InterpretML: A toolkit for understanding machine learning models*

Version: May 18, 2020

Summary

We introduce InterpretML, an open source Python toolkit for explaining black-box AI systems and training intelligible models. When AI systems are used in ways that impact people's lives, it is critically important that people understand the behavior of those systems. InterpretML helps users gain a better understanding of their model's overall behavior ("global explanation"), understand the reasons behind individual predictions ("local explanation"), and debug their model's predictions by exploring what similar data instances have received different outcomes. InterpretML incorporates state-of-the-art machine learning intelligibility techniques developed by Microsoft or well-proven third-party libraries. It creates a common API across the integrated libraries, making it possible to call and compare different intelligibility methods with the same set of function calls and data structures. InterpretML also applies optimizations that enable running real-world datasets at scale.

InterpretML's built-in interactive and exploratory visualization dashboard gives data scientists a wide range of insights about their dataset, model performance, and model explanations. InterpretML includes a new interpretability algorithm—the Explainable Boosting Machine, which is a highly intelligible and explainable—"glassbox"—model, with accuracy that's comparable to machine learning methods like random forests and boosted trees. InterpretML is a community-driven initiative and invites further expansion by researchers and data scientists.

Introduction: What is intelligibility in AI?

Transparency is one of Microsoft's six AI principles. **Intelligibility** is an important aspect of transparency and means that people should be able to understand, monitor, and respond to the technical behavior of AI systems. In machine learning circles, the term intelligibility is often, though not always, used interchangeably with the term **interpretability.**

When AI systems are used in ways that impact people's lives, it is critically important that people understand the behavior of those systems. In machine learning circles, intelligibility—or interpretability—means that people should be able to understand, monitor, and respond to the technical behavior of AI systems. It is a fundamentally human-centric concept. In practice, achieving

---

* This white paper was written by Mehrnoosh Sameki (Microsoft), Sarah Bird (Microsoft), and Kathleen Walker (Allovus Design Inc.), based on materials produced by the InterpretML team and others.

intelligibility can be complex and highly dependent on a host of variables and human factors, precluding anything resembling a one-size-fits-all approach. It is an area of cutting-edge interdisciplinary research, building on ideas from machine learning, psychology, human-computer interaction, and design.

The need for intelligibility can arise in an almost limitless range of scenarios and is a tool for achieving human objectives, such as ensuring the fairness of decisions or the reliability and safety of AI systems operating in physical environments. Intelligibility is critical at both stages of a machine learning life cycle; training time (pre-deployment) and post-deployment. The following examples illustrate a few of the many diverse reasons for intelligibility during training time:

- **Intelligibility can improve the robustness of an AI system by making it easier to identify and fix bugs.** Suppose an engineer believes that time of day should be an important and predictive input feature for a machine learning model she is building to predict which app the user will want to open next, yet adding this feature does not improve performance. Understanding how the model is using this feature will help the engineer determine how to proceed.
- **Intelligibility can help users decide how much to trust an AI system.** Suppose an AI system deployed in a high school predicts that a student is likely to drop out but his teacher suspects that the system's prediction is wrong. Providing the teacher with an explanation for the prediction, including a description of the factors most influential in that prediction, allows the teacher to better understand why the prediction was made, and even to reassess her opinion, before making a decision about how best to support the student's education.
- **Intelligibility can help designers explain the system to end users.** When designers can understand why specific predictions were made, they can reason about how that information should be passed on to the end users.
- **Intelligibility leads to more usable products for end users.** Suppose a machine vision system was offered that provided low-vision users with information about the people who are currently in the office. Providing information about when a colleague's face is outside the system's field of view could help the user understand how to adjust one's behavior in order to provide the best input for the system.
- **Intelligibility can uncover potential sources of unfairness.** Suppose an AI system is used by an employer to match candidates to jobs. By examining the characteristics of the data used to train its constituent models, the employer may be able to identify that it underrepresents qualified female candidates, which, if not corrected, could result in hiring recommendations that undermine the employer's objective of improving the diversity of its workforce.
- **Intelligibility can help demonstrate compliance with regulatory obligations.** Suppose a lender uses an AI system to support its consumer lending decisions. Examining the training data and understanding how the data influences the system's recommendations might help the lender's developers and regulators discover that the system is unintentionally using certain features, like zip code, as a proxy for race which the law excludes from consideration.

Intelligibility, beyond training time, can be a critical tool for inferencing or scoring in the following scenario:

- **Intelligibility can help explain prediction to people affected by the model.** Suppose a lender uses an AI system to support its consumer lending decisions. The loan applicant gets a rejection

from the model and contacts the lender to learn about the reasons behind the loan decision. The lender can potentially look at model explanation at inferencing time to understand how the model has rejected this person's loan application.

Introducing InterpretML

Intelligibility is critical to answering questions in scenarios such as model debugging (Why did my model make this mistake? How can I improve my model?), fairness assessment (What is my model's potential for fairness-related harms?), human-AI collaboration (How can I understand and trust the model's decisions?), regulatory compliance (Does my model satisfy legal requirements?); and high-risk applications, such as, for example, healthcare, financial services, or judicial environments.

One way of achieving intelligibility is to provide human-understandable **explanations** of predictions made by a machine learning model or actions taken by an AI system. InterpretML provides data scientists and business decision makers with seamless explanations for a model's overall behavior—global explanation (e.g., what features most importantly affect the overall decisions of a loan allocation model). It further provides explanations behind individual predictions—or local explanations (e.g., why a person's loan application has been rejected). Additionally, with InterpretML, one can observe model explanations for a subgroup of data points. Explanations across different subgroups of data are valuable, for example, when assessing potential fairness issues of over- or underrepresentation of groups of people.

InterpretML addresses intelligibility by **exposing many state-of-the-art interpretability algorithms under a unified API.** This API covers two major interpretability forms:

- Glassbox models, which are machine learning models that are inherently intelligible and explainable to the user. These include decision trees, linear models, and Explainable Boosting Machine (EBM) models, which provide lossless explanations and are editable by domain experts.
- Black-box interpretability methods that generate explanations for any machine learning pipeline, no matter how opaque it is. Some of these methods are model-specific and some are model-agnostic:
    - Model-agnostic explainers use different approximations to analyze the relationship between input feature and output predictions. They can explain any underlying black-box model and are applied post model-training. These techniques do not require access to model internals such as weights or any other info about model structure.
    - Model-specific explainers are designed to explain model behavior of a specific class of models e.g., neural networks. These methods often use internal model information (such as weights) to explain the model behavior.

InterpretML's interactive visualization dashboard offers **a rich visual experience for what-if analyses,** where users can perturb one or multiple features of a selected data point and investigate whether the outcome of their model changes as expected. This strong debugging capability can uncover many hard-to-see issues in a model. For example, in the case of a loan-allocation AI system, a user can perform a what-if analysis for a female loan applicant who has received a prediction of "rejection." By changing the

sex feature from “female” to “male,” the user can see whether this perturbation has an observable impact on the model’s prediction. If so, this potential fairness issue can be fully investigated and mitigated by a machine learning fairness toolkit such as Fairlearn.

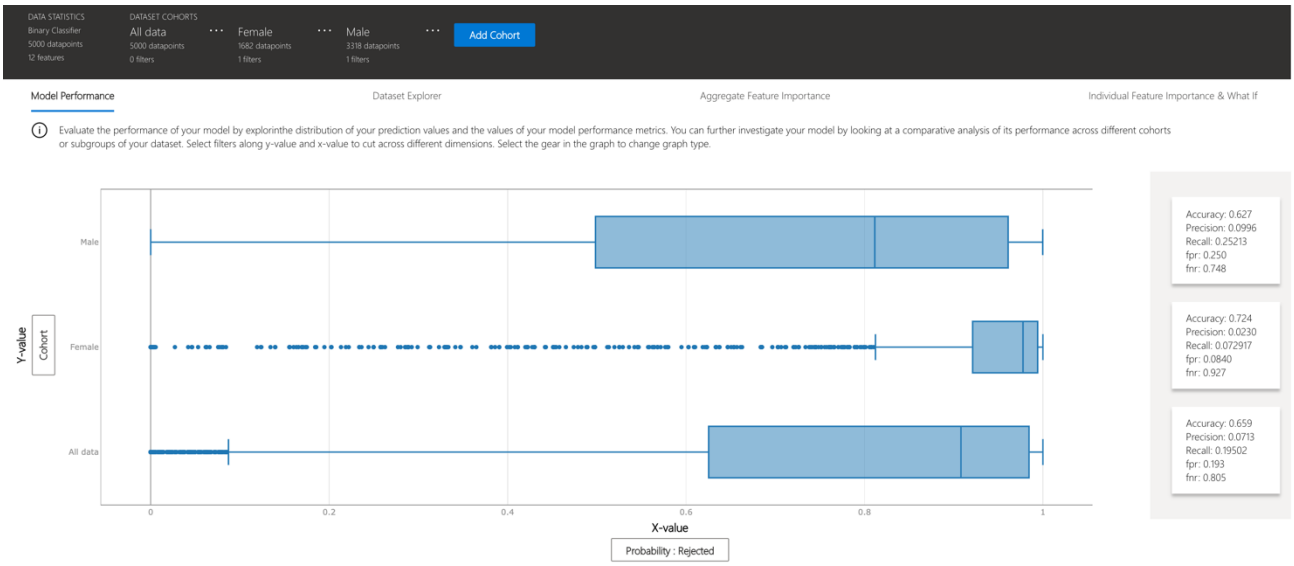
InterpretML also has another important debugging capability: **Counterfactual example analysis,** which computes the most similar data instances that have received different predictionsoutcomes. Counterfactual analysis generates explanations for individual outputs or predictions by identifying the smallest change to the input features that would cause the model or system to produce a desired output or prediction. For instance, in a loan-allocation model, feature-perturbed versions of the same loan application may show that the applicant would have received the loan, if, for example, the applicant’s income was $10,000 higher. This capability provides what-if explanations for model output and can be a useful complement to other explanation methods, both for end-users and model developers.

The interactive visualization dashboard

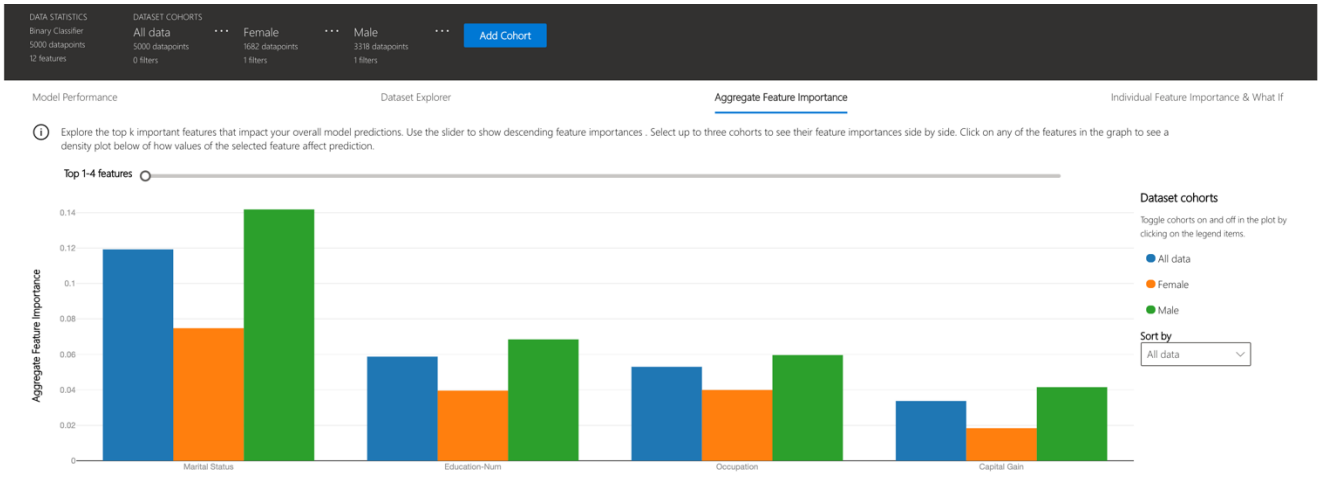
InterpretML’s dashboard provides interactive visualizations to help the user understand model performance for different subsets of data, explore model errors, and access dataset statistics and distributions. As mentioned, the dashboard also provides overall and individual explanations, using a variety of representations, and enables performing feature perturbations via what-if analysis for exploring how model predictions change as features are perturbed.

The visualization dashboard supports data filtering and cohort creation capabilities, enabling users to filter data and observe global and local feature-importance insights of a subset of data. They can further create cohorts (e.g., Age <= 35 and Age > 35) and compare model performance and global/local explanations of created subgroups.

Model performance tab: This tab shows model performance metrics across all data, and two subgroups of data (females and males). It further shows the distribution of rejection probability of loan applications of data points belonging to the female (second row) vs male (first row) group. In this case, females have consistently received higher prediction-probability scores of rejection compared to men.



Overall model explanations: This feature show the feature-importance values affecting the prediction for each subgroup (blue for all data, orange for females, green for males). It can be observed that marital status for instance is a lot more important for impacting predictions of the male group compared to the same feature impacting the prediction of the female group:



Individual feature importance factors: The user has clicked on a specific datapoint (from the female cohort) and is looking at the model prediction (reject with 68% probability) and the top factors

impacting that prediction (sex being the most important). The user can further perturb datapoint's feature values to observe how the model prediction changes.

Supported interpretability techniques

InterpretML introduces a new glassbox model, the Explainable Boosting Machine (EBM). EBM, developed by Microsoft Research, is an interpretable model that uses machine learning techniques like bagging, gradient boosting, and automatic interaction detection to breathe new life into traditional generalized additive models (GAMs). This makes EBMs as accurate as state-of-the-art techniques like random forests and gradient boosted trees. However, unlike these, which are black-box models, EBMs produce lossless explanations and are editable by domain experts. The following table summarizes all supported interpretability techniques in the Interpret repository:

Interpretability Technique	**Description**	**Type**
EBM	The decision tree can provide information about which concepts a model deems important, as well as provide an understanding how the concepts interact with each other.	glassbox model
Decision Tree	The decision tree can provide information about which concepts a model deems important, as well as provide an understanding how the concepts interact with each other.	glassbox model
Decision Rule List	A decision rule is a simple IF-THEN statement consisting of a condition (also called antecedent) and a prediction.	glassbox model
Linear Regression	A linear regression model predicts the target as a weighted sum of the feature inputs.	glassbox model

LogisticRegression	Logistic regression models the probabilities for classification problems with two possible outcomes. It's an extension of the linear regression model for classification problems.	glassbox model
SHAP Kernel Explainer	SHAP's Kernel explainer uses a specially weighted local linear regression to estimate SHAP values for **any model**.	Model-agnostic
SHAP Tree Explainer	SHAP's tree explainer, which focuses on polynomial time fast SHAP value estimation algorithm specific to **trees and ensembles of trees**.	Model-specific
LIME	LIME is an algorithm that can explain the predictions of any classifier or regressor in a faithful way, by approximating it locally with an interpretable model.	Model-agnostic
Morris Sensitivity Analysis	Sensitivity analysis provides insights into the importance of features by changing the feature values (or ignoring it) while all the other features staying constant, and seeing the output of the model.	Model-agnostic
Partial Dependence	The partial dependence method shows the marginal effect one or two features have on the predicted outcome of a machine learning model.	Model-agnostic

Interpret is extended by Interpret-Community, an experimental repository of additional interpretability methods and utility functions to handle real-world datasets and workflows:

Interpretability Technique	**Description**	**Type**
SHAP Deep Explainer	Based on the explanation from SHAP, Deep Explainer "is a high-speed approximation algorithm for SHAP values in deep learning models that builds on a connection with DeepLIFT described in the SHAP NIPS paper. **TensorFlow** models and **Keras** models using the TensorFlow backend are supported (there is also preliminary support for PyTorch)".	Model-specific
SHAP Linear Explainer	SHAP's Linear explainer computes SHAP values for a **linear model**, optionally accounting for inter-feature correlations.	Model-specific
Mimic Explainer (Global Surrogate)	Mimic explainer is based on the idea of training global surrogate models to mimic black-box models. A global surrogate model is an intrinsically interpretable model that is trained to approximate the predictions of **any black-box model** as accurately as possible. Data scientists can interpret the surrogate model to draw conclusions about the black-box model. You can use one of the following interpretable models as your surrogate model: LightGBM (LGBMExplainableModel), Linear Regression (LinearExplainableModel), Stochastic Gradient Descent explainable model (SGDExplainableModel), and Decision Tree (DecisionTreeExplainableModel).	Model-agnostic
Permutation Feature Importance Explainer (PFI)	Permutation Feature Importance is a technique used to explain classification and regression models that is inspired by Breiman's Random Forests paper (see section 10). At a high level, the way it works is by randomly shuffling data one feature at a time for the entire dataset and calculating how much the performance metric	Model-agnostic

	of interest changes. The larger the change, the more important that feature is. PFI can explain the overall behavior of **any underlying model** but does not explain individual predictions.	

InterpretML also provides intelligibility for text classification scenarios by supporting a variety of techniques that can explain BERT and RNN models (built via Pytorch). This involves a text-specific visualization dashboard that highlights and underlines important words in a document leading to a specific text classification class (e.g., classified as having a negative sentiment).

InterpretML as a community effort

AI is a rapidly evolving field, and interpretability in AI is all the more so. InterpretML is a community-driven open source project, to be shaped by people who seek to develop and deploy different intelligibility techniques in their model life cycle. Community members range from machine learning researchers seeking to contribute new intelligibility techniques to data scientists, developers, and business decision makers who aim to understand the impact of their AI systems on the lives of people affected by model predictions.

Where we want to go next

InterpretML will further invest on characterizing model errors and providing seamless end-to-end debugging scenarios augmented by intelligibility capabilities.

Contributing to InterpretML

Over the past few years, the machine learning transparency and interpretability landscape has been filled with techniques and OSS packages. However, having to call different explanation algorithms from a set of very distinct repositories (each with a different API), lack of unified visualizations to provide insights on model transparency, and lack of infrastructure to use trained explainers at inferencing time all have made the adoption of interpretability solutions slow and difficult. This effort started with the hope of accelerating adoption of advanced machine learning and deep learning algorithms across the academia and enterprise, particularly in regulated industries (e.g. finance, banking, insurance) with seamless access to different interpretability techniques and rich set of visualizations to enable interactive access to model predictions and explanations.

InterpretML is designed to make it simple for the research community to add new interpretability techniques comparable with the state-of-the-art techniques available in the toolkit.
Learn more at InterpretML on GitHub.