2.1 Open Source Interpretability Tools

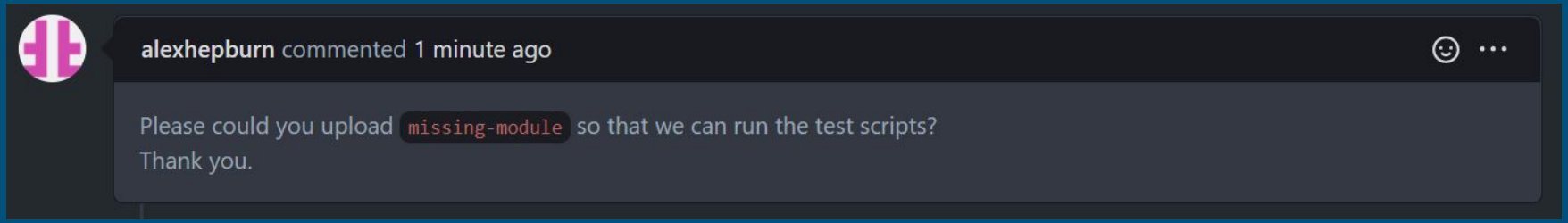
Using the Example of FAT Forensics

- Research software and reproducibility.
- Paperware.
- Interpretability, explainability and transparency toolkits.
- FAT Forensics and its design principles.

Alex Hepburn

Research Software

- Lucky to have it.
- Often not maintained.
- Sometimes lengthy Python scripts.
- Heavy and unnecessary dependencies (or simply unavailable).
- Over (or under) engineered.



Reproducibility

Software underpins much of research, but it tends to lack a good foundation.

- *Vapourware* -- promised but not delivered, i.e., non-existent.
- **Paperware** -- available but difficult to use by people outside of the core research (and/or development) team.
- *Software* -- documented, tested, usable and welcoming.

We need `scikit-learn` for Fairness, Accountability and **Transparency**.

Algorithmic Transparency

Individual explainers:

- LIME
  https://github.com/marcotcr/lime
- Local surrogates
  https://github.com/axa-rev-research/locality-interpretable-surrogate
- Anchor
  https://github.com/marcotcr/anchor
- PyCEbox
  https://github.com/AustinRochford/PyCEbox

Transparency/Interpretability/Explainability **packages**:

- Microsoft's InterpretML
  https://github.com/interpretml/interpret
- IBM's AI Explainability 360
  https://github.com/IBM/AIX360
- Oracle's Skater
  https://github.com/oracle/Skater
- ELI5
  https://github.com/TeamHG-Memex/eli5
- Yellowbrick
  https://github.com/DistrictDataLabs/yellowbrick

FAT Forensics

Algorithmic Fairness, Accountability and Transparency Toolkit

Origin

Creating a piece of software that covers fairness, accountability and transparency.

Team:

- ➔ Kacper Sokol -- Lead Developer
- ➔ Alex Hepburn -- Core Developer
- ➔ Peter Flach -- Principal Investigator
- ➔ Rafael Poyiadzi -- Developer
- ➔ Matthew Clifford -- Developer
- ➔ Raul Santos-Rodriguez -- Co-Investigator

Design and Development Principles

- Open sourced under the BSD 3-Clause licence.
- Minimal dependencies.
- Good software engineering practices:
  - unit testing;
  - continuous integration; and
  - consistent code styling and formatting.
- Complete and diverse documentation:
  - API reference;
  - online tutorials;
  - how-to guides; and
  - code examples.

Scope

Fairness

Data
Do some data points share the same unprotected features but different protected features?

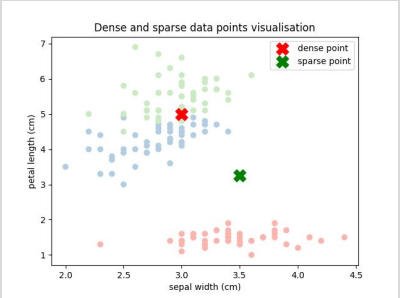
Models
Is there demographic parity between certain sub-groups?

Predictions
Are two data points that differ only in protected features treated differently?

Accountability

Data
Is there a sampling bias in the data according to some sub-groups?

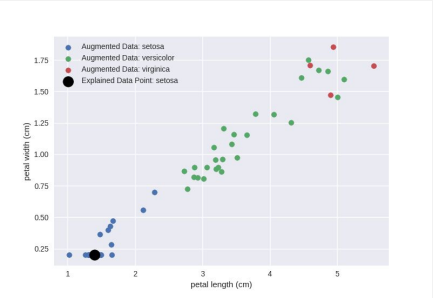
Models
Is there a systematic performance bias in the model?



Transparency

Predictions
Why is a decision made?

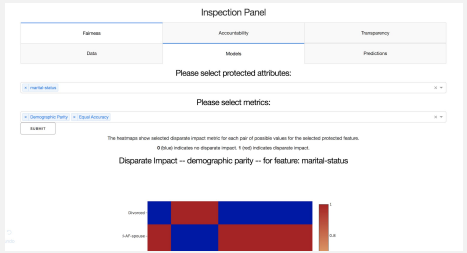
Models
What influence does each feature have on the model?



Use Modes

Deployment Mode

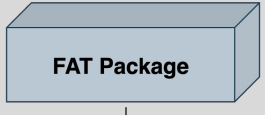
data in -- data out

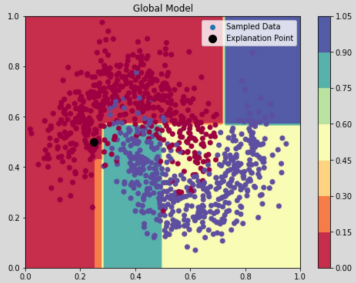


https://fatf.herokuapp.com/

Research Mode

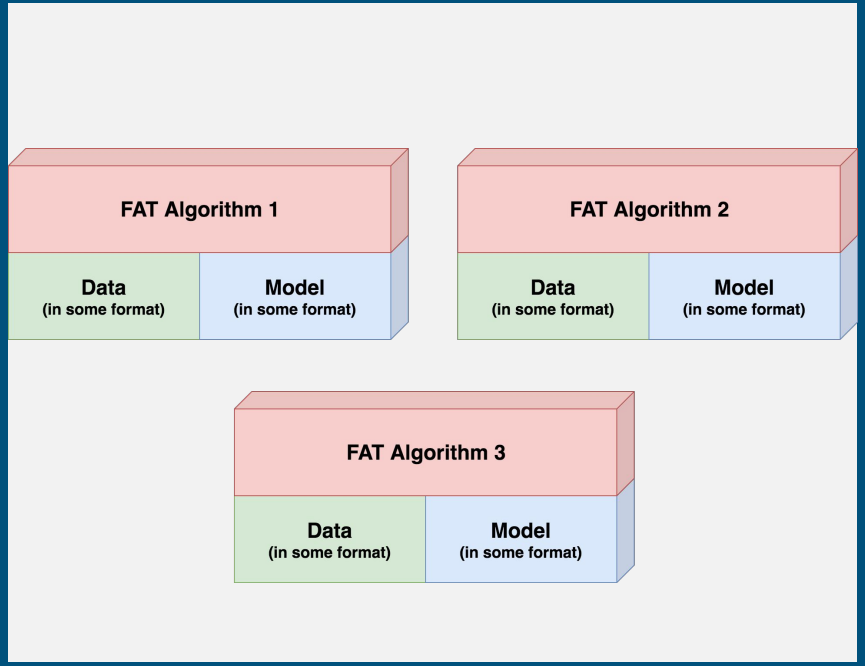
data in -- visualisations out



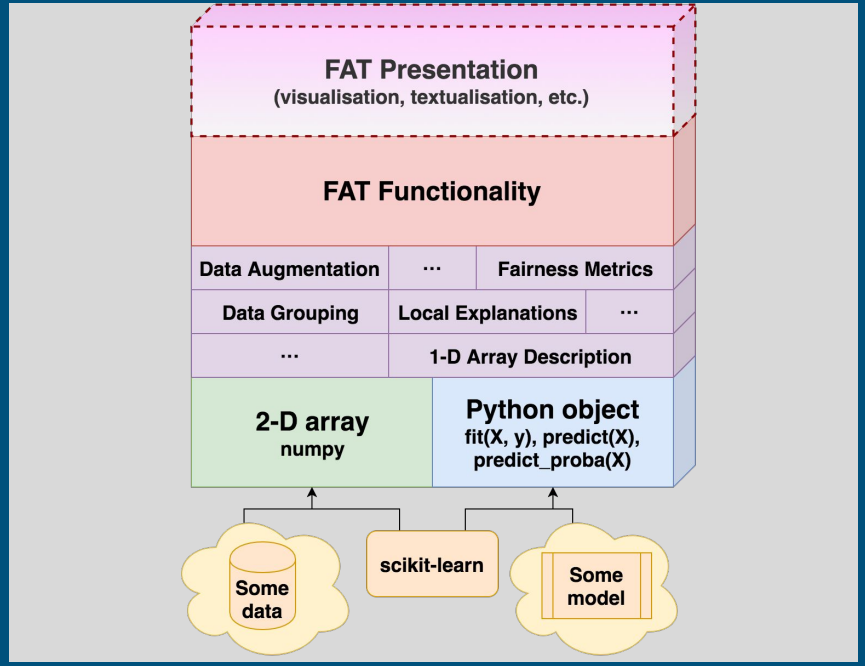


Modularity

Bespoke Code



Modular Design



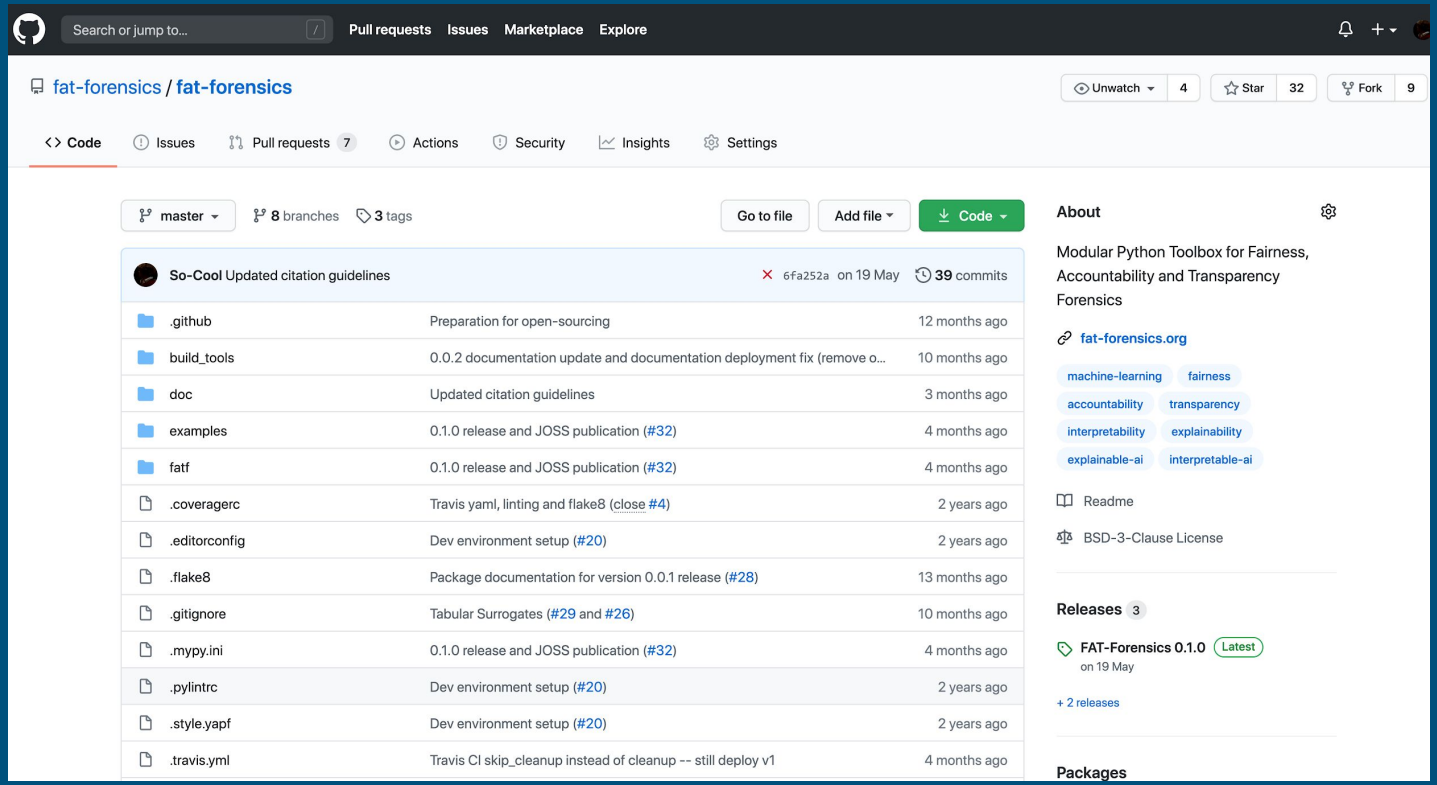
Implemented Functionality

	Fairness	Accountability	Transparency
Data/ Features	• Systemic Bias (disparate treatment labelling). • Sample size disparity (e.g., class imbalance).	• Sampling bias. • Data Density Checker.	• Data description.
Models	• Group-based fairness (disparate impact).	• Systematic performance bias.	• Partial dependence. • Individual conditional expectation.
Predictions	• Counterfactual fairness (disparate treatment).		• Counterfactuals. • Tabular bLIMEy (LIME alternative).

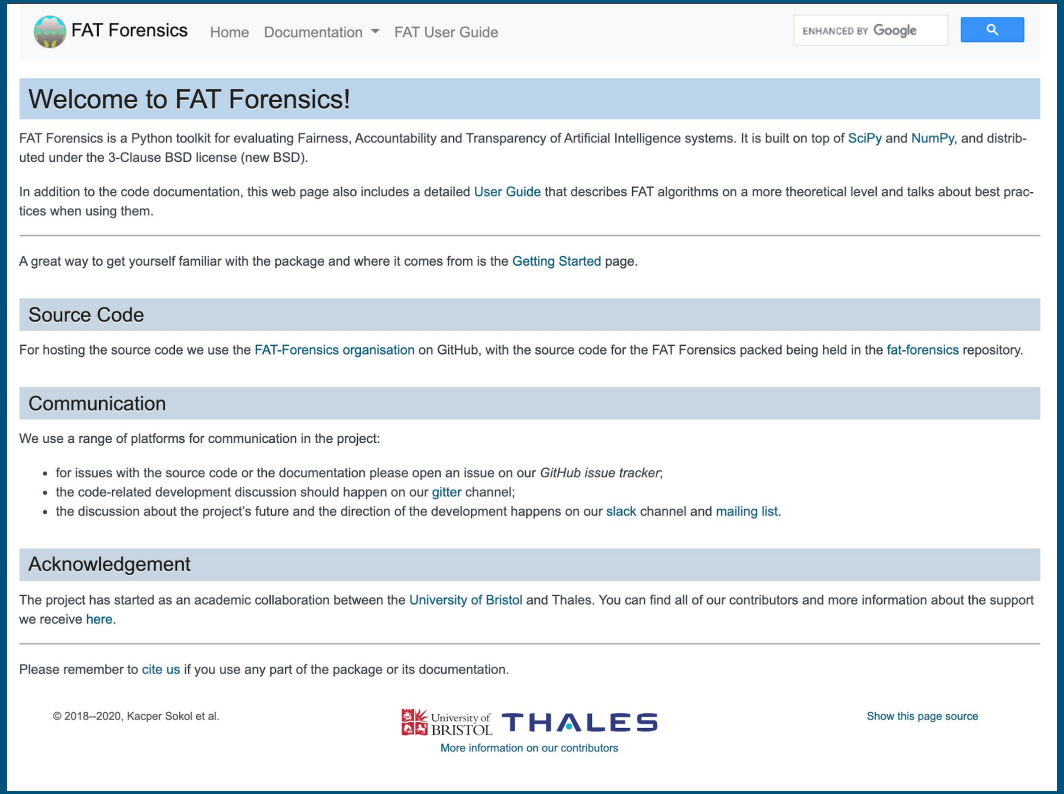
Planned Transparency Features

	Fairness	Accountability	Transparency
Data/ Features			• Bespoke Interpretable Representations
Models			• Permutation Importance • Decision Tree Explainer
Predictions			• Anchors • Image and Text Surrogates • Tree-specific Counterfactuals

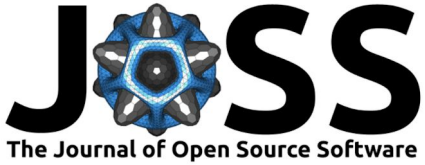
https://github.com/fat-forensics/fat-forensics



https://fat-forensics.org/



https://joss.theoj.org/papers/10.21105/joss.01904



FAT Forensics: A Python Toolbox for Implementing and Deploying Fairness, Accountability and Transparency Algorithms in Predictive Systems

Kacper Sokol[1], Alexander Hepburn[2], Rafael Poyiadzi[2], Matthew Clifford[2], Raul Santos-Rodriguez[2], and Peter Flach[1]

1 Department of Computer Science, University of Bristol **2** Department of Engineering Mathematics, University of Bristol

DOI: 10.21105/joss.01904

https://arxiv.org/abs/1909.05167

FAT Forensics: A Python Toolbox for Algorithmic Fairness, Accountability and Transparency

Kacper Sokol
K.Sokol@bristol.ac.uk
Department of Computer Science,
University of Bristol
Bristol, United Kingdom

Raul Santos-Rodriguez
enrsr@bristol.ac.uk
Department of Engineering
Mathematics, University of Bristol
Bristol, United Kingdom

Peter Flach
Peter.Flach@bristol.ac.uk
Department of Computer Science,
University of Bristol
Bristol, United Kingdom

Next Up

Hands-on Session Preparation

(Alex Hepburn)