

#WhatIfTool #SHAP

HANDS-ON TUTORIAL

# Probing ML Models for Fairness With the What-If Tool & SHAP

**James Wexler & Andrew Zaldivar**

With Sara Robinson, Mahima Pushkarna & Tolga Bolukbasi

<https://whatif-tool.dev>

<https://pair-code.github.io/what-if-tool/fat2020.html>

Google

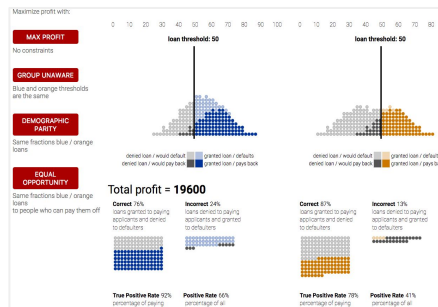
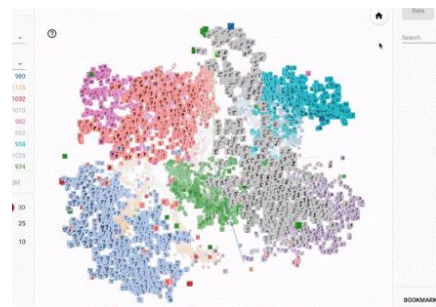
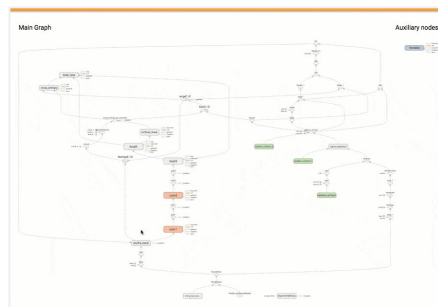
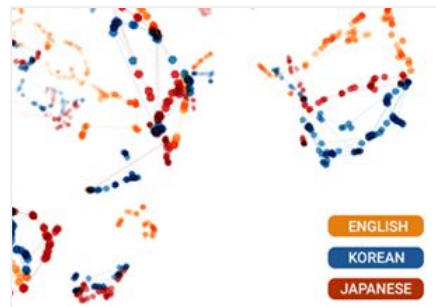
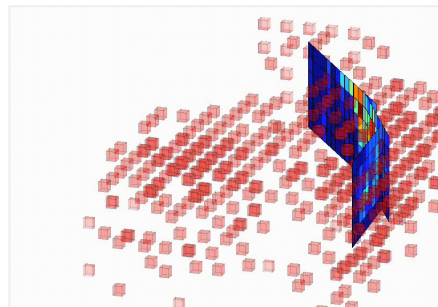
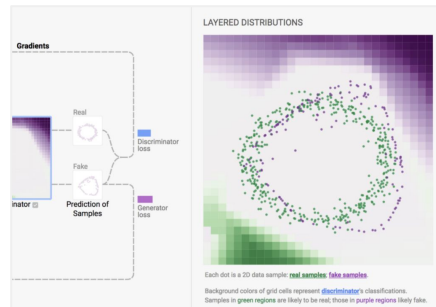


## #WhatIfTool #SHAP

Probing ML models for fairness with the What-If Tool & SHAP

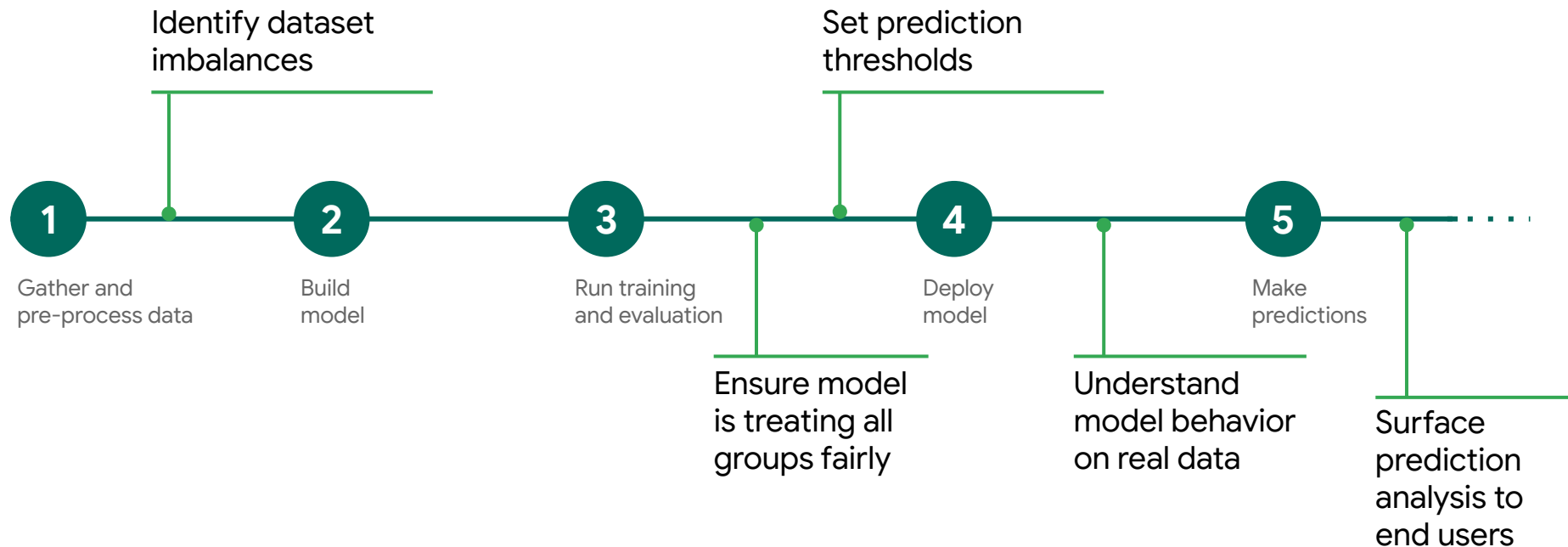
PAIR's mission is to conduct **human-centered research and design** to make **human-AI partnerships productive, enjoyable, and fair.**

We make technology.

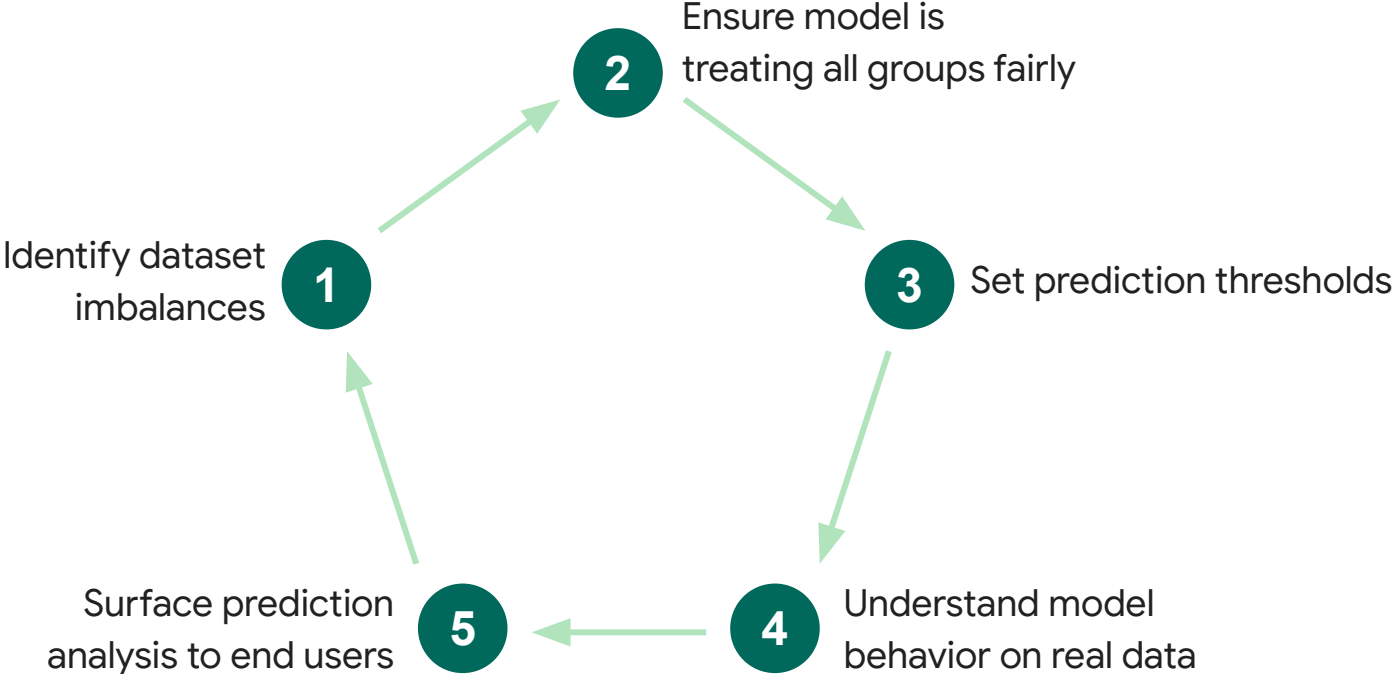


Above: A sampling of work from PAIR.

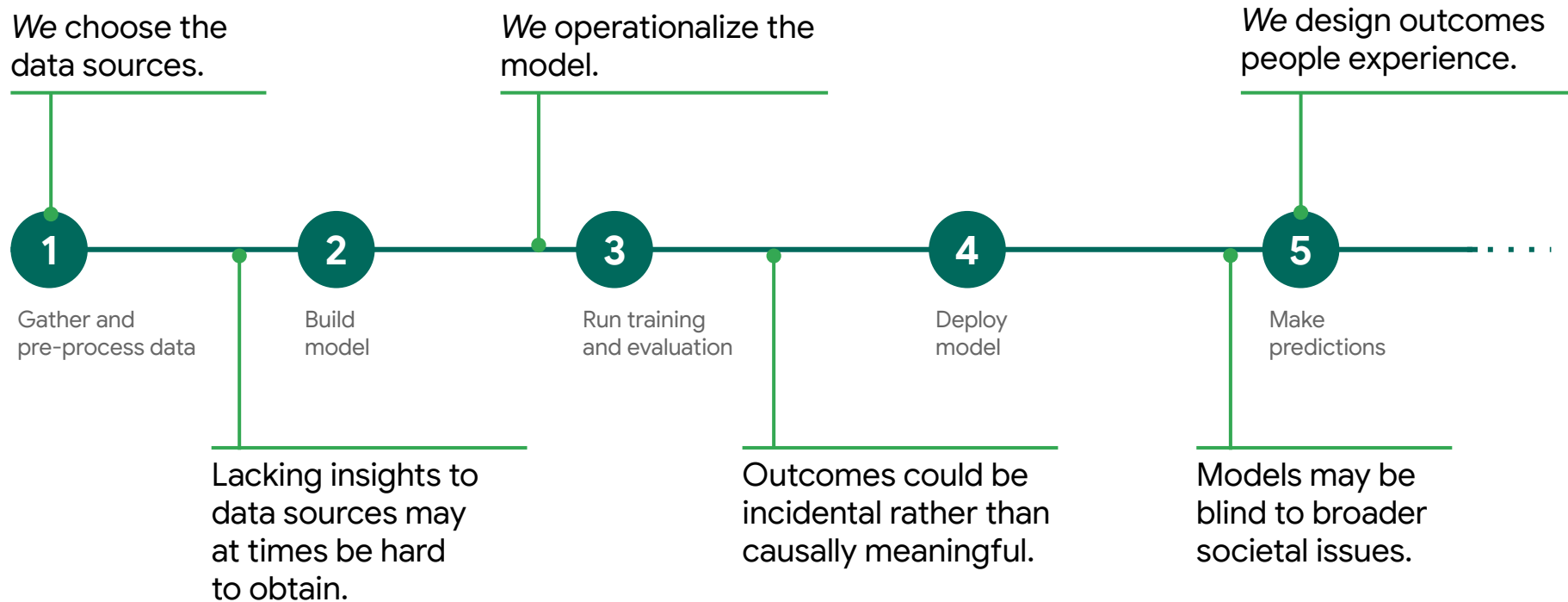
## Q: How does **explainability** fit into the ML lifecycle?



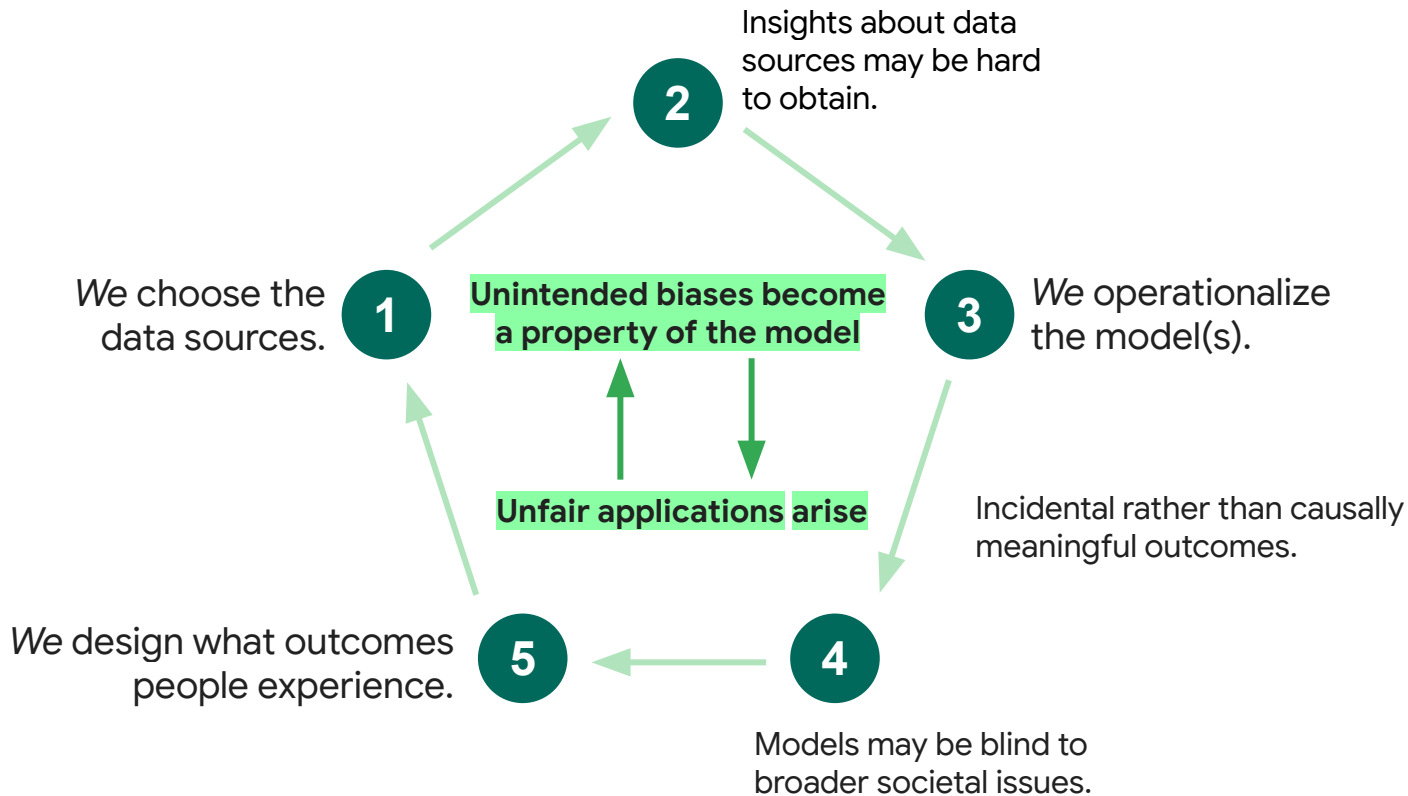
# The explainability process



## Q: How can biases be introduced into ML lifecycles?



# The *unfairness* process



# Algorithmic Unfairness: Some examples

## Representational Harm

When an ML system amplifies or reflects negative stereotypes about particular groups.

## Opportunity Denial

When an ML system negative impacts individuals' access to opportunities, resources, and overall quality of life.

## Disproportionate Failure

When the experience of interacting with an ML system is disproportionately failing for particular groups.





# Google's AI Principles

1. Be socially beneficial.

**2. Avoid creating or  
reinforcing unfair bias.**

3. Be built and tested for safety.

4. Be accountable to people.

5. Incorporate privacy  
design principles.

6. Uphold high standards  
of scientific excellence.

7. Be made available for uses  
that accord with these principles.





# There are many different interpretability approaches...

## Feature Attributions

Integrated gradients

SHAP

LIME

XRAI

## Model & Data Analysis

What-If Tool

Facets

Fairness Indicators

## Gradient & Concept Testing

TCAV

Grad-CAM

Guided  
Backpropagation

## Datapoint Inspection

Partial Dependence Plots

Counterfactuals

Ablation testing



We'll focus on **these two**:

## Feature Attributions

Integrated gradients

**SHAP**

LIME

XRAI

## Model & Data Analysis

**What-If Tool**

Facets

Fairness Indicators

## Gradient & Concept Testing

TCAV

Grad-CAM

Guided  
Backpropagation

## Datapoint Inspection

Partial Dependence Plots

Counterfactuals

Ablation testing



# How **does** my model perform...

classification accuracy / precision-recall curve / logarithmic loss /  
area under the curve / mean squared error / mean absolute error /  
F1 score / standard deviation / variance / confidence intervals /  
KL divergence / false positive rate / false negative rate /  
<insert **metric** here>

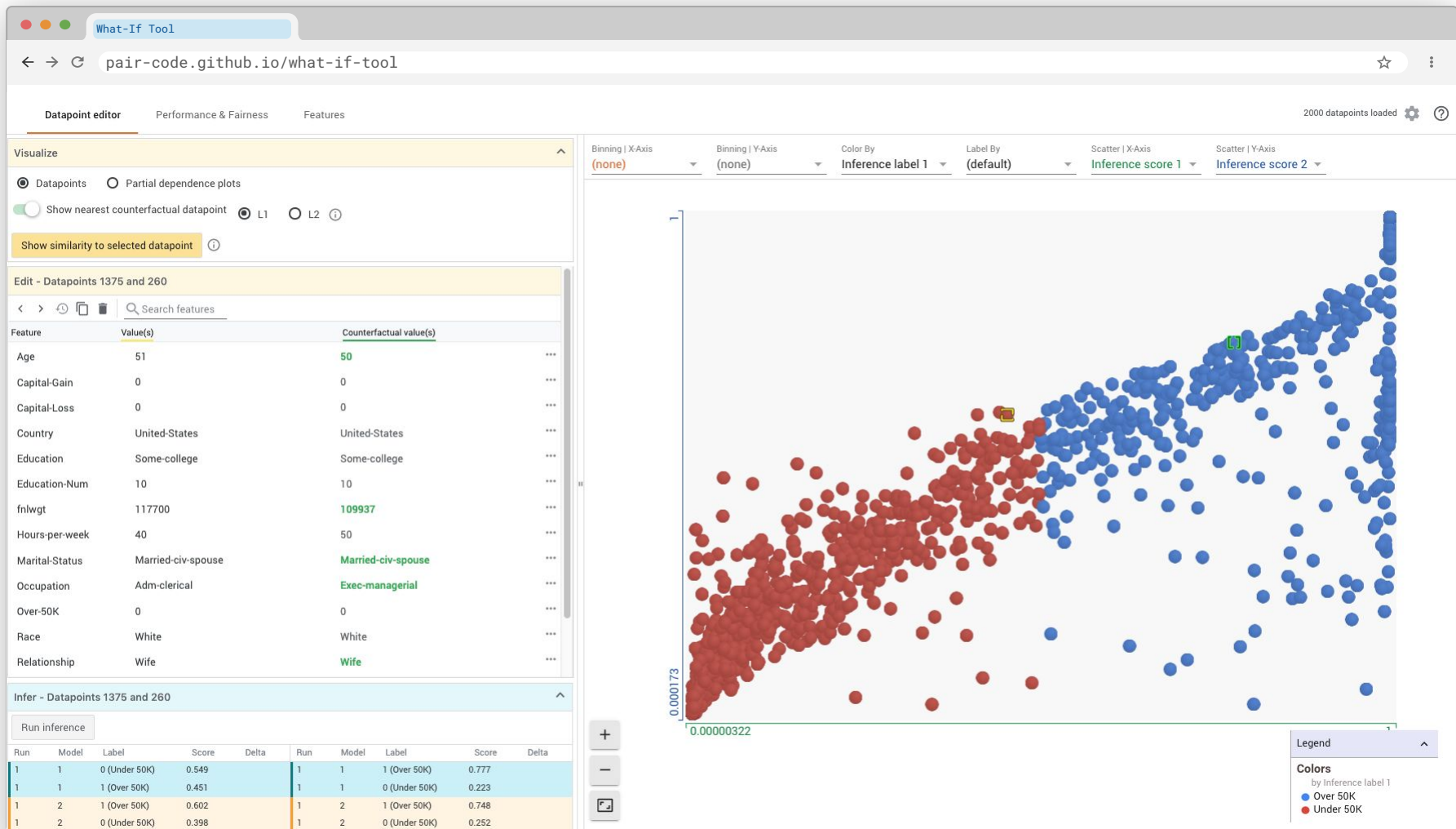
## How **might** my model perform...

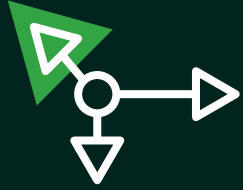
on subgroups in test data / on cross-slices in test data / on an individual data point / if a datapoint is perturbed / if model thresholds were different/ if optimized differently / across all values of a feature / when compared to a different model / on different data points within a neighborhood of data points / <insert **question** here>

# What if...

you could inspect  
machine learning models,  
with **minimal coding**  
required?



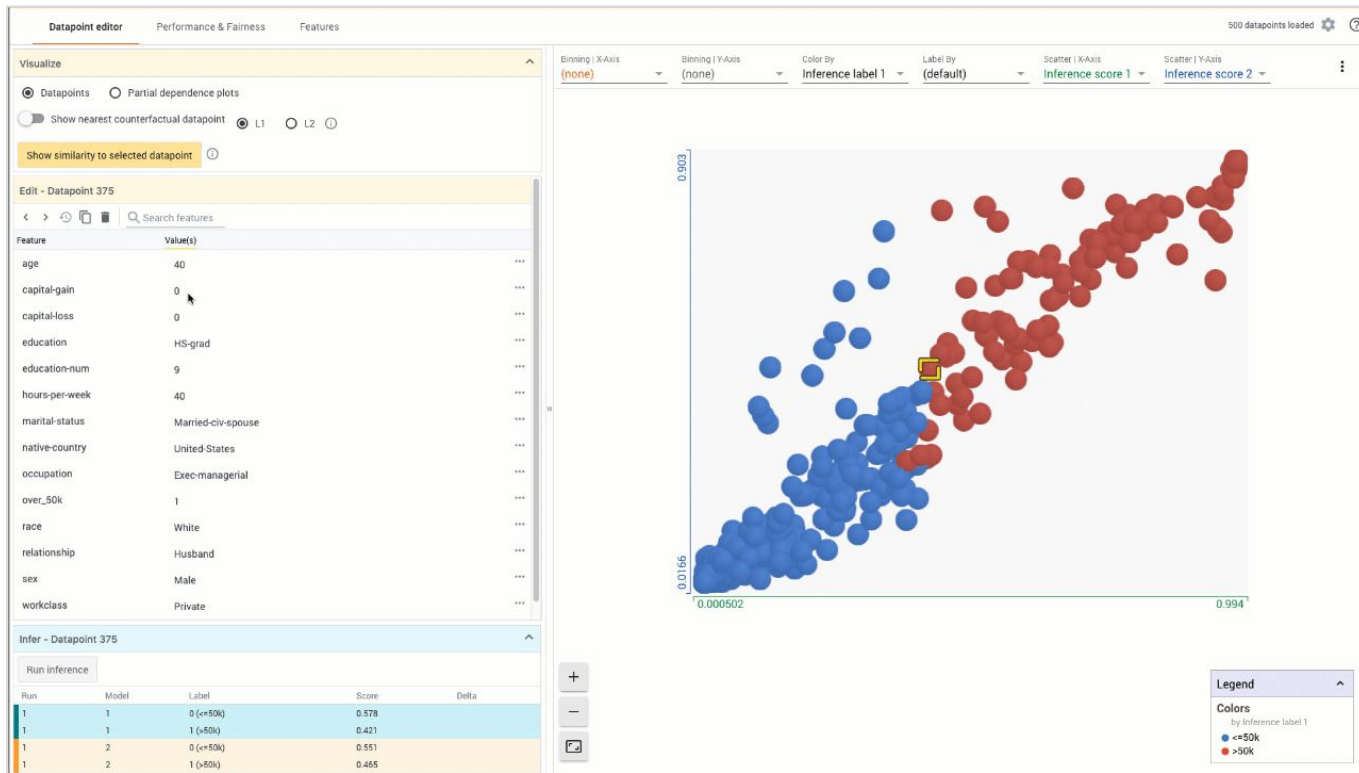




Supports *What-If* analysis

# Easily ask hypotheticals

Alter datapoints  
and see how model  
outputs change





# Counterfactuals

“What would have to change  
for me to have gotten the loan?”

$$\arg \min_{x'} \max_{\lambda} \lambda (f_w(x') - y')^2 + d(x_i, x')$$

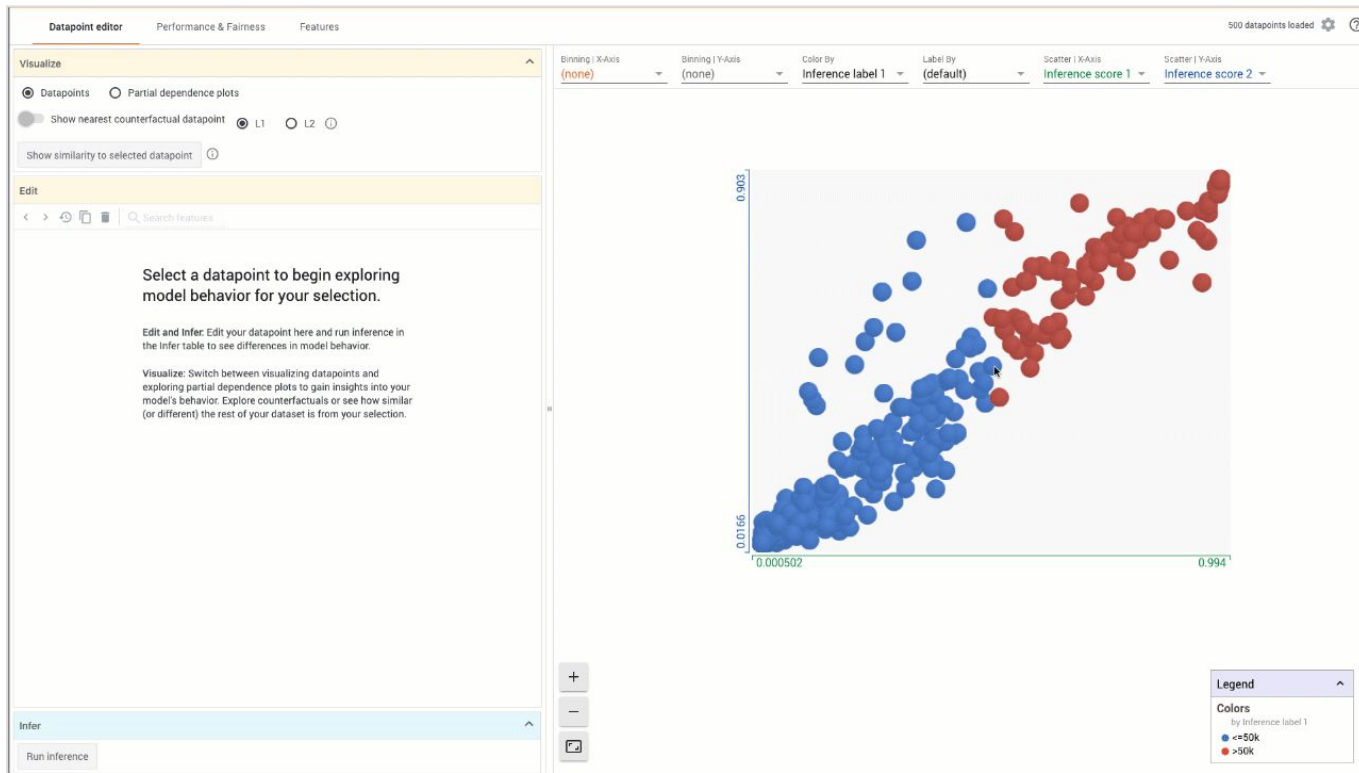
--  
Wachter et al. “Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR”

## Approaches

- Optimization problem to find hypothetical datapoint
- Search across real examples

# Explore decision boundaries

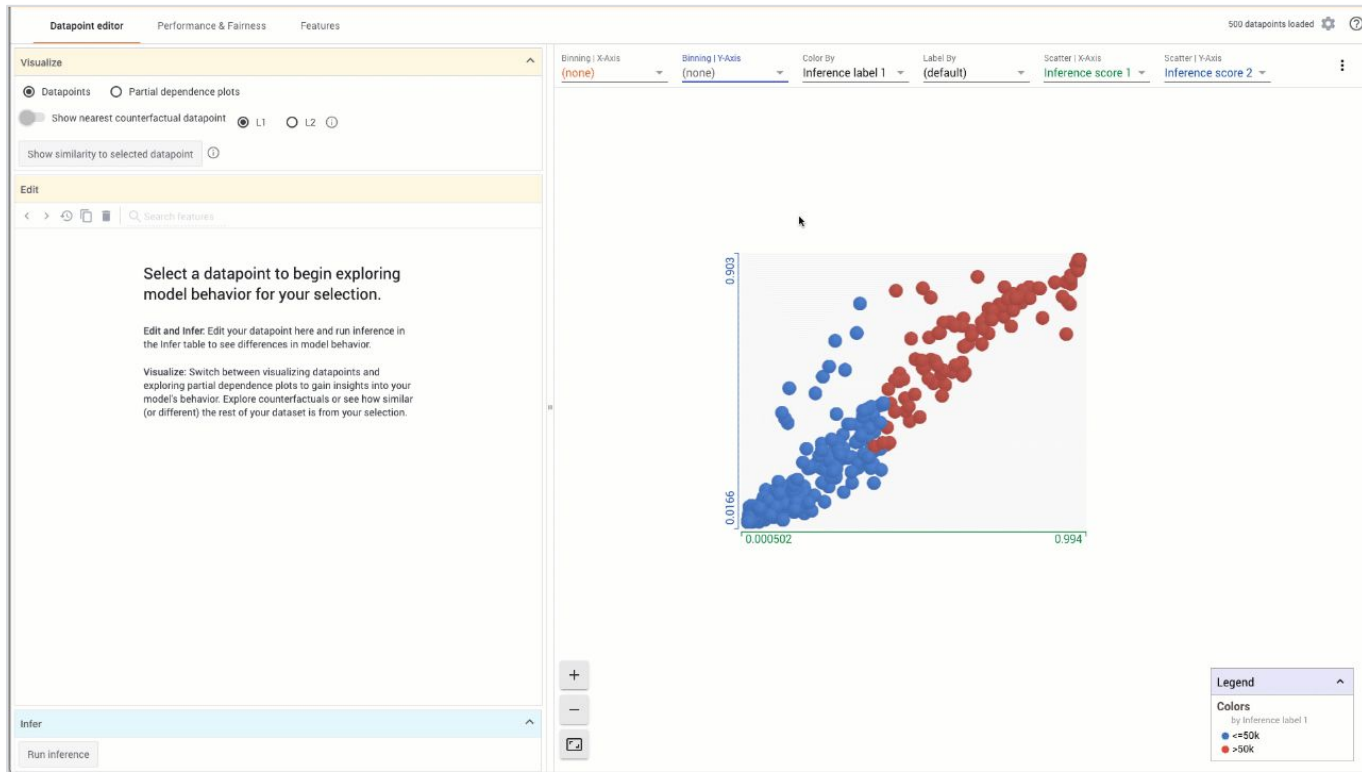
Translating  
counterfactual research  
into visual tooling within  
workflows.



**Above:** For classification problems, our counterfactual finding feature can identify the most similar datapoint (to a selection) in the loaded data that was classified differently by the model. For any dataset, L1 & L2 distances are available as *inbuilt similarity metrics*. However, users can specify custom metrics when invoking the tool.

# Scale up without changing user's mental models

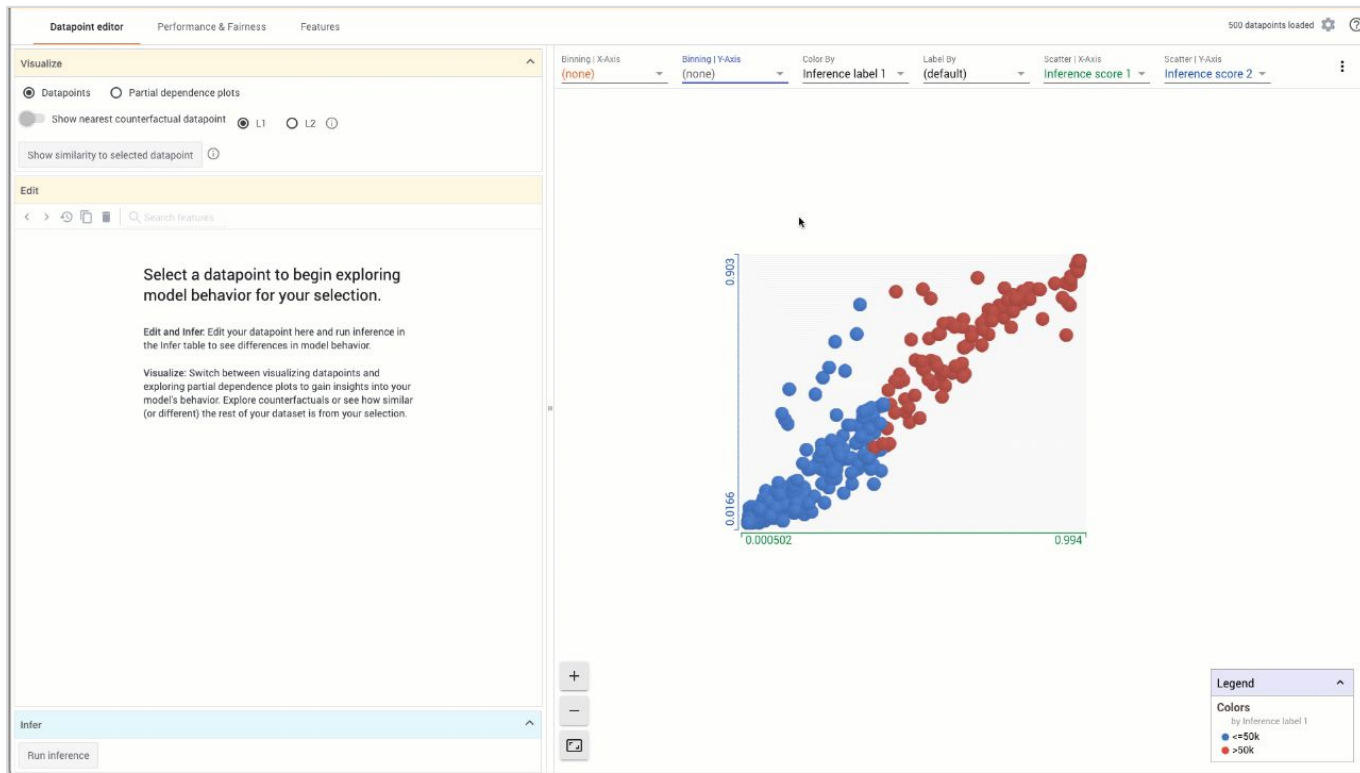
Compare performances of multiple models on the same simulation simultaneously.



**Above:** For classification problems, our counterfactual finding feature can identify the most similar datapoint (to a selection) in the loaded data that was classified differently by the model. For any dataset, L1 & L2 distances are available as *inbuilt similarity metrics*. However, users can specify custom metrics when invoking the tool.

# Support many workflows without coding

Create custom visualizations using dataset features and model scores.



# User-focused customizations

Ways to specify  
custom...

Models

Data

Distances

(eg. Similarity metrics)

Attributions

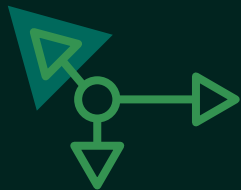
(eg. TCAV)

...

```
# This function extracts 'image/encoded' field, which is a reserved key for the
# feature that contains encoded image byte list. We read this feature into
# BytesIO and decode it back to an image using PIL.
# The model expects an array of images that are floats in range 0.0 to 1.0 and
# outputs a numpy array of (n_samples, n_labels)
def custom_predict(examples_to_infer):
    def load_byte_img(im_bytes):
        buf = BytesIO(im_bytes)
        return np.array(Image.open(buf), dtype=np.float64) / 255.

    ims = [load_byte_img(ex.features.feature['image/encoded'].bytes_list.value[0])
            for ex in examples_to_infer]
    preds = model.predict(np.array(ims))
    return preds
```

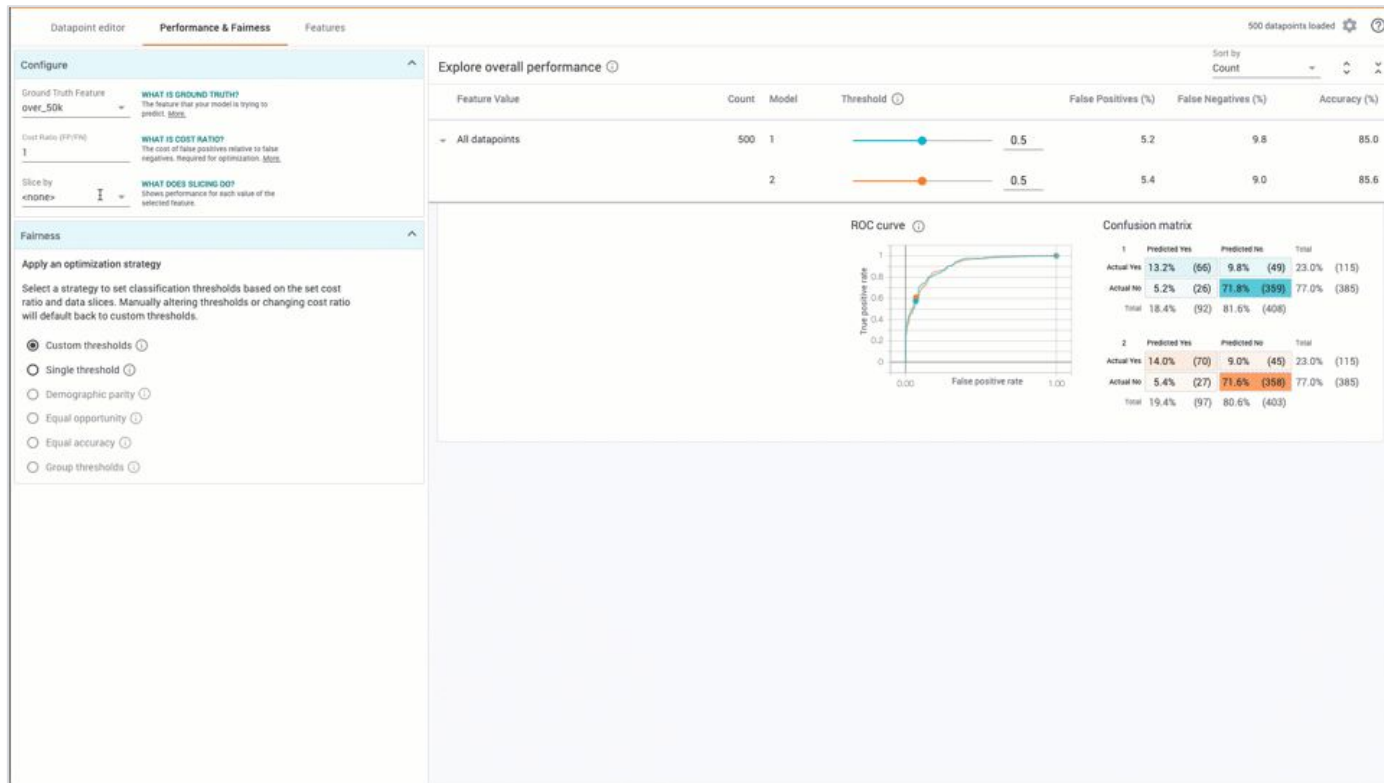
Above: Example of a custom predict function in colabotatory



Visualizes  
model performance

# Allow user-defined intersectional analysis

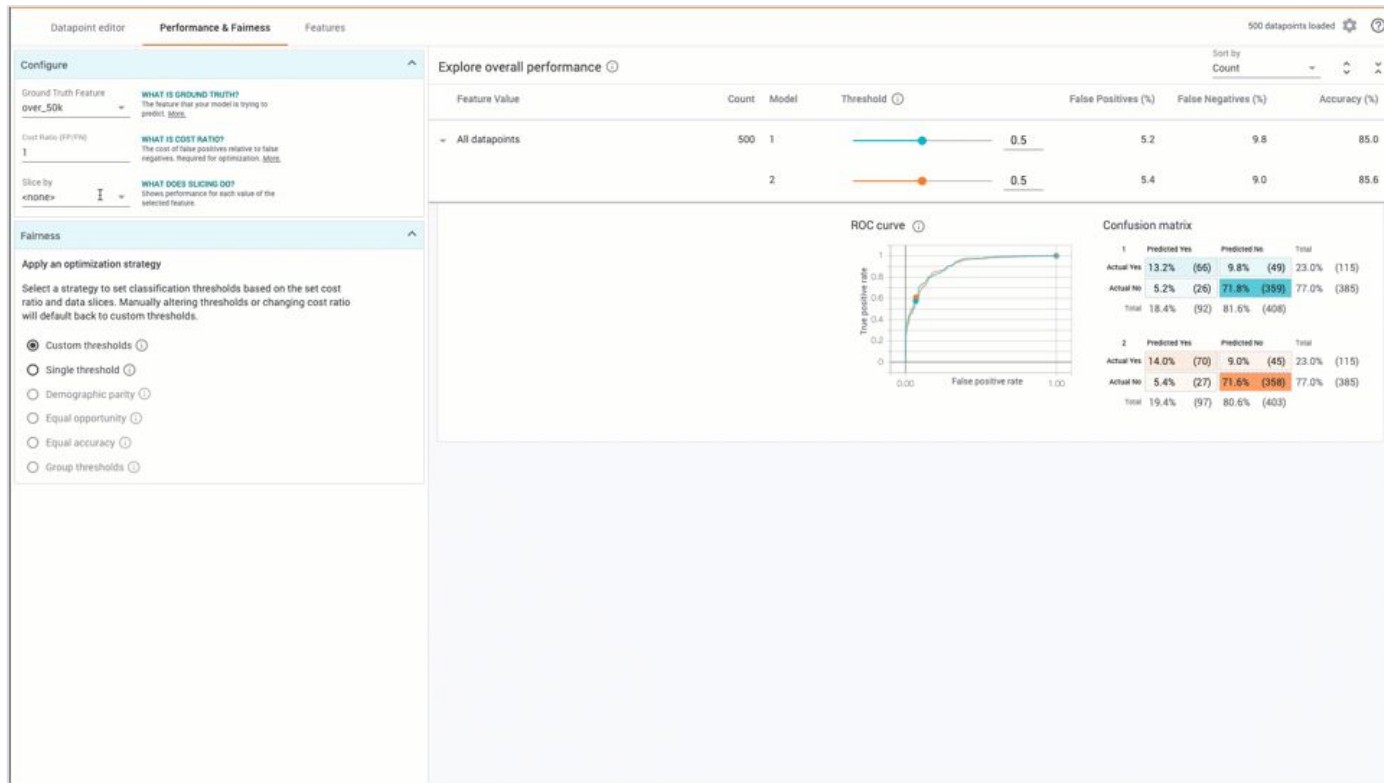
Evaluate performance on sub-groups of data rooted in feature values.



**Above:** Exploring model performance on different 'slices' of data given the values of specific features.

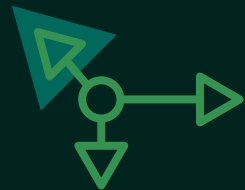
# Explore ML Fairness optimizations

Translating fairness research into visual tooling.



**Above:** Exploring model performance on different 'slices' of data given the values of specific features.





# Open-Source Tool

<https://whatif-tool.dev>

```
pip install witwidget
```



## Q: What is **feature attribution**?

A: The amount each feature in a model contributes to the model's prediction.

For example

**A model predicts you  
are 80% (0.8) likely to  
be approved for a loan.**

Feature attributions:

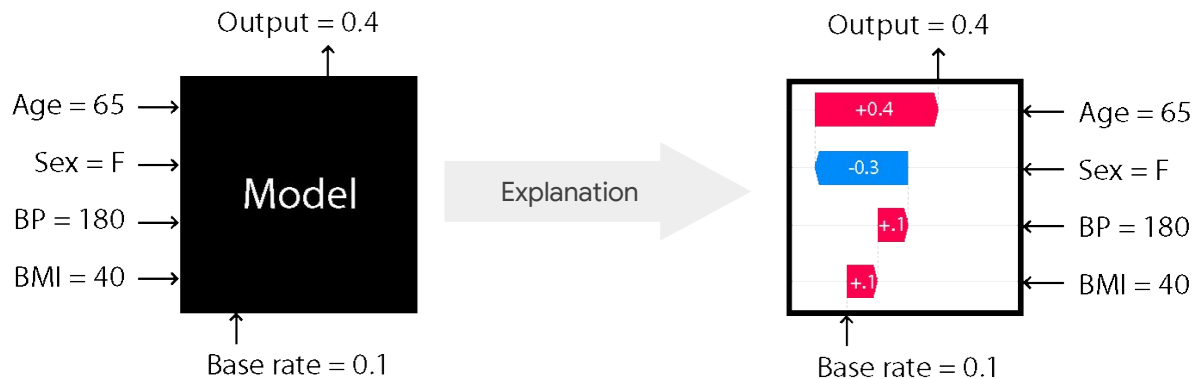
Age	0.3
-----	-----

Income	0.6
--------	-----

Credit Score	-0.1
--------------	------

## Q: What is **SHAP**?

A: An open source framework for inspecting any machine learning model through feature attributions.



# Q: How does SHAP work?

## What does SHAP return?

SHAP assigns importance values to each feature indicating **the effect that feature had** on the model prediction.

## How does SHAP calculate this?

SHAP approximates **the effect of removing a feature** from the model.

- Returns instance-level feature attributions along with global model-level feature importance.
- Works on image, text, and tabular models built with many different ML frameworks (TF, Scikit Learn, XGB, PyTorch),

--

Learn more: [papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf](https://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf)

## Q: How does SHAP work?

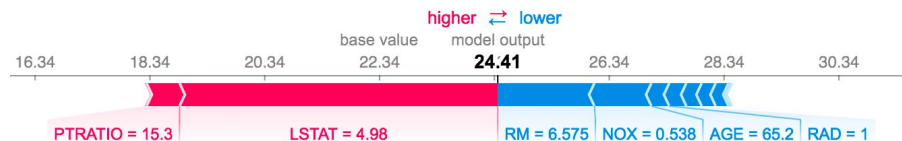
$$\phi_i(N, v) = \frac{1}{N!} \sum_{S \subseteq N \setminus \{i\}} |S|!(|N| - |S| - 1)! \left[ v(S \cup \{i\}) - v(S) \right]$$

Weighted average of the marginal contribution for an agent.

How much does adding or removing a single feature affect the prediction?

# Q: How does SHAP work?

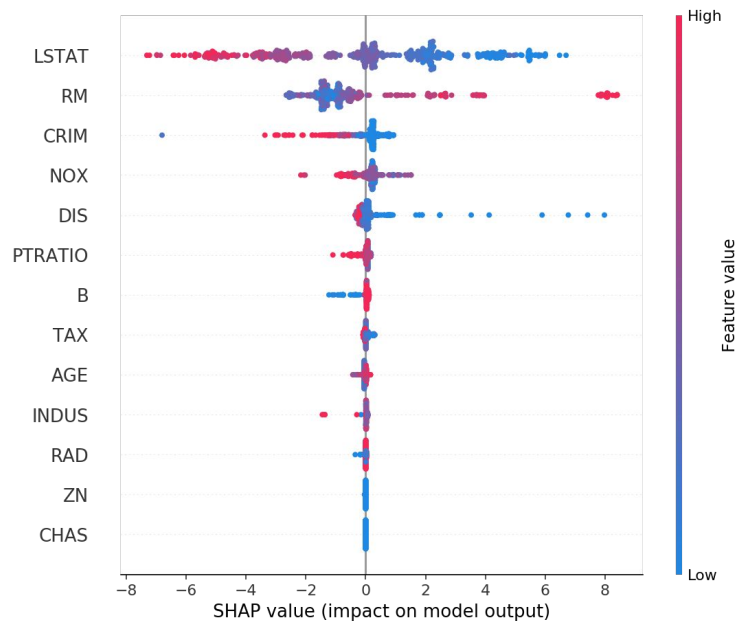
## Instance-level attributions



Shows how each feature changes the model output from the baseline.

Red features pushed the prediction up from the baseline, blue features pushed it down.

## Model-level attributions

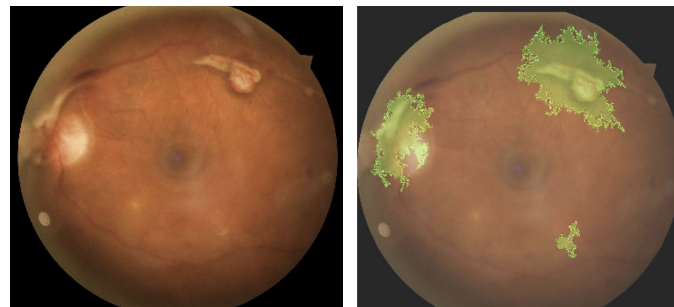


# Q: How might we interpret **image model attributions**?

## Animal Species Classification Model



## Diabetic Retinopathy (non-SHAP)





## Getting started with SHAP

```
import shap

# Create the explainer
explainer = shap.DeepExplainer(model, train_data.values[:200])

# Get attribution values
shap_values = explainer.shap_values(train_data.values[:5])
```

# Colab Notebook Exercise

Find link at:

<https://pair-code.github.io/what-if-tool/fat2020.html>



# Caveats

## Many approaches to interpreting models

Explainability is an emerging field with lots of ongoing research.

We've only shown a few methods.

Other techniques include: Integrated Gradients, LIME, SmoothGrad, etc...

Attribution techniques can be unreliable (see [The \(Un\)reliability of saliency methods](#) and related papers).

## “ML fairness” doesn't solve societal issues

Making a model more fair has no effect on issues that may have caused creation of a problematic dataset.

Fairer models can still be used to treat people unfairly.

The world is not static - model decisions affect future situations. See the [ML Fairness Gym](#) project.

# Discussion

## What did we discover?

Any interesting patterns in model behavior?

What were the performance disparities between groups?

What features had the largest effect on predictions?

What ways did you use the tool to find insights?

How does this speak to the larger issues with this data/task?

## Did anyone train a new model?

What differences in performance occurred?

What differences in attributions occurred?

How does this speak to the larger issues with this data/task?

Your feedback is important to us!  
Scan the code below:



## Thank You

More What-If Tool

People + AI Research

Get in touch

◀ This feedback form

[whatif-tool.dev](https://whatif-tool.dev)

[ai.google.com/pair](https://ai.google.com/pair)

[groups.google.com/g/what-if-tool](https://groups.google.com/g/what-if-tool)

[forms.gle/ugkNHkVqJspK7v9Q9](https://forms.gle/ugkNHkVqJspK7v9Q9)