



The latest news from Google AI

The What-If Tool: Code-Free Probing of Machine Learning Models

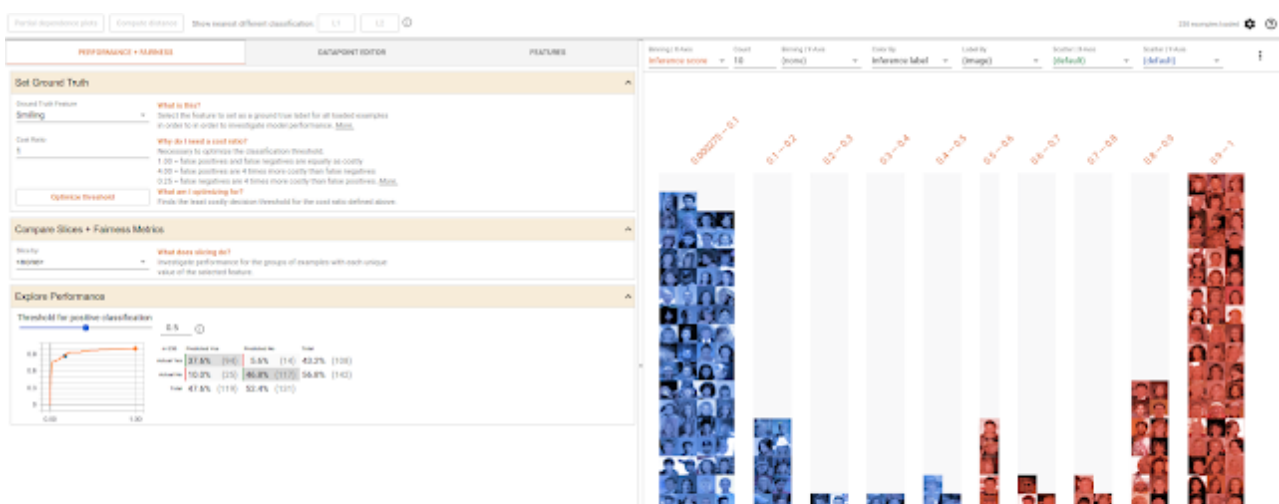
Tuesday, September 11, 2018

Posted by James Wexler, Software Engineer, Google AI

Building effective machine learning (ML) systems means asking a lot of questions. It's not enough to train a model and walk away. Instead, good practitioners act as detectives, probing to understand their model better: *How would changes to a datapoint affect my model's prediction?* *Does it perform differently for various groups—for example, historically marginalized people?* *How diverse is the dataset I am testing my model on?*

Answering these kinds of questions isn't easy. Probing “what if” scenarios often means writing custom, one-off code to analyze a specific model. Not only is this process inefficient, it makes it hard for non-programmers to participate in the process of shaping and improving ML models. One focus of the [Google AI PAIR initiative](#) is making it easier for a broad set of people to examine, evaluate, and debug ML systems.

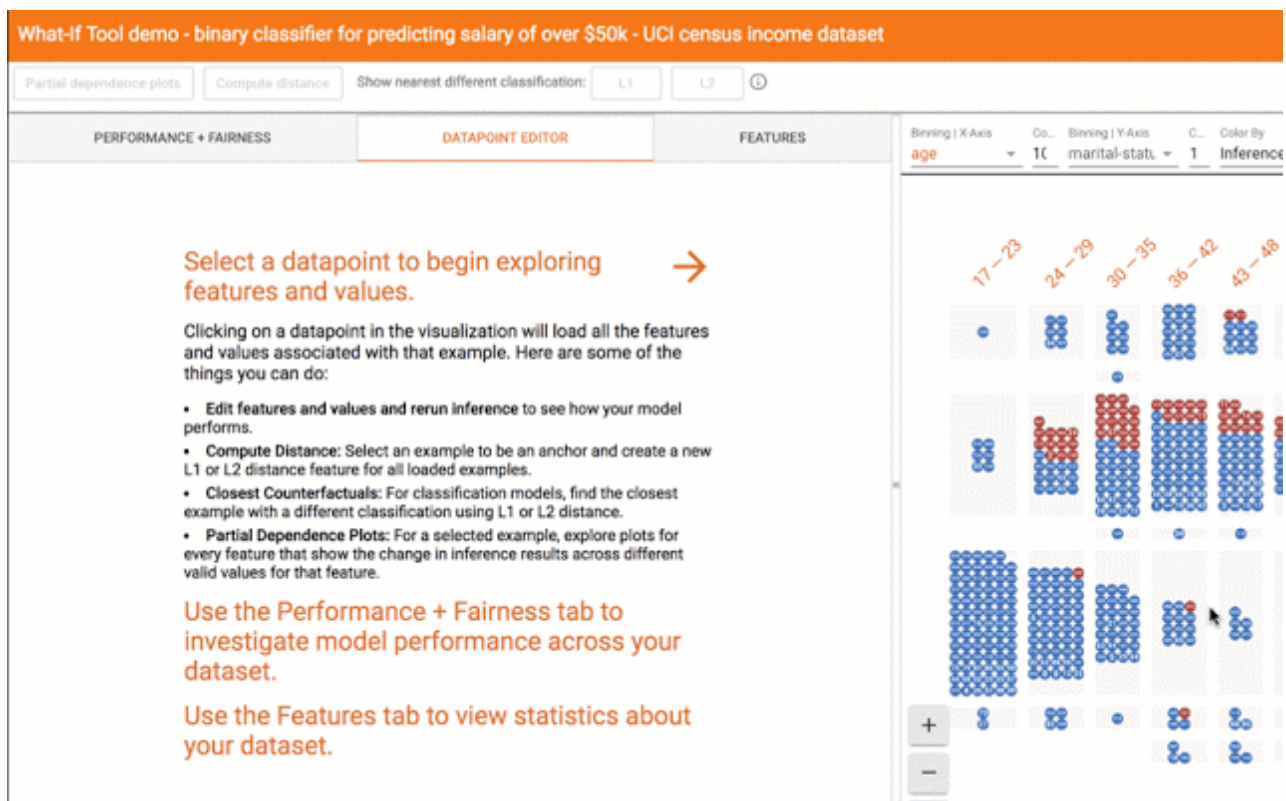
Today, we are launching the [What-If Tool](#), a new feature of the open-source [TensorBoard](#) web application, which let users analyze an ML model without writing code. Given pointers to a TensorFlow model and a dataset, the What-If Tool offers an interactive visual interface for exploring model results.





The What-If Tool, showing a set of 250 face pictures and their results from a model that detects smiles.

The What-If Tool has a large set of features, including visualizing your dataset automatically using [Facets](#), the ability to manually edit examples from your dataset and see the effect of those changes, and automatic generation of partial dependence plots which show how the model's predictions change as any single feature is changed. Let's explore two features in more detail.



Exploring what-if scenarios on a datapoint.

Counterfactuals

With a click of a button you can compare a datapoint to the most similar point where your model predicts a different result. We call such points "counterfactuals," and they can shed light on the decision boundaries of your model. Or, you can edit a datapoint by hand and explore how the model's prediction changes. In the screenshot below, the tool is being used on a binary classification model that predicts whether a person earns more than \$50k based on public census data from the [UCI census dataset](#). This is a benchmark prediction task used by ML researchers, especially when analyzing algorithmic fairness — a topic we'll get to soon. In this case, for the selected datapoint, the model predicted with 73% confidence that the person earns more than \$50k. The tool has automatically located the most-similar person in the dataset for which the model predicted earnings of less than \$50k and compares the two side-by-side. In this case, with just a minor difference in age and an occupation change, the model's prediction has flipped.

< Example ID: 399 >

Run inference

Search features

Feature name	Value	L1 neighbor value	
age	45	39	***
capital-gain	0	0	***
capital-loss	0	0	***
education	Bachelors	Bachelors	***
education-num	13	13	***
hours-per-week	40	40	***
marital-status	Married-civ-spouse	Married-civ-spouse	***
native-country	United-States	United-States	***
occupation	Exec-managerial	Transport-moving	***
over_50k	0	0	***
race	White	White	***
relationship	Husband	Husband	***
sex	Male	Male	***
workclass	Private	Private	***

Inference Results: Example 399

L1: Example 213

Run	Label	Score	Delta	Run	Label	Score	Delta
1	1 (>50k)	0.730		1	0 (<=50k)	0.546	
1	0 (<=50k)	0.270		1	1 (>50k)	0.452	

Comparing counterfactuals.

Analysis of Performance and Algorithmic Fairness

You can also explore the effects of different classification thresholds, taking into account constraints such as different [numerical fairness criteria](#). The below screenshot shows the results of a smile detector model, trained on the open-source [CelebA dataset](#) which consists of annotated face images of celebrities. Below, the faces in the dataset are divided by whether they have brown hair, and for each of the two groups there is an [ROC curve](#) and [confusion matrix](#) of the predictions, along with sliders for setting how confident the model must be before determining that a face is smiling. In this case, the confidence thresholds for the two groups were set automatically by the tool to optimize for [equal opportunity](#).

Compare Slices + Fairness Metrics

Slice by

Brown_Hair

What does slicing do?

Investigate performance for the groups of examples with each unique value of the selected feature.

Slice by (secondary)

<none>

Specify a second feature to slice the data by.

Optimize sliced thresholds for:

What am I optimizing for? [More...](#)

Group unaware

Demographic parity

Equal opportunity

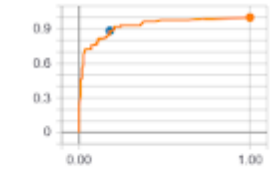
Equal accuracy

Group thresholds

Explore Performance

Not brown hair


0.48

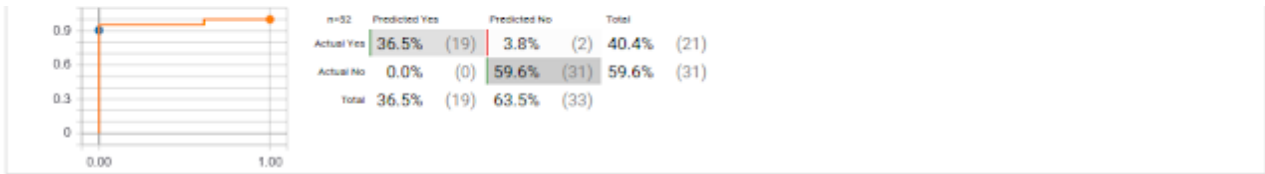


	Predicted Yes	Predicted No	Total
Actual Yes	38.9% (77)	5.1% (10)	43.9% (87)
Actual No	10.1% (20)	46.0% (91)	56.1% (111)
Total	49.0% (97)	51.0% (101)	

Brown hair

0.89





Comparing the performance of two slices of data on a smile detection model, with their classification thresholds set to satisfy the “equal opportunity” constraint.

Demos

To illustrate the capabilities of the What-If Tool, we’ve released a set of demos using pre-trained models:

- **Detecting misclassifications:** A [multiclass classification model](#), which predicts plant type from four measurements of a flower from the plant. The tool is helpful in showing the decision boundary of the model and what causes misclassifications. This model is trained with the [UCI iris dataset](#).
- **Assessing fairness in binary classification models:** The [image classification model](#) for smile detection mentioned above. The tool is helpful in assessing algorithmic fairness across different subgroups. The model was purposefully trained without providing any examples from a specific subset of the population, in order to show how the tool can help uncover such biases in models. Assessing fairness requires careful consideration of the overall context — but this is a useful quantitative starting point.
- **Investigating model performance across different subgroups:** A [regression model](#) that predicts a subject’s age from census information. The tool is helpful in showing relative performance of the model across subgroups and how the different features individually affect the prediction. This model is trained with the [UCI census dataset](#).

What-If in Practice

We tested the What-If Tool with teams inside Google and saw the immediate value of such a tool. One team quickly found that their model was incorrectly ignoring an entire feature of their dataset, leading them to fix a previously-undiscovered code bug. Another team used it to visually organize their examples from best to worst performance, leading them to discover patterns about the types of examples their model was underperforming on.

We look forward to people inside and outside of Google using this tool to better understand ML models and to begin assessing fairness. And as the code is open-source, we welcome contributions to the tool.

Acknowledgments

The What-If Tool was a collaborative effort, with UX design by Mahima Pushkarna, Facets updates by Jimbo Wilson, and input from many others. We would like to thank the Google teams that piloted the tool and provided valuable feedback and the TensorBoard team for all their help.



