only test scores, but how many spots were available per position and other factors, anyone who achieved a score of over 70% in the combine was assigned the favourable label. The dataset was then doubled in size as some algorithms were unable to produce meaningful results.

[1] Adult Dataset: `https://archive.ics.uci.edu/ml/datasets/adult-` last accessed 12/11/19

[2] Bank Dataset: `https://archive.ics.uci.edu/ml/datasets/adult-` last accessed 12/11/19

[3] Compas Dataset: `https://github.com/propublica/compas-analysis-` last accessed 12/11/19

[4] German Dataset: `'https://archive.ics.uci.edu/ml/datasets/Statlog+\%28German+Credit+Data\%29'-` last accessed 12/11/19

[5] Ricci Dataset: `https://archive.ics.uci.edu/ml/datasets/adult-` last accessed 12/11/19

Chapter 7: Implementation

7.1 Design

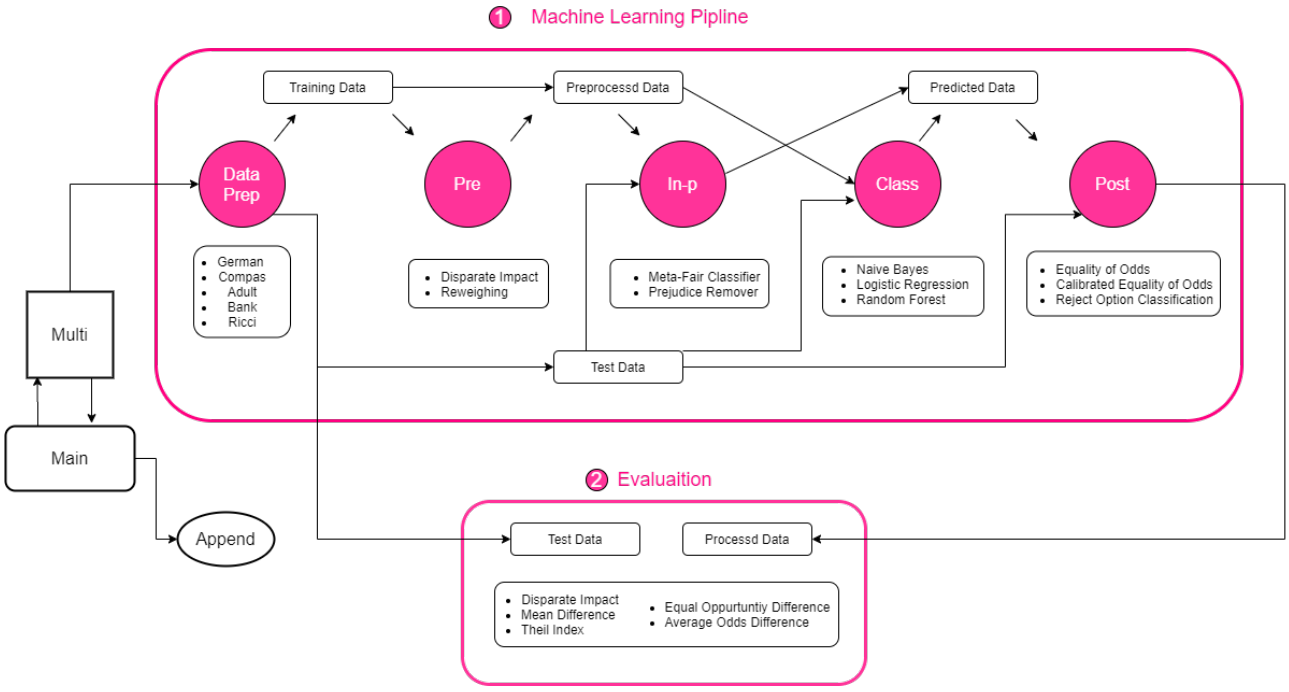


Figure 7.1: Data Life Cycle, consisting of machine learning pipeline and evaluation

7.1.1 Machine Learning Pipeline

1. Dataset and protected attributes are chosen based on user input. The dataset is divided with a standard 70/30 split in *Data Prep*.
2. The test set is copied and sent to the end of the machine learning pipeline to be used in evaluation.
3. The training set is passed to *Pre* where a preprocessing algorithm is either applied or not.
4. The training data is passed on to either *In-p* or *Class* where it used to train a model. The testing data is classified, creating a predicted dataset either using a fairness aware classifier in *In-p* or a standard scikit-learn classifier in *Class*.
5. The predicted data is received by *Post* where a post-processing algorithm is either applied or not. The post-processing algorithm evaluates and alters labels in the predicted dataset, based on the true labels in the original test data.
6. The processed dataset is evaluated by being compared to the original, unmitigated test dataset.

7.1.2 Fair Test

Ensuring data is not tampered with at any stage of the pipeline is paramount in ensuring a fair test. All datasets bar *Ricci* are contained within AIF360 so no cleaning was necessary. Data is split into training and test sets initially. The original test set is passed to the end of the machine learning pipeline, to be compared with the pipeline's outputs. It was ensured that this copy of the test set was at no stage involved in the machine learning process.

Models should only be fitted on the training data which includes the target value or label, never the testing data. The model is then used to predict the target value for the test set, based on its training. It is imperative that the model does not have access to the test datasets target value at any stage of the prediction process.

In order to provide accurate and fair results which would allow a large number of fairness aware machine learning algorithms to be compared, each algorithm was tested using the same datasets, with the dataset split and the resultant dataset evaluated by the same measures.

The main consideration when creating the pipeline was its re-usability. The pipeline would be required to run hundreds of rows of various combinations of Bias Mitigation Algorithms.The output of each function needed to be compatible as input to not only various other functions, but oftentimes itself. Multiple preprocessing and Post-Processing algorithms were often utilised. Bias mitigation algorithms, fairness metrics and most datasets were provided by AIF360, making it the basis of the framework for this project.

7.2 AIF360

All data is required to be in the Dataset class or a subclass in order to utilise AIF360s fairness metrics and bias mitigation algorithms. This class is vital as it identifies featues, labels, privileged classes, unprivileged classes and protected attributes for the mitigation algorithms. All datasets except *Ricci* were imported directly from AIF360 and are part of the *BinaryLabelDataset* subclass. This subset only has two labels, unfavourable or favourable. This means those datasets were cleaned and prepared adequately, already. Converting *Ricci* to the proper format was simple.

```
BinaryLabelDataset(favorable_label='1',
                   unfavorable_label='0',
                    df=df,
                    label_names=['Promotion'],
                    protected_attribute_names=['Race'],
                    unprivileged_protected_attributes=['0'])
```

AIF360s *BinaryLabelDataset* can only work with numerical data so the categorical data was converted

The *BinaryLabelDataset* class is used in every part of the fairness aware machine learning pipeline. Pre-processing algorithms are fitted on and transform the training dataset.

```
DI = DisparateImpactRemover(repair_level=1.0, sensitive_attribute = sens)
dataset_t = DI.fit_transform(data)
```

AIF360s in-processing algorithms work similarly to standard scikit-learn classifiers, another example of its flexibility and usability. Logistic Regression, Random Forest and Naive Bayes are the three classifiers used, as they are popular, intuitive and easy to implement. The appropriate model was chosen, the model was fitted on the training input samples and the target values, and subsequently used to predict target values for the test set.

```
MFC = MetaFairClassifier(tau=0, sensitive_attr= sens, type = 'sr')
MFC = MFC.fit(data)
data_pred = MFC.predict(dataset_test)
```

Post-processing algorithms are fitted on the dataset containing true labels, i.e the original test dataset, and the dataset containing predicted labels in order to compute new predicted labels.

```
ROC = RejectOptionClassification(privileged_groups = privileged_groups,
                                 unprivileged_groups = unprivileged_groups)
ROC = ROC.fit(dataset_test, data_pred)
```

AIF360s evaluation metrics are performed using the *Metric* Class and its subclasses. *Disparate Impact* and *Mean Difference* require the subclass *BinaryLabelDatasetMetric* class, which evaluates a single *BinaryLabelDataset*, the predicted dataset in this case.

```
metric = BinaryLabelDatasetMetric(dataset,
                                  unprivileged_groups=unprivileged_groups,
                                  privileged_groups=privileged_groups)
mean_diff = metric.mean_difference()
dis_impact = metric.disparate_impact()
```

In contrast *Theil Index*, *Average Odds Difference* and *Equal Opportunity Difference* require the *ClassificationMetric* class which examines two datasets as input. The first dataset inputted is the the one containing true labels, and the second contains the predicted labels, i.e. the original test data and the predicted data.

```
classified_met = ClassificationMetric(cm_dataset,
                                      classified_dataset,
                                      unprivileged_groups=unprivileged_groups,
                                      privileged_groups=privileged_groups)
theil = classified_met.theil_index()
av_odds = classified_met.average_odds_difference()
eq_opp_diff = classified_met.equal_opportunity_difference()
```

7.3 Evaluation Metrics

In addition to these fairness metrics are a number of performance metrics. The performance measures utilise the same formulas as the fairness metrics such as true positive rate etc. The key difference between accuracy and fairness metrics is that the fairness metrics compare classification differences strictly between privileged and unprivileged groups, whereas performance metrics deal with overall classifier performance i.e how many predictions are correct. Accuracy calculates the number of correct predictions, divided by total. Precision is the fraction of relevant instances among the total retrieved whereas recall is the fraction of the relevant instances retrieved.

An ROC curve is a plot of true positive rate vs false positive rate. That is, the number of true predictions returned, divided by total true values (also known as Recall), while false positive rate is the ratio of true predictions returned that were false, divided by total false values. The ROC curve is a plot of how these two measures change over different classification thresholds. The threshold in this context being how similar the value has to be to the positive class to be assigned to it. Making comparisons between two different classifiers was done using the Area Under the Curve (AUC). A better classifier will have a curve closer to the top left corner of the plot, giving it a larger AUC, closer to 1. Most classification performance metrics rely heavily on the threshold chosen. This is not the case with AUC as it is an average over all thresholds. It performs a 'broader' view, almost a summary of classifier performance, and is often the metric of choice in binary classification problems, although this can be case specific. Although accuracy, precision and recall have also been considered, it is the main performance metric on which algorithm will be evaluated in the project .

7.4 Parallelisation

As this project required hundreds of combinations of machine learning algorithms to be run, there was a significant time cost. In order to speed this process up, a parallel solution was explored. A key strength of Python is its extensive libraries, and this is no different when it comes to parallel programming. Two multithreading modules are available: *multithreading* and *multiprocessing*. Usually, threading involves dividing the workload into sections, where each thread is assigned a subsection of work and is a separate processor/processor core.

7.4.1 Multithreading v Multiprocessing in Python

Threading allows the user to run parts of a program concurrently. Unfortunately Python is not a powerful tool for threading, as threads merely appear to be executing at the same time, but in fact, only one thread is ever being executed at any given time. Python cannot access multiple processor cores due to the *GIL* or 'Global Interpreter Lock', which limits the language to execution on one core. It can in fact be detrimental to utilise this module for CPU intensive tasks, as run time t will result in total execution time plus time taken to switch between threads.The multithreading module should be utilised for I/O heavy tasks like querying a database, as there is a lot of time spent idle and threads can be run concurrently, saving time, but not in parallel. The multiprocessing module does not abide by these restrictions.

The multiprocessing module on the other hand allows the user to bypass the *GIL* limitations, by running Python multiple times essentially. It enables Python to utilise multiple CPU cores and is preferred for CPU intensive tasks which make it perfect for this project.

The number of processes should not be set to a number greater than the number of cores on the host computer as processes would then compete for CPU resources, causing slowdown.

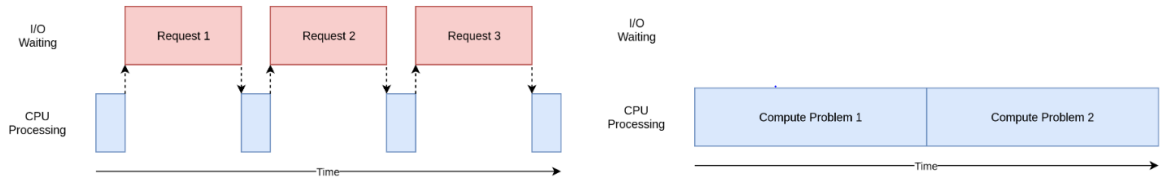


Figure 7.2: The graph on the right displays an I/O-bound problem. The blue box indicates the CPU being engaged, while the red represents time waiting for completion of I/O operations. Pythons multithreading module is more adept at these tasks. The left graph explains a CPU-bound problem, with the CPU being engaged throughout. Pythons multiprocessing module is a better fit to handle this situation.[2]

7.4.2 Queue and Deadlock

The multithreading and multiprocessing modules, although extremely different in practice, are quite similar in implementation in Python albeit with some slight nuances, including multiprocessing utilising a slightly different *multiprocessing queue*, which allows data to be transferred between processes.

[2]Python Concurrency Graphs: `https://realpython.com/python-concurrency/` - last accessed 08/05/2020

The input list is a list of lists, split in main so each process gets a subsection of the workload. Each sub-list is used as input for the various machine learning functions i.e. 0,1,2 will choose the first protected attribute, the second dataset and the third preprocessing algorithm. In *Pre*, *In-p*, Class and Post, *0* corresponds to 'skipping that process', returning the original inputted dataset unchanged. All data has to run through either the *In-p* or *Class* function i.e has to be classified, but cannot run through both. For each sub-list in the list, the pipeline is run, eventually putting the metrics into the queue at the end. A queue is the only data structure viable for this as it is thread and process safe, i.e many threads are accessing it sequentially but it uses a first come first serve basis, to avoid deadlock. However after an initial small number of test cases running as expected, when the workload was increased, deadlock occurred.

It was discovered that the queue had to be emptied in *main* before the *.join* method was called. This method instructs the main process to wait for the sub-processes, otherwise the sub-process would run through the entire program, on their own, which is usually detrimental when their workload has to be unionised in some way. In this case the processes had to wait for one another before the data was appended to the dataframe. If the queue is not emptied, a process will be blocked in the OS while trying to *put* something in the queue, waiting for another process to *get* something from the queue. In instances when there is a small amount of data, the processes can be joined and then the data collected and this was indeed the case here, with the pipeline initially running smoothly with a small amount of training data, but entering deadlock when burdened with a larger amount.

Chapter 8: **Evaluation**

The core of this project is to benchmark the various algorithms available that claim to mitigate bias in machine learning. How the classifiers are evaluated is a key part in this challenge. The evaluation is split into two main sections, the first analyses each bias mitigation algorithm separately, and then in combination with one another. They are judged on two fronts, a fairness metric and a more general evaluation method. Five separate fairness metrics are considered against one overall performance metric, AUC. The second section highlights the top fifteen combinations for two different criteria, rank and score.

In section one every bias mitigation algorithm was evaluated, by the same methods as to ensure a fair test. Every algorithm was implemented on five separate datasets, and then judged on five separate fairness metrics. The graphs best representing the overall trends were selected for this section, appearing in alphabetical order, with the entire results available in the Appendix. The graphs plot AUC against each of the five fairness metrics, in order *Disparate Impact*, *Mean Difference*, *Theil Index*, *Average Odds Difference* and *Equal Opportunity Difference*. AUC is on the x-axis with the fairness metric on the y-axis. All fairness metrics have an ideal value of 0. (*Disparate Impact* is usually 1, those values have been taken from one and the absolute value graphed), meaning the closer the points are to the x-axis, the better they score on fairness. AUC has a target value of 1, so the further the points are to the right, the more accurate the classifier. A line of best fit is drawn (for pre-processing and post-processing). Ideally there should be a negative correlation, with the line starting higher on the left and lower on the right. This correlation indicates a relationship between classification accuracy and fairness, which is a very desirable trait in the field.

There are two main criteria on which the results will be judged. The first is the line of regression, in which a negative correlation is the desired value. The second is that there is a tangible gain in fairness. The blue dots on the chart represent the fairness score and AUC of the corresponding unmitigated classifiers which will be used as a benchmark. For example, reweighing with Naive Bayes would be represented as a blue point, while Naive Bayes without reweighing is in red. The red points being closer to the x-axis than the blue indicates a gain in fairness, while the opposite represents a loss. Following, red points further to the right indicates more desirable AUC score overall. However as most bias mitigation algorithms add constraints to a classifier, a gain in AUC overall is unlikely. Overall change in AUC is unlikely during pre-processing as it adjusts the training data only.

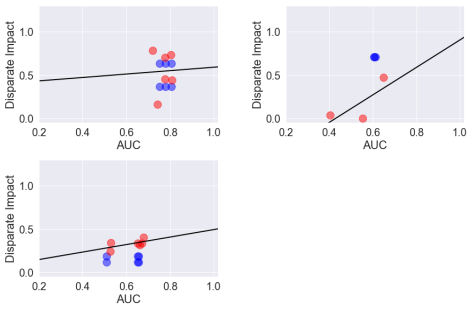
To begin the algorithms were evaluated alone before being implemented in tandem with one another. *Meta Fair Classifier - Statistical Ratio* performed identically to *Meta Fair Classifier -False Discovery Ratio* across all tests, and will simply be referred to as *Meta Fair Classifier*. Logistic Regression was chosen as the benchmark for all in-processing algorithms as the prejudice remover algorithm is based on Logistic Regression.

8.1 Pre-processing

8.1.1 Disparate Impact Remover

Disparate Impact

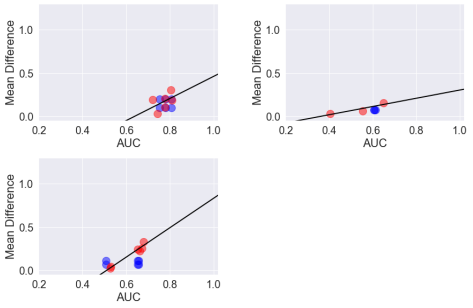
Figure 8.1: Adult, Bank and Compas Datasets



Disparate Impact Remover returns mixed results with regard to *Disparate Impact*, the very metric it is aimed at satisfying. There were no negative correlations between AUC and Disparate Impact across the five datasets tested. No notable improvement is recorded in *Adult* or *German*, while every row in the *Compas* and *Ricci* dataset returned a poorer *Disparate Impact* than the scikit-learn equivalent. *Bank* performed well however, with every row having a lower *disparate impact* score. *Compas'*, where the algorithm performs poorly, original disparate impact is much lower to begin with than *Bank*, notably. AUC is mostly left unchanged, apart from *Bank* and *Ricci*, where the mitigated results return a poorer score.

Mean Difference

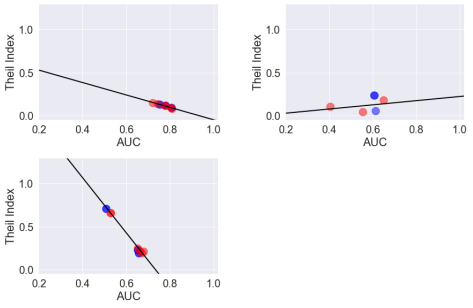
Figure 8.2: Adult, Bank and Compas Datasets



Mean Differences returns similar results to *Disparate Impact* with no negative correlations once again. *Compas* shows a slight improvement in results compared to *Disparate Impact*, while *Bank* performs poorer. No datasets returns substantially fairer results, including *Bank*.

Theil Index

Figure 8.3: Adult, Bank and Compas Datasets

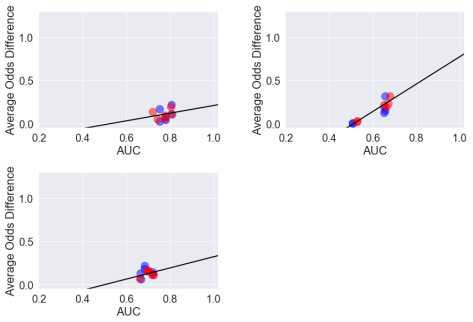


Results are once again mixed for *Theil Index*. Two datasets return a negative correlation between AUC and *Theil Index*, *Adult* and *Compas*. *Bank* has an undesirable positive correlation, but every row returned a more optimal score than the original, closer to the x-axis. *Compas* has a strong negative correlation, but this is mostly due to one outlier, that has a large *Theil Index* originally. Apart from *Bank*, the mitigated results are quite similar to the unmitagted across all datasets.

Average Odds Difference

Disparate Impact Remover performs poorly across all datasets on *Average Odds Difference*, with no desired correlations or significant gains in fairness.

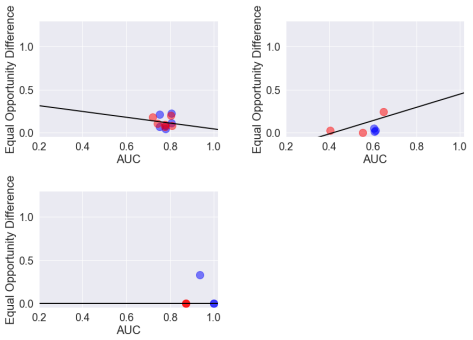
Figure 8.4: Adult, Compas, German Datasets



Equality Opportunity Difference

Two negative correlations are observed when *Equal Opportunity Difference* is measured, in *Adult* and *Ricci*. The remaining three datasets perform similarly to before, with results clustered closely and hard to draw any significant conclusions from. Across all datasets there are no strong gains in fairness, apart from *Ricci* and to some degree *Adult*, where mitigated results are normalising toward the line of best fit.

Figure 8.5: Adult, Bank, Ricci Datasets

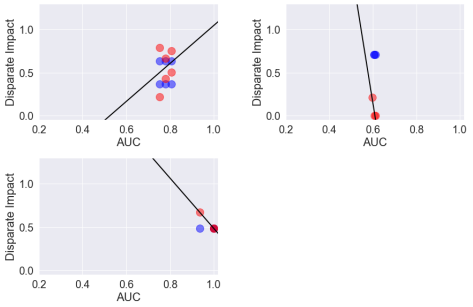


8.1.2 Reweighing

Disparate Impact

Reweighing also returns mixed results with regard to *Disparate Impact*. *Adult*, *Compas* and *German* show no substantial change in fairness. *Reweighing* is extremely effective on the *Bank* dataset with every mitigated point closer to the x-axis, than those unmitigated. *Ricci* returns a negative correlation here, although there isn't a sizeable increase in fairness.

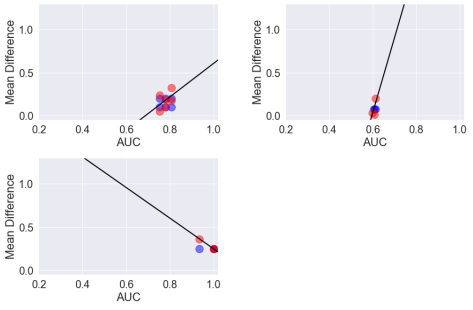
Figure 8.6: Adult, Bank and Ricci Datasets



Mean Difference

Mean Differences performs similarly to *Disparate Impact Remover* on *Mean Differences*, with no desired negative correlations. No datasets returns substantially more fair data, with results scattered. There is a slight loss in fairness in *Ricci*.

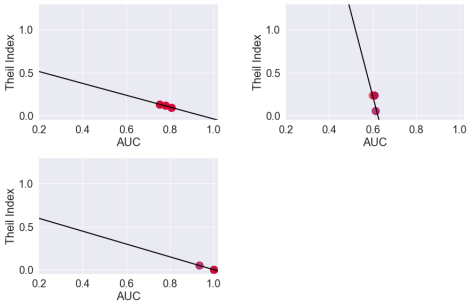
Figure 8.7: Adult, Bank and Ricci Datasets



Theil Index

Although there are three negative correlations, almost every result is unchanged from the original value.

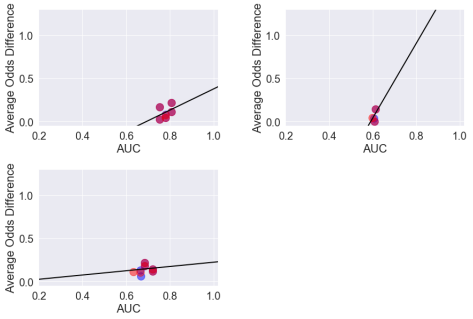
Figure 8.8: Adult, Bank and Ricci Datasets



Average Odds Difference

Reweighing performs poorly across all datasets on *Average Odds Difference*, with no desired correlations or significant gain in fairness, just as *Disparate Impact Remover* did.

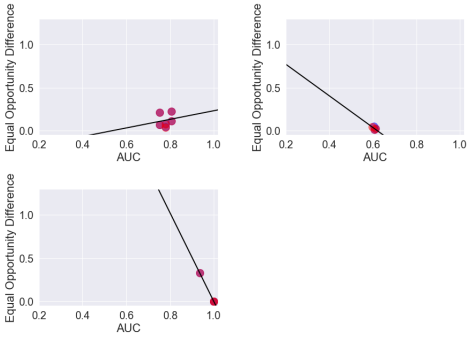
Figure 8.9: Adult, Bank, German Datasets



Equality Opportunity Difference

Equal Opportunity Difference presents two desires correlations in *Bank* and *Ricci*, although it is similar to before as no significant gain in fairness is made, apart from *Bank* where there is a small gain.

Figure 8.10: Adult, Bank, Ricci Datasets



Summary

Both pre-processing algorithms performed poorly across nearly all fairness metrics and datasets, with a few exceptions, mostly in the *Bank* Dataset. These algorithms work solely on the training data, with the assumption that eliminating bias here will train a fairer model, but these results were not seen further down the machine learning pipeline, with vast majority of fairness scores unchanged. AUC remained unchanged throughout.

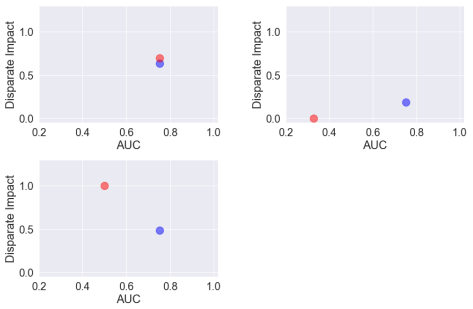
8.2 In-processing

8.2.1 Prejudice Remover

Disparate Impact

Prejudice Remover performs well on *Disparate Impact*. Two datsets, *Bank* and *Compas* show a significant increase in fairness compared to the unmitigated classifier. *Ricci* is the only datset where *Disparate Impact* gets significantly worse. There are significant losses in AUC across four of the five datasets, excluding *Adult*, whose fairness score also remains unchanged, along with *German*. This loss in AUC in all except *Adult*, remains constant throughout *Prejudice Remover* testing.

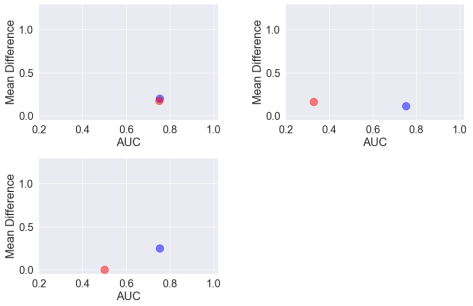
Figure 8.11: Adult, Compas, Ricci Datasets



Mean Difference

No datasets undergo a significant change in mean difference, apart from *Ricci* where the mitigated dataset performs better than the unmitigated.

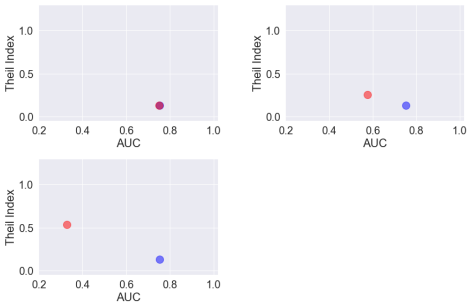
Figure 8.12: Adult, Compas and Ricci Datasets



Theil Index

Adult shows no significant change in fairness, whereas the remaining four have a worse *Theil index* than the equivalent original score.

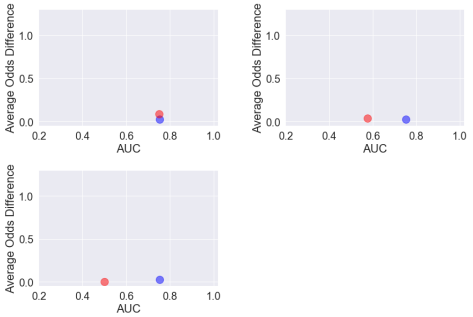
Figure 8.13: Adult, Bank and Compas Datasets



Average Odds Difference

Prejudice Remover performs poorly across all datasets on Average Odds Difference, with no measurable gain in fairness, apart from *Ricci*.

Figure 8.14: Adult, Bank and Ricci Datasets



Equality Opportunity Difference

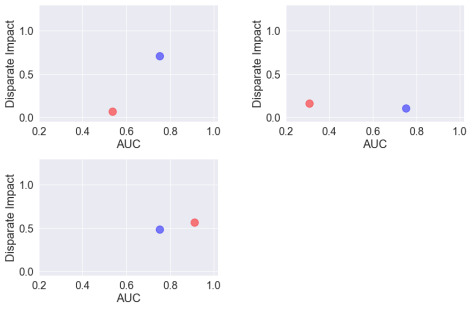
Equal Opportunity Difference is similar to before, with no serious gains in fairness, apart from a slight gain in *Ricci*.

8.2.2 Meta Fair Classifier

Disparate Impact

Bank and *Compas* have a much improved fairness score, while *German*, *Adult* and *Ricci* show slight disimprovement. *Adult* exhibits similar AUC in mitigated and unmitigated results once again, while *Ricci* has a slight gain in AUC. The remaining three datasets all experience a heavy loss in this area. These AUC scores once again remain consistent throughout this algorithms testing.

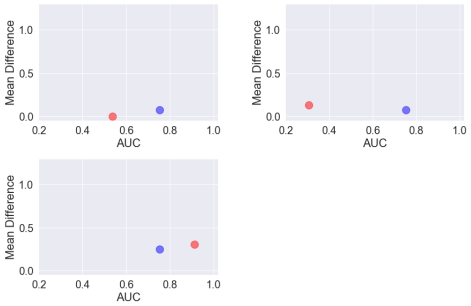
Figure 8.15: Bank, German and Ricci Datasets



Mean Difference

The algorithm performs similarly on *Mean Difference* as it does on *Disparate Impact*. *Adult*, not pictured, undergoes no significant change, *Bank* and *Compas'* gains in fairness are offset by poorer AUCs. *German* also has a poorer AUC, but no gain in fairness. *Ricci* has a stronger AUC but slightly poorer fairness.

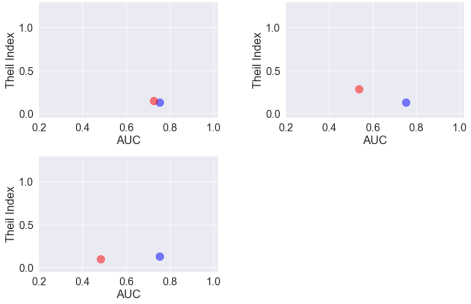
Figure 8.16: Bank, German and Ricci Datasets



Theil Index

Results are similar to before. Once again *Adult* is mostly unchanged. *Bank*, *Compas* and *German* all experience a worse AUC, with gains in fairness in the former two. *Ricci* experiences a stronger AUC and fairness here.

Figure 8.17: Adult, Bank and Compas Datasets



Average Odds Difference

Average Odds Difference returns identical results to *Mean Difference* for *Prejudice Remover*.

Equality Opportunity Difference

Results are once again identical to *Mean Difference* .

8.2.3 Summary

Both in-processing algorithms display larger gains in fairness than the pre-processing algorithms, with the poorer in-processing algorithm, *Prejudice Remover*, performing similarly to the stronger pre-processing algorithm, disparate impact. *Meta Fair Classifier* strongly outperforms its counterpart here, with fairness gains in two datasets across all five metrics tested, those datasets being *Bank* and *Compas*. *Prejudice Remover* performed better on *Disparate Impact* than any algorithm tested so far, with two datsets experiencing a substantially more desirable score here.

Both in-processing algorithms post significant losses in AUC across the board, apart from the *Adult* dataset. Utilising a large dataset could offset this loss, as *Adult* is by far the largest dataset on test, however there were no gains in fairness in the *Adult* dataset for any metric. Overall a stronger performance fairness-wise compared to pre-processing, somewhat off-set by the losses in AUC.

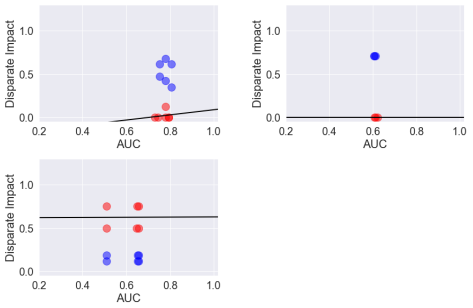
8.3 Post-processing

8.3.1 Equalized Odds Post-Processing

Disparate Impact

Overall *Equalized Odds Post-Processing* performs extremely well on Disparate Impact. Four of the five datasets return fairer mitigated results overall, (apart from one outlier in *German*). *Compas* is the exception here, with every point scores worse on disparate impact than the unmitigated equivalent. Notably *Compas* has the lowest original disparate impact. AUC is slightly poorer in *Adult* and slightly stronger in *Bank*, while being equal to the unmitigated score everywhere else, this will remain constant throughout, yet again.

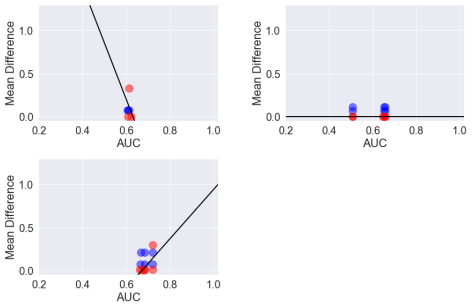
Figure 8.18: Adult, Bank, Compas Datasets



Mean Difference

Equalized Odds Post-Processing performs well on *Mean Difference* also, There are strong gains in fairness across the board, with almost every combination scoring lower than the original. Four negative correlations are present in *Adult*, *Bank*, *Compas* and *Ricci*. The exception to this is *German*, which is likely due to an outlier. There are strong gains in fairness here apart from one point.

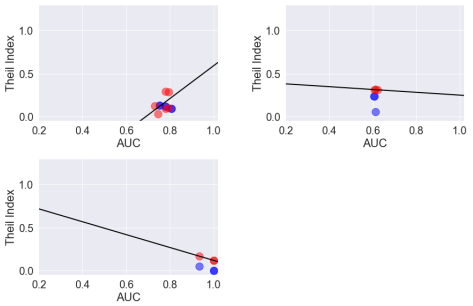
Figure 8.19: Bank, Compas and German Datasets



Theil Index

Theil Index provides more mixed results. There are four negative correlations, which is a strong result, However this is deceiving as fairness scores are universally poorer than the unmitigated dataset. Apart from *Adult*, every combination of every dataset returns a higher *Theil* than the original, unmitigated value.

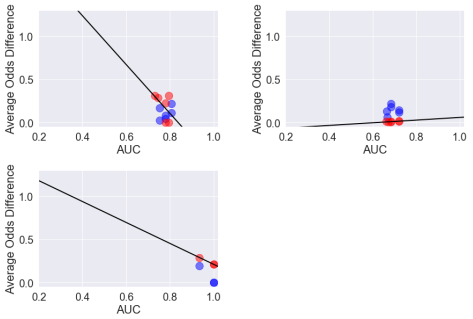
Figure 8.20: Adult, Bank and Ricci Datasets



Average Odds Difference

Mixed results for *Average Odds Difference*, aswell Once again there is a strong number of negative correlations with all five datasets achieving this. *Adult* and *Bank* experience some gains in fairness, but some losses. *Compas* and *German* results are strong, with every combination returning a stronger fairness score. *Ricci* performs poorly with every row receiving a worse fairness score than the unmitigated original.

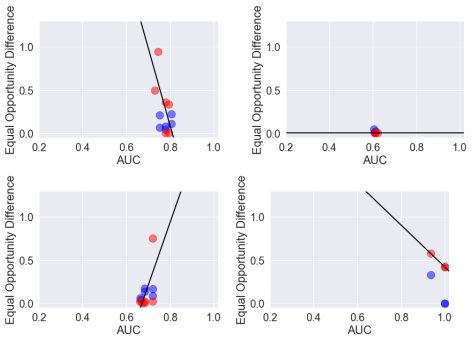
Figure 8.21: Adult, German, Ricci Datasets



Equality Opportunity Difference

There are two negative correlations here with, with two more almost horizontal. Although *Adult* has the desired correlation, a strong gain in fairness is lacking. The algorithm performs well on the *Bank*, *Compas* and *German* datasets with every combination achieving a stronger fairness score. *German* would have a stronger correlation, but for one point. *Ricci* is the outlier once again, with point points achieving a worse fairness score than the original, unmitigated versions.

Figure 8.22: Adult, Bank, German and Ricci Datasets

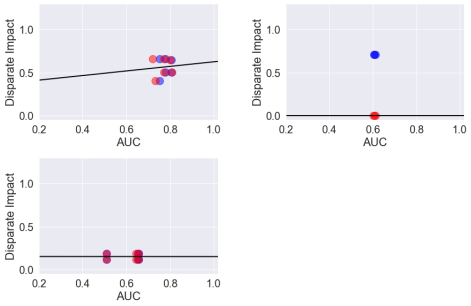


8.3.2 Calibrated Equalized Odds Post-Processing

Disparate Impact

Calibrated Equalized Odds Post-Processing neither performs strongly or weakly on Disparate Impact, with no overall change in most cases, apart from *Bank*. The algorithm perform well on the *Bank* dataset with all combinations achieving a perfect fairness score. AUC remains identical and will for every metric.

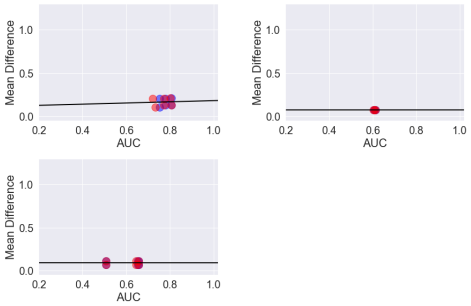
Figure 8.23: Adult, Bank and Compas Datasets



Mean Difference

The algorithm performs similarly on Mean Difference as it does on *Disparate Impact*, as seen in Fig. 8.24. This time, however, there is no real fairness change to any dataset, including *Bank*.

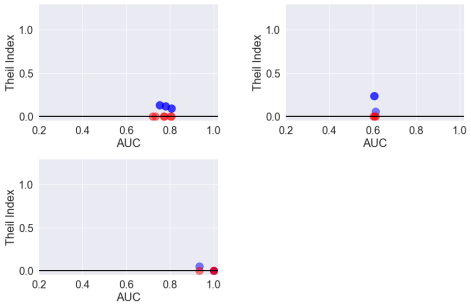
Figure 8.24: Adult, Bank and Compas Datasets



Theil Index

Calibrated Equalized Odds performs strongly on *Theil Index* as shown in Fig 8.25, with every combination of mitigated results posting the strongest fairness score possible of 0, across all datasets. All correlations reflect these strong results, as they are horizontal.

Figure 8.25: Adult, Bank and Ricci Datasets



Average Odds Difference

Calibrated Equalized Odds posts identical results with regard to average odds difference as it did on *Theil Index* with every combination achieving the perfect score.

Equality Opportunity Difference

Once again the perfect fairness score is achieved universally.

8.3.3 Reject Option Classification

Identical Results to *Calibrated Equalized Odds*.

8.3.4 Summary

The post-processing algorithms achieve the strongest results by a considerable distance. *Equalized Odds Post-Processing* performs stronger on *Disparate Impact* and *Mean Difference* than any other algorithm tester, including *Prejudice Remover*. There were universal gains in fairness across almost every dataset on these two metrics. Performance was also strong on *Average Odds Difference* and *Equality of Odds* with a number of datasets in both these metrics posting better scores when paired with the algorithm. The exception was *Theil Index* where scores were poorer.

On the other hand *Calibrated Equalized Odds* and *Reject Option Classification* results are mostly unchanged on *Disparate Impact* and *Mean Difference*, but transformed *Theil Index*, *Average Odds Difference* and *Equal Opportunity*, with perfect scores on these metrics, without exception. This was all achieved with no major loss in AUC, compared to the in-processing algorithms.

From all three individual categories, post-processing returned the strongest results by a sizeable margin. *Equalized Odds Post-Processing* was the strongest performer in *Disparate Impact* and *Mean Difference*, while *Calibrated Equalized Odds* and *Reject Option Classification* mitigated *Theil Index*, *Average Odds Difference* and *Equal Opportunity Difference* perfectly.

8.4 Pre-Processing & In-Processing

There was no significant difference to the in-processing algorithms when paired with pre-processing, therefore this section was moved to the Appendix.

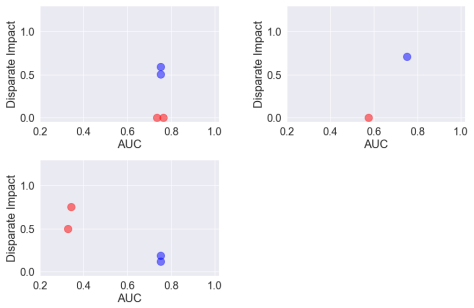
8.5 In-Processing & Post-Processing

8.5.1 Prejudice Remover & Equalized Odds Post-Processing

Disparate Impact

Fairness results are mixed with improvement in *Adult*, *Bank* and *German*, no change in *Ricci* and a loss in *Compas*. AUC losses are present in all datasets bar *Adult*, which is common across all metrics.

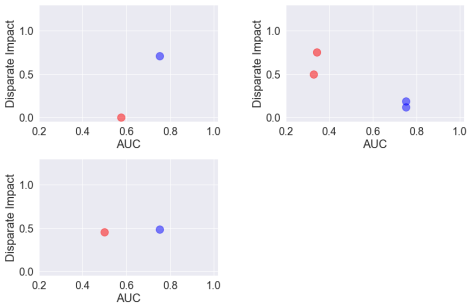
Figure 8.26: Adult, Bank and Compas Datasets



Mean Difference

Every combination across all datasets has a stronger fairness score here.

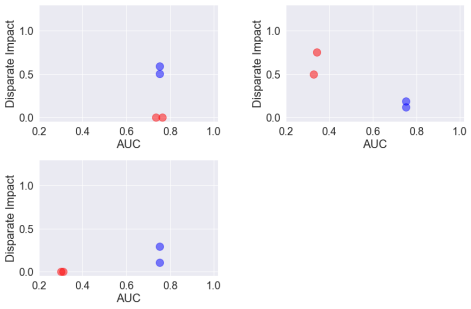
Figure 8.27: Bank,Compas and Ricci Datasets



Theil Index

Theil Index is the opposite, with poorer fairness scores across the board.

Figure 8.28: Adult, Compas and German Datasets



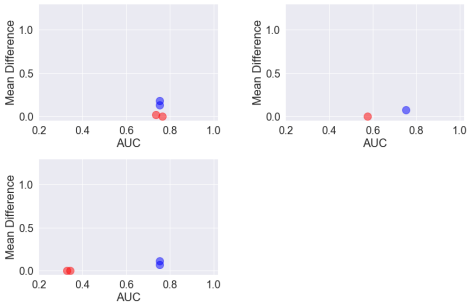
Average Odds Difference

Apart from *Adult*, mitigated fairness results are stronger than unmitigated scores.

Equality Opportunity Difference

Identical to *Average Odds Difference*, with much improved fairness scores.

Figure 8.29: Adult, Bank and Compas Datasets



8.5.2 Meta Fair Classifier & Equalized Odds Post-Processing

Identical Results to *Prejudice Remover & Equalized Odds Post-Processing* apart from gains in *Ricci* AUC here across all metrics, and a gain in *Disparate Impact* for *Ricci*

8.5.3 Summary

Equalized Odds Post-Processing performed well when paired with either of the in-processing models, with it posting moderately more desirable scores with *Meta Fair Classifier*. On *Disparate Impact* and *Mean Difference* it performed just as well as it did in the individual category. However with the in-processing algorithms there were superior performances on average odds difference and equality of opportunity difference, making this combination the most well rounded so far, even with its poor performance on *Theil index*. This comes at the cost of all combinations experiences a

loss in AUC across all data sets, apart from *Adult*, just as the in-processing algorithms experienced when utilised on their own.

8.5.4 Prejudice Remover & Calibrated Equalized Odds Post-Processing

Disparate Impact

Poor results here for Disparate Impact, with only one dataset exhibiting improvement in fairness scores, just as *Calibrated Equalized Odds* performed individually.

Mean Difference

Identical poor results to *Calibrateds* individual performance.

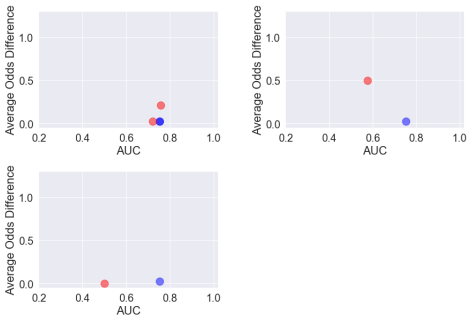
Theil Index

Strong results with all datasets except *Bank* show noticeable improvements in fairness, once again adding the in-processing algorithm has not amended these results.

Average Odds Difference

Significant decreases in fairness, with *Ricci* and to some degree, *Adult* being the only exception.

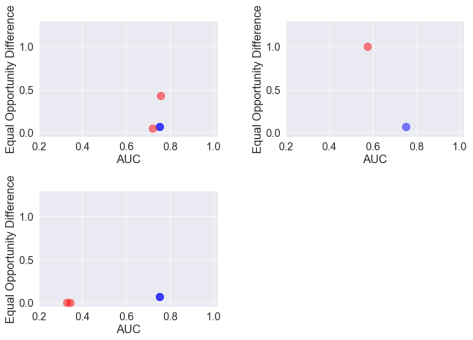
Figure 8.30: Adult,Bank and Ricci Datasets



Equality Opportunity Difference

Improvement in fairness in all datasets except *Adult* and *Bank*. There is a significant decrease in *Equal Opportunity Difference* in *Bank*.

Figure 8.31: Adult,Bank and Compas Datasets



8.5.5 Meta Fair Classifier & Calibrated Equalized Odds Post-Processing

Similar performance to *Prejudice Remover & Calibrated Equalized Odds Post-Processing*.

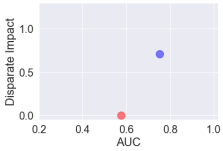
8.5.6 Summary

Unlike *Equalized Odds Post-Processing* did, pairing *Calibrated Equalized Odds* when paired an in-processing algorithm proves detrimental to its performance with a standard classifier. its results on the *Mean Difference* and *Disparate Impact* metrics were poor just as before, but where it performed exceptionally on *Theil*, *Average Odds* and *Equality of Odds* before, it was hampered here. It did not perform nearly as well on these metrics as *Calibrated* performed individually, especially on *Average Odds Difference*. When paired with the loss in AUC, *Equalized Odds Post-Processing* should not be paired with an in-processing algorithm, according to these tests.

8.5.7 Prejudice Remover & Reject Option Classification

Disparate Impact

Figure 8.32: Bank Dataset



Fairness is equal for all datasets apart from *Bank*, where there is a significant improvement in *Disparate Impact*. There are AUC losses in every dataset, apart from *Adult*, just as before.

Mean Difference

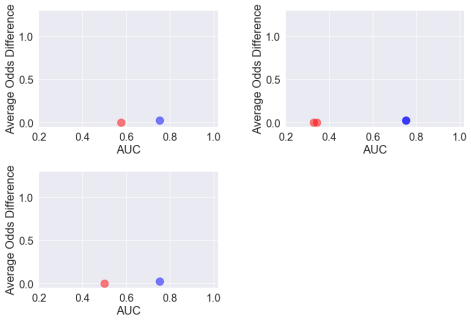
There is no difference in Mean Difference for mitigated and unmitigated results, just as in the individual performance.

Theil Index

Theil exhibits more desirable results across the board, which is the same as before.

Average Odds Difference

Figure 8.33: Bank,Compas and Ricci Datasets



Mitigated *Average Odds Difference* is similar/slight increase compared to unmitigated, as seen in Fig 8.33.

Equality Opportunity Difference

Identical to *Average Odds Difference*.

8.5.8 Meta Fair Classifier & Reject Option Classification

Identical Results to *Prejudice Remover & Reject Option Classification* apart from loss in *Ricci* AUC here across all metrics,

8.5.9 Summary

Pairing *Reject Option Classification* with an in-processing algorithm did not reveal the same loss in fairness as *Calibrated Equalized Odds* did, even though these algorithms performed identically when paired with a standard scikit-learn classifier.

However unlike, *Equalized Odds Post-Processing*, there was no benefit to *Reject Option Classification* and either in-processing algorithm working in tandem, meaning it should probably be avoided to prevent the significant loss is AUC.

8.6 Pre-processing & In-processing & Post-Processing

Paring a pre-processing algorithm with the previous combinations did not unearth any notable differences and was thus moved to the Appendix.

8.7 Rank and Score Analysis

8.7.1 Performance Rank

This table shows the top fifteen combinations for the performance rank category. Performance Rank was computed by ranking each performance metric (not fairness) individually, adding these ranks and ranking the resulting number. The performance ranks utilised were Accuracy, Precision, Recall, and AUC. Accuracy, Precision and Recall are three of the most popular and commonly implemented metrics to evaluate classifier performance in machine learning.

The rank was then averaged for the same combinations across different datasets. For this reason, the dataset column is not important in the table, however it displays a variety of datasets in the results. The score column refers to the number of fairness metrics that returned more optimal results than the original unmitigated results. For example a score of three, means three of the five fairness metrics returned more desirable scores (closer to zero for all except disparate impact where one is the goal), when the bias mitigation algorithm was applied compared to without. Scores of zero were excluded as they would imply a 'lazy' algorithm that is simply returning a dataset with a high starting accuracy. Time denotes run-time in seconds.

	Pre	In-p	Post	Classifier	Perf-Rank	Dataset	Score	Time
0	di	mfc-fdr	roc	-	1.0	Adult	3	94.45
1	rw	-	eop	Random Forest	1.0	Ricci	2	0.29
2	-	-	eop	Random Forest	1.0	Ricci	2	0.18
3	di	-	-	Random Forest	1.0	German	1	0.58
4	rw	-	roc	Logistic Regression	1.0	Bank	3	46.87
5	-	-	roc	Logistic Regression	1.0	Bank	3	47.41
6	rw	-	-	Logistic Regression	1.0	Bank	3	0.8
7	rw	-	eop	Logistic Regression	1.0	Bank	2	0.84
8	-	-	eop	Logistic Regression	1.0	Bank	2	0.67
9	-	-	cpp	Logistic Regression	1.0	Bank	3	0.86
10	rw	-	cpp	Logistic Regression	1.0	Bank	3	0.97
11	di	-	roc	Logistic Regression	1.0	Compas	3	42.1
12	di	-	eop	Logistic Regression	1.0	Compas	3	13.03
13	di	-	cpp	Logistic Regression	1.0	Compas	3	13.4
14	-	-	roc	Random Forest	2.0	German	3	6.68

Table 8.1: Table containing the top 15 combinations for Performance Rank

All but one of the top fifteen performers utilise a scikit-learn classifier as opposed to a fairness aware in-processing model. The singular in-processing algorithm classifier is meta fair classifier which out-performed prejudice remover in the previous section also. Logistic Regression dominates, with more than double the number of occurrences than Random Forest. There are no instances of Naive Bayes. This correlates with the poor AUC score returned from the in-processing scores in the last section as that metric is intrinsically linked to the other performance metrics. Thirteen of the fifteen combinations contain a post-processing algorithm, with ten pre-processing algorithms present. There is an even distribution of both *Disparate Impact Remover* and *Reweighing* in pre-processing and *Equalized Odds Post-Processing* and *Reject Option Classification* in post-processing.

The only row that contains a pre-processing algorithm, but no post-processing, returns the lowest score of one, reiterating the pre-processing algorithms' poor performance overall. The longest run-time was the one combination containing an in-processing algorithm, highlighting this as a potential pitfall for these algorithms.

8.7.2 Fairness Rank

Fairness Rank was calculated by precisely the same method as Performance Rank, except with the five fairness metrics that were used in the last section, as opposed to the performance metrics. The rank was averaged across different datasets, just as it was before and scores of zero were omitted.

	Pre	In-p	Post	Classifier	Fair-Rank	Dataset	Score	Time
0	-	pr	roc	-	1.0	Adult	3	186.95
1	di	mfc-sr	cpp	-	1.0	Compas	3	366.9
2	-	-	cpp	Random Forest	1.0	German	3	0.34
3	di	mfc-sr	roc	-	1.0	German	3	9.09
4	di	mfc-fdr	roc	-	1.0	German	3	8.98
5	di	-	cpp	Random Forest	1.0	German	3	0.57
6	rw	-	roc	Random Forest	1.0	German	3	6.17
7	rw	-	roc	Logistic Regression	1.0	Bank	3	46.87
8	-	-	roc	Logistic Regression	1.0	Bank	3	47.41
9	-	-	roc	Random Forest	1.0	German	3	6.76
10	rw	pr	roc	-	1.0	Bank	3	357.89
11	-	-	cpp	Logistic Regression	1.0	Bank	3	0.86
12	rw	-	cpp	Logistic Regression	1.0	Bank	3	0.97
13	di	-	roc	Naive Bayes	1.0	Bank	3	50.59
14	-	-	roc	Naive Bayes	1.0	Bank	3	48.71

Table 8.2: Table containing the top 15 combinations for Fairness Rank

In-processing algorithms are more frequent than in the performance test, but still the table is dominated by unmitigated classifiers, at a 2:1 ratio. Logistic Regression is now tied with Random Forest and there are two instances of Naive Bayes. *Meta Fair Classifier* is the most frequently appearing in-processing algorithm once again. *Reject Option Classification* is the most frequently occurring post-processing algorithm. *Equalized Odds* surprisingly, does not feature, as it outperformed *Calibrated Equalized Odds* post-processing in the previous tests. The three longest run-times were once again rows involving in-processing algorithms.

8.7.3 Overall Rank

Overall Rank is calculated by combining fairness metrics and performance metrics ranks. It was produced identically to the previous tables

	Pre	In-p	Post	Classifier	Ovr-Rank	Dataset	Score	Time
0	-	-	cpp	Random Forest	1.0	Adult	3	2.91
1	rw	-	roc	Random Forest	1.0	Bank	3	49.01
2	rw	mfc-sr	roc	-	1.0	Adult	3	99.64
3	-	mfc-sr	roc	-	2.0	German	3	9.21
4	rw	-	cpp	Random Forest	2.0	German	3	0.3
5	rw	-	-	Random Forest	2.0	Adult	3	9.82
6	-	mfc-fdr	roc	-	3.0	Compas	3	421.39
7	di	-	roc	Random Forest	3.0	Compas	3	38.82
8	-	-	roc	Random Forest	4.0	Adult	3	32.15
9	di	-	-	Random Forest	4.0	German	1	0.58
10	rw	mfc-fdr	roc	-	5.0	Bank	3	111.05
11	di	-	cpp	Random Forest	5.0	Adult	3	6.39
12	rw	mfc-sr	-	-	5.0	Bank	4	61.92
13	rw	mfc-fdr	-	-	5.0	Adult	2	440.41
14	-	mfc-sr	-	-	5.0	Adult	2	499.93

Table 8.3: Table containing the top 15 combinations for Overall Rank

Post-processing and pre-processing algorithms are present in ten of the top fifteen results for overall rank, while in-processing makes seven appearances. *Meta Fair Classifier* is the solitary in-processing algorithm, solidifying its victory over prejudice remover. *Reweighing* beats out *Disparate impact Remover* in the pre-processing category, while post-processing is exclusively *Reject Option Classification* and *Calibrated Equalized Odds Post-Processing* once again, just as it was for fairness rank. Random Forest is the sole classifier present.

One pre-processing algorithm appears individually, with the lowest score here at 1, informing us of its relative impracticality, when used individually, and the difference when paired with other algorithms is present but not substantial. The five longest run times are combinations containing in-processing algorithms, further identifying this as a concern.

8.7.4 Score

	Pre	In-p	Post	Classifier	Ovr-Rank	Dataset	Score	Time
0	di	-	-	Naive Bayes	18.0	Bank	5	6.94
1	-	mfc-fdr	-	-	5.0	Compas	5	545.28
2	-	mfc-sr	-	-	5.0	Compas	5	840.78
3	rw	mfc-fdr	-	-	5.0	Compas	5	852.32
4	rw	mfc-sr	-	-	5.0	Compas	5	532.47
5	di	mfc-fdr	cpp	-	33.0	Adult	5	46.71
6	rw	mfc-sr	cpp	-	25.0	Bank	5	52.31
7	rw	mfc-fdr	cpp	-	28.0	Bank	5	52.31
8	-	mfc-fdr	cpp	-	24.0	Bank	5	53.84
9	-	mfc-sr	cpp	-	31.0	Bank	5	50.52
10	rw	pr	eop	-	50.0	Ricci	4	4.52
11	-	pr	eop	-	51.0	Ricci	4	5.53
12	di	pr	cpp	-	56.0	Compas	4	97.58
13	rw	-	-	Naive Bayes	24.0	Bank	4	0.25

Table 8.4: Table containing the top 15 combinations for Score

Every combination that achieved a score of either four or five is presented in fig 2. Overall, twelve out of the fourteen rows contain an in-processing classifier. The remaining two rows consist of a pre-processing algorithm only, along with Naive Bayes contrasting with the results from the previous table. *Disparate Impact Remover* has a comparatively low rank to its high score, implying its mitigated scores outperformed the original scores, but not by a large margin. *Meta Fair Classifier* outperformed prejudice remover for the most part, just as it has in the previous section. In-processing algorithms once again exhibit the largest run-time.

8.8 Evaluation Conclusions

Between the three varieties of bias mitigation algorithms tested, the best results were universally achieved by the post-processors. Both pre-processing algorithms proved to be ineffective when used on their own with no significant fairness gains. In-processing algorithms fared better but only marginally, coupled with colossal AUC losses. These were not an issue for the post-processing algorithms, whose mitigated results outperformed the other other twos sets of algorithms substantially. *Equalized Odds Post-Processing* was the strongest performer on two fairness metrics, while the other two post-processing algorithms were the joint-strongest on the remaining three. They performed equally well on the performance rankings and score categories also.

The post-processing algorithms were also very effective when paired with in-processing and to some extent, either pre-processing algorithm also. *Meta-Fair Classifier* outperformed *Prejudice Remover* here. **Reject Option Classification paired with Meta-Fair Classifier and a pre-processing algorithm** is the strongest choice if *Theil Index*, *Average Odds Difference* or *Equality of Opportunity Difference* are the metrics that should be satisfied, otherwise **Equalized Odds Difference paired with Meta-Fair Classifier and a pre-processing algorithm** should be used. If the loss in performance is a concern, use without the in-processing algorithm. Calibrated Equalized Odds should be considered here as it performed impressively on its own but poorly in combination with the in-processing algorithms. Logistic Regression or Random Forest should be chosen over Naive

Bayes, although the latter was the only classifier to appear in the score category.

Chapter 9: **Summary and Conclusions**

Machine learning algorithms are making important decisions that affect the lives of an increasing number of people, and they are only becoming more popular. These algorithms are being implemented in a vast number of sectors in modern society and the number continues to grow. Fairness in machine learning as a result, is becoming a popular topic of discussion in the industry. Googles head of AI, John Giannandrea[1] explained that his main concern with AI is not super-intelligent killer robots that are displayed in movies and television but "The real safety question, if you want to call it that, is that if we give these systems biased data, they will be biased," The Obama administration released a report in 2014[2] entitled "Big Data: Seizing opportunities, preserving values" highlighting the potential for discrimination in big data. Bias in machine learning is a huge challenge and only increasing. Machine learning models at their core are a form of statistical discrimination, so treating historically disadvantaged groups differently is a particularly damming error, bringing into question their whole purpose in the first place.

IBM's Kush Varshney gave an interesting analogy at an AIF360 tutorial presentation[3]. He compared AI today to the processed foods industry a few decades ago. Both industries where mistrusted by the public to a certain extent. Heinz did a great deal to earn the general public's trust, including factory tours to show off their sanitary conditions, new legislation to ensure ingredients had to be fresh and labelled on the package and clear bottles to show what was inside. Varshney believes that in order for artificial intelligence and machine learning to become accepted and trusted by society, the industry will need to be more transparent with it's algorithms and models, and detailing exactly how they work. There is a distinct lack of transparency in current solutions to the problem.

There is no one metric or algorithm that will satisfy every definition of fairness. In fact, even more worryingly, certain metrics seem to be directly opposed to one another. What metric or mitigation algorithm should be used in what context? This is a huge question, and one that goes beyond the scope of this project. However there is little to no information out there comparing the current solutions. Obtaining a better understanding of how these algorithms work, and indeed, interact with one another is key. **50** combinations of bias mitigation algorithms and classifiers were applied to **5** popular datasets to evaluate them based on **5** fairness metrics and **4** performance metrics. The most efficient combination was found to be a **pre-processing algorithm**, used alone or potentially in combination with other algorithms. Fairness is deeply dependent on context and there is not enough context in the field at the moment. Comparing algorithms results with the others will provide a deeper understanding of how these algorithms work and which situations they should or shouldn't be used. Fairness is difficult to define, however, gaining a better insight into the current solutions to algorithmic bias is essential in understanding what aspect of fairness the algorithms are trying to enforce. Being open and conveying this to the lay user is paramount in gaining widespread acceptance for artificial intelligence and machine learning

[1] MIT Technology Review: `https://www.technologyreview.com/s/608986/forget-killer-robotsbias-is-the-real-ai-danger/`- last accessed on 17/11/2019

[2] White House Report on Big Data: `https://obamawhitehouse.archives.gov/sites/default/files/microsites/ostp/2016_0504_data_discrimination.pdf`- last accessed on 17/11/2019

[3] IBM Heinze Metaphor: `https://www.youtube.com/watch?v=HxUqZSB-LFU&list=PLzcaObIwh3-TWcwgHMha7AsmJtn-4J2dx&index=4&t=0s`- roughly 4:55 in the video, last accessed on 17/11/2019

Chapter 10: **Limitations**

Second Protected Attribute: Some datasets contain two protected attributes, for example sex and age. When bias is rectified with regard to one, could it potentially be increased in another? Further research must be carried out.

Multi-class Classification: This project only dealt with binary classification problems, i.e. the label could only be one of two choices. There are plenty of real-world examples were this isn't the case and this again is the grounds for further work.

AIF360 scikit-learn: AIF360 was updated with scikit-learn functionality during this project, so this was not considered.

Lay-user friendly results: Fairness in machine learning is starting to affect everyone, not just computer scientists. Creating an API or website where these results could be easily interpreted by a lay user would go a long way into quashing any doubts about these algorithms validity, and increase public perception of the field at large.

Fairness Limits: The ability to conceptualise, mitigate and even define fairness is limited by our understanding of fairness in the legal system, politics and society in large. Fairness in some ways is not a set definition but an ever-changing concept that is difficult to define in general, never-mind mathematically in mathematical terms.

Chapter 11: **Appendix**

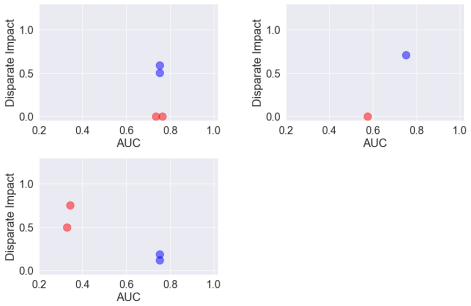
11.1 Pre-Processing & In-Processing

11.1.1 Disparate Impact Remover and Prejudice Remover

Disparate Impact

Adult, *Bank* and *German* have significant increases in fairness, with *Bank* and *German* at the expense of AUC. *Compas* and *Ricci* also experience a significant loss in AUC, with *Compas* fairness scores worse than the unmitigated version also, while *Ricci* stays the same.

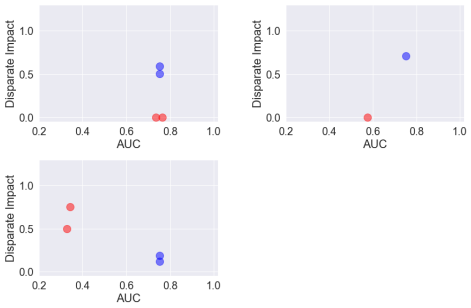
Figure 11.1: Adult, Bank and Compas Datasets



Mean Difference

Similar. *Ricci* and *Compas* has better fairness this time.

Figure 11.2: Adult, Bank and Compas Datasets



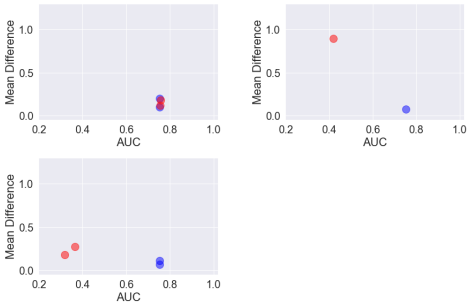
Theil Index

Poor , Losses in AUC and fairness in all except *Adult*, as seen in 11.3.

Figure 11.3: Adult, Bank and Compas Datasets

Average Odds Difference

Results Identical to mean difference



Equality Opportunity Difference

Results identical to mean difference .

11.1.2 Disparate Impact Remover and Meta Fair Classifier

Results here are identical to before, apart from *Bank* fairness score, and *Ricci* AUC, being much stronger.

11.1.3 Reweighing and Prejudice Remover

Results here are identical to *Disparate Impact Remover and Prejudice Remover*, apart from *Bank* fairness score being improved. *Ricci* has an improved fairness on *Mean Difference*, *Average Odds Difference* and *Equal Opportunity Difference*, and a worse fairness on the remaining two metrics.

11.1.4 Reweighing and Meta Fair Classifier

Results here are identical to *Disparate Impact Remover with Meta Fair Classifier*, apart from increases in *Compas* fairness score across the board.

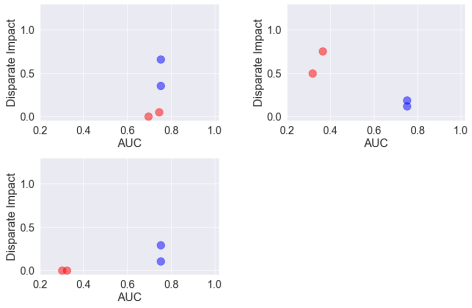
11.2 Pre-processing & In-processing & Post-Processing

11.2.1 Disparate Impact Remover &Prejudice Remover & Equalized Odds Post-Processing

Disparate Impact

There are increases in fairness in *Adult*, *Bank* and *German*, while there are decreases in *Compas* and *Ricci*. AUC is similar in *Adult*, and poorer in all other datasets tested.

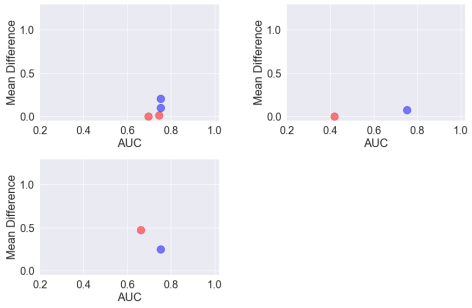
Figure 11.4: Adult, Compas and German Datasets



Mean Difference

There are increases in*Mean Difference* in all datasets apart from *Ricci*. .

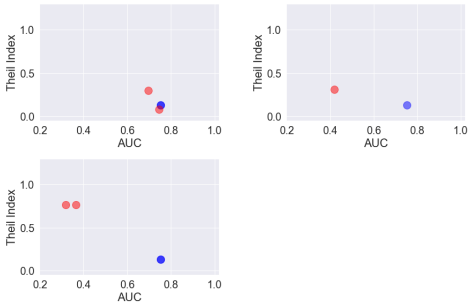
Figure 11.5: Adult, Bank and Ricci Datasets



Theil Index

Mitigated results here are universally worse than the original results..

Figure 11.6: Adult, Bank and Compas Datasets



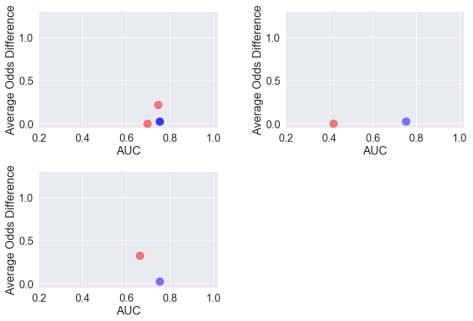
Average Odds Difference

There are similar/fairness gains in all datasets bar *Ricci*.

Equality Opportunity Difference

Identical to *Average Odds Difference*.

Figure 11.7: Bank,Compas and Ricci Datasets



11.2.2 Disparate Impact Remover Meta Fair Classifier & Equalized Odds Post-Processing

Identical Results to *Disparate Impact Remover Prejudice Remover & Equalized Odds Post-Processing* apart from loss in *Ricci* AUC and *Ricci* fairness here across all metrics.

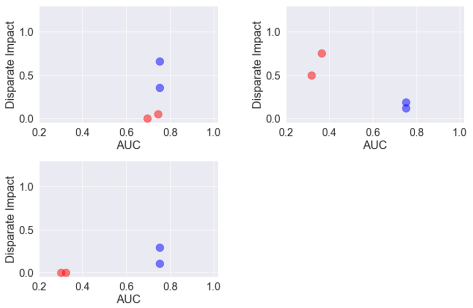
Overall there is no tangible benefit to adding disparate impact when the combination without disparate impact outperformed it across the *Ricci* dataset and was identical otherwise.

11.2.3 Disparate Impact Remover &Prejudice Remover & Calibrated Equalized Odds Post-Processing

Disparate Impact

There are gains made in Disparate Impact in *Adult*, *Bank* and *Compas*. Once again, losses in AUC in all, apart from *Adult*.

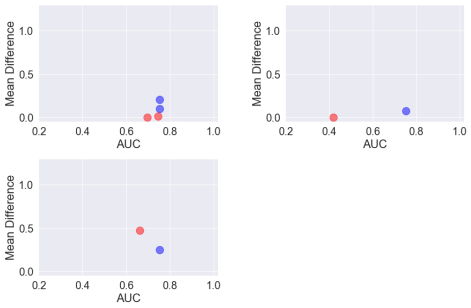
Figure 11.8: Adult, Compas and German Datasets



Mean Difference

The combination performs poorly here with losses in fairness across the board, apart from slight gains in *Adult* .

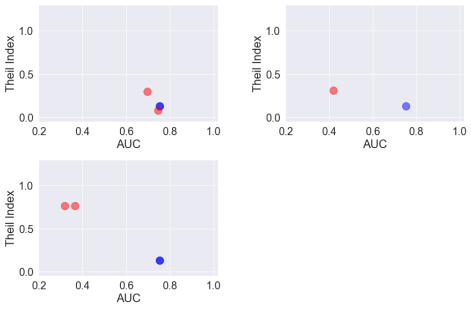
Figure 11.9: Adult, Bank and Ricci Datasets



Theil Index

Theil Index returns more desirable values in the *Compas*, *German* and *Ricci* datasets.

Figure 11.10: Adult, Bank and Compas Datasets



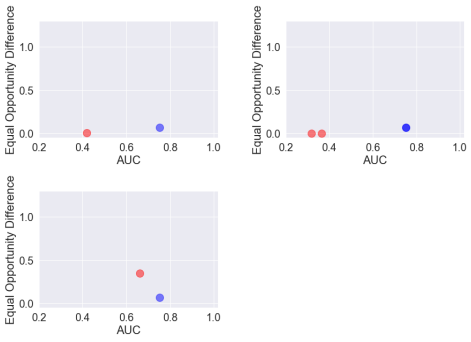
Average Odds Difference

Identical to mean difference

Equality Opportunity Difference

Improved fairness scores in *Compas*, *German* and *Ricci* datasets, similar to Theil Index.

Figure 11.11: Bank,Compas and Ricci Datasets

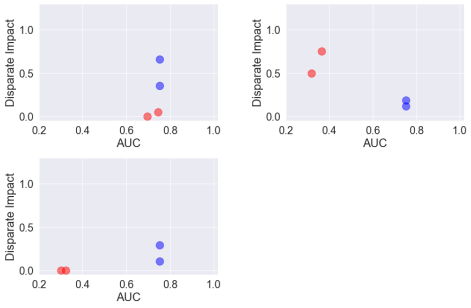


11.2.4 Disparate Impact Remover & Meta Fair Classifier & Calibrated Equalized Odds

Disparate Impact

Increases in fairness in one, losses in AUC in all apart from *Adult*.

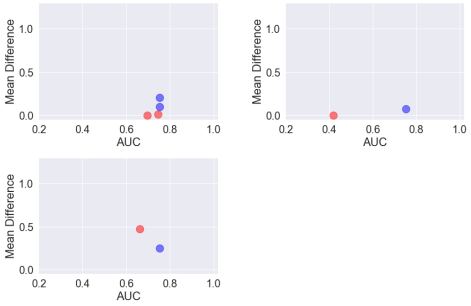
Figure 11.12: Adult, Compas and German Datasets



Mean Difference

Losses/equal in fairness in all except *Adult* , losses in AUC in all except *Adult*.

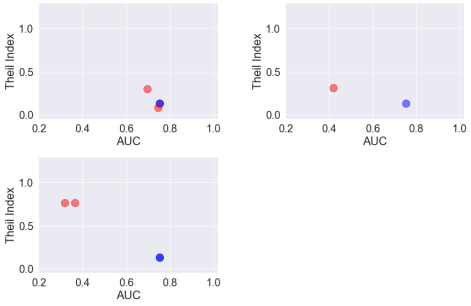
Figure 11.13: Adult, Bank and Ricci Datasets



Theil Index

Worse AUC in all except *Adult Ricci* , except *Adult*, better fairness in all

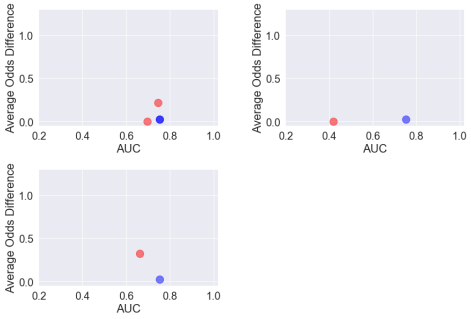
Figure 11.14: Adult, Bank and Compas Datasets



Average Odds Difference

Fairness gains in all except *Adult* and *Ricci*,

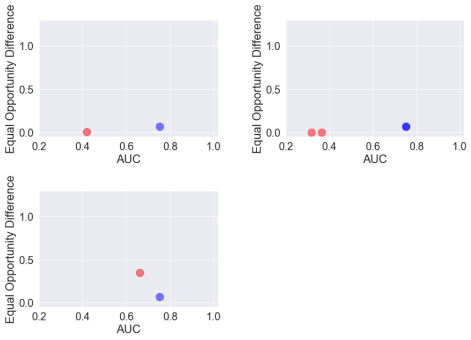
Figure 11.15: Bank,Compas and Ricci Datasets



Equality Opportunity Difference

fairness gains in all excpet *Adult*. (better fairness in last three, worse AUC in all except adult)

Figure 11.16: Bank,Compas and Ricci Datasets

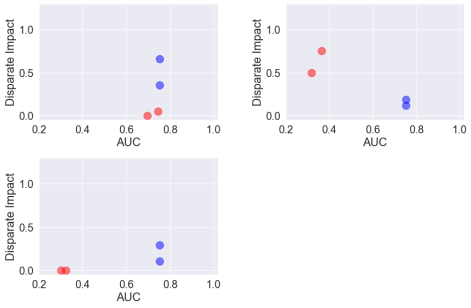


11.2.5 Disparate Impact Remover &Prejudice Remover & Reject Option Classification

Disparate Impact

Better fairness in *Bank*, equal in rest.

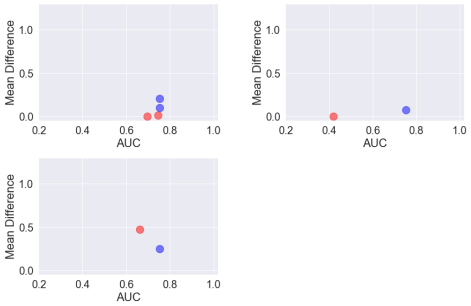
Figure 11.17: Adult, Compas and German Datasets



Mean Difference

Equal or worse fairness.

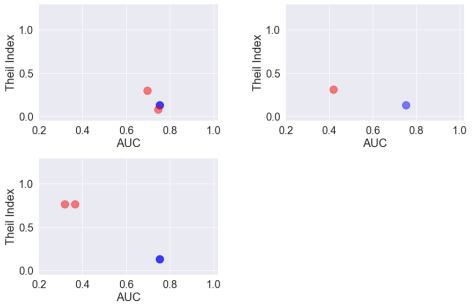
Figure 11.18: Adult, Bank and Ricci Datasets



Theil Index

Better in all except *Bank*.

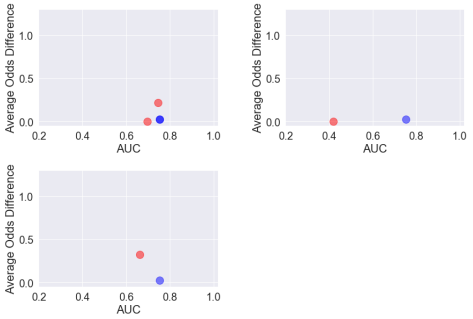
Figure 11.19: Adult, Bank and Compas Datasets



Average Odds Difference

Identical to *Theil*.

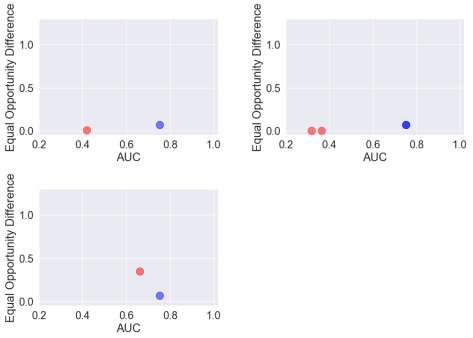
Figure 11.20: Bank,Compas and Ricci Datasets



Equality Opportunity Difference

Identical to *Theil*.

Figure 11.21: Bank,Compas and Ricci Datasets



11.2.6 Disparate Impact Remover &Meta Fair Classifier & Reject Option Classification

Loss in fairness in *Bank* across all metrics apart from Disparate Impact, loss in Ricci AUC.

11.2.7 Reweighing

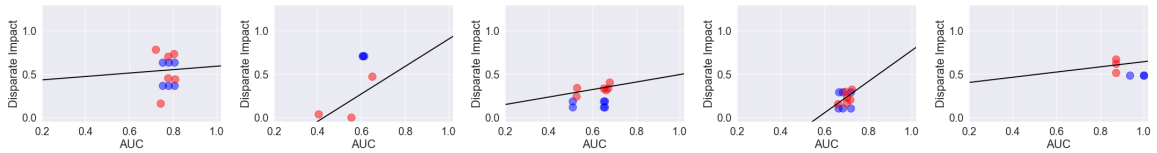
Reweighing performed identically to Identical to *Disparate Impact Remover* when paired with an in-processing and post-Processing algorithm across all tests.

11.3 Pre-processing

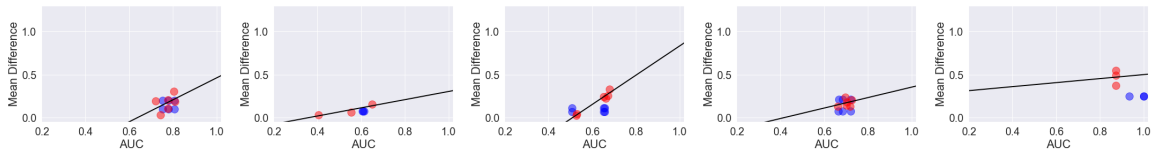
Dataset Order is alphabetically: *Adult*, Bank, Compas, *German*, *Ricci*

11.3.1 Disparate Impact Remove

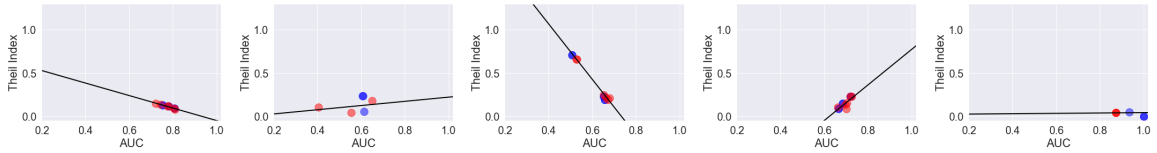
Disparate Impact



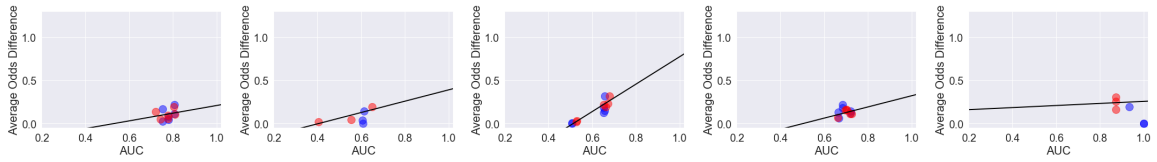
Mean Difference



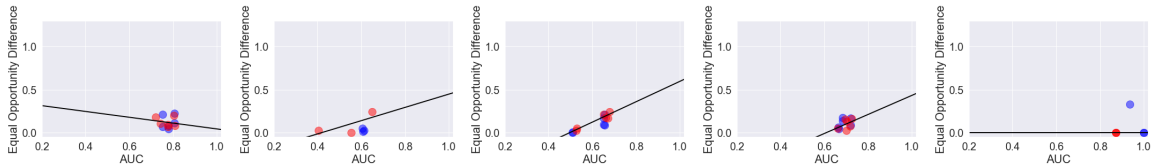
Theil Index



Average Odds Difference

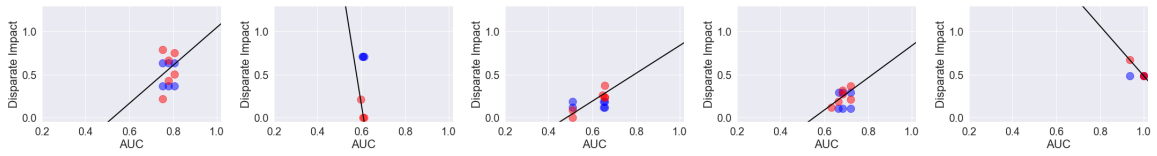


Equality Opportunity Difference

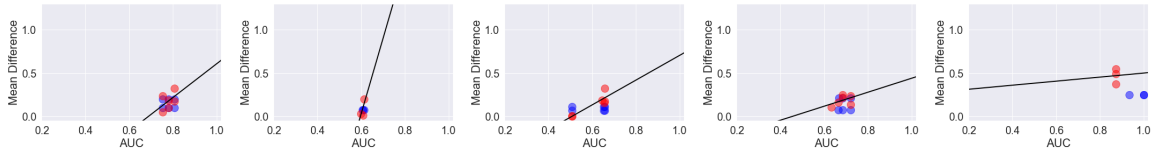


11.3.2 Reweighing

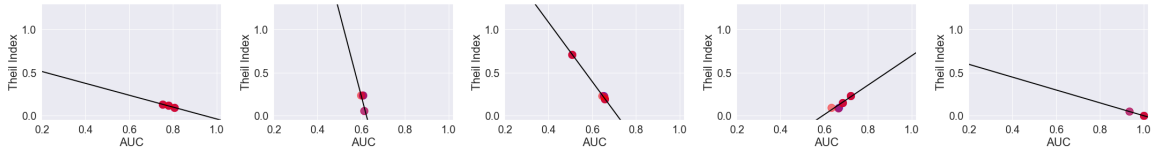
Disparate Impact



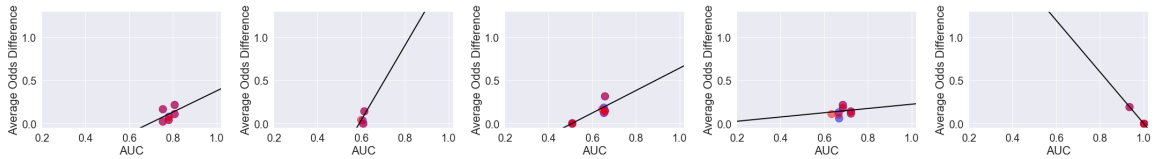
Mean Difference



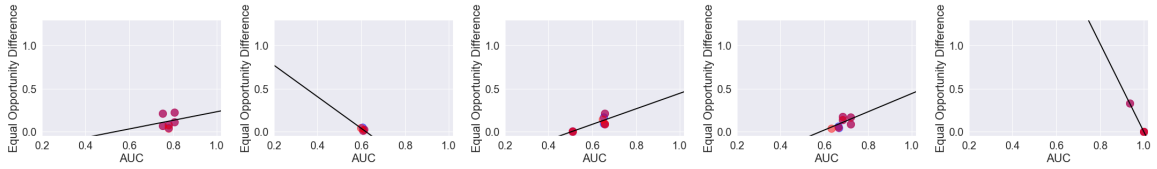
Theil Index



Average Odds Difference



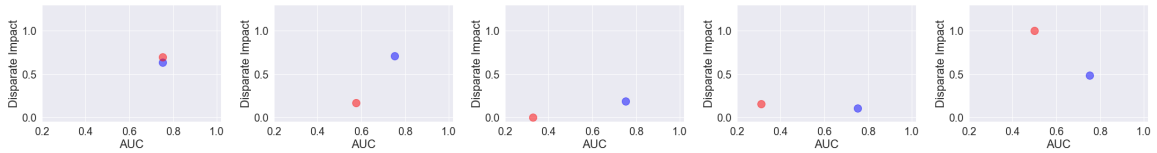
Equality Opportunity Difference



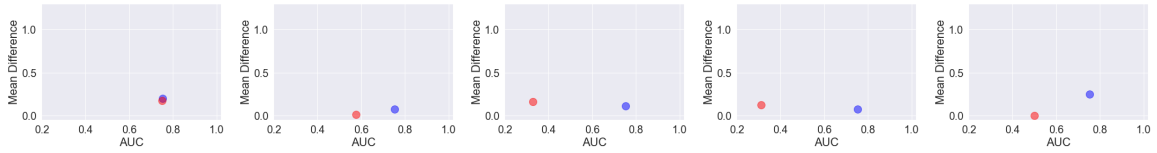
11.4 In-Processing

11.4.1 Prejudice Remover

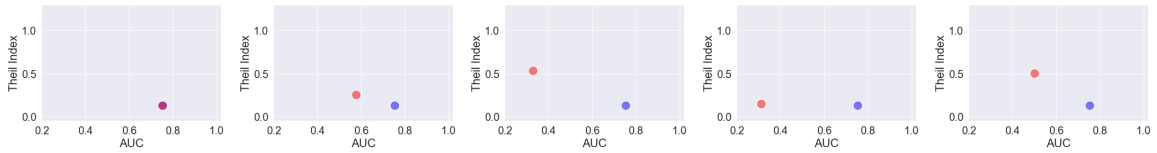
Disparate Impact



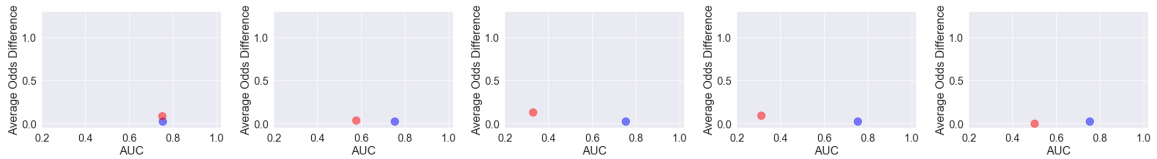
Mean Difference



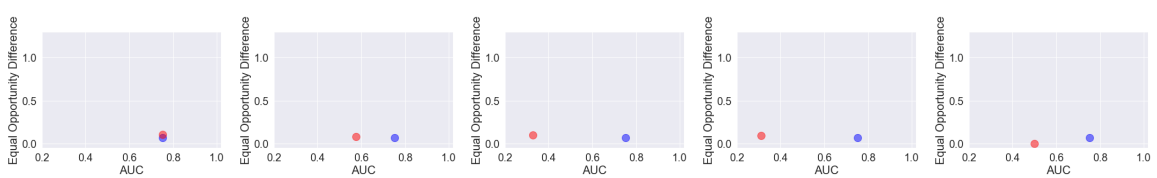
Theil Index



Average Odds Difference

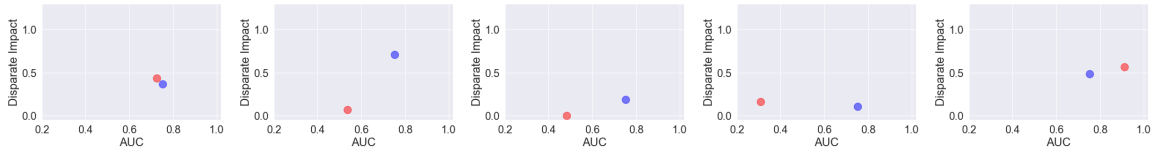


Equality Opportunity Difference

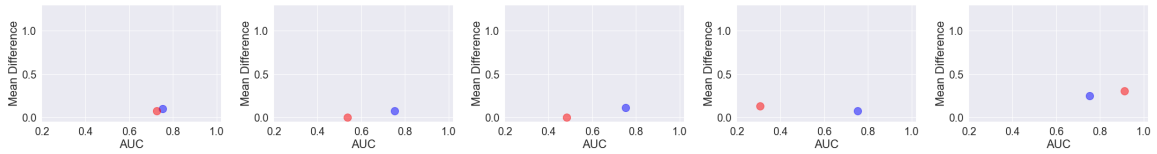


11.4.2 Meta Fair Classifier - False Discovery Ratio

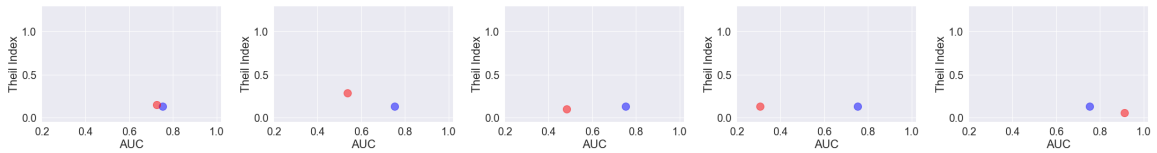
Disparate Impact



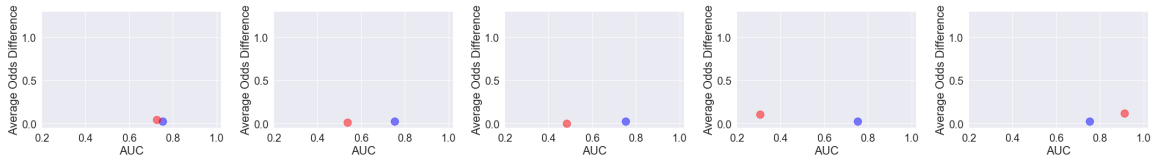
Mean Difference



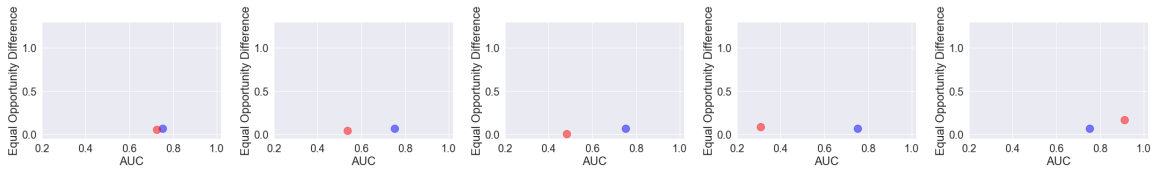
Theil Index



Average Odds Difference

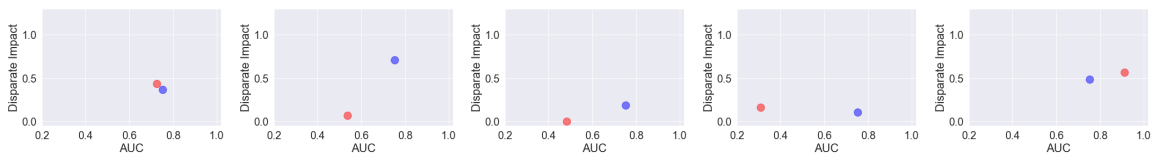


Equality Opportunity Difference

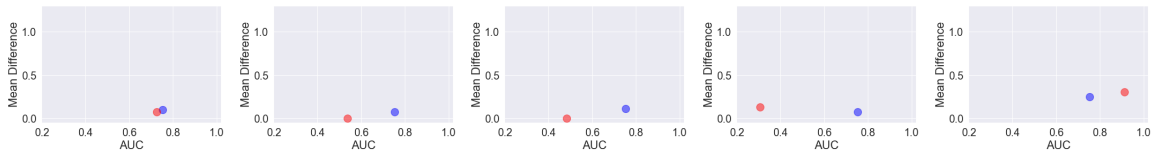


11.4.3 Meta Fair Classifier - Statistical Ratio

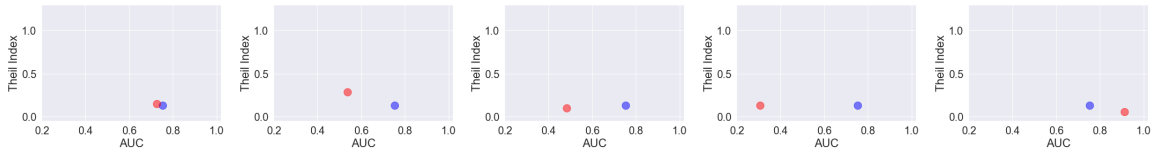
Disparate Impact



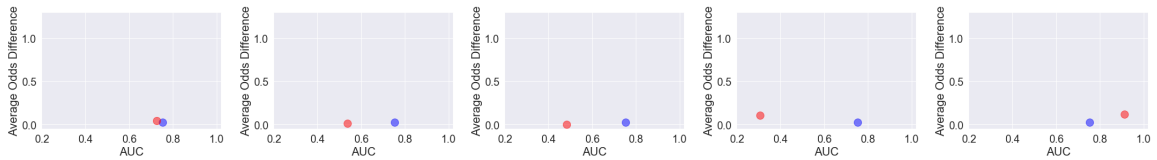
Mean Difference



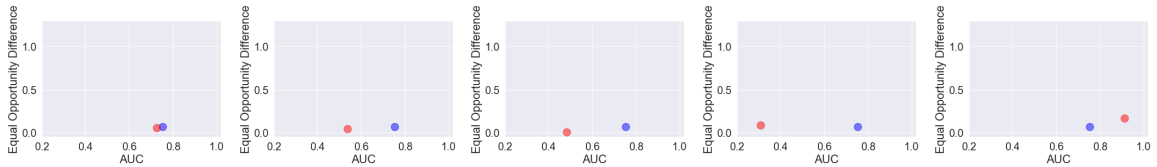
Theil Index



Average Odds Difference



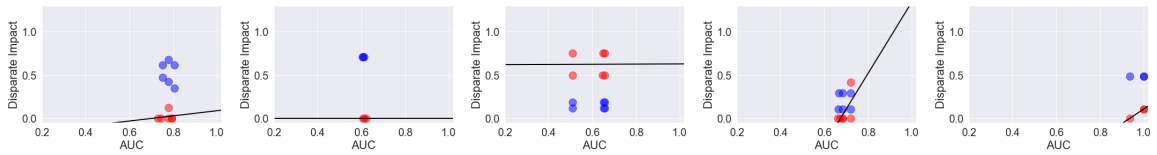
Equality Opportunity Difference



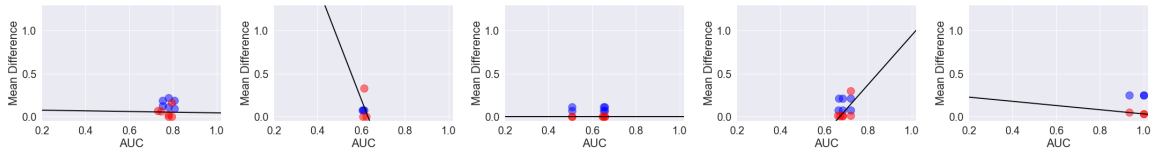
11.5 Post-Processing

11.5.1 Equalized Odds Post-Processing

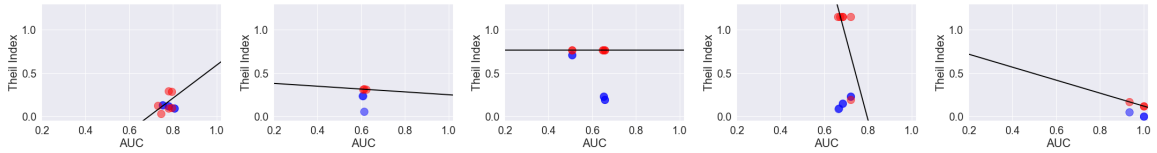
Disparate Impact



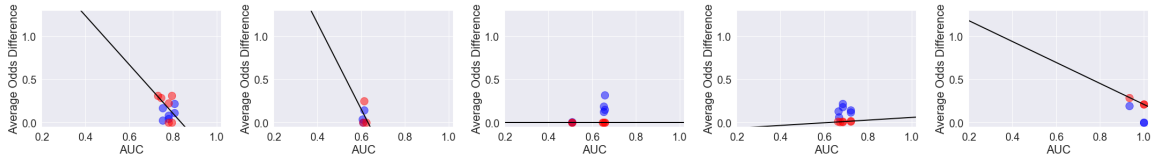
Mean Difference



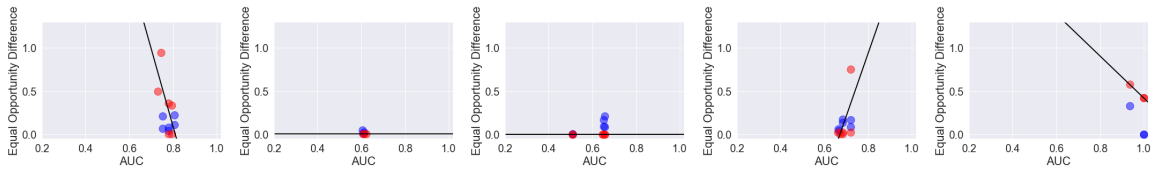
Theil Index



Average Odds Difference

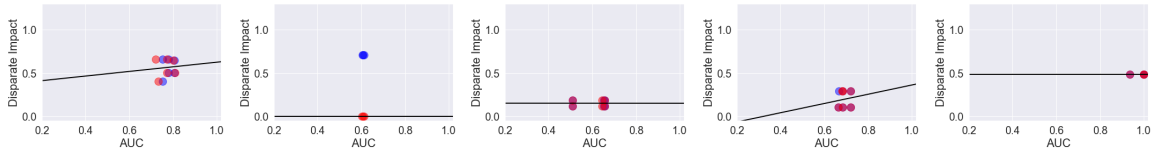


Equality Opportunity Difference

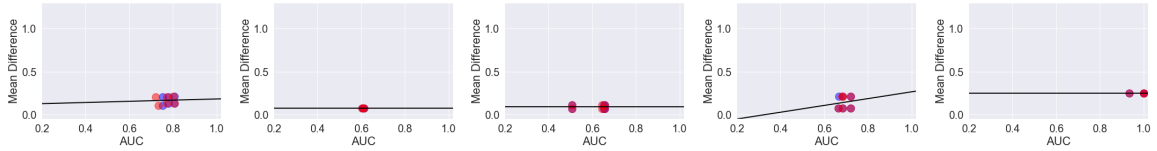


11.5.2 Calibrated Equal Odds Post-Processing

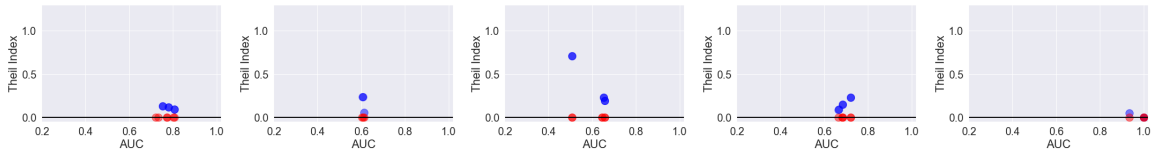
Disparate Impact



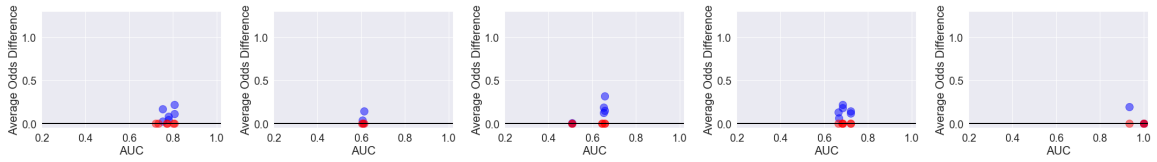
Mean Difference



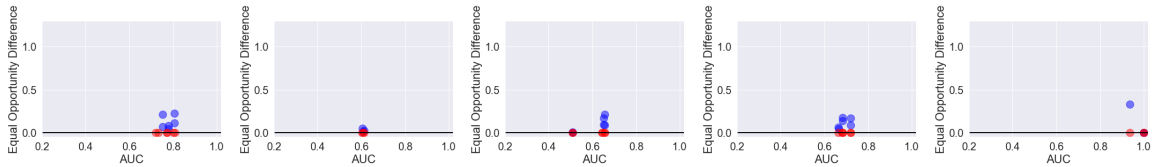
Theil Index



Average Odds Difference

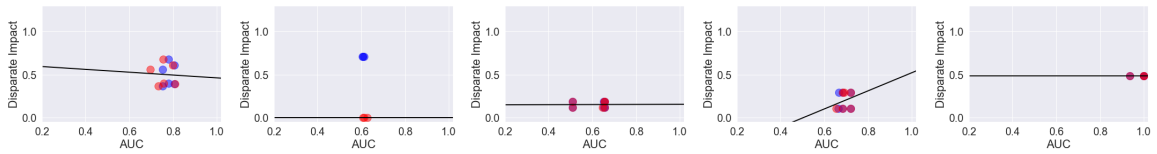


Equality Opportunity Difference

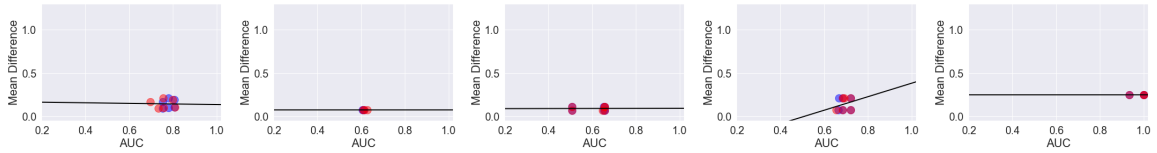


11.5.3 Reject Option Classification

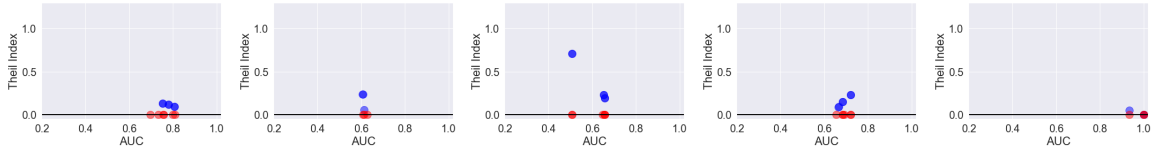
Disparate Impact



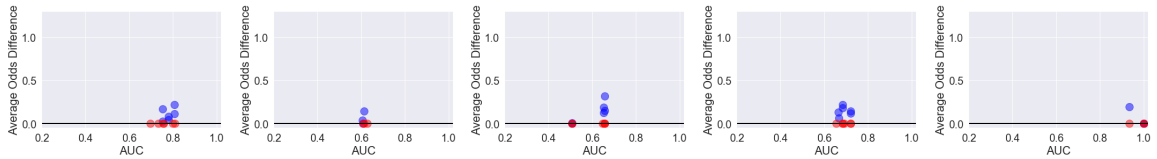
Mean Difference



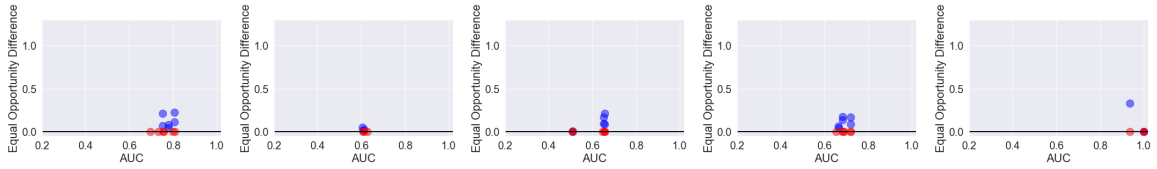
Theil Index



Average Odds Difference



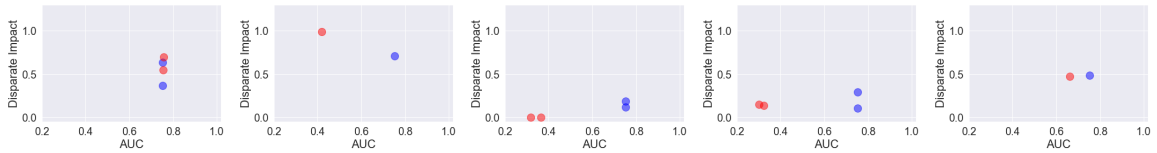
Equality Opportunity Difference



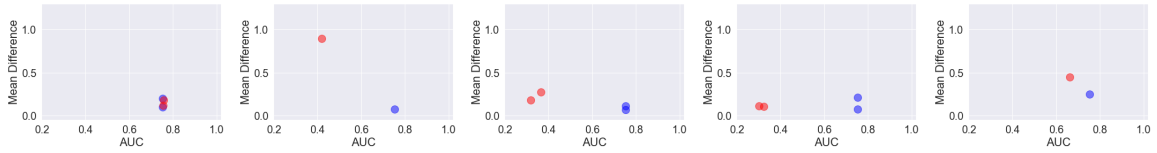
11.6 Pre-processing & In-processing

11.6.1 Disparate Impact Prejudice Remover

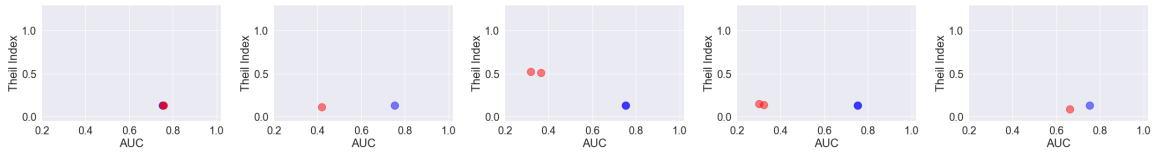
Disparate Impact



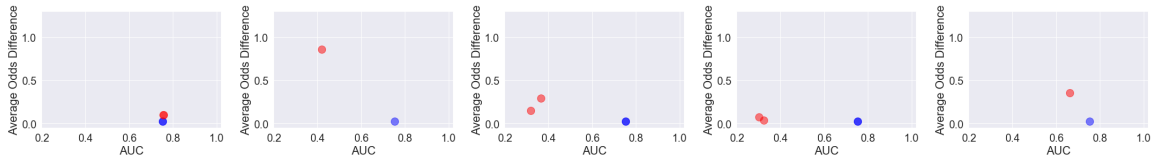
Mean Difference



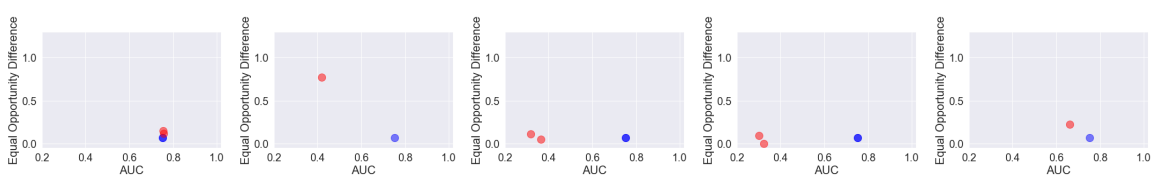
Theil Index



Average Odds Difference

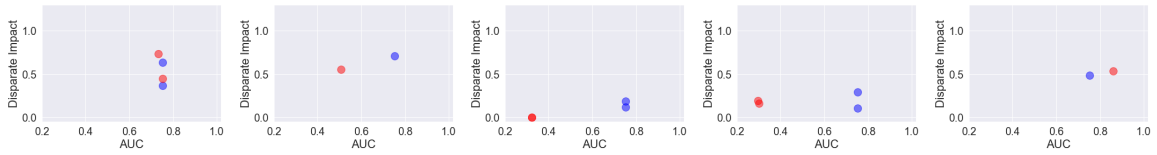


Equality Opportunity Difference

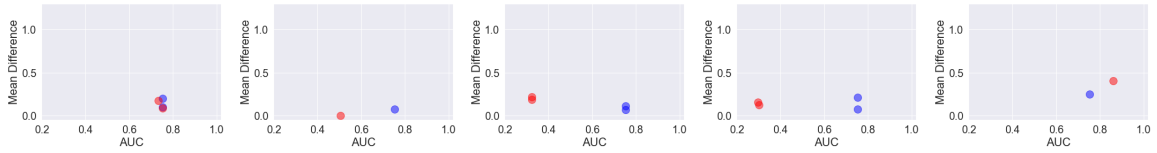


11.6.2 Disparate Impact & Meta Fair Classifier - False Discovery ratio

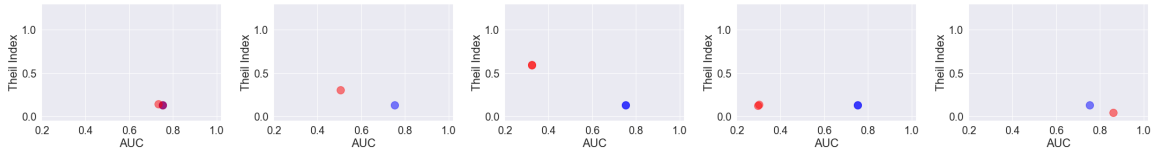
Disparate Impact



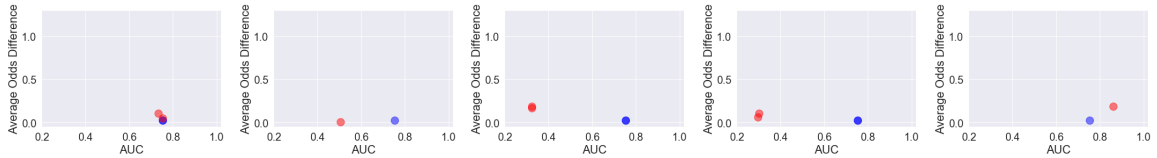
Mean Difference



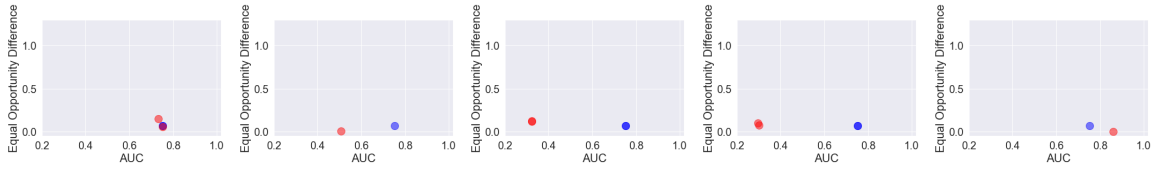
Theil Index



Average Odds Difference

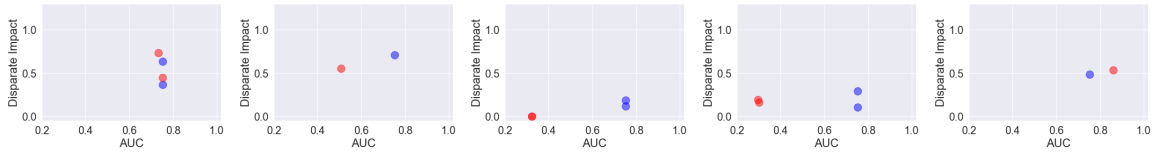


Equality Opportunity Difference

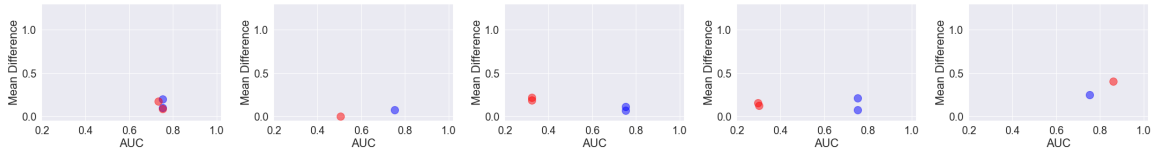


11.6.3 Disparate Impact & Meta Fair Classifier - Statistical Ratio

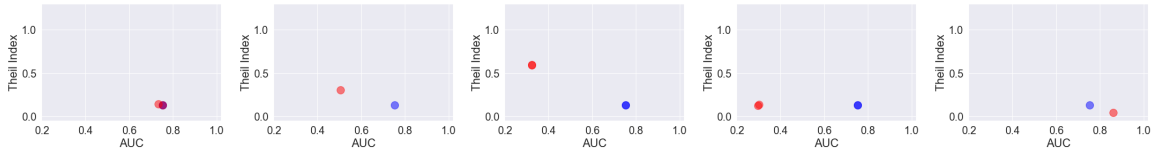
Disparate Impact



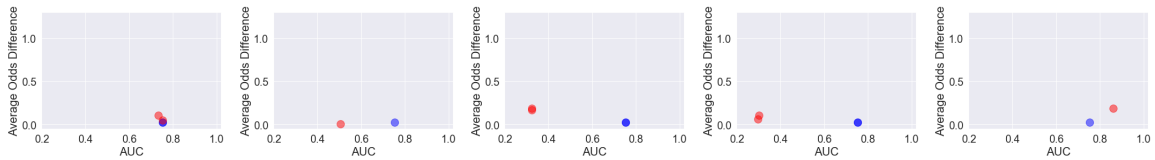
Mean Difference



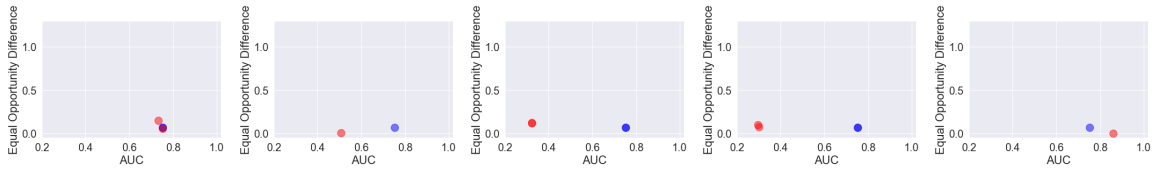
Theil Index



Average Odds Difference

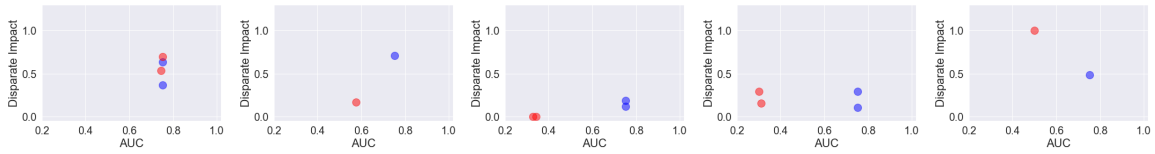


Equality Opportunity Difference

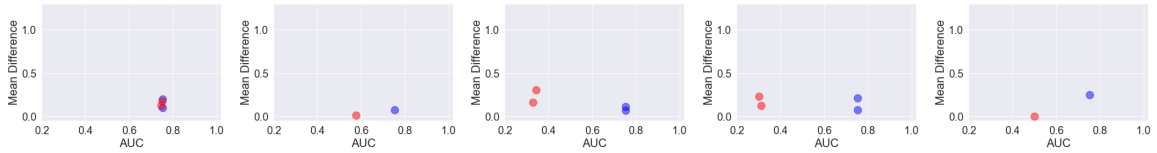


11.6.4 Reweighing & Prejudice Remover

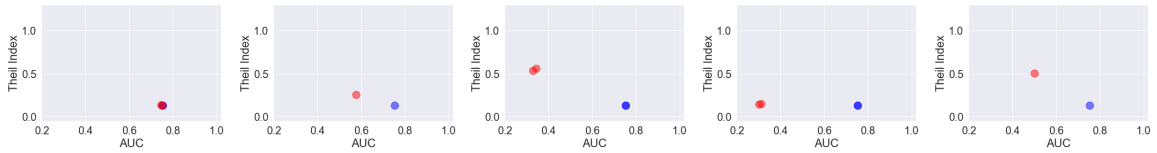
Disparate Impact



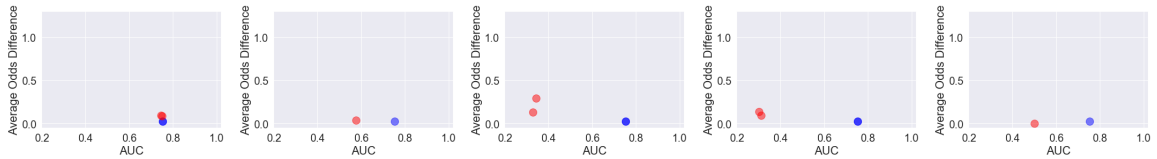
Mean Difference



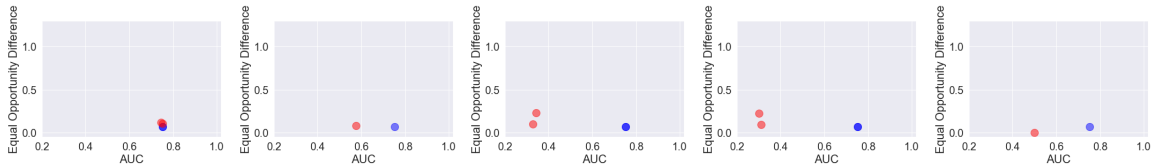
Theil Index



Average Odds Difference

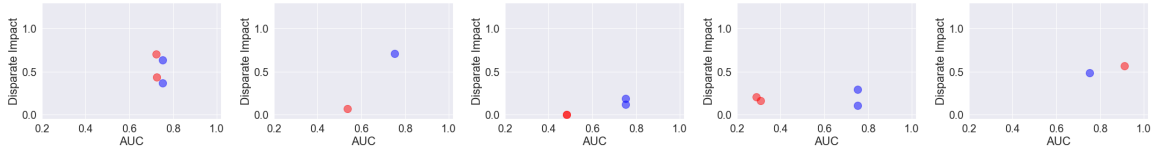


Equality Opportunity Difference

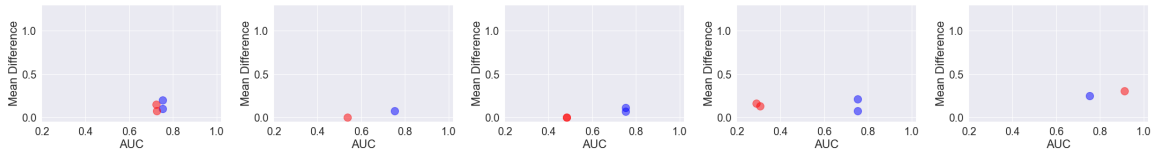


11.6.5 Reweighing & Meta Fair Classifier - False Discovery ratio

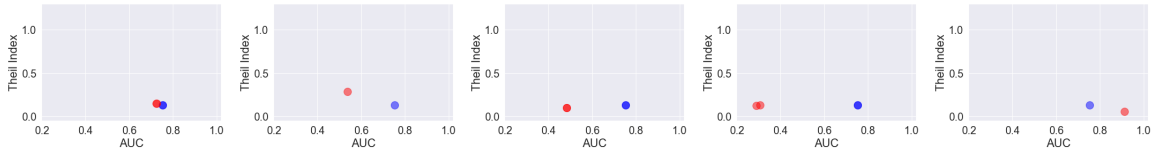
Disparate Impact



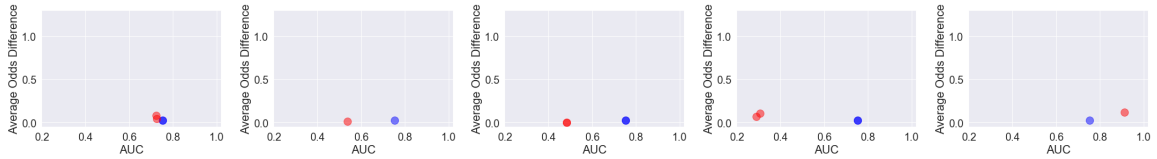
Mean Difference



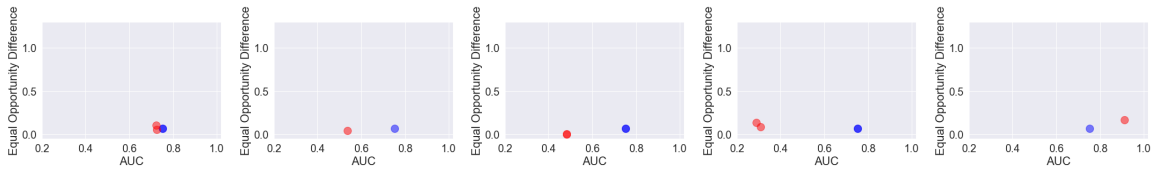
Theil Index



Average Odds Difference

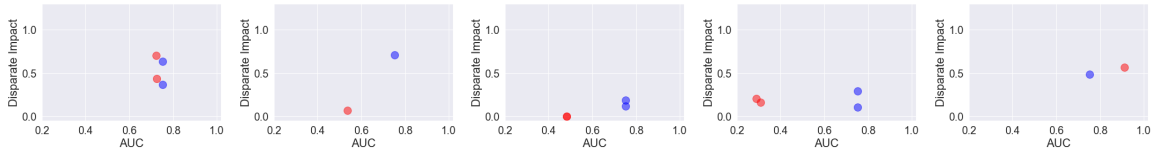


Equality Opportunity Difference

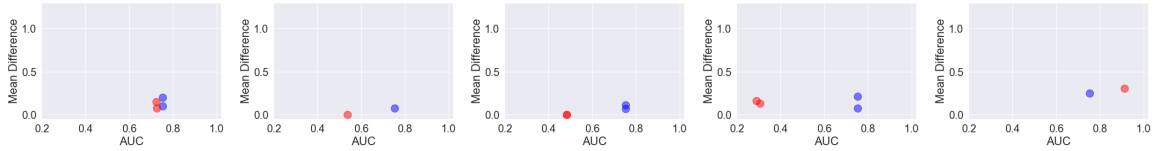


11.6.6 Reweighing & Meta Fair Classifier - Statistical Ratio

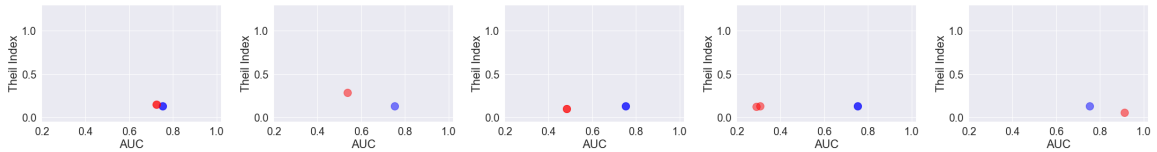
Disparate Impact



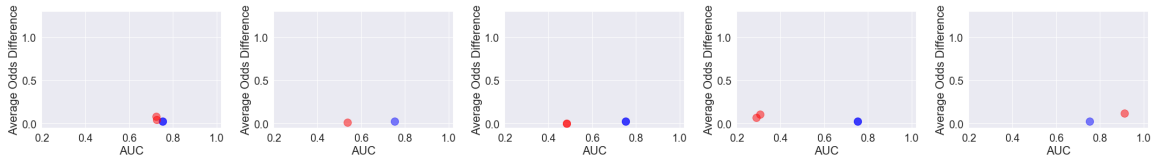
Mean Difference



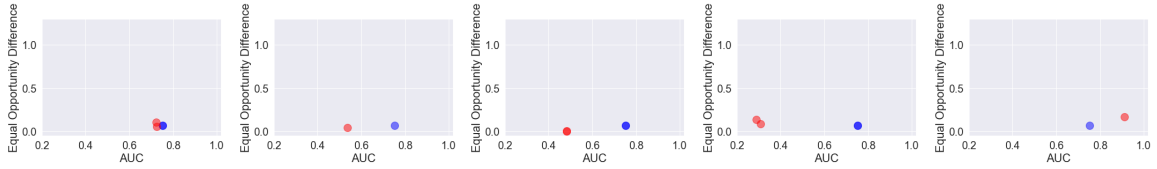
Theil Index



Average Odds Difference



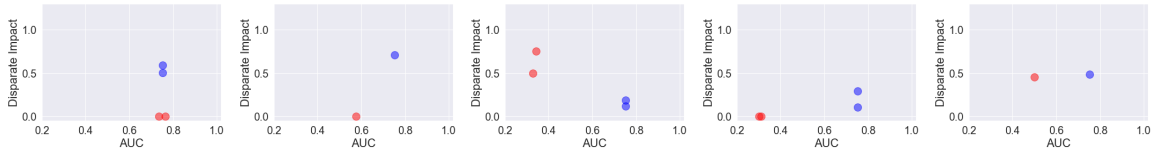
Equality Opportunity Difference



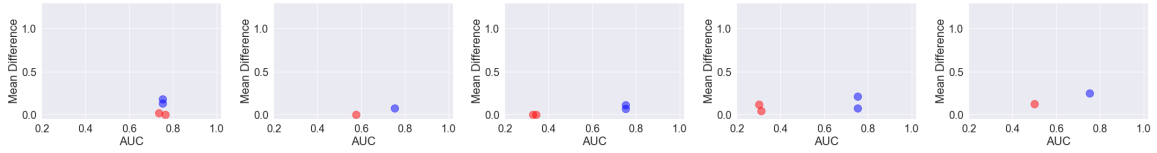
11.7 In-Processing & Post-processing

11.7.1 Prejudice Remover & Equalized Odds Post-Processing

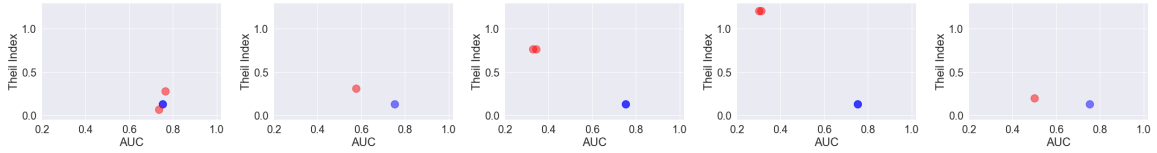
Disparate Impact



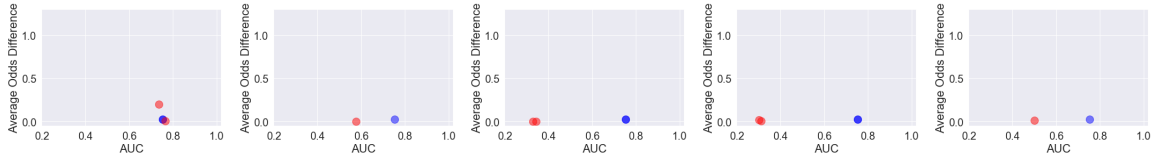
Mean Difference



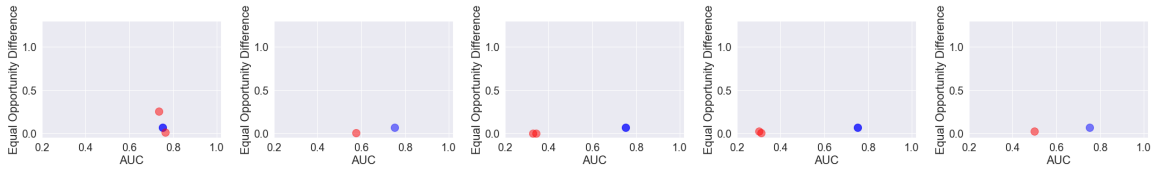
Theil Index



Average Odds Difference

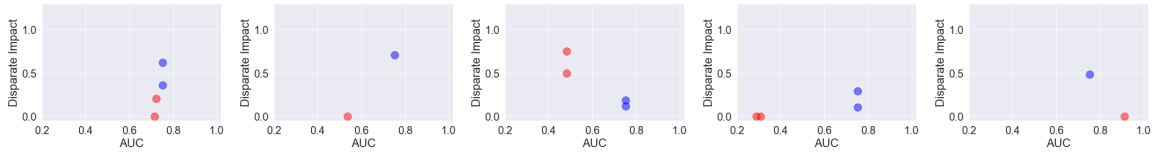


Equality Opportunity Difference

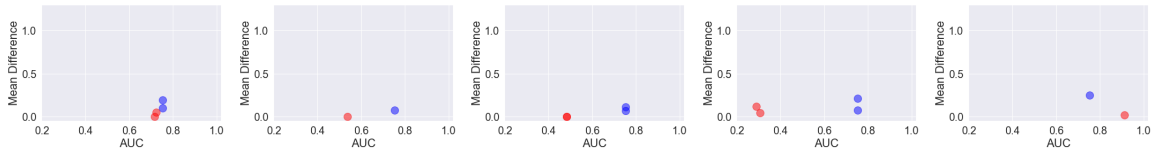


11.7.2 Meta Fair Classifier & Equalized Odds Post-Processing

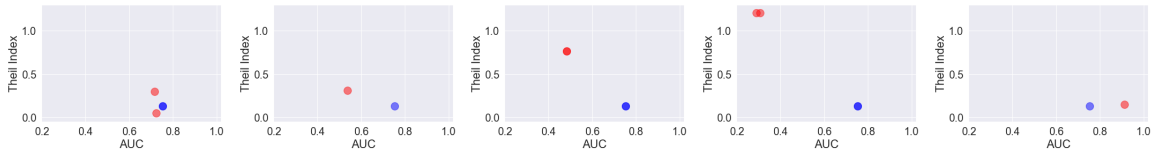
Disparate Impact



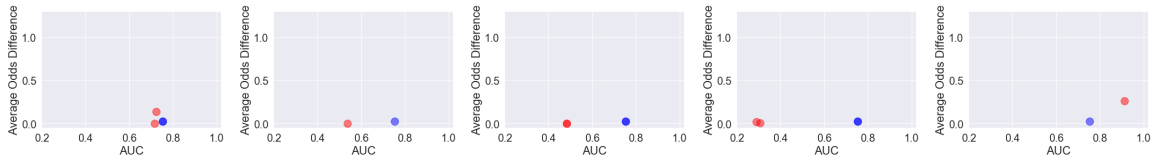
Mean Difference



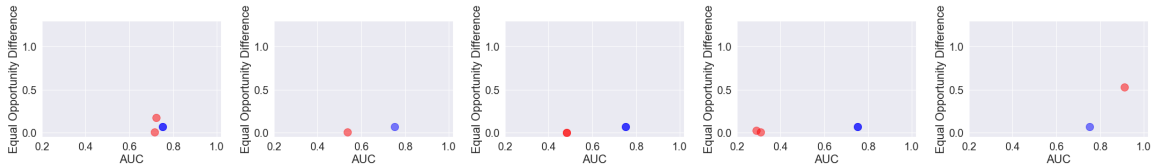
Theil Index



Average Odds Difference

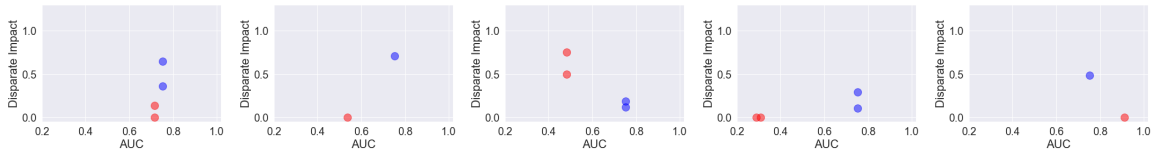


Equality Opportunity Difference

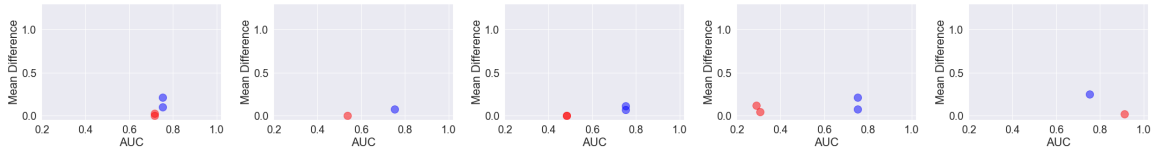


11.7.3 Meta Fair Classifier - Statistical Ratio & Equalized Odds Post-Processing

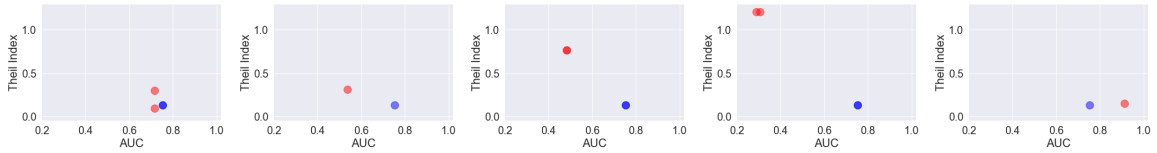
Disparate Impact



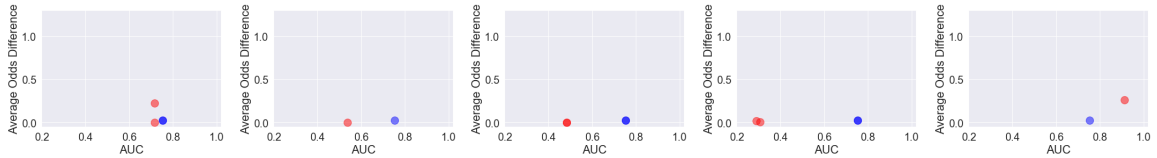
Mean Difference



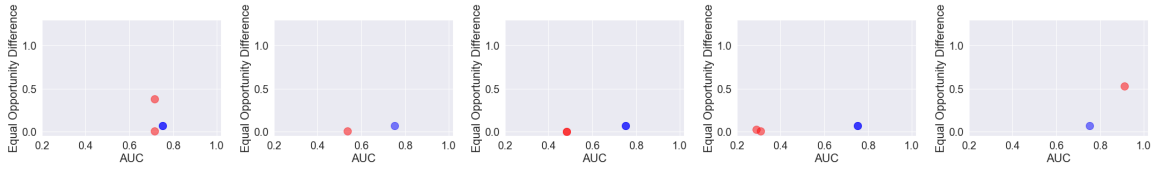
Theil Index



Average Odds Difference

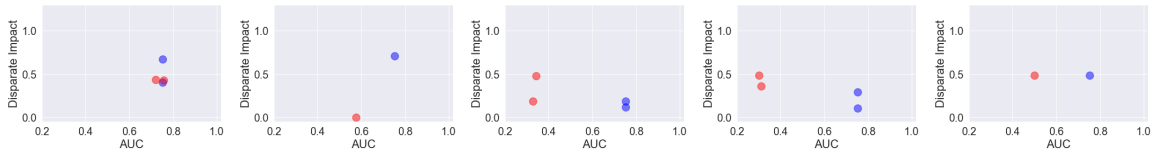


Equality Opportunity Difference

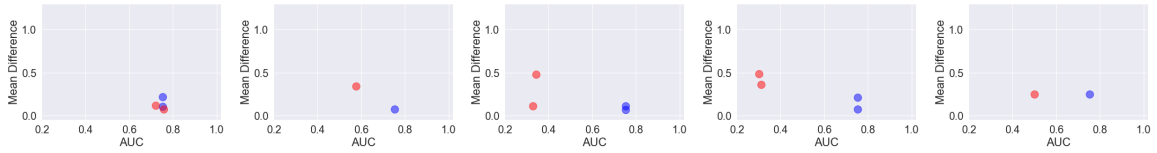


11.7.4 Prejudice Remover & Calibrated Equalized Odds Post-Processing

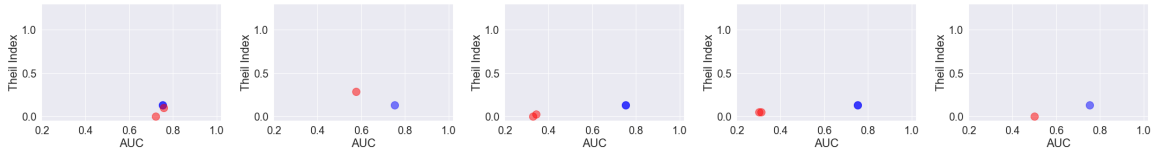
Disparate Impact



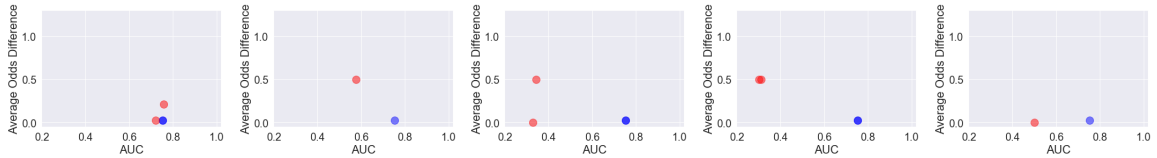
Mean Difference



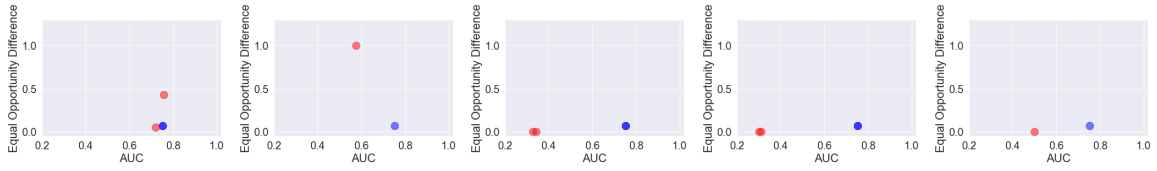
Theil Index



Average Odds Difference

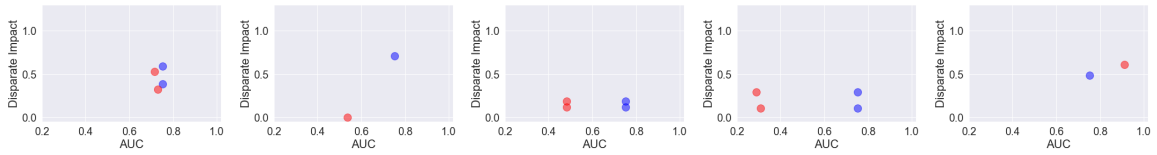


Equality Opportunity Difference

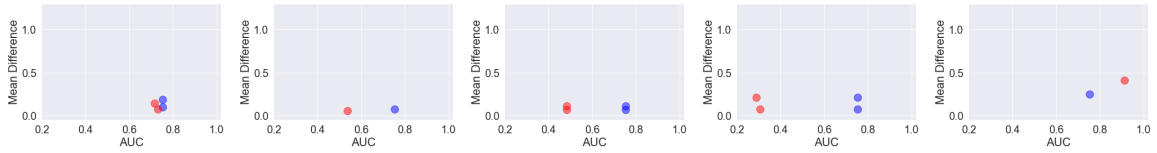


11.7.5 Calibrated Equal Odds & Meta Fair Classifier - False Discovery ratio

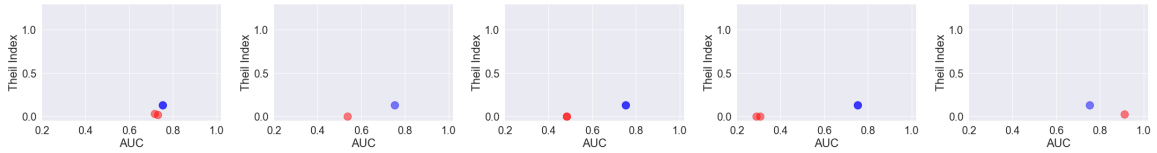
Disparate Impact



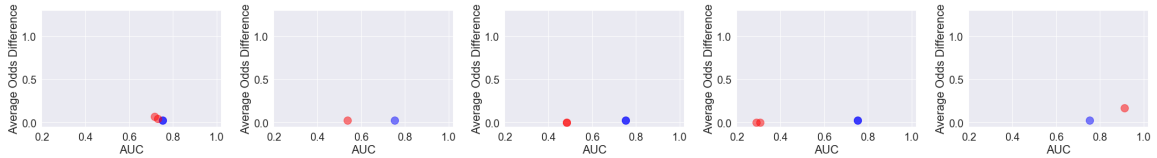
Mean Difference



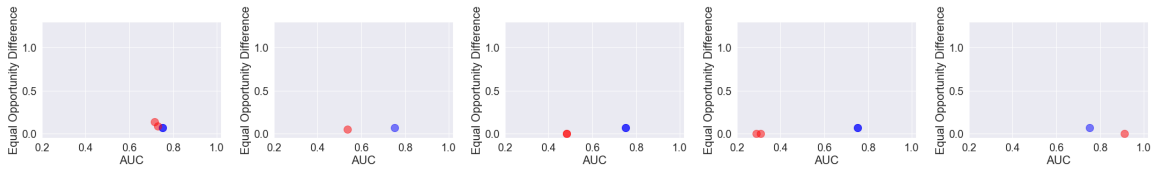
Theil Index



Average Odds Difference

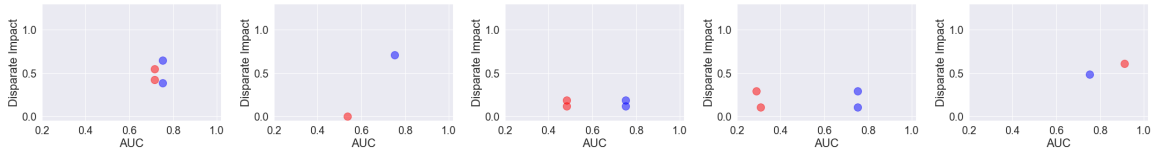


Equality Opportunity Difference

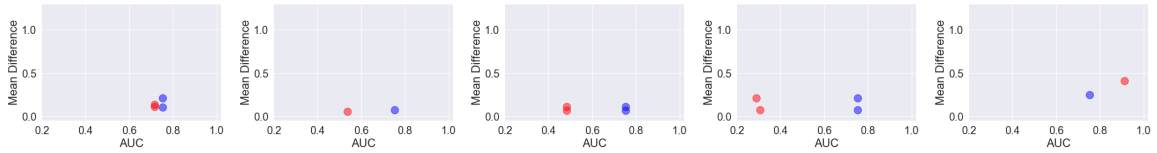


11.7.6 Calibrated Equal Odds & Meta Fair Classifier - Statistical Ratio

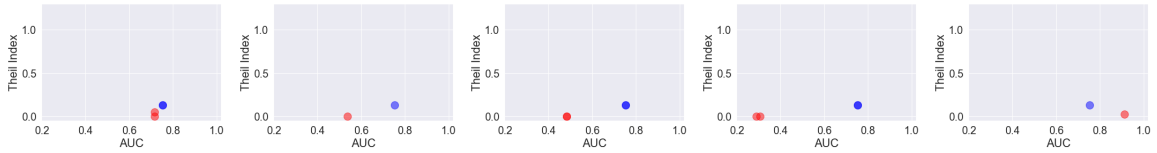
Disparate Impact



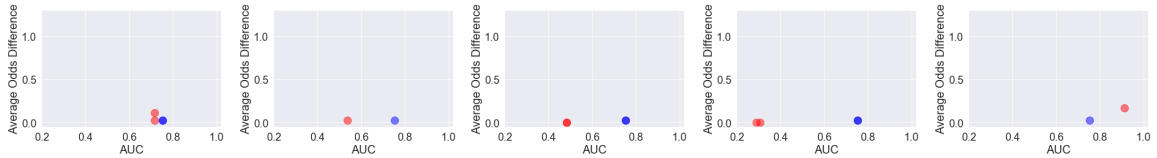
Mean Difference



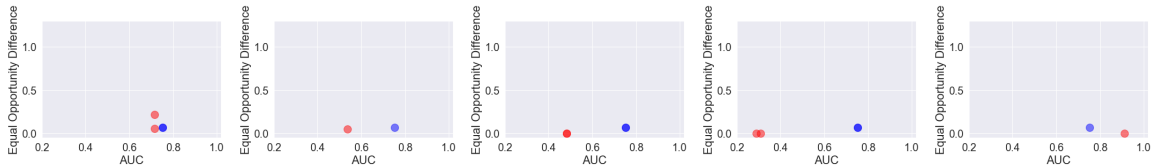
Theil Index



Average Odds Difference

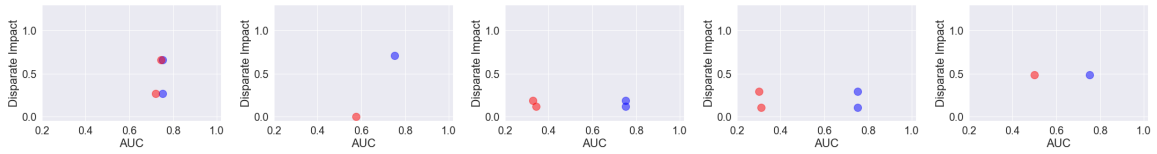


Equality Opportunity Difference

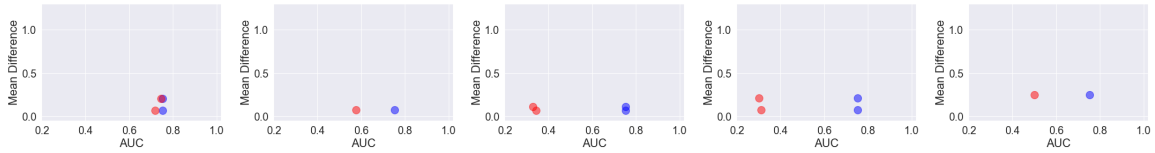


11.7.7 Reject Option Classification & Prejudice Remover

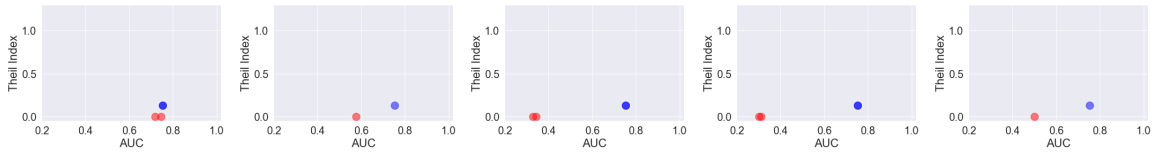
Disparate Impact



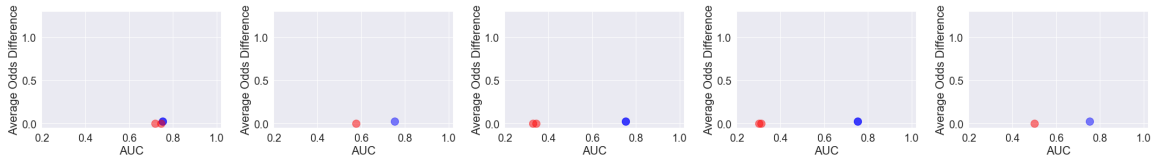
Mean Difference



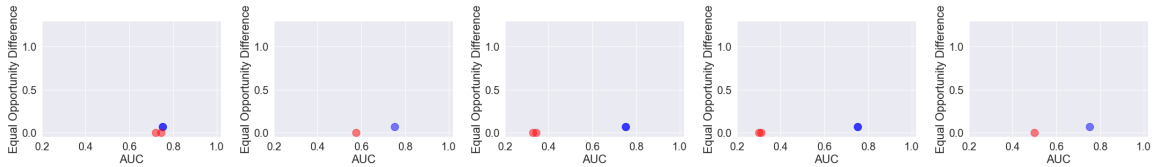
Theil Index



Average Odds Difference



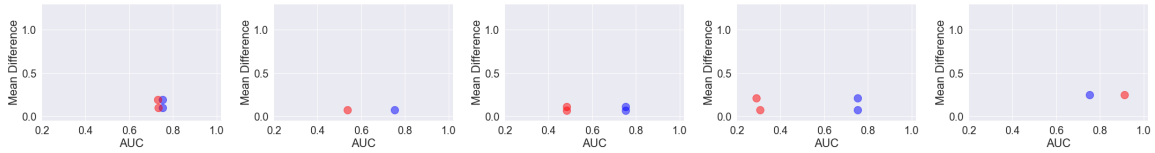
Equality Opportunity Difference



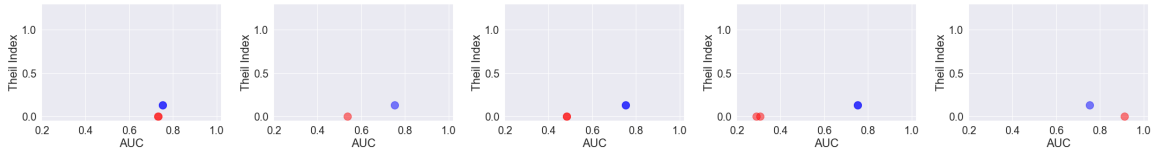
11.7.8 Reject Option Classification & Meta Fair Classifier - False Discovery ratio

Disparate Impact

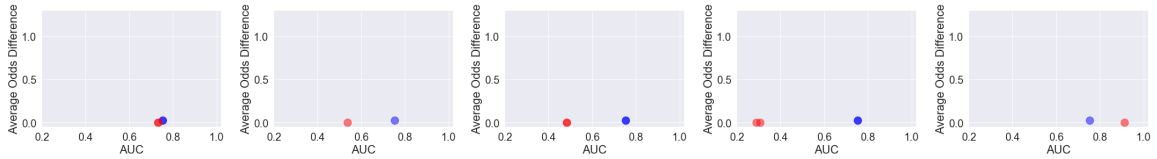
Mean Difference



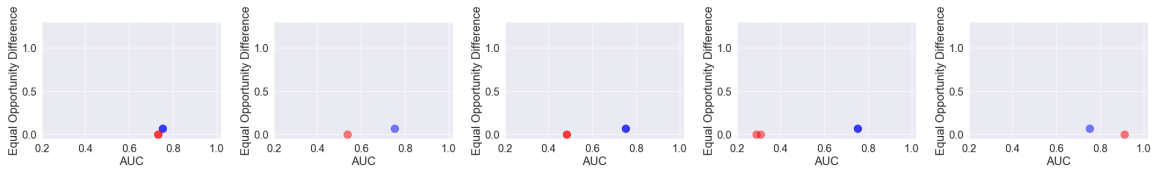
Theil Index



Average Odds Difference

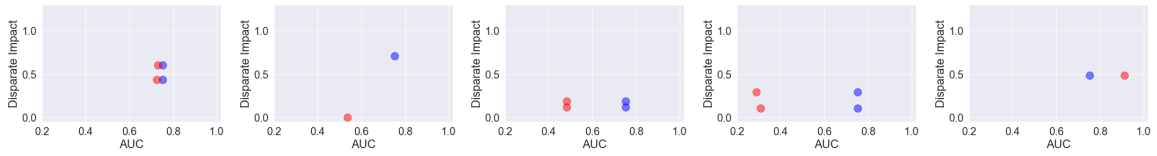


Equality Opportunity Difference

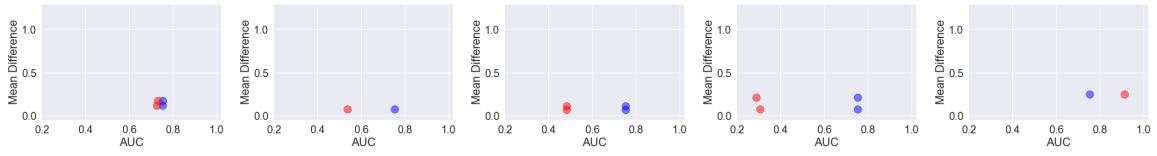


11.7.9 Reject Option Classification & Meta Fair Classifier - Statistical Ratio

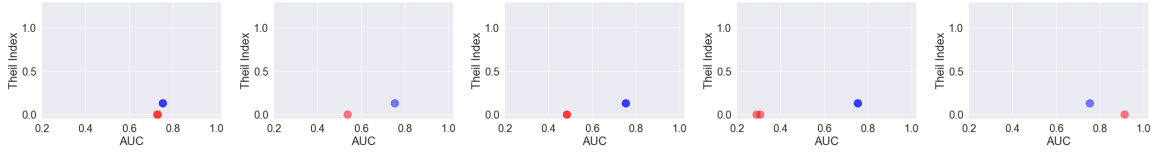
Disparate Impact



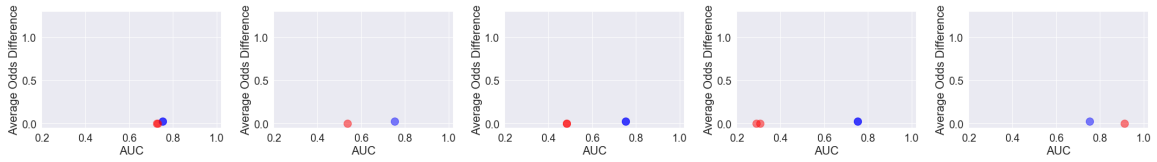
Mean Difference



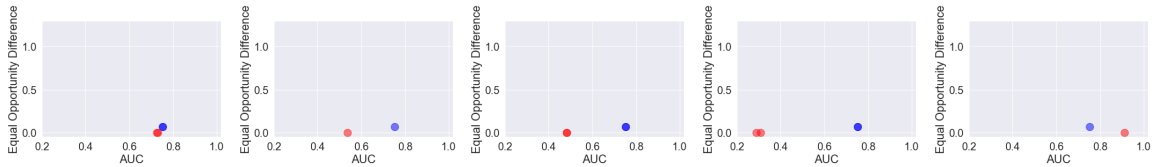
Theil Index



Average Odds Difference



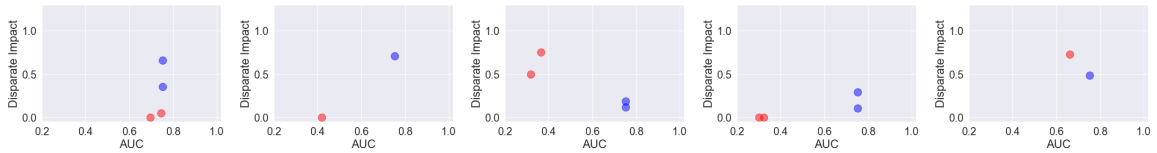
Equality Opportunity Difference



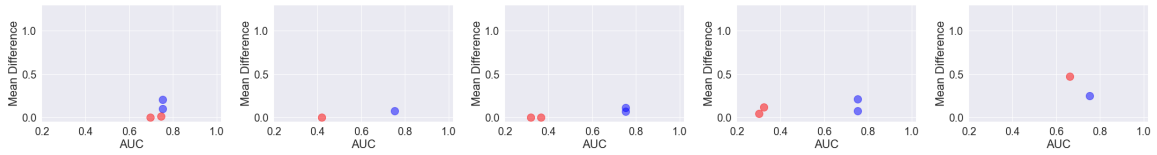
11.8 Pre & In & Post

11.8.1 Disparate Impact Remover & Prejudice Remover & Equalized Odds Post-Processing

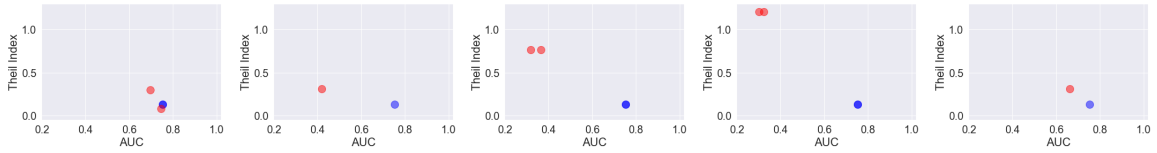
Disparate Impact



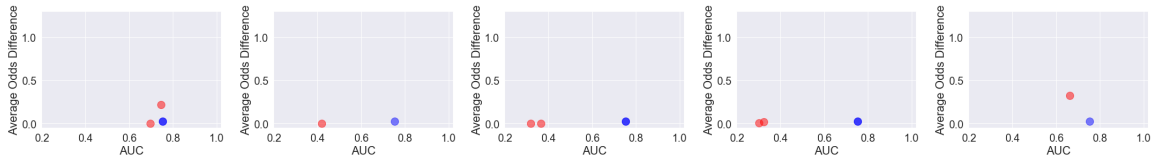
Mean Difference



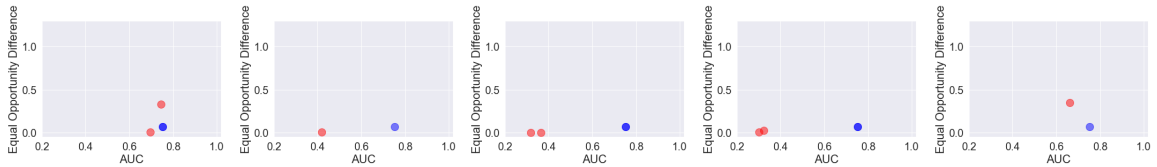
Theil Index



Average Odds Difference

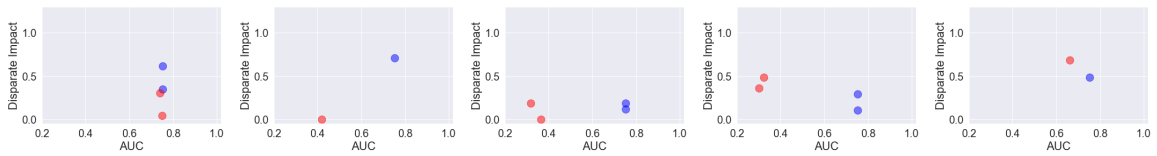


Equality Opportunity Difference

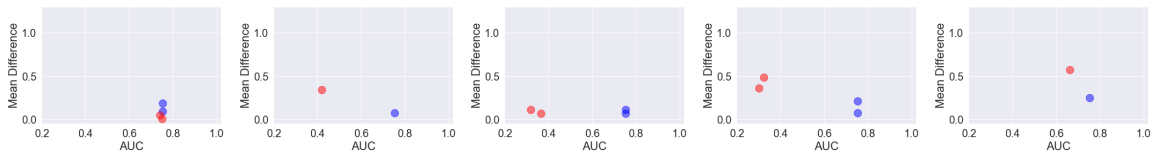


11.8.2 Disparate Impact Remover & Prejudice Remover & Calibrated Equalized odds Post-Processing

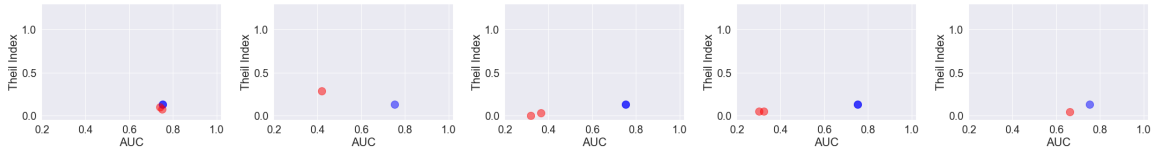
Disparate Impact



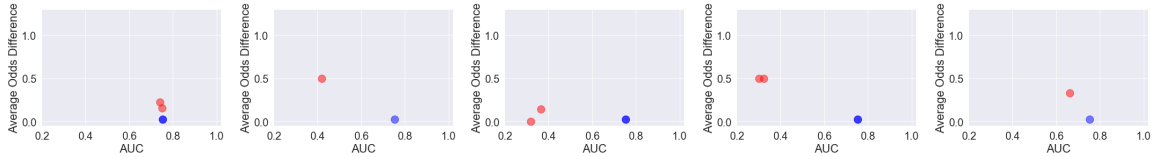
Mean Difference



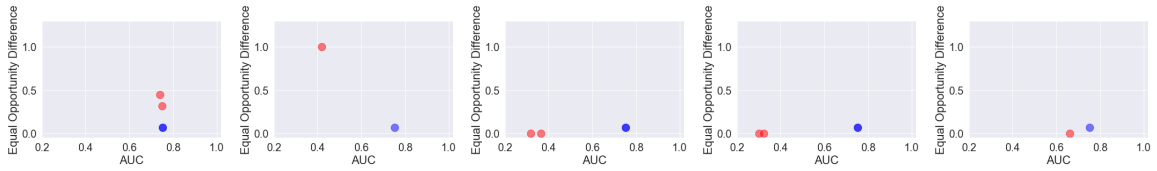
Theil Index



Average Odds Difference

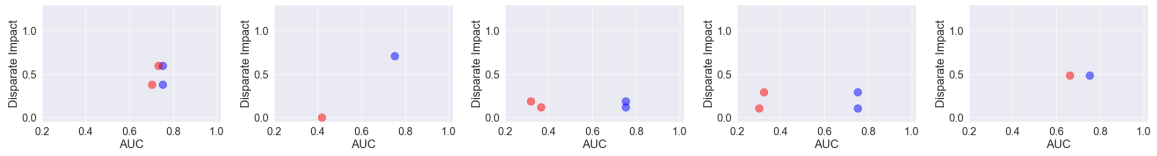


Equality Opportunity Difference

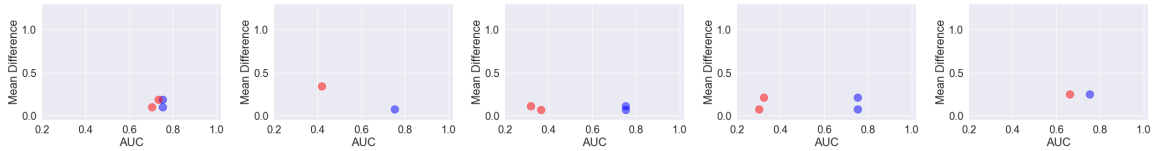


11.8.3 Disparate Impact Remover & Prejudice Remover & Reject Option Classification

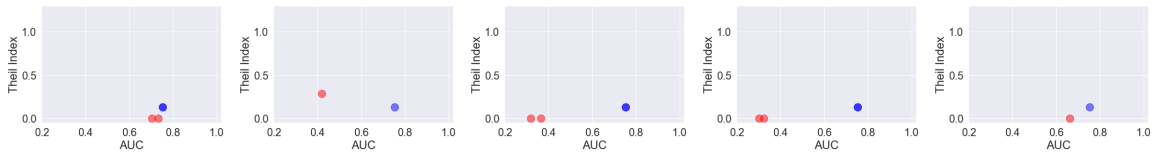
Disparate Impact



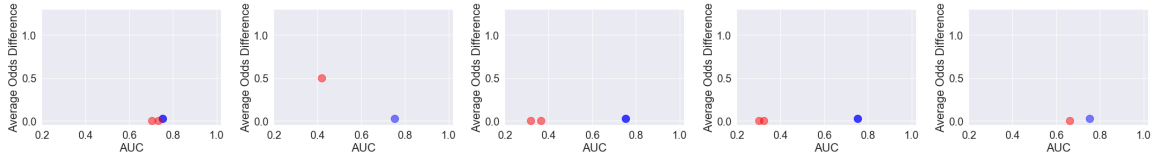
Mean Difference



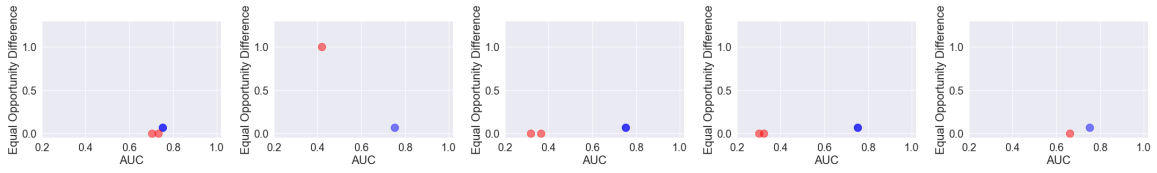
Theil Index



Average Odds Difference

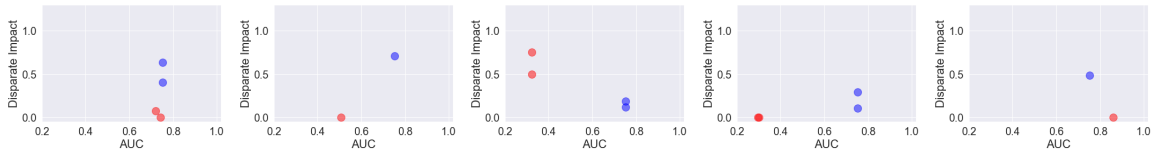


Equality Opportunity Difference

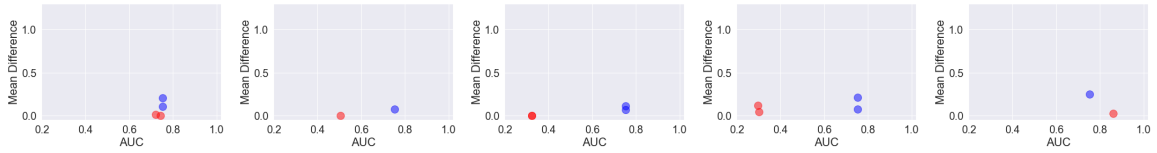


11.8.4 Disparate Impact Remover & Meta Fair Classifier - False Discovery Ratio & Equalized Odds Post-Processing

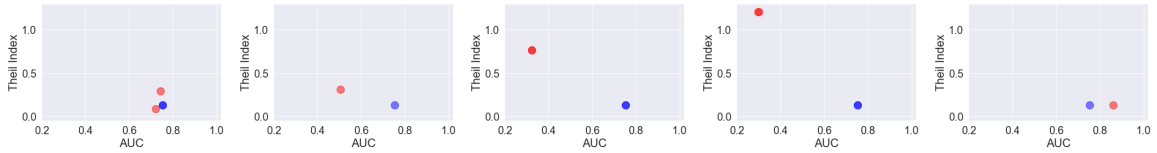
Disparate Impact



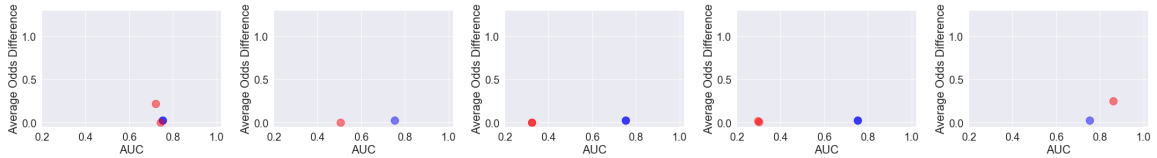
Mean Difference



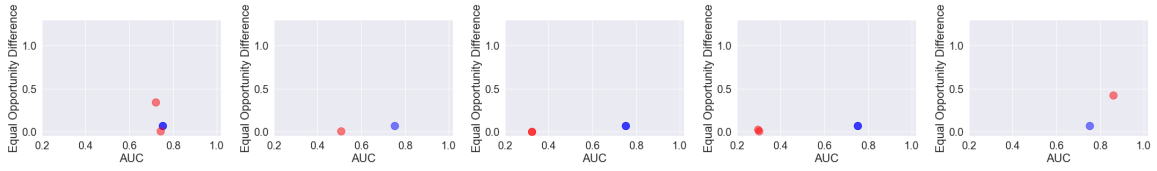
Theil Index



Average Odds Difference

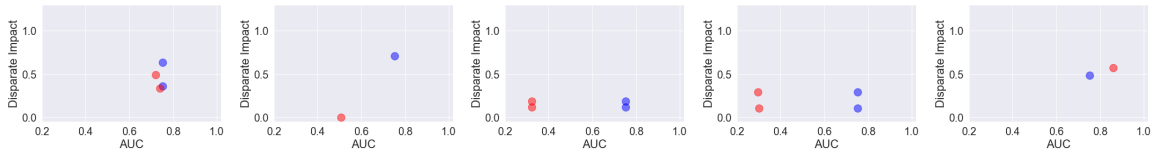


Equality Opportunity Difference

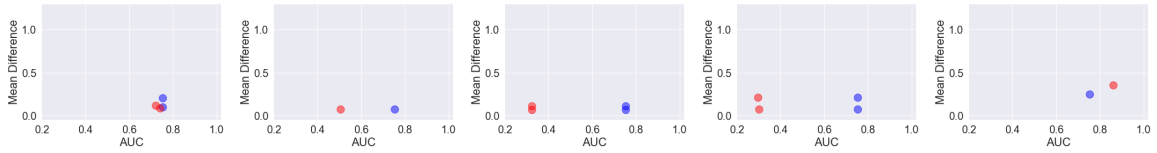


11.8.5 Disparate Impact Remover & Meta Fair Classifier - False Discovery Ratio & Calibrated Equalized odds Post-Processing

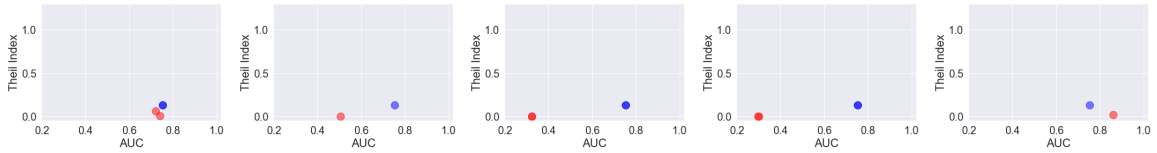
Disparate Impact



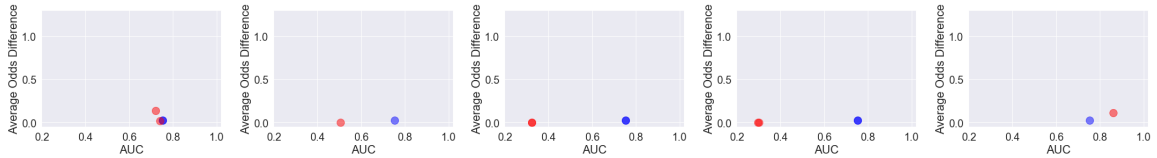
Mean Difference



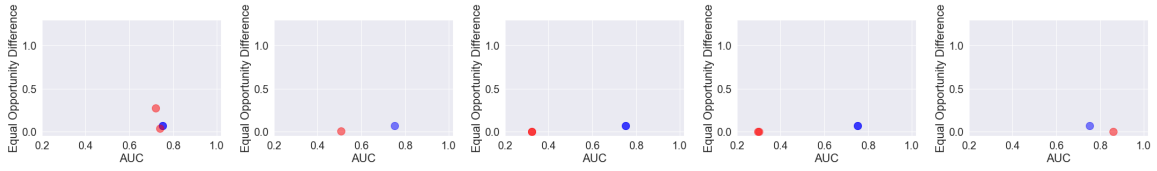
Theil Index



Average Odds Difference

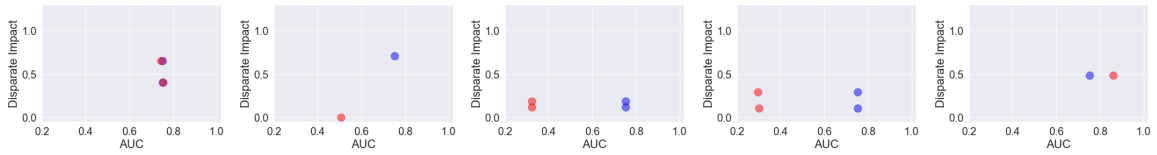


Equality Opportunity Difference

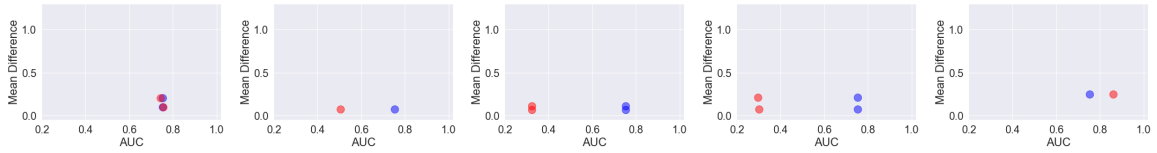


11.8.6 Disparate Impact Remover & Meta Fair Classifier - False Discovery Rate & Reject Option Classification

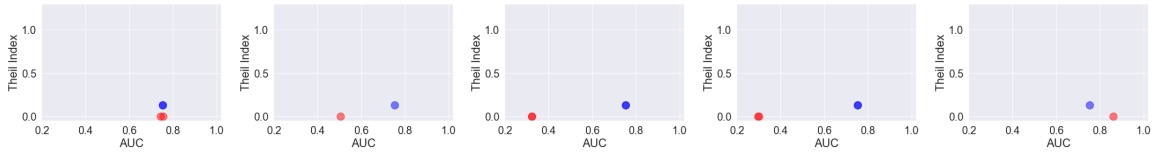
Disparate Impact



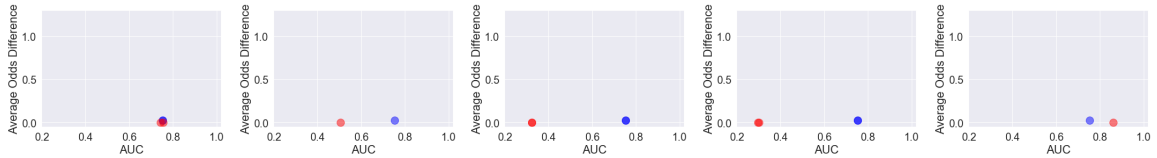
Mean Difference



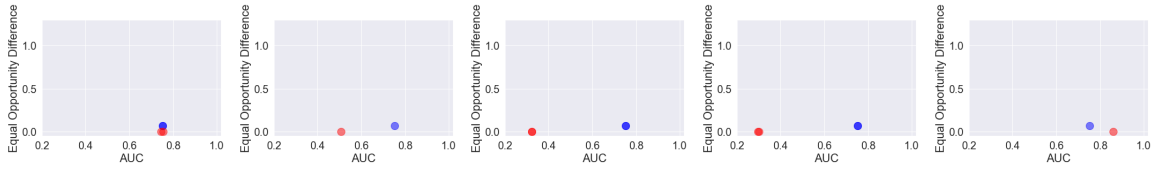
Theil Index



Average Odds Difference

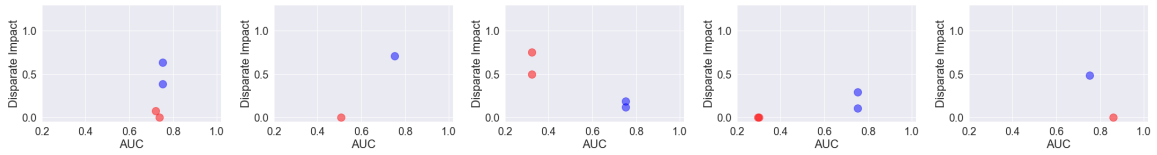


Equality Opportunity Difference

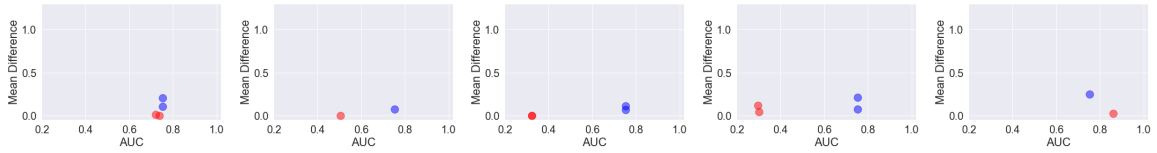


11.8.7 Disparate Impact Remover & Meta Fair Classifier - Statistical Ratio & Equalized Odds Post-Processing

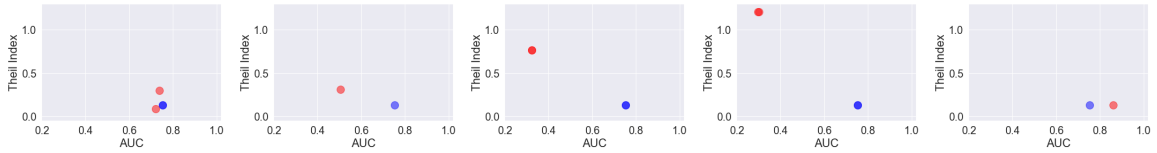
Disparate Impact



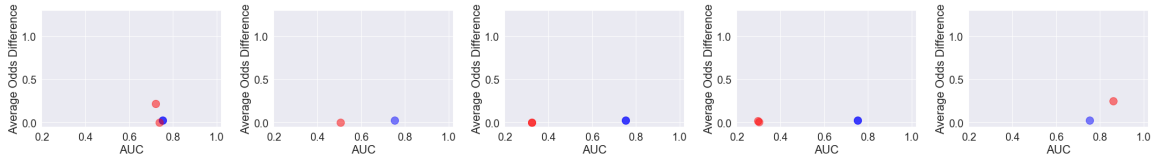
Mean Difference



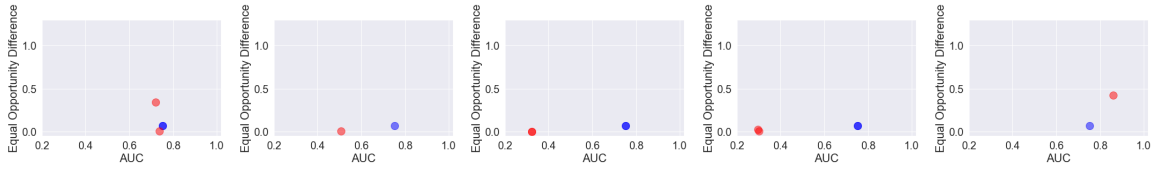
Theil Index



Average Odds Difference

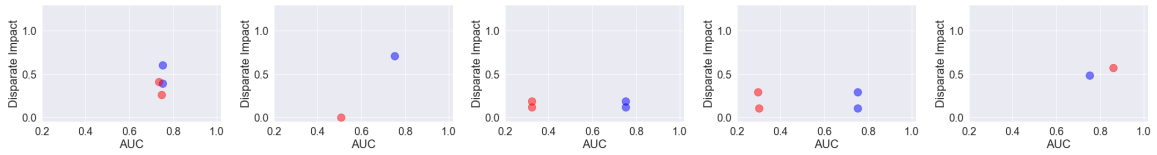


Equality Opportunity Difference

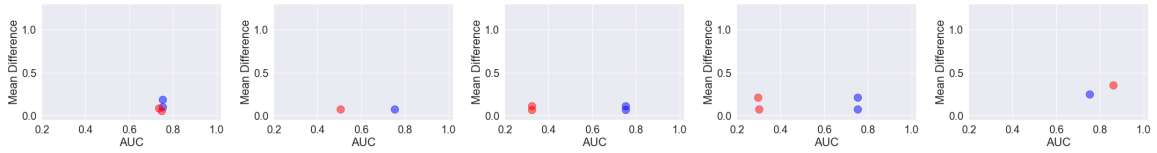


11.8.8 Disparate Impact Remover & Meta Fair Classifier - Statistical Ratio & Calibrated Equalized Odds Post-Processing

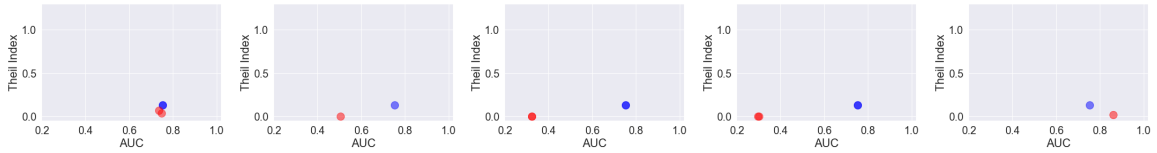
Disparate Impact



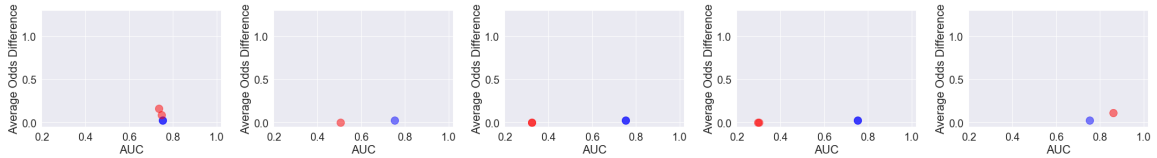
Mean Difference



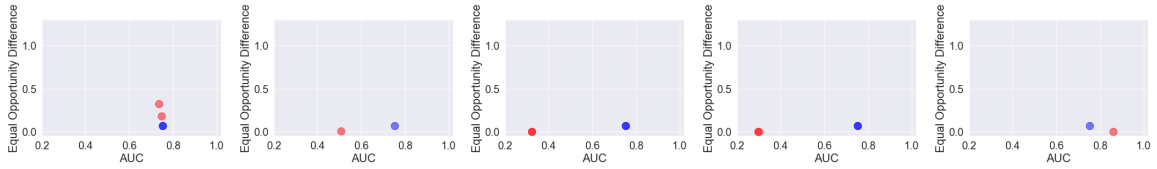
Theil Index



Average Odds Difference

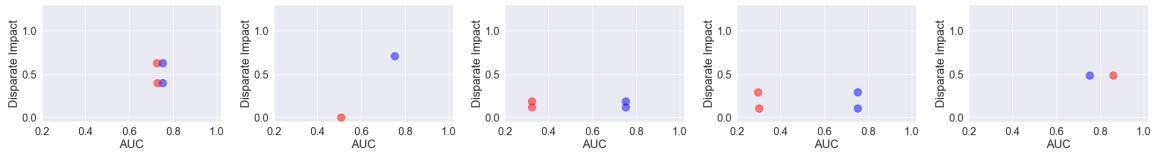


Equality Opportunity Difference

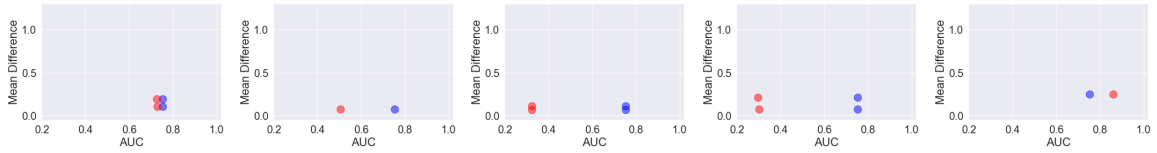


11.8.9 Disparate Impact Remover & Meta Fair Classifier - Statistical Ratio & Reject Option Classification

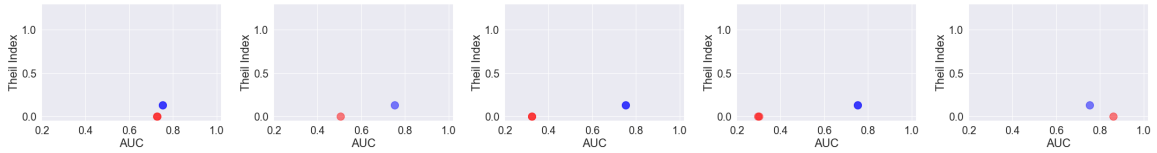
Disparate Impact



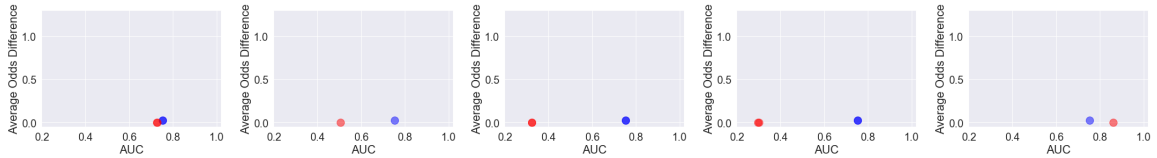
Mean Difference



Theil Index



Average Odds Difference



Equality Opportunity Difference