Algorithm auditing with jfa: : **CHEAT SHEET**

Basics

jfa is an R package that facilitates algorithm auditing. The package also enables users to create a prior probability distribution to perform Bayesian algorithm auditing using these functions.

The package provides one function that allow users to easily apply Bayesian or classical probability theory in their algorithm audit workflow.

Installation

Installing the package can be done via:
`install.packages('jfa')`

Loading the package can be done via:
`library(jfa)`

Example

The blue code blocks next to the function descriptions provide a working example of the intended use.

The data for this example can be loaded via:
`data('compas')`

Select an appropriate metric for assessing algorithmic fairness

`jfa::fairness_selection()`

This function is designed to help auditors to select the appropriate fairness metric to perform algorithm auditing. It presents the user with four questions in an easily understandable manner, requiring no statistical background or in-depth knowledge of the fairness measures. The path through the decision-making workflow can be visualized using the `plot` function.

- `q1:` Is the information on the true values of the classification relevant in your context?
- `q2:` In what type of classification are you interested?
- `q3:` Can the negative class be considered as everything not positive or is it well-defined?
- `q4:` What are the errors with the highest cost?

```
fairness_selection(
  q1 = 1,
  q2 = 1,
  q3 = 2,
  q4 = 3
)
```

Test the equality of fairness metrics among protected groups

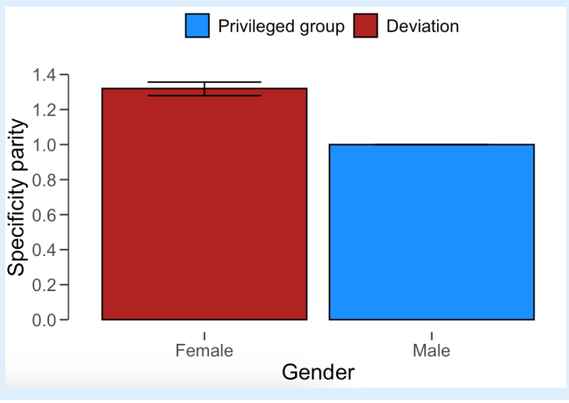
`jfa::model_fairness()`

This function assesses fairness in algorithmic decision-making systems by computing and testing the equality of one of several model-agnostic fairness metrics between protected classes. The metrics are computed based on a set of true labels and the predictions of an algorithm. The ratio of these metrics between any unprivileged protected class and the privileged protected class is called parity. The `prior` argument can be used to specify a prior distribution to perform Bayesian inference.

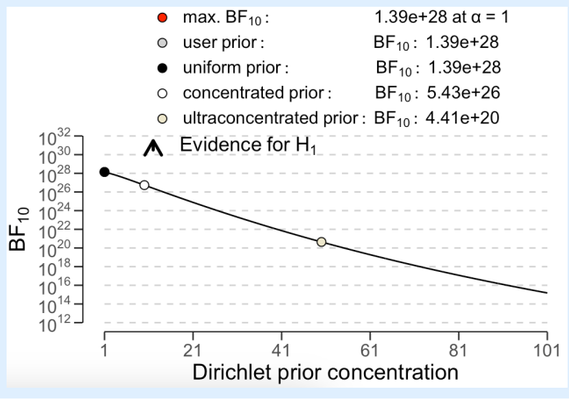
- `protected:` Specifies the variable that contains the protected groups / sensitive attribute.
- `target`: Specifies the variable that contains the target (i.e., to be predicted) variable.
- `predictions:` Specifies the predictions of the algorithm for the `target` variable.
- `privileged:` Specifies the privileged group in the `protected` variable.
- `positive:` Specifies the positive class in the `target` variable.
- `metric:` Specifies which fairness metric to compute.
- `prior:` Specifies the concentration parameter of the Dirichlet prior distribution.

```
model_fairness(
  data = compas,
  protected = 'Gender',
  target = 'TwoYrRecidivism',
  predictions = 'Predicted',
  privileged = 'Male',
  positive = 'yes',
  metric = 'sp',
  prior = FALSE, ...)
```

```
# does not require 'prior = TRUE'
plot(..., type = 'estimates')
```




```
# requires 'prior = TRUE'
plot(..., type = 'robustness')
```




```
# requires 'prior = TRUE'
plot(..., type = 'sequential')
```