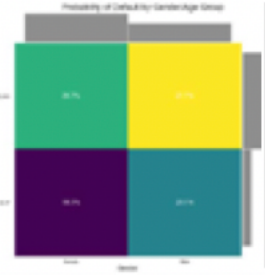
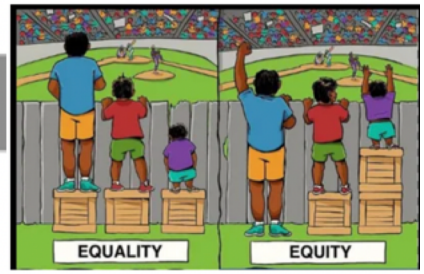
Brandenburgische
Technische Universität
Cottbus - Senftenberg

Explainable Machine Learning

Bias Mitigation





Content

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Bias Mitigation

- Bias and Fairness
- Detecting bias
- Mitigating bias

Mission

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- In 2008, there was a global financial crisis.
- Financial bubble has burst

- US banks wanted to make higher profits.
- -> home credits to people with **low incomes** and **insufficient guarantees**
- insurance policies with other banks against losses in case the borrowers could not pay the credits

Mission

Brandenburgische
Technische Universität
Cottbus - Senftenberg

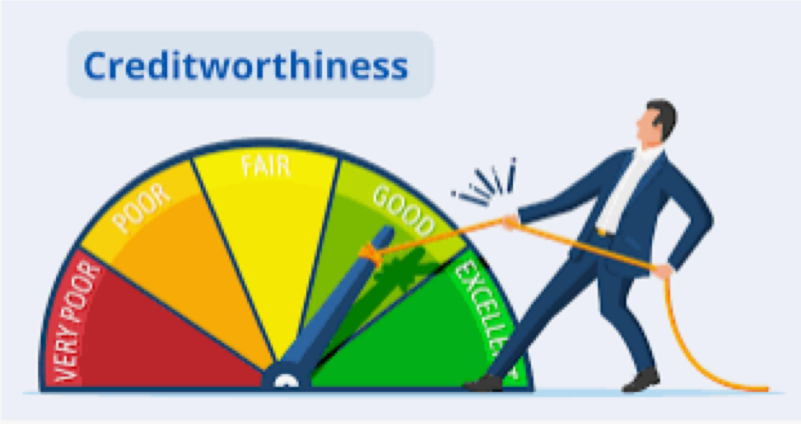
- Many people could not pay off the credits
- The banks received money from the insurance company (other bank)
- -> other **banks collapsed**

- Because many people had to sell their houses
- -> selling price went down.
- -> the banks got less than they had given as credits (expected interest rates)
- -> more banks collapsed

Mission

Brandenburgische
Technische Universität
Cottbus - Senftenberg

We want to develop a program that reasonably **checks** the **creditworthiness** of bank customers

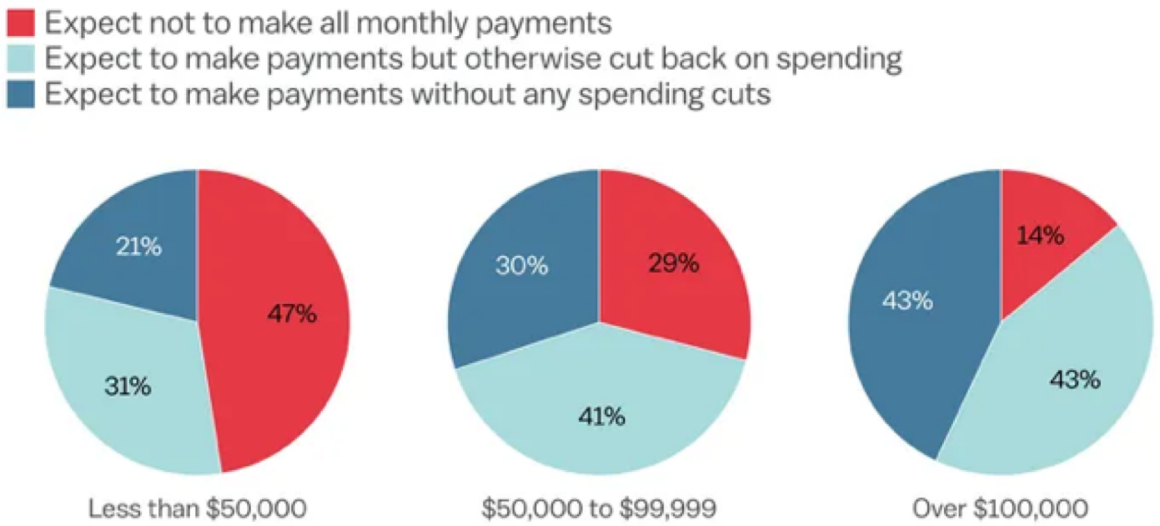


Approach

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Experience:

- younger people are less able to repay credits
- Gender differences?



student loan borrowers

Vox

Approach

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Identify unfair bias
- Mitigate the bias effects using the **AI Fairness 360**
  (An **Extensible Toolkit** for **Detecting**, Understanding,
  and **Mitigating** Unwanted **Algorithmic Bias)**
- Identify the causal effects
- Assess the heterogeneous treatment effects
- Ensure that our conclusions are robust

Data

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- ```
  ccdefault_all_df =
  pd.read_csv('./data/credit/creditworthiness.csv',sep=',',encoding='latin-1')
  ```

• CC_LIMIT_CAT: ordinal; the credit card limit (_CC_LIMIT) separated into **eight more or less equally distributed quartiles**

• EDUCATION: nominal; the customer's educational attainment level (0: Other, 1:High School, 2: Undergraduate, 3: Graduate)

• MARITAL_STATUS: nominal; the customer's marital status (0: Other, 1: Single,2: Married)

• GENDER**:** nominal; the gender of the customer (**1: Male, 2: Female**)

• AGE GROUP: binary; denoting if the customer belongs to a privileged age group

(**1:privileged (26-47 years old), 0: underprivileged (every other age)**)

• pay_status_1… pay_status_6: ordinal; the repayment status for the previous six periods from April, pay_status_6, to August 2005, pay_status_1

(-1: pay duly, 1: payment is 1 month delayed, 2: payment is 2 months delayed… 8: 8 months delayed, 9: 9 months and above)

• paid_pct_1… paid_pct_6: continuous; what percentage of the bill due each month from April, paid_pct_6, to August 2005, paid_pct_1**, was paid**

• bill1_over_limit: continuous; the last bill's ratio in August 2005 over the corresponding credit limit

• IS_DEFAULT: binary; target; whether the customer has delayed repayment

• _TREATMENT: nominal; the intervention or policy prescribed to each customer (-1: not part of the experiment, 0: Control group, 1: Lower Credit Limit, 2: Payment Plan, 3: Payment Plan and Credit Limit)

. . .
  ```

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Definition of fairness

of the European Union High Level Expert Group on Artificial Intelligence

1. Lawful—respecting all applicable laws and regulations
2. Ethical—respecting ethical principles and values
3. Robust—both from a technical perspective while taking into account its social environment

- https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai

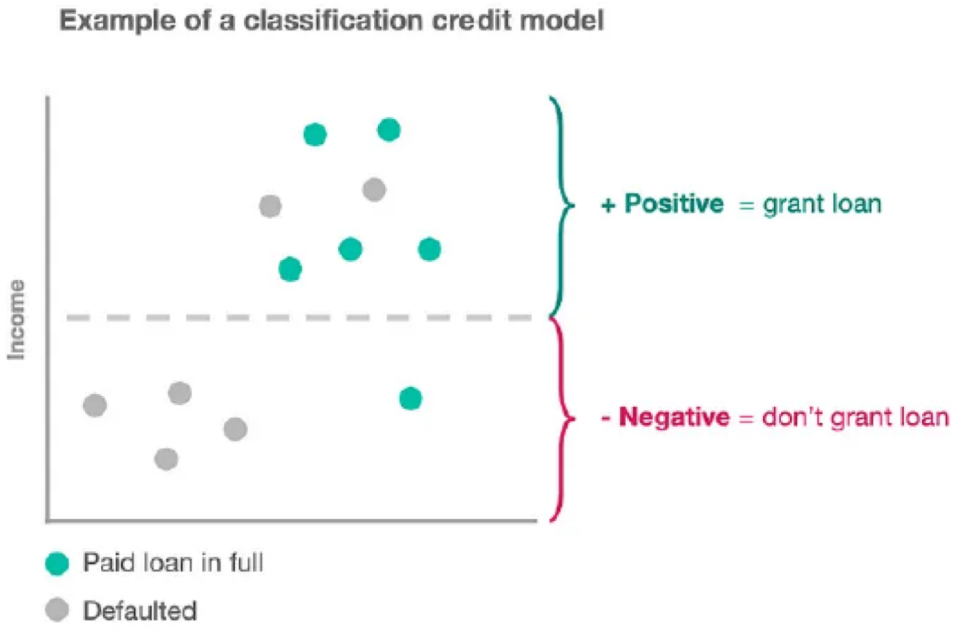
Brandenburgische
Technische Universität
Cottbus - Senftenberg

How can we detect and prevent discriminatory outcomes of ML?

Example 1

Paid credit:
those who paid in full their credits

Defaulted:
those who were not able to pay their credits and defaulted



Brandenburgische
Technische Universität
Cottbus - Senftenberg

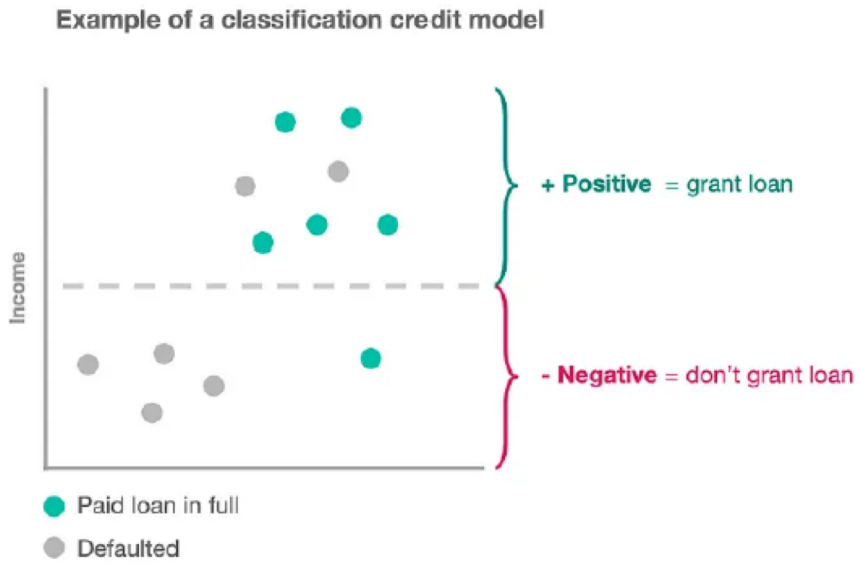
Mathematical Fairness Methods

- Threshold
- Demografic Parity
- Equal Opportunity
- Equalised Odds

Brandenburgische
Technische Universität
Cottbus - Senftenberg

How can we detect and prevent discriminatory outcomes of ML?

Example: Threshold



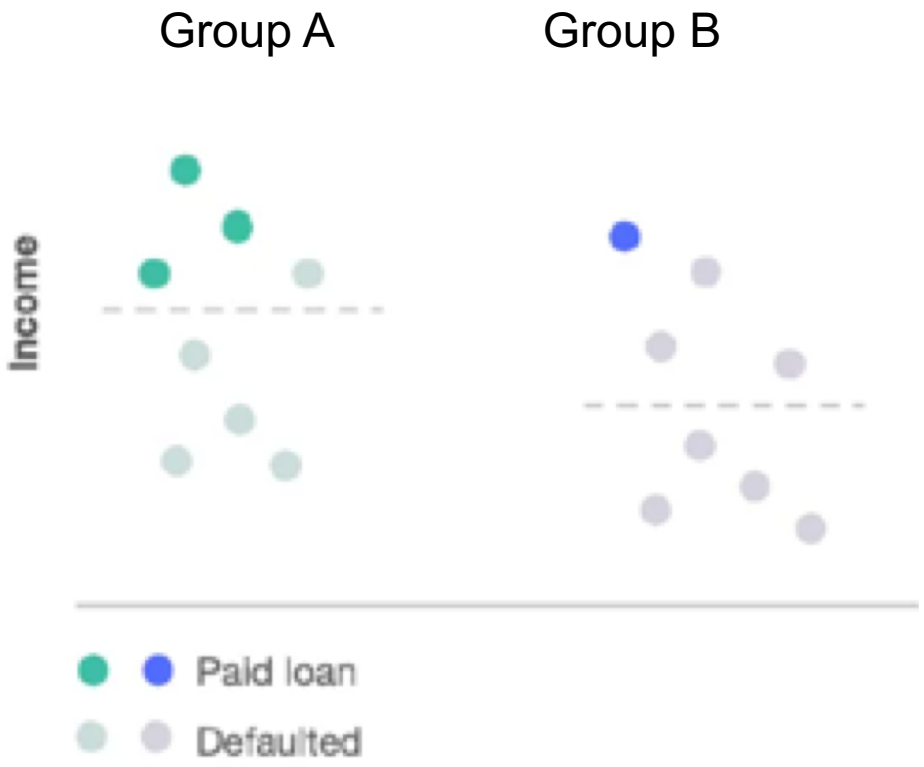
no groups

The model based purely on *income*
Threshold decides who gets a credit in future

-> **not fair:** one person, who paid their credits don't get a credit in future

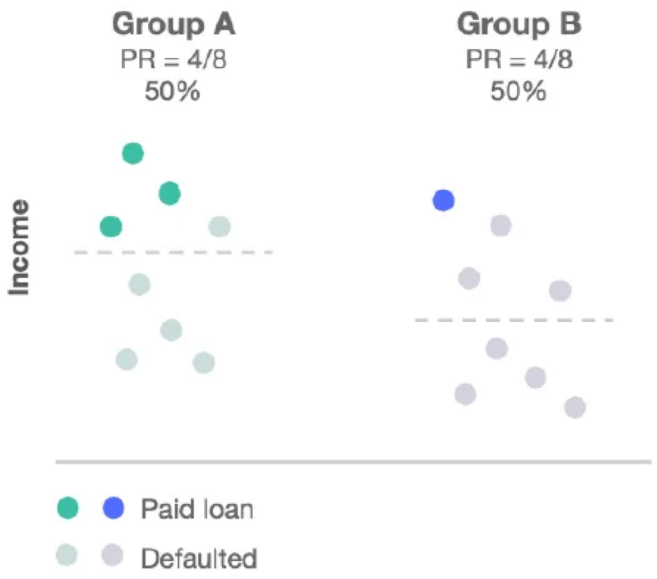
Brandenburgische
Technische Universität
Cottbus - Senftenberg

First Example of Mathematical Fairness:



Brandenburgische
Technische Universität
Cottbus - Senftenberg

Mathematical Fairness:
Demographic Parity



$$P(\hat{Y} \mid A = 0) = P(\hat{Y} \mid A = 1)$$

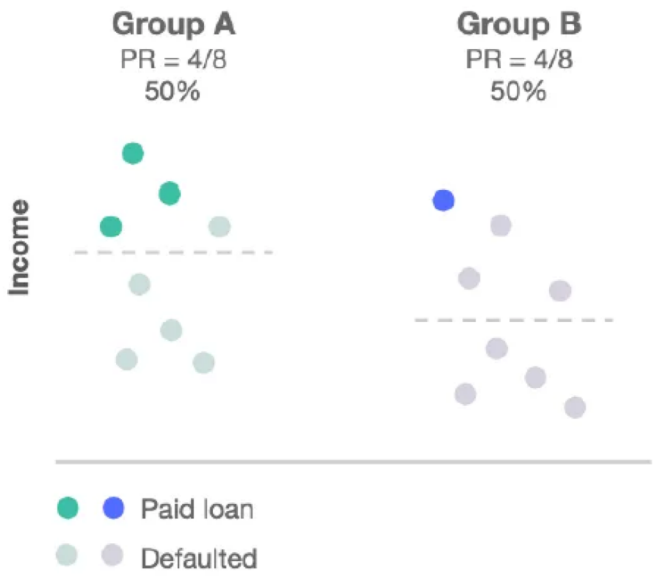
Positive Rate = (TP+FP)/(TP+FP+TN+FN)
Positive Rate (A=0) = Positive Rate (A=1)
A=0: Group A, A=1: Group B

protected
class (e.g. gender) should receive the positive outcome at equal rates.

A positive outcome is the preferred decision, such as “getting to university”, “getting a credit”

Brandenburgische
Technische Universität
Cottbus - Senftenberg

First Example of Mathematical Fairness:
Demographic Parity



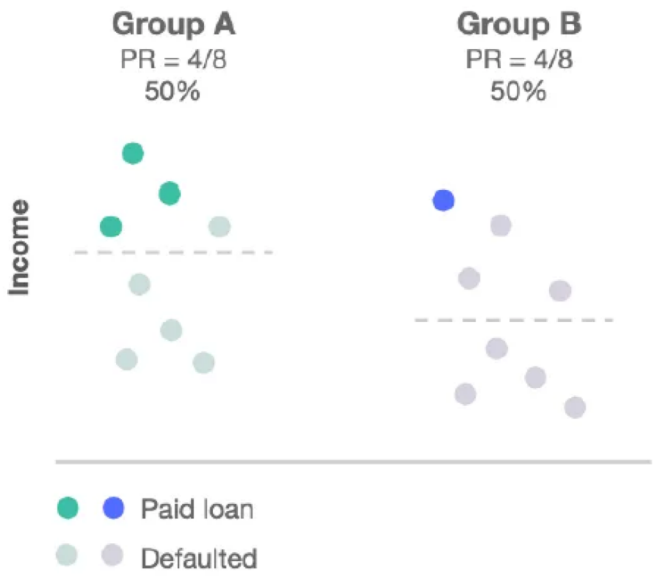
What are
- Advantages and
- Disadvantages?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

First Example of Mathematical Fairness:

Demographic Parity

$$P(\hat{Y} \mid A=0) = P(\hat{Y} \mid A=1)$$



Positive Rate (A=0) = Positive Rate (A=1)

Percentage of people getting a credit in group A is equal to percentage in group B

Advantage: Demographic parity is meaningful in political decisions

Disadvantage:
- FP not observed
- FN nor observed

(Person might not be qualified
Can result in high financial losses)

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Demographic Parity

We should use Demographic Parity, when

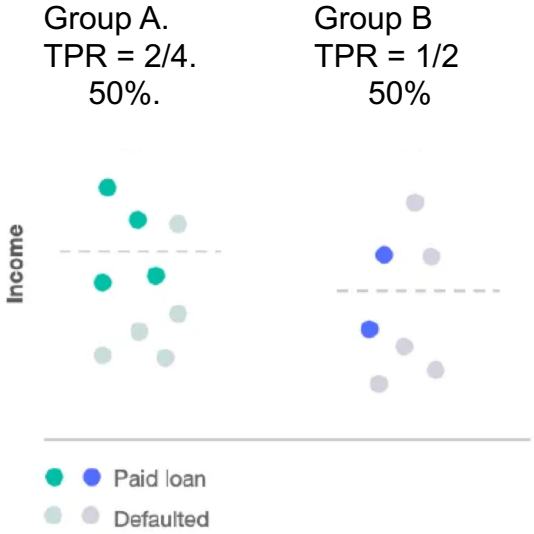
- We want to change the state of our current world to improve it (e.g.: more minority groups getting more rights)
- We are aware of historical biases may have affected the quality of our data
- We have a plan in place to support the unprivileged group and to prevent there enforcement of historical biases

- -> Oxford University: students from disadvantaged background

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Second Example of Mathematical Fairness:

Equal Opportunity



$$P(\hat{Y} = 1 \mid A = 0, Y = 1) = P(\hat{Y} = 1 \mid A = 1, Y = 1)$$

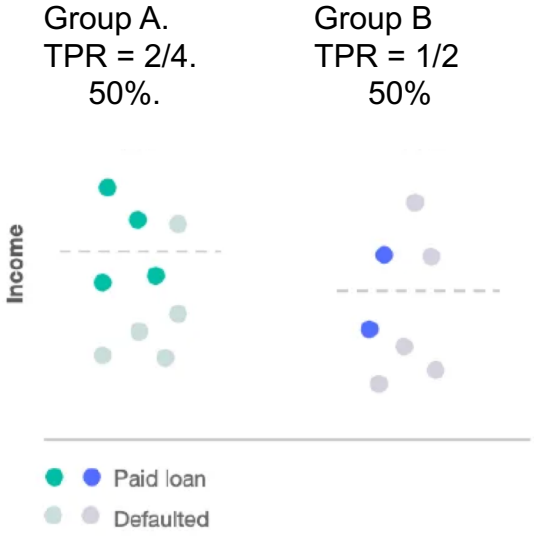
True Positive Rate = TP / (TP + FN)

True Positive Rate (A=0) = True Positive Rate (A=1)

What are
- Advantages and
- Disadvantages?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Second Example of Mathematical Fairness:
Equal Opportunity



$$P(\hat{Y} = 1 \mid A = 0, Y = 1) = P(\hat{Y} = 1 \mid A = 1, Y = 1)$$

True Positive Rate (A=0) = True Positive Rate (A=1)

Advantage:
Equal Opportunity states that each group should get the positive outcome at equal rates, **assuming** that people in this group **qualify for it**.

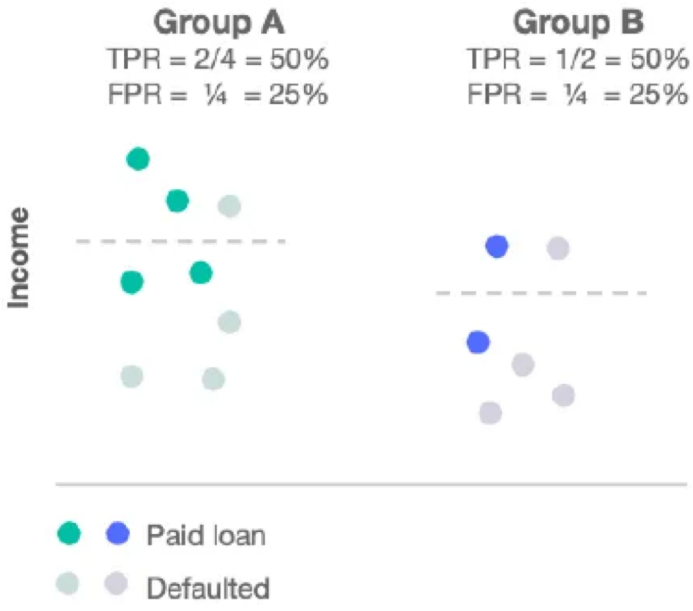
Disadvantage:
- FP not observed

(False positives can cause costs)

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Third Example of Mathematical Fairness:

Equalised Odds



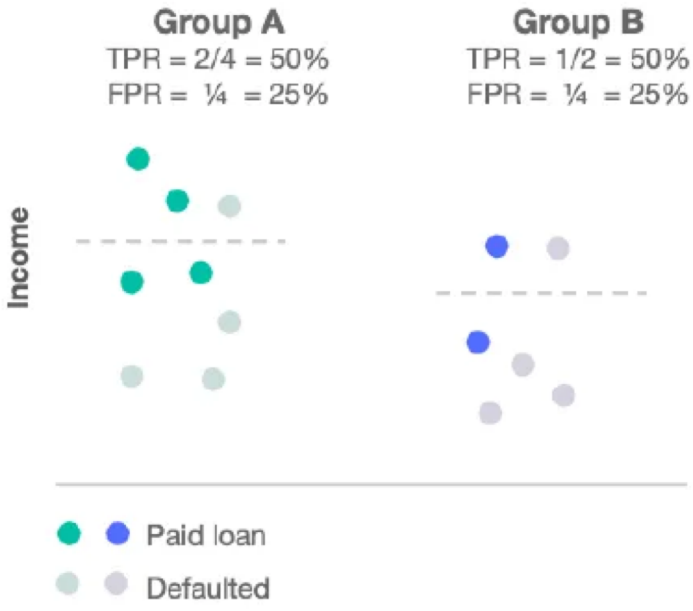
$$P(\hat{Y} = 1 \mid A = 0, Y = y) = P(\hat{Y} = 1 \mid A = 1, Y = y), y \in \{0, 1\}$$

True Positive Rate (A=0) = True Positive Rate (A=1)
False Positive Rate (A=0) = False Positive Rate (A=1)
FP/(FP+TN)

What are
- Advantages and
- Disadvantages?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Third Example of Mathematical Fairness:
Equalised Odds



$$P(\hat{Y} = 1 \mid A = 0, Y = y) = P(\hat{Y} = 1 \mid A = 1, Y = y), y \in \{0, 1\}$$

True Positive Rate (A=0) = True Positive Rate (A=1)
False Positive Rate (A=0) = False Positive Rate (A=1)

False Positive Rate = FP / (FP + TN)

Advantage:
People have equal opportunities in their group

Disadvantage:
- Model could be at risk of not achieving optimal accuracy

Brandenburgische
Technische Universität
Cottbus - Senftenberg

All fairness models have advantages and disadvantages

When should we use which fairness model?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Summery Mathematical Fairness

Model	When to use
Demographic Parity	We are conscious of **historical biases** of our data We have a **plan in place to support the unprivileged group** and to prevent the reinforcement of historical biases
Equal Opportunity	Introducing **False Positives are not costly** to the user nor the company The **target variable** is **not** considered **subjective**
Equalised Odds	The **target variable** is **not** considered **subjective** There **remains a profit** for the company

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Bias detection

- What bias features can we have?
  - Differences in gender, race, rural,…
- What types fairness do we want to achieve?
  - Legal fairness
  - Ethical fairness
  - Outcome fairness

Data
Preparation

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Split data into bias and causal impact

_TREATMENT: nominal; the intervention or policy prescribed to each customer
(-1: not part of the experiment, 0: Control group, 1: Lower Credit Limit, 2: Payment Plan, 3: Payment Plan and Credit Limit)

```
ccdefault_bias_df = ccdefault_all_df[ccdefault_all_df._TREATMENT < 1]
ccdefault_causal_df = ccdefault_all_df[ccdefault_all_df._TREATMENT >= 0]
```

-

Data
Bias Detection

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Credit creditworthiness: IS_DEFAULT = 1

What feature can have a bias?

How can we detect the bias?

Data

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- ```
  ccdefault_all_df =
  pd.read_csv('./data/credit/creditworthiness.csv',sep=',',encoding='latin-1')
  ```

• CC_LIMIT_CAT: ordinal; the credit card limit (_CC_LIMIT) separated into **eight more or less equally distributed quartiles**

• EDUCATION: nominal; the customer's educational attainment level (0: Other, 1:High School, 2: Undergraduate, 3: Graduate)

• MARITAL_STATUS: nominal; the customer's marital status (0: Other, 1: Single,2: Married)

• GENDER**:** nominal; the gender of the customer (**1: Male, 2: Female**)

• AGE GROUP: binary; denoting if the customer belongs to a privileged age group

(**1:privileged (26-47 years old), 0: underprivileged (every other age)**)

• pay_status_1… pay_status_6: ordinal; the repayment status for the previous six periods from April, pay_status_6, to August 2005, pay_status_1

(-1: pay duly, 1: payment is 1 month delayed, 2: payment is 2 months delayed… 8: 8 months delayed, 9: 9 months and above)

• paid_pct_1… paid_pct_6: continuous; what percentage of the bill due each month from April, paid_pct_6, to August 2005, paid_pct_1**, was paid**

• bill1_over_limit: continuous; the last bill's ratio in August 2005 over the corresponding credit limit

• IS_DEFAULT: binary; target; whether the customer has delayed repayment

• _TREATMENT: nominal; the intervention or policy prescribed to each customer (-1: not part of the experiment, 0: Control group, 1: Lower Credit Limit, 2: Payment Plan, 3: Payment Plan and Credit Limit)

. . .
  ```

Data
Bias Detection

Brandenburgische
Technische Universität
Cottbus - Senftenberg

How many women and how many men repay the credit late?

```
print("all repayment delays\n",
ccdefault_bias_df[ccdefault_bias_df.IS_DEFAULT==1]._TREATMENT.\
            value_counts())

print("number of women/men\n",ccdefault_bias_df.GENDER.value_counts())
print("number of women/men with repayment delay\n",
ccdefault_bias_df[ccdefault_bias_df.IS_DEFAULT==1].GENDER.\
            value_counts())


print("Percentage of women/men with repayment
delay\n",ccdefault_bias_df[ccdefault_bias_df.IS_DEFAULT==1].GENDER.\
            value_counts()/ccdefault_bias_df.GENDER.value_counts())
```

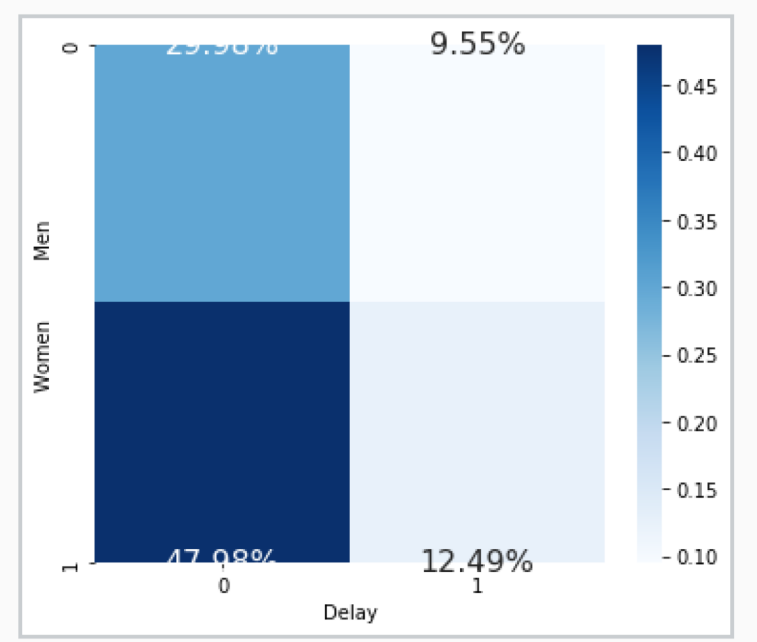
Data
Bias Detection

Brandenburgische
Technische Universität
Cottbus - Senftenberg

How many women and how many men repay the credit late?

```
cf_matrix = metrics.confusion_matrix(ccdefault_bias_df.GENDER==2,\
                                     ccdefault_bias_df.IS_DEFAULT)
print("cf_matrix\n",cf_matrix)
```

men with default: 9.55%
women with default: 12.49%



Bias Mitigation Bias_Mitigation.py

Data
Bias Detection

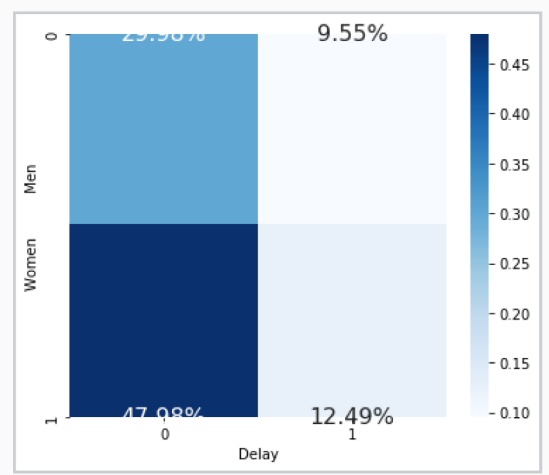
Brandenburgische Technische Universität Cottbus - Senftenberg

GENDER: (1: Male, 2: Female)

How many women and how many men repay the credit late?

```
all repayment delays
 _TREATMENT
-1    6335
 0      96
Name: count, dtype: int64
number of women/men
 GENDER
2    17644
1    11534
Name: count, dtype: int64
number of women/men with repayment delay
 GENDER
2    3644
1    2787
Name: count, dtype: int64
Percentage of women/men with repayment delay
 GENDER
2    0.206529
1    0.241633
```

all delays



Bias

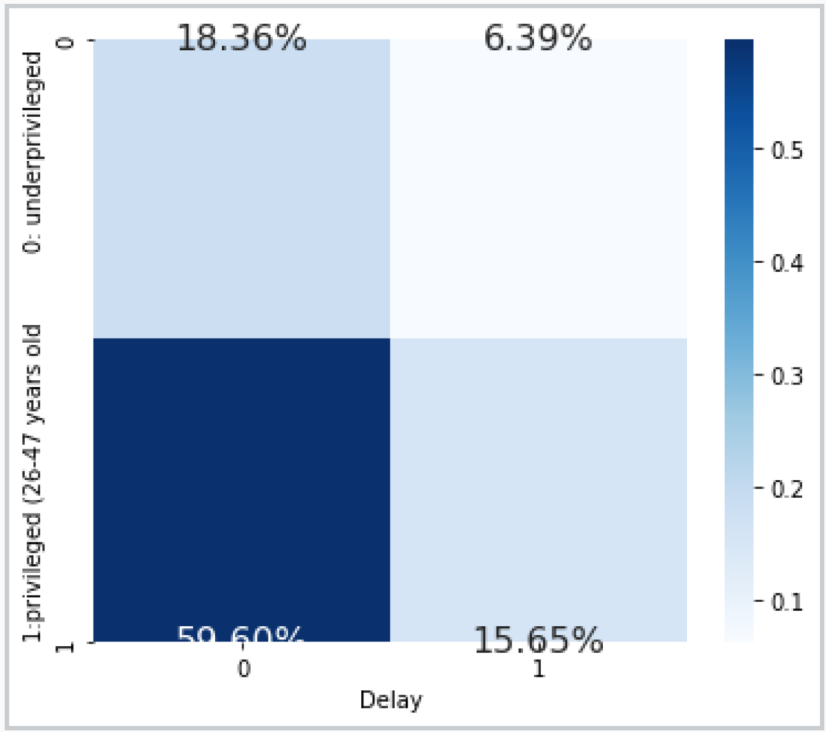
Bias Mitigation Bias_Mitigation.py

Data
Bias Detection

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- AGE_GROUP repayment delay

AGE GROUP: binary; denoting if the customer belongs to a privileged age group (1:privileged (26-47 years old), 0: underprivileged (every other age))



Bias

```
number of AGE_GROUP
 AGE_GROUP
1    21957
0     7221
Name: count, dtype: int64
number of AGE_GROUP with repayment delay
 AGE_GROUP
1    4566
0    1865
Name: count, dtype: int64
Percentage of AGE_GROUP with repayment delay
 AGE_GROUP
1    0.207952
0    0.258274
Name: count, dtype: float64
cf_matrix
 [[ 5356  1865]
 [17391  4566]]
```


Bias Mitigation Bias_Mitigation.py

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

-> larger groups have a greater impact on the target

We want to quantify dataset bias on the training data

- Define the target
- Split data

```
rand = 9
os.environ['PYTHONHASHSEED']=str(rand)
np.random.seed(rand)


y = ccdefault_bias_df['IS_DEFAULT']
X = ccdefault_bias_df.drop(['IS_DEFAULT'], axis=1).copy()
X_train, X_test, y_train, y_test =\
    model_selection.train_test_split(X, y, test_size=0.25, random_state=rand)
```

Data
Quantifying dataset bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Objective: get fairness attributes

- library AIF360
  - create abstracts datasets
  - these classes include the data converted to a NumPy array
  - Store attributes related to fairness

  - RegressionDataset() for regression
  - BinaryLabelDataset() for binary classification

Data
Quantifying dataset bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

BinaryLabelDatasets

- rationalize the analysis and modelling process
- dataset structure facilitates model training validation, and testing
- **Imbalance Handling** -> model is not biased toward the majority class

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

```
train_ds = BinaryLabelDataset(df=X_train.join(y_train),\
                      label_names=['IS_DEFAULT'],\
                      protected_attribute_names=['AGE_GROUP', 'GENDER'],
                      favorable_label=0, unfavorable_label=1)

test_ds = BinaryLabelDataset(df=X_test.join(y_test),\
                      label_names=['IS_DEFAULT'],\
                      protected_attribute_names=['AGE_GROUP', 'GENDER'],
                      favorable_label=0, unfavorable_label=1)
```

Maps protected attributes to binary privileged/unprivileged values (1/0)

- **favorable_label** (*float*) – Label value which is considered favorable (i.e. “positive”).
  - What we are trying to predict—the positive class: `favorable_label`
  - (not whether it is a favorable outcome)
- **protected_attributes** (*numpy.ndarray*) – A subset of features for which fairness is desired.
- **protected_attribute_names** (*list*(*str*)) – A subset of feature_names corresponding to protected_attributes.
- **privileged_protected_attributes** (*list*(*numpy.ndarray*)) – A subset of protected attribute values which are considered privileged from a fairness perspective.

our favorable class: no default (IS_DEFAULT == 0)

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

- Next, we create arrays for unprivileged_groups and privileged_groups

- Datasets in AGE_GROUP=1 (`privileged age group`) have a **lower probability of default** your credit payments
- -> so they are privileged_groups, and vice versa

```
unprivileged_groups=[{'GENDER': 2}]
privileged_groups=[{'GENDER': 1}]
```

unprivileged femals (GENDER: 2)

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

- Metrics for quantifying bias

```
metrics_train_ds = BinaryLabelDatasetMetric(train_ds,\
                           unprivileged_groups=unprivileged_groups,\
                           privileged_groups=privileged_groups)
metrics_test_ds = BinaryLabelDatasetMetric(test_ds,\
                           unprivileged_groups=unprivileged_groups,\
                           privileged_groups=privileged_groups)

print ("Difference in mean outcomes TEST between unprivileged and privileged groups = ",
metrics_test_ds.mean_difference())
```

```
Difference in mean outcomes TEST between unprivileged and privileged groups =  -7.01
```

Difference: 7%

Brandenburgische
Technische Universität
Cottbus - Senftenberg

How can we quantify the dataset bias?

Data
Quantifying dataset bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Statistical parity difference (SPD)
- Disparate impact (DI)
- Smoothed empirical differential (SED)

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

- Statistical parity difference (SPD)
  - difference between the mean probability of favorable outcomes between underprivileged and privileged groups.
  - 0: preferable, <0: bad, >0: better

$$Pr(Y = f|D = \text{unprivileged}) - Pr(Y = f|D = \text{privileged})$$

```
print('Statistical Parity Difference (SPD):\t\t\t%.4f' %\
      metrics_train_ds.statistical_parity_difference())

Statistical Parity Difference (SPD):    -0.0437
```

Data
Quantifying dataset bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Disparate impact (DI):
  - exactly like SPD
  - except it's the ratio not the difference
  - 1: preferable, <1: bad, >1: better

$$\frac{Pr(Y = f|D = \text{unprivileged})}{Pr(Y = f|D = \text{privileged})}$$

```
print('Disparate Impact (DI):\t\t\t\t\t%.4f' %\
      metrics_train_ds.disparate_impact())
```

```
Disparate Impact (DI):          0.9447
```

Data
Quantifying dataset bias

Brandenburgische Technische Universität Cottbus - Senftenberg

- Smoothed empirical differential (SED)
  - SED calculates the differential in the probability of favorable and unfavorable outcomes between intersecting groups divided by features.
  - minimum ratio of smoothed probability for favorable and unfavorable outcomes between the groups
  - between 0 and 1
  - ideal value should be close to 0

```
print('Smoothed Empirical Differential Fairness (SEDF):\t%.4f' %\
      metrics_train_ds.smoothed_empirical_differential_fairness())
```

```
Smoothed Empirical Differential Fairness (SEDF):  0.3514
```

Brandenburgische
Technische Universität
Cottbus - Senftenberg

We have quantified the **data bias**

Now we want to quantify the **model bias**

What is a model bias?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

1. Model **without** bias mitigation

```
import tensorflow as tf
from catboost import CatBoostClassifier
rand = 9
np.random.seed(rand)
tf.random.set_seed(rand)

orig_plt_params = plt.rcParams
sns.set()

cb_mdl = CatBoostClassifier(iterations=500, learning_rate=0.5, depth=8)
fitted_cb_mdl = cb_mdl.fit(X_train, y_train, verbose=False)
y_test_cb_pred = fitted_cb_mdl .predict(X_test)
```

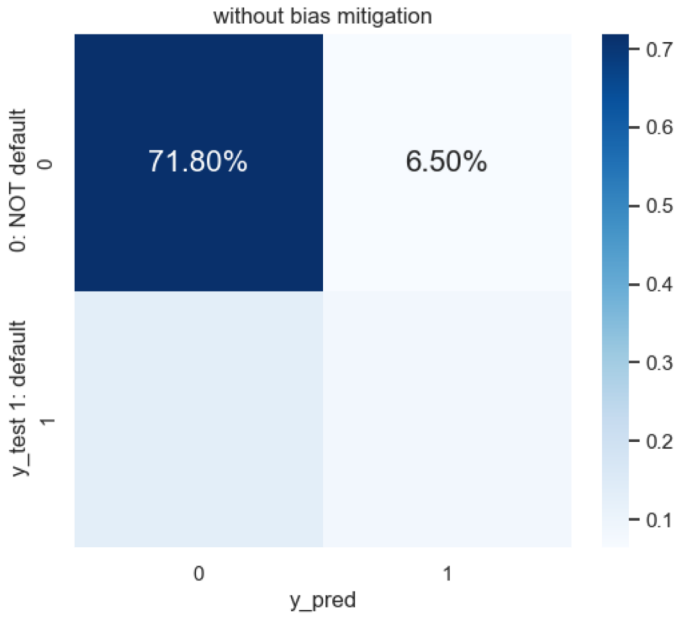
Confusion matrix

```
cf_matrix = metrics.confusion_matrix(y_test,\
                                     y_test_cb_pred)
print("CatBoostClassifier without bias mitigation\n",cf_matrix)
```

Data
Quantifying model bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Result




```
confusion matrix
 [[0.71802605 0.06497601]
 [0.13296779 0.08403016]]
```

Statistical Parity Difference (SPD): -0.0437 (0)
Disparate Impact (DI): 0.9447 (1)
Smoothed Empirical Differential Fairness (SEDF): 0.3514 (0)
Difference in mean outcomes TRAIN between unprivileged and privileged groups in percent -4.37
Difference in mean outcomes TEST between unprivileged and privileged groups in percent -7.01

CatBoostClassifier without bias mitigation
[[5238 474]
[970 613]]

accuracy CatBoostClassifier without bias mitigation
0.8

Data
Quantifying model bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

We want to give larger weights to positive datasets
(in our case: 0 – not default) (model bias)

```
# class_weights = [0.8,0.2]
test_pred_ds = test_ds.copy(deepcopy=True)
cb_mdl = CatBoostClassifier(iterations=500, learning_rate=0.5, depth=8,class_weights = [0.8,0.2])
fitted_cb_mdl = cb_mdl.fit(X_train, y_train, verbose=False)

y_test_cb_pred = fitted_cb_mdl .predict(X_test)


cf_matrix = metrics.confusion_matrix(y_test,\
                                     y_test_cb_pred)
print("POSW CatBoostClassifier\n",cf_matrix)
```

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Results

Standard weights vs. larger weights to positive datasets

CatBoostClassifier without bias mitigation

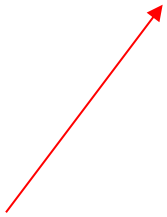
[[5238 474]
[970 613]]

accuracy CatBoostClassifier without bias mitigation
0.8

POSW CatBoostClassifier

[[5447 265]
[1106 477]]

POSW accuracy CatBoostClassifier
0.81



Better accuracy
More persons that repay credits (TP)
Less persons that don't repay credits (TN)

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Results

Standard weights vs. larger weights to positive datasets

Statistical Parity Difference (SPD):
-0.0437
Disparate Impact (DI):
0.9447
Smoothed Empirical Differential Fairness (SEDF):
0.3514
Difference in mean outcomes TEST between
unprivileged and privileged groups in percent
-7.01

POSW Statistical Parity Difference (SPD):
-0.0452
Disparate Impact (DI):
0.9503
Smoothed Empirical Differential Fairness (SEDF):
0.7026
POSW Difference in mean outcomes TEST between
unprivileged and privileged groups in percent
-4.52

SPD: worse (0: preferable, <0: bad, >0: better)
DI: better (1: preferable, <1: bad, >1: better)
SEDF: worse (best: 0)
Better Difference in mean outcomes

Data
Mitigating bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- We have quantified the data bias and (possible) model bias
- How can we mitigate the bias?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Phases for Mitigating Bias

- Preprocessing
- In-processing
- Post-processing

- Bias mitigation methods can hurt predictive performance

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Mitigating bias phases

- Preprocessing:
  - detect and remove bias from the training data before training the model
  - tackle bias at the source

- In-processing:
  - mitigate bias during the model training
  - Are highly dependent on the model
  - require hyperparameter tuning to calibrate fairness metrics
  - Impact on the model
  - -> explanation more difficult

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Mitigating bias phases

- Post-processing:
  - No impact on the model
  - We make **manual adjustments** to achieve parity with false positives
  - Advantage:
    - tackle outcome unfairness
    - where it can have the **greatest impact**
  - Disadvantage:
    - Impact is **disconnected from the rest of the model development**, it can deform things

Data
Pre-processing bias mitigation methods

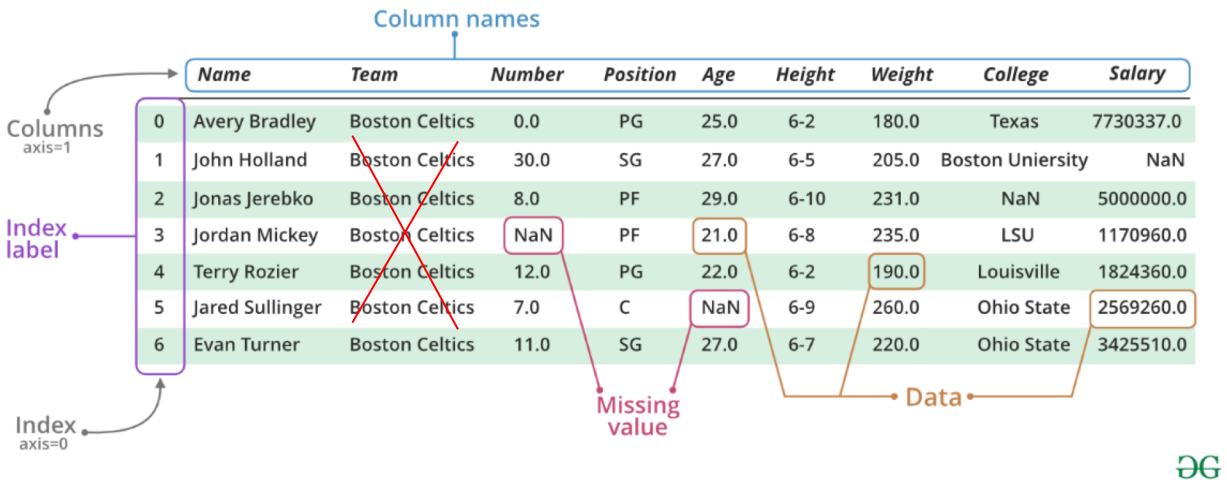
Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Unawareness/Suppression
- Feature Engineering
- Balancing/Resampling
- Relabelling
- Learning fair representation
- Optimization pre-processing for discrimination prevention
- Reweighting
- Disparate impact remover

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Unawareness/Suppression

- remove biased features from the dataset



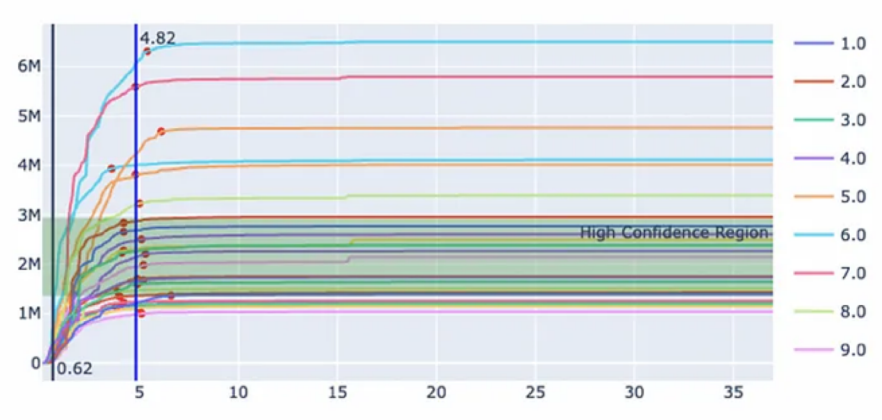
	Name	Team	Number	Position	Age	Height	Weight	College	Salary
0	Avery Bradley	Boston Celtics	0.0	PG	25.0	6-2	180.0	Texas	7730337.0
1	John Holland	Boston Celtics	30.0	SG	27.0	6-5	205.0	Boston Uniersity	NaN
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0	6-10	231.0	NaN	5000000.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0	6-8	235.0	LSU	1170960.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0	6-2	190.0	Louisville	1824360.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN	6-9	260.0	Ohio State	2569260.0
6	Evan Turner	Boston Celtics	11.0	SG	27.0	6-7	220.0	Ohio State	3425510.0

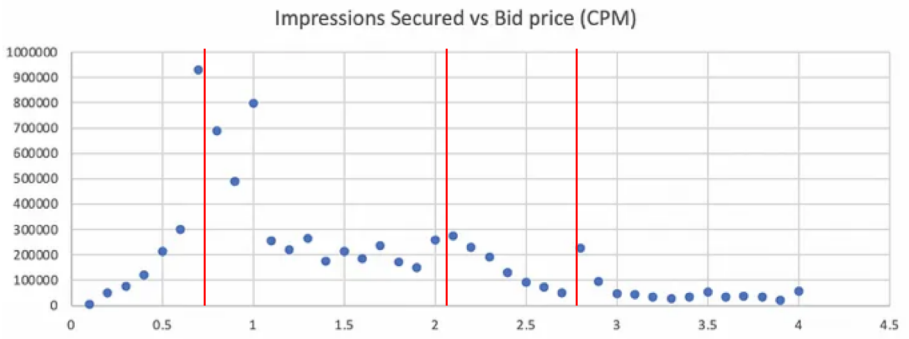
Data
Pre-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Feature Engineering

- Use guardrails for monotonicity





Data

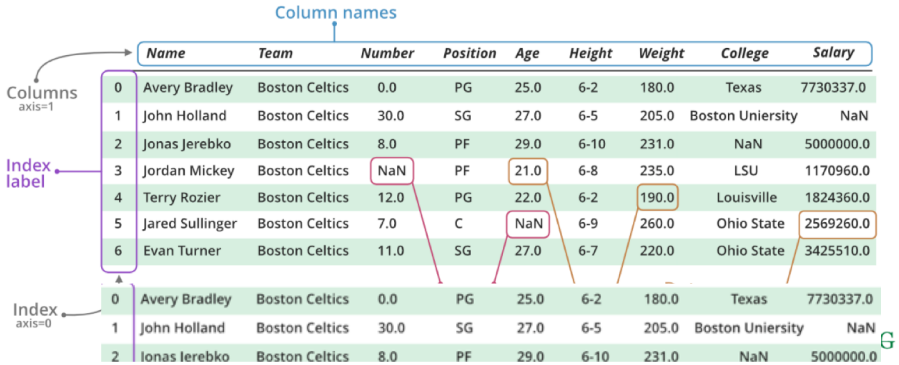
Pre-processing bias mitigation methods

Brandenburgische Technische Universität Cottbus - Senftenberg

Balancing/Resampling

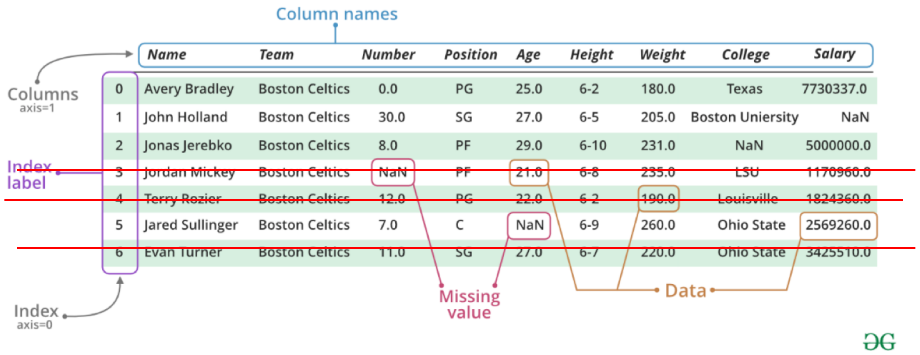
Upsampling

- Creates a certain amount



Downsampling

- just taking a certain perce



It's always preferable to downsample than upsample if we have enough data

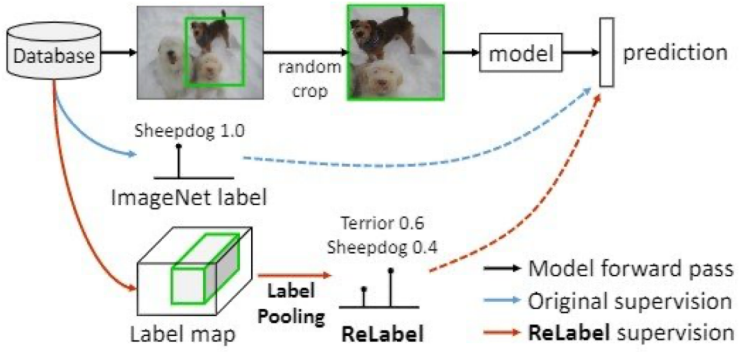
Brandenburgische
Technische Universität
Cottbus - Senftenberg

Relabelling/Massaging

- changes the labels for observations that appear to be most biased by ranking

e.g.

- change IS_DEFAULT for the best women 1-> 0
- Text label bad -> good
- useimage sections



Data
Pre-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Learning fair representations

the proportion of members in a protected group receiving positive classification is identical to the proportion in the population as a whole

individual fairness (similar individuals should be treated similarly)

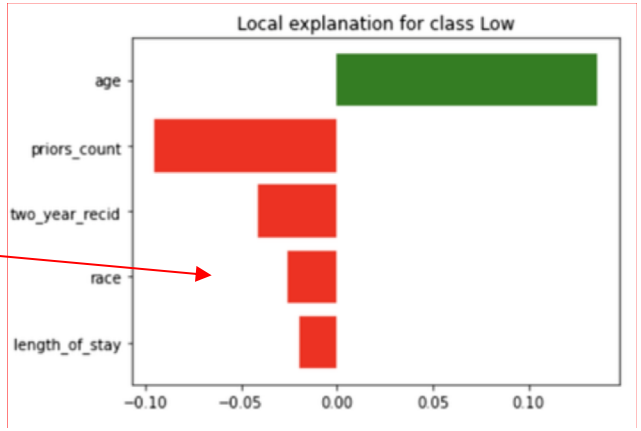
Data
Pre-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Optimized preprocessing for discrimination prevention `aif360.algorithms.preprocessing.OptimPreproc`

- learns a probabilistic transformation that
  - edits the features and
  - labels

impact -> 0



F. P. Calmon, D. Wei, B. Vinzamuri, K. Natesan Ramamurthy, and K. R. Varshney. “Optimized Pre-Processing for Discrimination Prevention.” Conference on Neural Information Processing Systems, 2017.

Data
Pre-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Reweighting

- Gives new weight for each observation

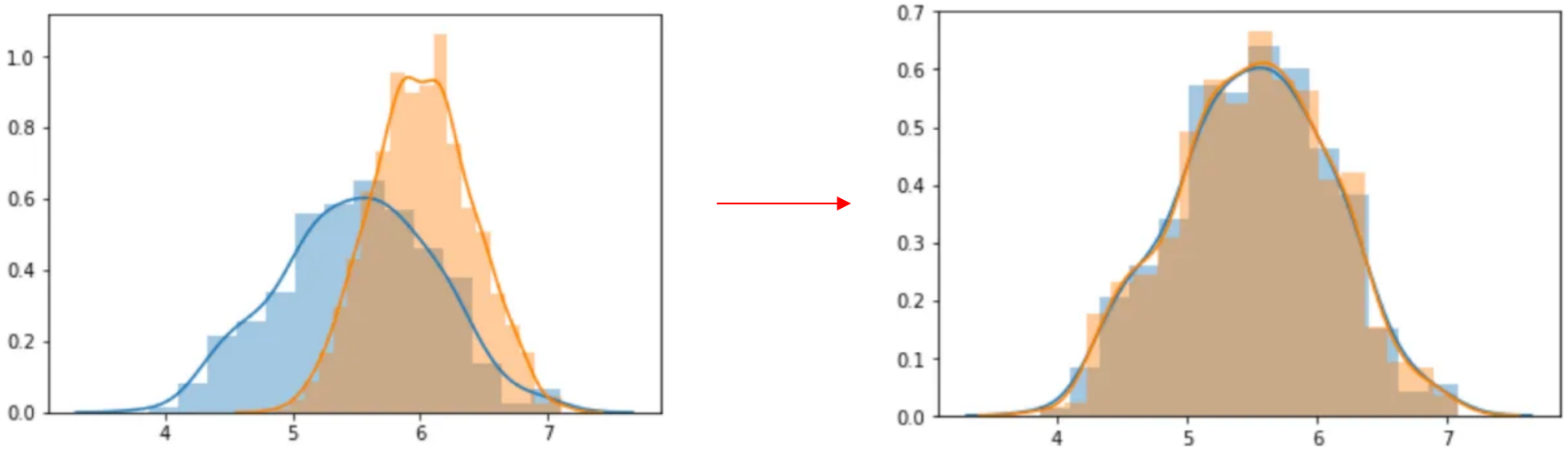
		...		
1.0	1836.0	...	0.280160	1.0
1.0	27014.0	...	0.015600	1.0
1.0	388.0	...	0.789300	1.0
1.0	11077.0	...	0.374805	0.0
1.0	28424.0	...	0.370619	0.0
		...		
1.051049	1836.0	...	0.280160	1.0
1.051049	27014.0	...	0.015600	1.0
0.870666	388.0	...	0.789300	1.0
0.986366	11077.0	...	0.374805	0.0
0.986366	28424.0	...	0.370619	0.0

Data
Pre-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Disparate impact remover

- edits values, which will be used as features, to increase fairness between the groups



Brandenburgische
Technische Universität
Cottbus - Senftenberg

Transformation using reweighting()

reweighting: weights the examples in each (group, label) combination differently to ensure fairness before classification

```
reweigher = reweighting(unprivileged_groups=unprivileged_groups,\
                        privileged_groups=privileged_groups)


reweigher.fit(train_ds)
train_rw_ds = reweigher.transform(train_ds)
```

Data

Pre-processing bias mitigation methods

reweighting

Brandenburgische Technische Universität Cottbus - Senftenberg

train_ds

	instance weights	features	...	labels
			...	
		ID	...	bill1_over_limit
instance names			...	
1835	1.0	1836.0	...	1.0
27013	1.0	27014.0	...	1.0
387	1.0	388.0	...	1.0
11076	1.0	11077.0	...	0.0
28423	1.0	28424.0	...	0.0
...	...	...	...	...
23208	1.0	23209.0	...	0.0
517	1.0	518.0	...	0.0
23804	1.0	23805.0	...	0.0
21393	1.0	21394.0	...	1.0
25647	1.0	25648.0	...	1.0

	instance weights	features	...	labels
			...	
		ID	...	bill1_over_limit
instance names			...	
1835	1.051049	1836.0	...	1.0
27013	1.051049	27014.0	...	1.0
387	0.870666	388.0	...	1.0
11076	0.986366	11077.0	...	0.0
28423	0.986366	28424.0	...	0.0
...	...	...	...	...
23208	0.986366	23209.0	...	0.0
517	0.986366	518.0	...	0.0
23804	1.044141	23805.0	...	0.0
21393	0.870666	21394.0	...	1.0
25647	1.051049	25648.0	...	1.0

[21883 rows x 21 columns]

Before reweighting

train_ds

```
train_ds =
    BinaryLabelDataset(df=X_tr
    ain.join(y_train),\
    label_names=['IS_DEFAUL
    T'],\
    protected_attribute_names=
    ['AGE_GROUP',
    'GENDER'],
    favorable_label=0,
    unfavorable_label=1)
```

After reweighting

train_rw_ds

Data

Pre-processing bias mitigation methods

Reweighting

Brandenburgische Technische Universität Cottbus - Senftenberg

instance names		ID	CC_LIMIT_CAT	EDUCATION	
1835	1.0	1836.0	1.0	2.0	
27013	1.0	27014.0	1.0	2.0	
387	1.0	388.0	2.0	1.0	
11076	1.0	11077.0	5.0	3.0	
28423	1.0	28424.0	5.0	3.0	
...	...	...	...	...	
23208	1.0	23209.0	4.0	0.0	
517	1.0	518.0	1.0	2.0	
23804	1.0	23805.0	2.0	1.0	
21393	1.0	21394.0	5.0	3.0	
25647	1.0	25648.0	3.0	2.0	

instance names		ID	CC_LIMIT_CAT	EDUCATION	
1835	1.075665	1836.0	1.0	2.0	
27013	0.902960	27014.0	1.0	2.0	
387	0.902960	388.0	2.0	1.0	
11076	0.980374	11077.0	5.0	3.0	
28423	0.980374	28424.0	5.0	3.0	
...	...	...	...	...	
23208	0.980374	23209.0	4.0	0.0	
517	1.031549	518.0	1.0	2.0	
23804	0.980374	23805.0	2.0	1.0	
21393	0.902960	21394.0	5.0	3.0	
25647	1.075665	25648.0	3.0	2.0	

CC_LIMIT_CAT

Before reweighting

train_ds

```
train_ds =
    BinaryLabelDataset(df=X_tr
    ain.join(y_train),\
    label_names=['IS_DEFAUL
    T'],\
    protected_attribute_names=
    ['AGE_GROUP',
    'GENDER'],
    favorable_label=0,
    unfavorable_label=1)
```

After reweighting

train_rw_ds

Brandenburgische
Technische Universität
Cottbus - Senftenberg

What is the impact of reweighting on our Data?

Brandenburgische
Technische Universität
Cottbus - Senftenberg

What is the impact of reweighting on our data?

```
metrics_train_rw_ds = BinaryLabelDatasetMetric(train_rw_ds,\
                              unprivileged_groups=unprivileged_groups,\
                              privileged_groups=privileged_groups)


print('Reweigher (REW) Statistical Parity Difference (SPD):\t\t\t%.4f' %\
      metrics_train_rw_ds.statistical_parity_difference())
print('REW Disparate Impact (DI):\t\t\t\t%.4f' %\
      metrics_train_rw_ds.disparate_impact())
print('REW Smoothed Empirical Differential Fairness (SEDF):\t%.4f' %\
      metrics_train_rw_ds.smoothed_empirical_differential_fairness())

np.abs(train_ds.instance_weights.mean() - train_rw_ds.instance_weights.mean()) < 1e-6
```

Row data

Statistical Parity Difference (SPD):	-0.0437 (0)
Disparate Impact (DI):	0.9447 (1)
Smoothed Empirical Differential Fairness (SEDF):	0.3514 (0)

Reweigher (REW)

Statistical Parity Difference (SPD):	-0.0000 (0)
REW Disparate Impact (DI):	1.0000 (1)
REW Smoothed Empirical Differential Fairness (SEDF):	0.1942 (0)

Brandenburgische
Technische Universität
Cottbus - Senftenberg

What is the impact of reweighting on our Prediction?

Data

Pre-processing bias mitigation methods

Reweighting

Brandenburgische Technische Universität Cottbus - Senftenberg

What is the impact of reweighting on our prediction?

We initialize our model with the **new weights** from `reweigher`

```
cb_mdl = CatBoostClassifier(iterations=500, learning_rate=0.5, depth=8)
fitted_cb_mdl = cb_mdl.fit(X_train, y_train, verbose=False,
                           sample_weight=train_rw_ds.instance_weights)
y_test_cb_pred = fitted_cb_mdl .predict(X_test)


cf_matrix = metrics.confusion_matrix(y_test,\
                                     y_test_cb_pred)
```

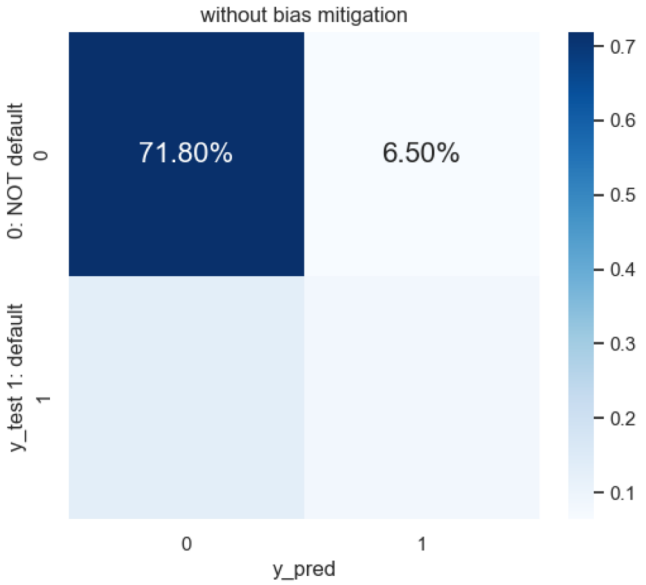
Data

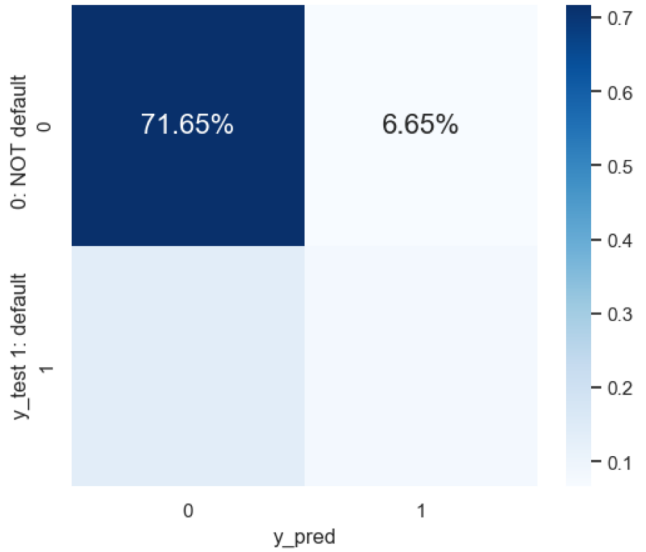
Pre-processing bias mitigation methods

Reweighting

Brandenburgische Technische Universität Cottbus - Senftenberg

Results without and with reweighting





CatBoostClassifier without bias mitigation
accuracy CatBoostClassifier 0.8

[[5238 474]
[970 613]]

confusion matrix
[[0.71802605 0.06497601]
[0.13296779 0.08403016]]

11 persons less get a credit (TN)
27 persons less get a credit that they do not repay (TP)

confusion matrix
[[0.71651816 0.06648389]
[0.13666895 0.08032899]]

REW CatBoostClassifier
[[5227 485]
[997 586]]

REW accuracy CatBoostClassifier
0.8

Bias Mitigation Bias_Mitigation.py

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Results original data vs. reweighting

Statistical Parity Difference (SPD):
-0.0437
Disparate Impact (DI):
0.9447
Smoothed Empirical Differential Fairness (SEDF):
0.3514
Difference in mean outcomes TEST between unprivileged and privileged groups in percent
-7.01

REW Statistical Parity Difference (SPD): -0.0285
REW Disparate Impact (DI):
0.9671
REW Smoothed Empirical Differential Fairness (SEDF):
0.4888
REW Difference in mean outcomes TEST between unprivileged and privileged groups in percent
-2.85

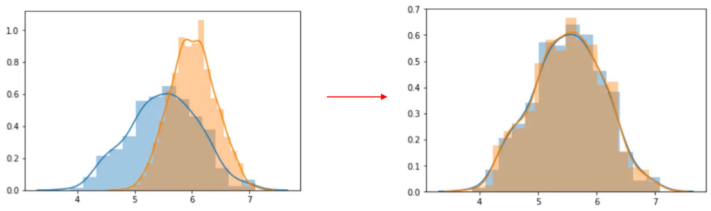
SPD: worse (0: preferable, <0: bad, >0: better)
DI: better (1: preferable, <1: bad, >1: better)
SEDF: worse (best: 0)
Better Difference in mean outcomes (less 4 %)
-> fairer

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Disparate impact remover method

AGE_GROUP

- edits features values
- adjusts the features -> equal distribution across all protected groups
- We delete the protected feature AGE_GROUP during the process
- save the protected feature
- at the end, we will need its index



Data

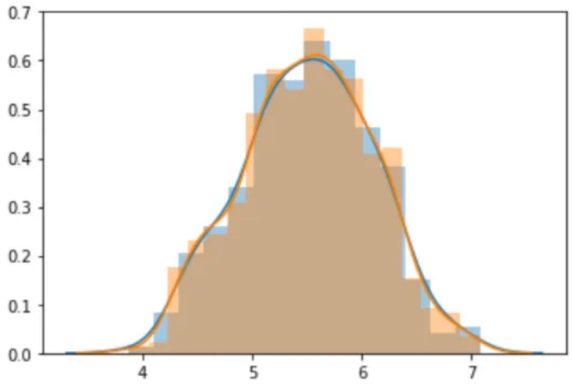
Pre-processing bias mitigation methods

Disparate impact remover method

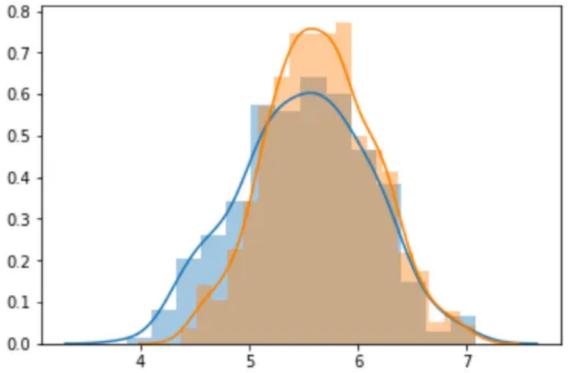
Brandenburgische Technische Universität Cottbus - Senftenberg

- **Disparate impact remover** requires a **repair level** between zero and one,
- -> how much you wish for the distributions of the groups to overlap
- so we need to find the optimal one

Repair value = 1.0



Repair value = 0.8



Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

- requires a repair level between zero and one, so we need to find the optimal one

```
protected_index = train_ds.feature_names.index('AGE_GROUP')
```

protected feature

```
di = np.array([])
train_dir_ds = None
test_dir_ds = None
lgb_dir_mdl = None
X_train_dir = None
X_test_dir = None
```

np.linespace(sart, end, number steps):

```
levels = np.hstack([np.linspace(0., 0.1, 41), np.linspace(0.2, 1, 9)])
```

```
a
 [0.     0.0025 0.005  0.0075 0.01   0.0125 0.015  0.0175 0.02   0.0225
 0.025  0.0275 0.03   0.0325 0.035  0.0375 0.04   0.0425 0.045  0.0475
 0.05   0.0525 0.055  0.0575 0.06   0.0625 0.065  0.0675 0.07   0.0725
 0.075  0.0775 0.08   0.0825 0.085  0.0875 0.09   0.0925 0.095  0.0975
 0.1   ]
b
 [0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1. ]
```

Disparate_Impact.py

Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

DI is a ratio, so it's a metric: 1: best, < 1, >1
-> iterate across different repair levels
continuously save the model whose DI is closest to one

Iterates between 2 values

Iteration to find the best Remover

```
for level in tqdm(levels):
    di_remover = DisparateImpactRemover(repair_level=level)
    train_dir_ds_i = di_remover.fit_transform(train_ds)
    test_dir_ds_i = di_remover.fit_transform(test_ds)

    X_train_dir_i = np.delete(train_dir_ds_i.features, protected_index,
axis=1)
    X_test_dir_i = np.delete(test_dir_ds_i.features, protected_index, axis=1)

    lgb_dir_mdl_i = CatBoostClassifier(iterations=500, learning_rate=0.5,
depth=8)
    lgb_dir_mdl_i.fit(X_train_dir_i, train_dir_ds_i.labels)

    test_dir_ds_pred_i = test_dir_ds_i.copy()
    test_dir_ds_pred_i.labels = lgb_dir_mdl_i.predict(X_test_dir_i)

    ...
```

fit: de-bias the data

train the model without the protected feature

Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

```
for level in tqdm(levels):

...

    metrics_test_dir_ds = BinaryLabelDatasetMetric(test_dir_ds_pred_i,\
                                  unprivileged_groups=unprivileged_groups,\
                                  privileged_groups=privileged_groups)
    di_i = metrics_test_dir_ds.disparate_impact()

    if (di.shape[0]==0) or (np.min(np.abs(di-1)) >= abs(di_i-1)):
        # print(abs(di_i-1))
        train_dir_ds = train_dir_ds_i
        test_dir_ds = test_dir_ds_i
        X_train_dir = X_train_dir_i
        X_test_dir = X_test_dir_i
        lgb_dir_mdl = lgb_dir_mdl_i

    di = np.append(np.array(di), di_i)
```

optimal DI is closest to one.
Therefore, as we iterate across different repair levels,
we will continuously save the model whose DI is closest to one

np.min(np.abs(di-1))
0.05386714351424127
abs(di_i-1)
0.055226288338036555

Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

```
print("di \n",di)
print("levels\n",levels)
#best values
```

→

```
di
 [0.93433017 0.94152953 0.94027652 0.94133757 0.93292371]
levels
 [0.   0.05 0.1  0.2  1.  ]
```

```
metrics_test_dir_ds = BinaryLabelDatasetMetric(test_dir_ds_pred,\
                                unprivileged_groups=unprivileged_groups,\
                                privileged_groups=privileged_groups)
test_dir_ds_pred_i.labels = lgb_dir_mdl.predict(X_test_dir)

print('DPI Statistical Parity Difference (SPD):\t\t\t%.4f' %\
      metrics_test_dir_ds.statistical_parity_difference())
print('DPI Disparate Impact (DI):\t\t\t\t%.4f' %\
      metrics_test_dir_ds.disparate_impact())
print('DPI Smoothed Empirical Differential Fairness (SEDF):\t%.4f' %\
      metrics_test_dir_ds.smoothed_empirical_differential_fairness())

print ("DPI Difference in mean outcomes TEST between unprivileged and
privileged groups in percent" ,
round(metrics_test_dir_ds.mean_difference()*100,2))
```

Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

With bias

DPI Statistical Parity Difference (SPD): -0.0549 (0)
DPI Disparate Impact (DI): 0.9375 (1)
DPI Smoothed Empirical Differential Fairness (SEDF): 0.7036 (0)
DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent -5.49

After Disparate Impact Remover

DPI Statistical Parity Difference (SPD): -0.0525
DPI Disparate Impact (DI): 0.9415
DPI Smoothed Empirical Differential Fairness (SEDF): 0.6636
DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent -5.25

```
levels = np.hstack([np.linspace([np.linspace(0., 0.1, 3), np.linspace(0.2, 1, 2)])
->
levels
 [0.   0.05 0.1  0.2  1.  ]
```

Data

Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

After Disparate Impact Remover

levels = [0. 0.05 0.1 0.2 1.]
DPI Statistical Parity Difference (SPD): -0.0525 (0)
DPI Disparate Impact (DI): 0.9415 (1)
DPI Smoothed Empirical Differential Fairness (SEDF): 0.6636 (0)
DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent -5.25

```
levels = np.hstack([np.linspace(0., 0.1, 41), np.linspace(0.2, 1, 9)])
->
levels =
 [0.    0.0025 0.005  0.0075 0.01   0.0125 0.015  0.0175 0.02   0.0225
 0.025  0.0275 0.03   0.0325 0.035  0.0375 0.04   0.0425 0.045  0.0475
 0.05   0.0525 0.055  0.0575 0.06   0.0625 0.065  0.0675 0.07   0.0725
 0.075  0.0775 0.08   0.0825 0.085  0.0875 0.09   0.0925 0.095  0.0975
 0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9    1.   ]
```

DPI Statistical Parity Difference (SPD): -0.0497
DPI Disparate Impact (DI): 0.9445
DPI Smoothed Empirical Differential Fairness (SEDF): 0.6800
DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent -4.97

Data

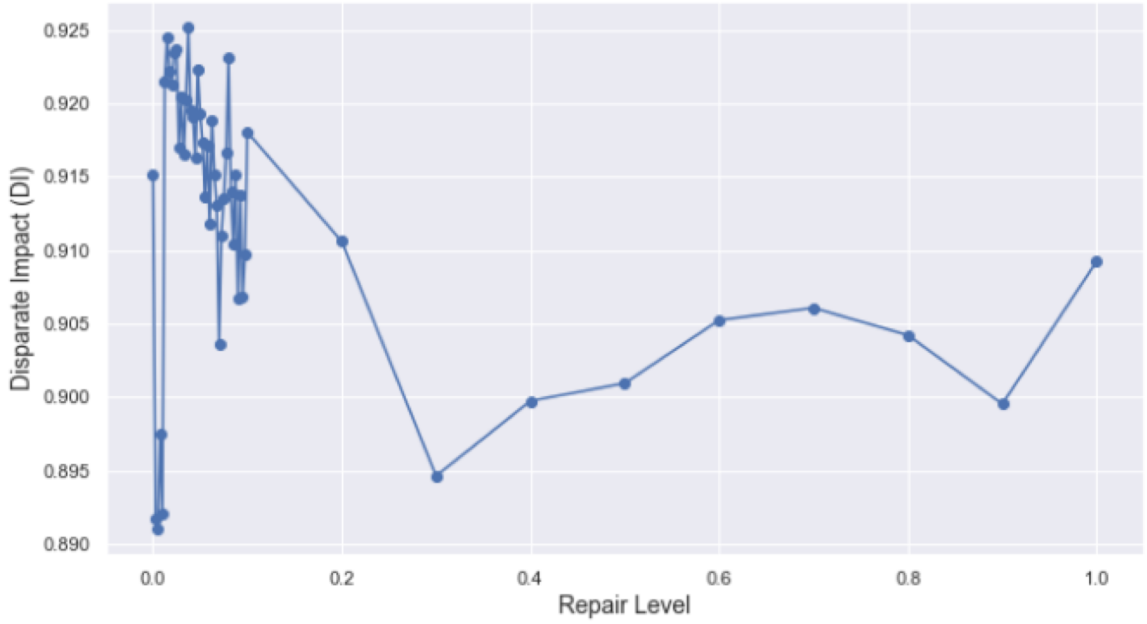
Pre-processing bias mitigation methods

Disparate impact remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

```
levels = np.hstack([np.linspace(0., 0.1, 41), np.linspace(0.2, 1, 9)])
->
levels =
 [0.     0.0025 0.005  0.0075 0.01   0.0125 0.015  0.0175 0.02   0.0225
 0.025  0.0275 0.03   0.0325 0.035  0.0375 0.04   0.0425 0.045  0.0475
 0.05   0.0525 0.055  0.0575 0.06   0.0625 0.065  0.0675 0.07   0.0725
 0.075  0.0775 0.08   0.0825 0.085  0.0875 0.09   0.0925 0.095  0.0975
 0.1    0.2    0.3    0.4    0.5    0.6    0.7    0.8    0.9    1.    ]
```

DPI Disparate Impact (DI): 0.9415 (1)



Disparate_Impact.py

Model
In-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Pre-precessing Methods

In-processing Methods

Post-processing Methos

Model
In-processing bias mitigation methods

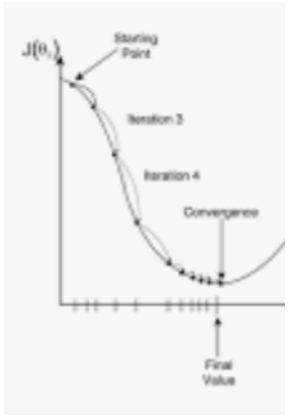
Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Cost-sensitive training
- Constraints
- Adversarial debiasing
- Exponentiated gradient reduction
- Prejudice remover regularizer
- Gerry fair classifier

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Cost-sensitive training

- takes the costs of prediction errors
- + `scale_pos_weight`, `class_weights` or `sample_weight`



+

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Constraints

- ensure that relationships between features and targets are restricted to a certain pattern,
- placing guardrails at the model level
  - Ordinalisation
  - Discretization
  - Interaction terms

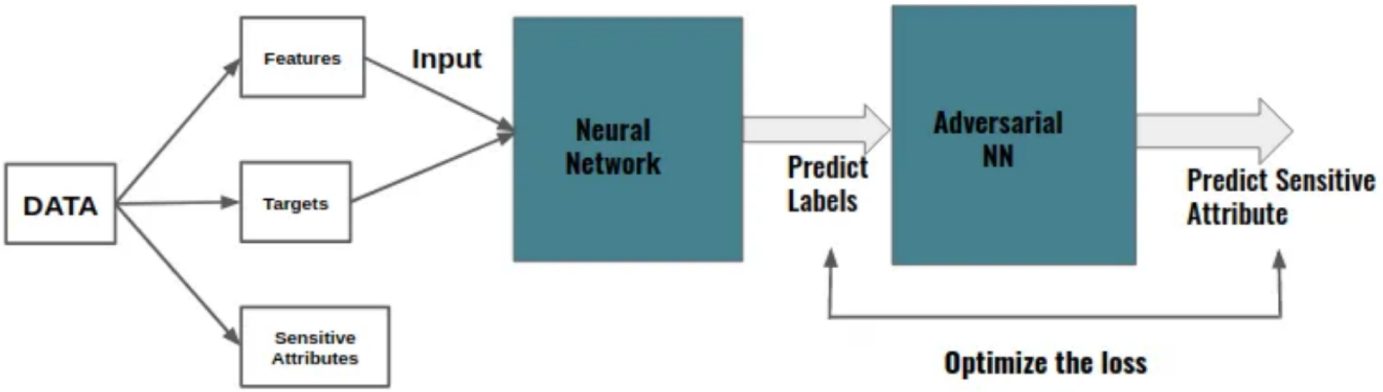
(F3)	6555
(M1)	2632
(F2)	857
(M2)	768
(F1)	131
(F7)	104
(MO3)	76
(F5)	7
(F6)	5
(NI0)	4
(CO3)	2
(TCX)	1

1
2
3
4
5
6
7
8
9
10
11
12

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Adversarial debiasing

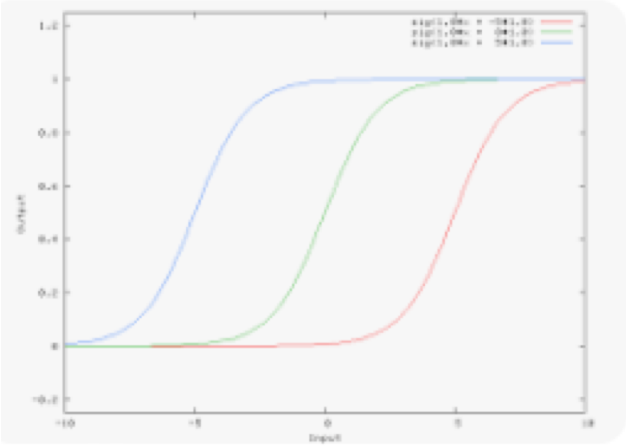
- model 1: normal prediction
- model 2: adversary model – input: output of model 1
  - predict sensitive feature
- minimize the loss of the prediction of both networks
- (similar to autoencoder)



Brandenburgische
Technische Universität
Cottbus - Senftenberg

Exponentiated gradient reduction

- uses demographic parity or equalized odds
- automates cost-sensitive optimization
- Adjusting Predictions:
  - changes weight of biased features

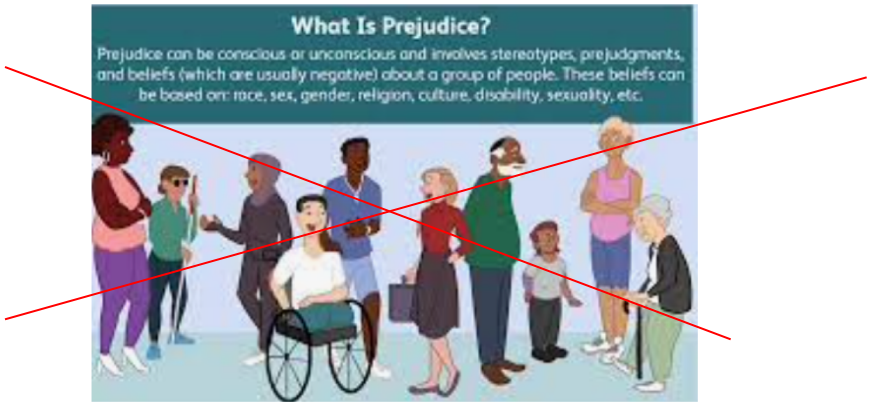


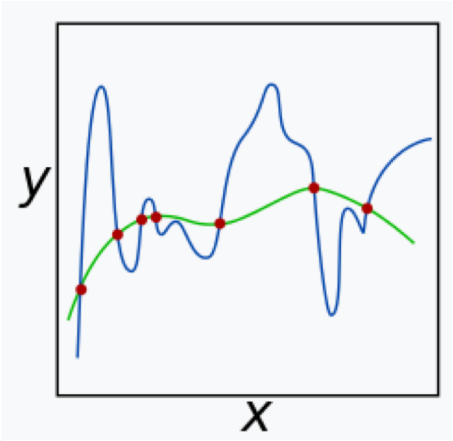
Model
In-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Prejudice remover regularizer

- defines prejudice:
  - statistical dependence between the sensitive and target variables
- Quantifying a prejudice index (PI)
- L2 (Ridge Regression)
  - PI are integrated into a custom regularization term



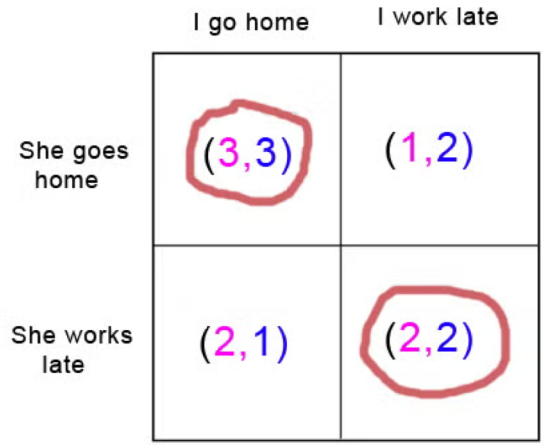


Brandenburgische
Technische Universität
Cottbus - Senftenberg

Gerry fair classifier

(From gerrymandering - Election manipulation)

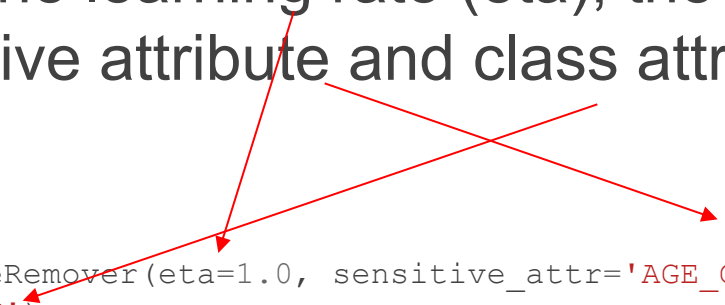
- fairness when subdivided into subgroups
- Nash equilibrium
  - two non-cooperative players with possibly contradictory aims reach a solution that partially satisfies both.



Brandenburgische
Technische Universität
Cottbus - Senftenberg

Prejudice Remover method

- Quantifying a prejudice index (PI)
- L2 (Ridge Regression)
  - PI are integrated into a custom regularization term.
- Initialisation with the learning rate (eta), the specific the sensitive attribute and class attribute (target).



```
log_pr_mdl = PrejudiceRemover(eta=1.0, sensitive_attr='AGE_GROUP',
class_attr='IS_DEFAULT')
```

Model

In-processing bias mitigation methods

Prejudice remover method

Brandenburgische Technische Universität Cottbus - Senftenberg

```
log_pr_mdl = PrejudiceRemover(eta=1.0, sensitive_attr='AGE_GROUP',
class_attr='IS_DEFAULT')
log_pr_mdl.fit(train_ds)
```

```
train_pred_pr_ds = log_pr_mdl.predict(train_ds)
test_pred_pr_ds = log_pr_mdl.predict(test_ds)
```

Result

DPI Statistical Parity Difference (SPD):	0.0480 (0)
DPI Disparate Impact (DI):	1.7315 (1) better than <1
DPI Smoothed Empirical Differential Fairness (SEDF):	0.8431 (0)

DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent 4.8

Model

In-processing bias mitigation methods

Gerry fair classifier method

Brandenburgische Technische Universität Cottbus - Senftenberg

Gerry fair classifier

- uses Nash equilibrium
- It only supports
  - linear models,
  - support vector machines (SVMs),
  - kernel regression
  - decision trees

SP: Statistical parity (Demographic Parity)
FP: False Positive -> 0
FN: False Negative -> 0

C: regularization strength
gamma: Fairness Approximation Paramater
fairness_def: approximation for early stopping

```
dt_gf_mdl = GerryFairClassifier(C=100, gamma=.005, fairness_def='FN',\
                                max_iters=50, printflag=True,\
                                predictor=tree.DecisionTreeRegressor(max_depth=3))
```

Model

In-processing bias mitigation methods

Gerry fair classifier method

Brandenburgische Technische Universität Cottbus - Senftenberg

```
dt_gf_mdl.fit(train_ds, early_termination=True)

train_pred_gf_ds = dt_gf_mdl.predict(train_ds, threshold=False)
test_pred_gf_ds = dt_gf_mdl.predict(test_ds, threshold=False)
```

Result

DPI Statistical Parity Difference (SPD):	-0.0548 (0)
DPI Disparate Impact (DI):	0.9388 (1)
DPI Smoothed Empirical Differential Fairness (SEDF):	0.6429 (0)

DPI Difference in mean outcomes TEST between unprivileged and privileged groups in percent -5.48

Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Pre-precessing Methods

In-processing Methods

Post-processing Methos

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Post-processing bias mitigation methods

- Prediction abstention
- Reject option classification
- Equalized odds postprocessing
- Calibrated equalized odds postprocessing

Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Prediction abstention

Imagine

- to answer a difficult math question
- If you're not sure about the answer
- -> to abstain from answering rather than providing a potentially incorrect response.
- prediction abstention:
  - the model chooses not to make a prediction when it's uncertain about its accuracy.

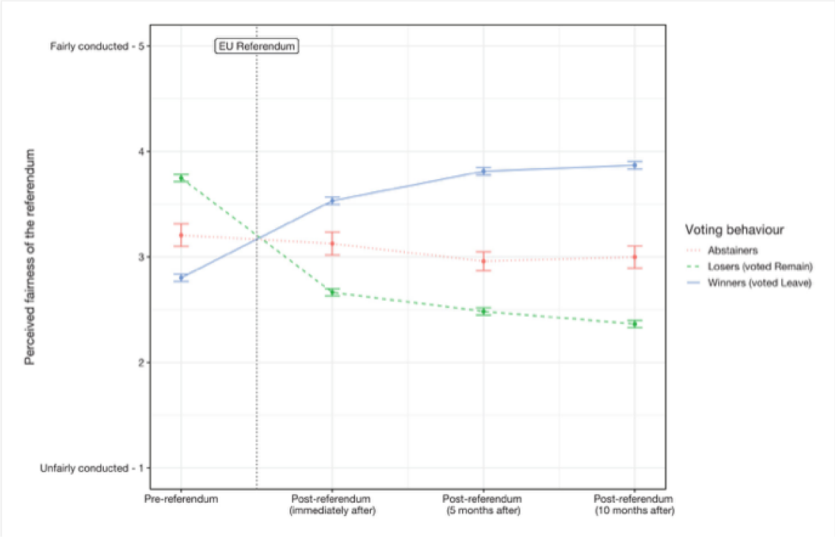
.

Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Prediction abstention

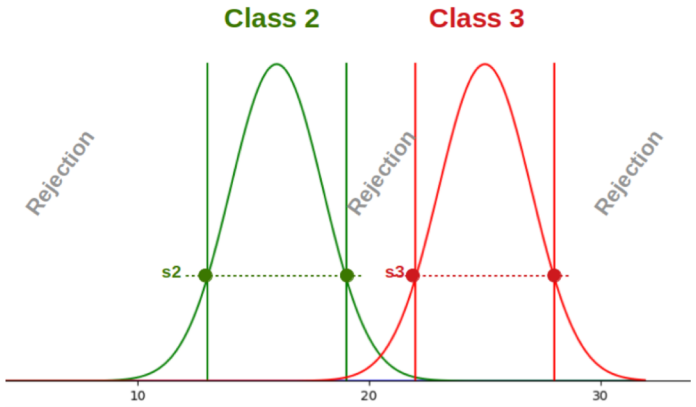
- measures the certainty of its decision
- If **confidence for a prediction <= threshold**
- -> **not to make a prediction** (abstains from providing an answer)
- **Consider fairness metrics when abstaining from predictions**
- avoid harmful or misleading predictions



Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Reject option classification



- similar to Prediction Abstention
- Confidence Threshold
- Confidence < threshold
  - -> "rejects" the prediction

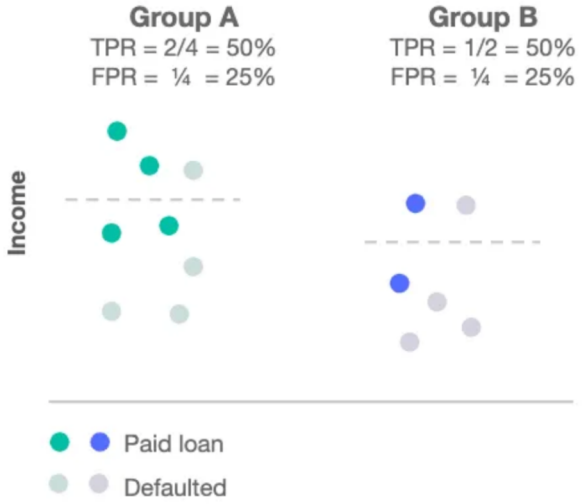
different rejection thresholds for different groups defined by sensitive attributes

Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Equalized odds postprocessing

- starts with prediction
- **predictive errors** are **balanced** across different groups
- Define groups by a protected attribute

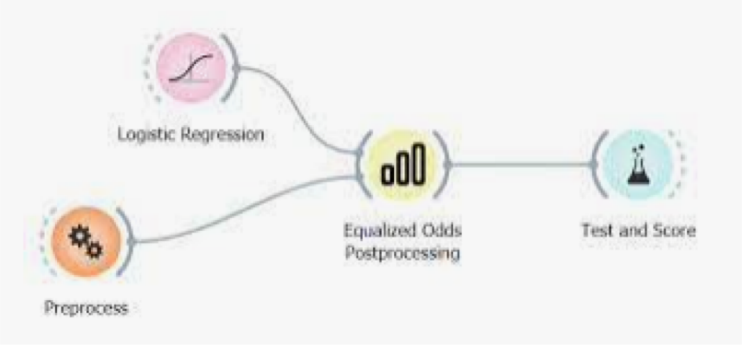


Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Equalized odds postprocessing

- adjusts the predictions:
  - -> **FPR and FNR** (wrongly classifying) approximately **equal** across different groups
- Balancing conflicting objectives:
  - modifying the predicted probabilities
  - -> might some loss in overall accuracy

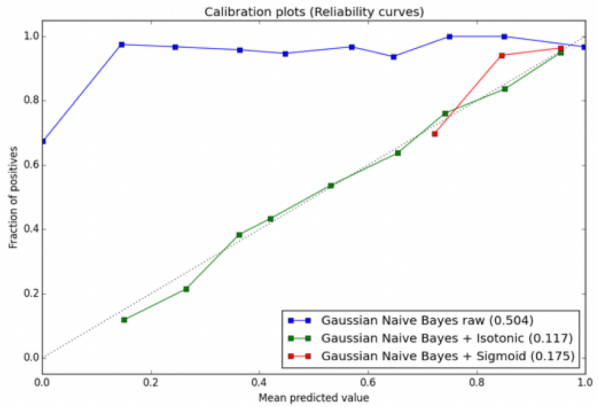


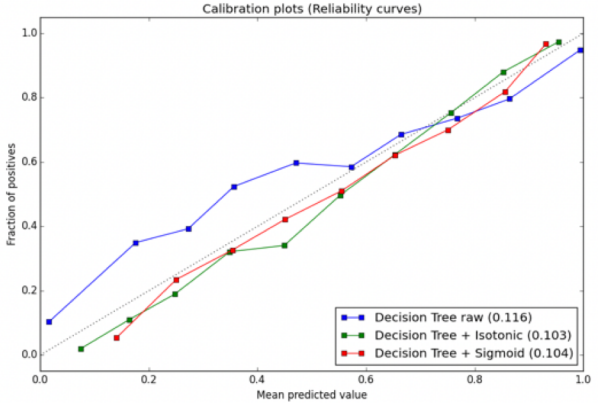
Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Calibrated equalized odds postprocessing

- Similar to Equalized Odds Postprocessing
- extra step to ensure
- -> predicted probabilities are well-calibrated.
-





Model
Post-processing bias mitigation methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Implement two Post-processing bias mitigation methods

- Equalized odds postprocessing
- Calibrated equalized odds postprocessing method

Brandenburgische Technische Universität Cottbus - Senftenberg

Equalized odds postprocessing

1. Normal classification

```
from catboost import CatBoostClassifier
cb_mdl = CatBoostClassifier(iterations=500, learning_rate=0.5, depth=8)
fitted_cb_mdl = cb_mdl.fit(X_train, y_train, verbose=False)
y_test_cb_pred = fitted_cb_mdl .predict(X_test)
```

2. Create BinaryLabelDatasets for test and predicted data

```
test_ds = BinaryLabelDataset(df=X_test.join(y_test),\
                    label_names=['IS_DEFAULT'],\
                    protected_attribute_names=['AGE_GROUP', 'GENDER'],
                    favorable_label=0, unfavorable_label=1)

X_posw = X_test.copy()
X_posw['IS_DEFAULT'] = y_test_cb_pred

y_test_cb_pred = BinaryLabelDataset(df=X_posw,\
                    label_names=['IS_DEFAULT'],\
                    protected_attribute_names=['AGE_GROUP', 'GENDER'],
                    favorable_label=0, unfavorable_label=1)
```

Model
Post-processing bias mitigation methods
Equalized odds postprocessing

Brandenburgische Technische Universität Cottbus - Senftenberg

3. EqOddsPostprocessing() is initialized with the groups we want to equalize odds

```
epp = EqOddsPostprocessing(privileged_groups=privileged_groups,\
                           unprivileged_groups=unprivileged_groups,\
                           seed=rand)
```

4. fit() needs two datasets: the original one (test_ds) and the dataset with predictions (test_pred_cb_ds).

```
epp = epp.fit(test_ds, y_test_cb_pred)
test_pred_epp_ds = epp.predict(y_test_cb_pred)
```

Model
Post-processing bias mitigation methods
Equalized odds postprocessing

Brandenburgische Technische Universität Cottbus - Senftenberg

Result

```
metrics_test_dir_ds = BinaryLabelDatasetMetric(test_pred_epp_ds,\
                                unprivileged_groups=unprivileged_groups,\
                                privileged_groups=privileged_groups)

print('EPP Statistical Parity Difference (SPD):\t\t\t%.4f' %\
      metrics_test_dir_ds.statistical_parity_difference())
print('EPP Disparate Impact (DI):\t\t\t\t%.4f' %\
      metrics_test_dir_ds.disparate_impact())
print('EPP Smoothed Empirical Differential Fairness (SEDF):\t%.4f' %\
      metrics_test_dir_ds.smoothed_empirical_differential_fairness())

print ("EPP Difference in mean outcomes TEST between unprivileged and privileged groups
in percent" ,round(metrics_test_dir_ds.mean_difference()*100,2))
```

EPP Statistical Parity Difference (SPD): -0.0195 (0)
EPP Disparate Impact (DI): 0.9774 (1)
EPP Smoothed Empirical Differential Fairness (SEDF): 0.3254 (0)
EPP Difference in mean outcomes TEST between unprivileged and privileged groups in percent -1.95

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Calibrated equalized odds postprocessing method

1. Normal classification

```
from catboost import CatBoostClassifier
cb_mdl = CatBoostClassifier(iterations=500, learning_rate=0.5, depth=8)
fitted_cb_mdl = cb_mdl.fit(X_train, y_train, verbose=False)
y_test_cb_pred = fitted_cb_mdl .predict(X_test)
```

2. Create Binary Datasets for test and predicted data

```
test_ds = BinaryLabelDataset(df=X_test.join(y_test),\
                    label_names=['IS_DEFAULT'],\
                    protected_attribute_names=['AGE_GROUP', 'GENDER'],
                    favorable_label=0, unfavorable_label=1)

X_posw = X_test.copy()
X_posw['IS_DEFAULT'] = y_test_cb_pred

y_test_cb_pred = BinaryLabelDataset(df=X_posw,\
                    label_names=['IS_DEFAULT'],\
                    protected_attribute_names=['AGE_GROUP', 'GENDER'],
                    favorable_label=0, unfavorable_label=1)
```

Model
Post-processing bias mitigation methods
Calibrated equalized odds postprocessing method

Brandenburgische
Technische Universität
Cottbus - Senftenberg

3. CalibratedEqOddsPostprocessing() needs the attribute cost_constraint

```
cpp = CalibratedEqOddsPostprocessing(privileged_groups=privileged_groups,\
                                     unprivileged_groups=unprivileged_groups,\
                                     cost_constraint="fpr",
                                     seed=rand)
```

```
self.cost_constraint = cost_constraint
if self.cost_constraint == 'fnr':
    self.fn_rate = 1
    self.fp_rate = 0
elif self.cost_constraint == 'fpr':
    self.fn_rate = 0
    self.fp_rate = 1
elif self.cost_constraint == 'weighted':
    self.fn_rate = 1
    self.fp_rate = 1
```

4. fit() needs two datasets: the original one (test_ds) and the dataset with predictions (test_pred_cb_ds).

```
cpp = cpp.fit(test_ds, y_test_cb_pred)
test_pred_cpp_ds = cpp.predict(y_test_cb_pred)
```

Model
Post-processing bias mitigation methods
Calibrated equalized odds postprocessing method

Brandenburgische
Technische Universität
Cottbus - Senftenberg

Result

CPP Statistical Parity Difference (SPD):	0.0278 (0)
CPP Disparate Impact (DI):	1.0317 (1)
CPP Smoothed Empirical Differential Fairness (SEDF):	0.4924 (0)

CPP Difference in mean outcomes TEST between unprivileged and privileged groups in percent 2.78

Model
Comparison Mitigation Bias Methods

Brandenburgische
Technische Universität
Cottbus - Senftenberg

		Pre-Processing		In-Processing		Post-Processing	
	with bias	RW	DIR	PRM	GFC	EOP	CEOP
SPD (0)	0.0549	0.0363	-0.0525	0.0480	0.0548	**-0.0195**	0.0278
DI (1)	0.9375	0.9579	0.9415	1.7315	0.9388	**0.9774**	1.0317
SED (0)	0.7036	0.4322	0.6636	0.8431	0.6429	**0.3254**	0.4924
Difference mean %	-5.49	-3.63	-5.25	4.8	-5.48	**-1.95**	2.78

RW: Reweighing
DIR: Disparate impact remover method
PRM: Prejudice remover method
GFC: Gerry fair classifier method
EOP: Equalized odds postprocessing
CEOP: Calibrated equalized odds postprocessing method

Summery Mitigation Bias

Brandenburgische
Technische Universität
Cottbus - Senftenberg

- Fairness Bias
- Detecting bias
- Mitigating bias