O'REILLY®

An Introduction to Machine Learning Interpretability

An Applied Perspective on Fairness, Accountability, Transparency, and Explainable AI

Patrick Hall & Navdeep Gill

REPORT

SECOND EDITION

An Introduction to Machine Learning Interpretability

An Applied Perspective on Fairness, Accountability, Transparency, and Explainable AI

Patrick Hall and Navdeep Gill

Beijing • Boston • Farnham • Sebastopol • Tokyo O'REILLY®

An Introduction to Machine Learning Interpretability, Second Edition
by Patrick Hall and Navdeep Gill

Printed in the United States of America.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Development Editor: Nicole Tache
Production Editor: Deborah Baker
Copyeditor: Christina Edwards
Proofreader: Charles Roumeliotis

Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

April 2018: First Edition
August 2019: Second Edition

Revision History for the Second Edition

2019-08-19: First Release
2019-10-04: Second Release

978-1-098-11547-0

[LSI]

Table of Contents

An Introduction to Machine Learning Interpretability

Understanding and trusting models and their results is a hallmark of good science. Analysts, engineers, physicians, researchers, scientists, and humans in general have the need to understand and trust models and modeling results that affect our work and our lives. For decades, choosing a model that was transparent to human practitioners or consumers often meant choosing straightforward data sources and simpler model forms such as linear models, single decision trees, or business rule systems. Although these simpler approaches were often the correct choice, and still are today, they can fail in real-world scenarios when the underlying modeled phenomena are nonlinear, rare or faint, or highly specific to certain individuals. Today, the trade-off between the accuracy and interpretability of predictive models has been broken (and maybe it never really existed). The tools now exist to build accurate and sophisticated modeling systems based on heterogeneous data and machine learning algorithms and to enable human understanding and trust in these complex systems. In short, you can now have your accuracy and interpretability cake...and eat it too.

To help practitioners make the most of recent and disruptive breakthroughs in debugging, explainability, fairness, and interpretability techniques for machine learning, this report defines key terms, introduces the human and commercial motivations for the techni-

1 Cynthia Rudin, "Please Stop Explaining Black Box Models for High-Stakes Decisions," arXiv:1811.10154, 2018, *https://arxiv.org/pdf/1811.10154.pdf*.

ques, and discusses predictive modeling and machine learning from an applied perspective, focusing on the common challenges of business adoption, internal model documentation, governance, validation requirements, and external regulatory mandates. We'll also discuss an applied taxonomy for debugging, explainability, fairness, and interpretability techniques and outline the broad set of available software tools for using these methods. Some general limitations and testing approaches for the outlined techniques are addressed, and finally, a set of open source code examples is presented.

Definitions and Examples

To facilitate detailed discussion and to avoid ambiguity, we present here definitions and examples for the following terms: *interpretable*, *explanation*, *explainable machine learning* or *artificial intelligence*, *interpretable* or *white-box models*, *model debugging*, and *fairness*.

Interpretable and explanation

> In the context of machine learning, we can define *interpretable* as "the ability to explain or to present in understandable terms to a human," from "Towards a Rigorous Science of Interpretable Machine Learning" by Doshi-Velez and Kim. (In the recent past, and according to the Doshi-Velez and Kim definition, *interpretable* was often used as a broader umbrella term. That is how we use the term in this report. Today, more leading researchers use *interpretable* to refer to directly transparent modeling mechanisms as discussed below.) For our working definition of a *good explanation* we can use "when you can no longer keep asking why," from "Explaining Explanations: An Approach to Evaluating Interpretability of Machine Learning" by Gilpin et al. These two thoughtful characterizations of *interpretable* and *explanation* link explanation to some machine learning process being interpretable and also provide a feasible, abstract objective for any machine learning explanation task.

2 Finale Doshi-Velez and Been Kim, "Towards a Rigorous Science of Interpretable Machine Learning," arXiv:1702.08608, 2017, *https://arxiv.org/pdf/1702.08608.pdf*.

3 Leilani H. Gilpin et al., "Explaining Explanations: An Approach to Evaluating Interpretability of Machine Learning," arXiv:1806.00069, 2018, *https://arxiv.org/pdf/1806.00069.pdf*.

Explainable machine learning

Getting even more specific, explainable machine learning, or explainable artificial intelligence (XAI), typically refers to post hoc analysis and techniques used to understand a previously trained model or its predictions. Examples of common techniques include:

Reason code generating techniques

In particular, local interpretable model-agnostic explanations (LIME) and Shapley values.[4]

Local and global visualizations of model predictions

Accumulated local effect (ALE) plots, one- and two-dimensional partial dependence plots, individual conditional expectation (ICE) plots, and decision tree surrogate models.[5][6][7][8][9]

XAI is also associated with a group of DARPA researchers that seem primarily interested in increasing explainability in sophisticated pattern recognition models needed for military and security applications.

Interpretable or white-box models

Over the past few years, more researchers have been designing new machine learning algorithms that are nonlinear and highly accurate, but also directly interpretable, and interpretable as a term has become more associated with these new models.

---

4 Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, "'Why Should I Trust You?': Explaining the Predictions of Any Classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM (2016): 1135–1144. *https://oreil.ly/2OQyGXx*.

5 Scott M. Lundberg and Su-In Lee, "A Unified Approach to Interpreting Model Predictions," in I. Guyon et al., eds., *Advances in Neural Information Processing Systems* 30 (Red Hook, NY: Curran Associates, Inc., 2017): 4765–4774. *https://oreil.ly/2OWsZYf*.

6 Daniel W. Apley, "Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models," arXiv:1612.08468, 2016, *https://arxiv.org/pdf/1612.08468.pdf*.

7 Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *The Elements of Statistical Learning*, Second Edition (New York: Springer, 2009). *https://oreil.ly/31FBpoe*.

8 Alex Goldstein et al., "Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation," *Journal of Computational and Graphical Statistics* 24, no. 1 (2015), *https://arxiv.org/pdf/1309.6392.pdf*.

9 Osbert Bastani, Carolyn Kim, and Hamsa Bastani, "Interpreting Blackbox Models via Model Extraction," arXiv:1705.08504, 2017, *https://arxiv.org/pdf/1705.08504.pdf*.

Examples of these newer Bayesian or constrained variants of traditional black-box machine learning models include explainable neural networks (XNNs), explainable boosting machines (EBMs), monotonically constrained gradient boosting machines, scalable Bayesian rule lists, and super-sparse linear integer models (SLIMs).[9] In this report, interpretable or white-box models will also include traditional linear models, decision trees, and business rule systems. Because interpretable is now often associated with a model itself, traditional black-box machine learning models, such as multilayer perceptron (MLP) neural networks and gradient boosting machines (GBMs), are said to be uninterpretable in this report. As explanation is currently most associated with post hoc processes, unconstrained, black-box machine learning models are usually also said to be at least partially explainable by applying explanation techniques after model training. Although difficult to quantify, credible research efforts into scientific measures of model interpretability are also underway. The ability to measure degrees implies interpretability is not a binary, on-off quantity. So, there are shades of interpretability between the most transparent white-box model and the most opaque black-box model. Use more interpretable models for high-stakes applications or applications that affect humans.

Model debugging

Refers to testing machine learning models to increase trust in model mechanisms and predictions. Examples of model debugging techniques include variants of sensitivity (i.e., "What if?")

---

10 Joel Vaughan et al., "Explainable Neural Networks Based on Additive Index Models," arXiv:1806.01933, 2018, *https://arxiv.org/pdf/1806.01933.pdf*.

11 Hongyu Yang, Cynthia Rudin, and Margo Seltzer, "Scalable Bayesian Rule Lists," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017, *https://arxiv.org/pdf/1602.08610.pdf*.

12 Berk Ustun and Cynthia Rudin, "Supersparse Linear Integer Models for Optimized Medical Scoring Systems," *Machine Learning* 102, no. 3 (2016): 349–391, *https://oreil.ly/31CyzjV*.

13 Microsoft Interpret GitHub Repository: *https://oreil.ly/2z275YJ*.

14 Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl, "Quantifying Interpretability of Arbitrary Machine Learning Models Through Functional Decomposition," arXiv: 1904.03867, 2019, *https://arxiv.org/pdf/1904.03867.pdf*.

15 Debugging Machine Learning Models: *https://debug-ml-iclr2019.github.io*.

analysis, residual analysis, prediction assertions, and unit tests to verify the accuracy or security of machine learning models. Model debugging should also include remediating any discovered errors or vulnerabilities.

Fairness

Fairness is an extremely complex subject and this report will focus mostly on the more straightforward concept of disparate impact (i.e., when a model's predictions are observed to be different across demographic groups, beyond some reasonable threshold, often 20%). Here, fairness techniques refer to disparate impact analysis, model selection by minimization of disparate impact, remediation techniques such as disparate impact removal preprocessing, equalized odds postprocessing, or several additional techniques discussed in this report.[16] The group Fairness, Accountability, and Transparency in Machine Learning (FATML) is often associated with fairness techniques and research for machine learning, computer science, law, various social sciences, and government. Their site hosts useful resources for practitioners such as full lists of relevant scholarship and best practices.

Social and Commercial Motivations for Machine Learning Interpretability

> *The now-contemplated field of data science amounts to a superset of the fields of statistics and machine learning, which adds some technology for "scaling up" to "big data." This chosen superset is motivated by commercial rather than intellectual developments. Choosing in this way is likely to miss out on the really important intellectual event of the next 50 years.*
>
> —David Donoho

16 Michael Feldman et al., "Certifying and Removing Disparate Impact," in *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM (2015): 259–268. *https://arxiv.org/pdf/1412.3756.pdf*.

17 Moritz Hardt et al., "Equality of Opportunity in Supervised Learning," in *Advances in Neural Information Processing Systems* (2016): 3315–3323. *https://oreil.ly/2KyRdnd*.

18 David Donoho, "50 Years of Data Science," Tukey Centennial Workshop, 2015, *http://bit.ly/2GQOh1J*.

Among many other applications, machine learning is used today to make life-altering decisions about employment, bail, parole, and lending. Furthermore, usage of AI and machine learning models is likely to become more commonplace as larger swaths of the economy embrace automation and data-driven decision making. Because artificial intelligence, and its to-date most viable subdiscipline of machine learning, has such broad and disruptive applications, let's heed the warning from Professor Donoho and focus first on the intellectual and social motivations for more interpretability in machine learning.

Intellectual and Social Motivations

Intellectual and social motivations boil down to trust and understanding of an exciting, revolutionary, but also potentially dangerous technology. Trust and understanding are overlapping, but also different, concepts and goals. Many of the techniques discussed in this report are helpful for both, but better suited to one or the other. Trust is mostly related to the accuracy, fairness, and security of machine learning systems as implemented through model debugging and disparate impact analysis and remediation techniques. Understanding is mostly related to the transparency of machine learning systems, such as directly interpretable models and explanations for each decision a system generates.

Human trust of machine learning models

As consumers of machine learning, we need to know that any automated system generating a decision that effects us is secure and accurate and exhibits minimal disparate impact. An illustrative example of problems and solutions for trust in machine learning is the Gender Shades project and related follow-up work. As part of the Gender Shades project, an accuracy and disparate impact problem was discovered and then debugged in several commercial facial recognition systems. These facial recognition systems exhibited highly disparate levels of accuracy across men and women and across skin tones. Not only were these cutting-edge models wrong in many cases, they were consistently wrong more often for women and people with darker skin tones. Once Gender Shades researchers pointed out these problems, the organizations they targeted took remediation steps including creating more diverse training datasets and devising ethical standards for machine learning projects. In

most cases, the result was more accurate models with less disparate impact, leading to much more trustworthy machine learning systems. Unfortunately, at least one well-known facial recognition system disputed the concerns highlighted by Gender Shades, likely damaging their trustworthiness with machine learning consumers.

Hacking and adversarial attacks on machine learning systems are another wide-ranging and serious trust problem. In 2017, researchers discovered that slight changes, such as applying stickers, can prevent machine learning systems from recognizing street signs. These physical adversarial attacks, which require almost no software engineering expertise, can obviously have severe societal consequences. For a hacker with more technical expertise, many more types of attacks against machine learning are possible. Models and even training data can be manipulated or stolen through public APIs or other model endpoints. So, another key to establishing trust in machine learning is ensuring systems are secure and behaving as expected in real time. Without interpretable models, debugging, explanation, and fairness techniques, it can be very difficult to determine whether a machine learning system's training data has been compromised, whether its outputs have been altered, or whether the system's inputs can be changed to create unwanted or unpredictable decisions. Security is as important for trust as accuracy or fairness, and the three are inextricably related. All the testing you can do to prove a model is accurate and fair doesn't really matter if the data or model can be altered later without your knowledge.

Human understanding of machine learning models

Consumers of machine learning also need to know exactly how any automated decision that affects us is made. There are two intellectual drivers of this need: one, to facilitate human learning from machine learning, and two, to appeal wrong machine learning decisions. Exact explanation of machine-learned decisions is one of the most fundamental applications of machine learning interpretability technologies. Explanation enables humans to learn how machine

19 Kevin Eykholt et al., "Robust Physical-World Attacks on Deep Learning Visual Classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018): 1625-1634. *https://oreil.ly/2yX8W11*.

20 Patrick Hall, "Proposals for Model Vulnerability and Security," O'Reilly.com (Ideas), March 20, 2019. *https://oreil.ly/308qKm0*.

learning systems make decisions, which can satisfy basic curiosity or lead to new types of data-driven insights. Perhaps more importantly, explanation provides a basis for the appeal of automated decisions made by machine learning models. Consider being negatively impacted by an erroneous black-box model decision, say for instance being wrongly denied a loan or parole. How would you argue your case for appeal without knowing how model decisions were made? According to the *New York Times*, a man named Glenn Rodríguez found himself in this unfortunate position in a penitentiary in upstate New York in 2016. Without information about exactly why a proprietary black-box model was mistakenly recommending he remain in prison, he was unable to build a direct case to appeal that decision. Like the problems exposed by the Gender Shades study, the inability to appeal automated decisions is not some far-off danger on the horizon, it's a present danger. Fortunately, the technologies exist today to explain even very complex model decisions, and once understanding and trust can be assured, broader possibilities for the use of machine learning come into view.

Guaranteeing the promise of machine learning

One of the greatest hopes for data science and machine learning is simply increased convenience, automation, and organization in our day-to-day lives. Even today, we are beginning to see fully automated baggage scanners at airports and our phones are constantly recommending new music (that we might actually like). As these types of automation and conveniences grow more common, consumers will likely want to understand them more deeply and machine learning engineers will need more and better tools to debug these evermore present decision-making systems. Machine learning also promises quick, accurate, and unbiased decision making in life-changing scenarios. Computers can theoretically use machine learning to make objective, data-driven decisions in critical situations like criminal convictions, medical diagnoses, and college admissions, but interpretability, among other technological advances, is needed to guarantee the promises of correctness and objectivity. Without extremely high levels of trust and understanding in machine learning decisions, there is no certainty that a machine learning system is

21 Rebecca Wexler, "When a Computer Program Keeps You in Jail," *New York Times*, June 13, 2017, *https://oreil.ly/2TyHIr5*.

not simply relearning and reapplying long-held, regrettable, and erroneous human biases. Nor are there any assurances that human operators, or hackers, have not forced a machine learning system to make intentionally prejudicial or harmful decisions.

Commercial Motivations

Companies and organizations use machine learning and predictive models for a very wide variety of revenue- or value-generating applications. Just a few examples include facial recognition, lending decisions, hospital release decisions, parole release decisions, or generating customized recommendations for new products or services. Many principles of applied machine learning are shared across industries, but the practice of machine learning at banks, insurance companies, healthcare providers, and in other regulated industries is often quite different from machine learning as conceptualized in popular blogs, the news and technology media, and academia. It's also somewhat different from the practice of machine learning in the technologically advanced and less regulated digital, ecommerce, FinTech, and internet verticals.

In commercial practice, concerns regarding machine learning algorithms are often overshadowed by talent acquisition, data engineering, data security, hardened deployment of machine learning apps and systems, managing and monitoring an ever-increasing number of predictive models, modeling process documentation, and regulatory compliance. Successful entities in both traditional enterprise and in modern digital, ecommerce, FinTech, and internet verticals have learned to balance these competing business interests. Many digital, ecommerce, FinTech, and internet companies, operating outside of most regulatory oversight, and often with direct access to web-scale, and sometimes unethically sourced, data stores, have often made web data and machine learning products central to their business. Larger, more established companies tend to practice statistics, analytics, and data mining at the margins of their business to optimize revenue or allocation of other valuable assets. For all these reasons, commercial motivations for interpretability vary across industry verticals, but center around improved margins for previ-

22 Patrick Hall, Wen Phan, and Katie Whitson, *The Evolution of Analytics* (Sebastopol, CA: O'Reilly Media, 2016). *https://oreil.ly/2Z3eBxk*.

ously existing analytics projects, business partner and customer adoption of new machine learning products or services, regulatory compliance, and lessened model and reputational risk.

Enhancing established analytical processes

For traditional and often more-regulated commercial applications, machine learning can enhance established analytical practices, typically by increasing prediction accuracy over conventional but highly interpretable linear models. Machine learning can also enable the incorporation of unstructured data into analytical pursuits, again leading to more accurate model outcomes in many cases. Because linear models have long been the preferred tools for predictive modeling, many practitioners and decision-makers are simply suspicious of machine learning. If nonlinear models—generated by training machine learning algorithms—make more accurate predictions on previously unseen data, this typically translates into improved financial margins...but only if the model is accepted by internal validation teams, business partners, and customers. Interpretable machine learning models and debugging, explanation, and fairness techniques can increase understanding and trust in newer or more robust machine learning approaches, allowing more sophisticated and potentially more accurate models to be used in place of previously existing linear models.

Regulatory compliance

Interpretable, fair, and transparent models are simply a legal mandate in certain parts of the banking, insurance, and healthcare industries. Because of increased regulatory scrutiny, these more traditional companies typically must use techniques, algorithms, and models that are simple and transparent enough to allow for detailed documentation of internal system mechanisms and in-depth analysis by government regulators. Some major regulatory statutes currently governing these industries include the Civil Rights Acts of 1964 and 1991, the Americans with Disabilities Act, the Genetic Information Nondiscrimination Act, the Health Insurance Portability and Accountability Act, the Equal Credit Opportunity Act (ECOA), the Fair Credit Reporting Act (FCRA), the Fair Housing Act, Federal Reserve SR 11-7, and European Union (EU) Greater

23 Fast Forward Labs—Interpretability. *https://oreil.ly/301LuM4*.

Data Privacy Regulation (GDPR) Article 22. These regulatory regimes, key drivers of what constitutes interpretability in applied machine learning, change over time or with political winds.

Tools like those discussed in this report are already used to document, understand, and validate different types of models in the financial services industry (and probably others). Many organizations are now also experimenting with machine learning and the reason codes or adverse actions notices that are mandated under ECOA and FCRA for credit lending, employment, and insurance decisions in the United States. If newer machine learning approaches are used for such decisions, those decisions must be explained in terms of adverse action notices. Equifax's NeuroDecision is a great example of constraining a machine learning technique (an MLP) to be interpretable, using it to make measurably more accurate predictions than a linear model, and doing so in a regulated space. To make automated credit-lending decisions, NeuroDecision uses modified MLPs, which are somewhat more accurate than conventional regression models and also produce the regulator-mandated adverse action notices that explain the logic behind a credit-lending decision. NeuroDecision's increased accuracy could lead to credit lending in a broader portion of the market than previously possible, such as new-to-credit consumers, increasing the margins associated with the preexisting linear model techniques.[26] Shapley values, and similar local variable importance approaches we will discuss later, also provide a convenient methodology to rank the contribution of input variables to machine learning model decisions and potentially generate customer-specific adverse action notices.

Adoption and acceptance

For digital, ecommerce, FinTech, and internet companies today, interpretability is often an important but secondary concern. Less-traditional and typically less-regulated companies currently face a greatly reduced burden when it comes to creating transparent and

24 Andrew Burt, "How Will the GDPR Impact Machine Learning?" O'Reilly.com (Ideas), May 16, 2018, *https://oreil.ly/304nxDI*.

25 Hall et al., *The Evolution of Analytics*.

26 Bob Crutchfield, "Approve More Business Customers," Equifax Insights Blog, March 16, 2017, *https://oreil.ly/2NcWnar*.

trustworthy machine learning products or services. Even though transparency into complex data and machine learning products might be necessary for internal debugging, validation, or business adoption purposes, many newer firms are not compelled by regulation to prove their models are accurate, transparent, or nondiscriminatory. However, as the apps and systems that such companies create (often based on machine learning) continue to change from occasional conveniences or novelties into day-to-day necessities, consumer and government demand for accuracy, fairness, and transparency in these products will likely increase.

Reducing risk

No matter what space you are operating in as a business, hacking of prediction APIs or other model endpoints and discriminatory model decisions can be costly, both to your reputation and to your bottom line. Interpretable models, model debugging, explanation, and fairness tools can mitigate both of these risks. While direct hacks of machine learning models still appear rare, there are numerous documented hacking methods in the machine learning security literature, and several simpler insider attacks that can change your model outcomes to benefit a malicious actor or deny service to legitimate customers.[27] You can use explanation and debugging tools in white-hat hacking exercises to assess your vulnerability to adversarial example, membership inference, and model stealing attacks. You can use fair (e.g., learning fair representations, LFR) or private (e.g., private aggregation of teaching ensembles, PATE) models as an active measure to prevent many attacks.[29] Also, real-time disparate impact monitoring can alert you to data poisoning attempts to

27 Marco Barreno et al., "The Security of Machine Learning," *Machine Learning* 81, no. 2 (2010): 121–148. *https://oreil.ly/31JwoLL*.

28 Reza Shokri et al., "Membership Inference Attacks Against Machine Learning Models," IEEE Symposium on Security and Privacy (SP), 2017, *https://oreil.ly/2Z22LHI*.

29 Nicholas Papernot, "A Marauder's Map of Security and Privacy in Machine Learning: An Overview of Current and Future Research Directions for Making Machine Learning Secure and Private," in *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, ACM, 2018, *https://arxiv.org/pdf/1811.01134.pdf*.

30 Rich Zemel et al., "Learning Fair Representations," in *International Conference on Machine Learning* (2013): 325-333. *https://oreil.ly/305wjBE*.

31 Nicolas Papernot et al., "Scalable Private Learning with PATE," arXiv:1802.08908, 2018, *https://arxiv.org/pdf/1802.08908.pdf*.

change your model behavior to benefit or harm certain groups of people. Moreover, basic checks for disparate impact should always be conducted if your model will affect humans. Even if your company can't be sued for noncompliance under FCRA or ECOA, it can be called out in the media for deploying a discriminatory machine learning model or violating customer privacy. As public awareness of security vulnerabilities and algorithmic discrimination grows, don't be surprised if a reputational hit in the media results in customers taking business elsewhere, causing real financial losses.

A Machine Learning Interpretability Taxonomy for Applied Practitioners

A heuristic, practical, and previously defined taxonomy is presented in this section. This taxonomy will be used to characterize the interpretability of various popular machine learning and statistics techniques used in commercial data mining, analytics, data science, and machine learning applications. This taxonomy describes approaches in terms of:

- Their ability to promote understanding and trust
- Their complexity
- The global or local scope of information they generate
- The families of algorithms to which they can be applied

Technical challenges as well as the needs and perspectives of different user communities make characterizing machine learning interpretability techniques a subjective and complicated task. *Many* other authors have grappled with organizing and categorizing a variety of general concepts related to interpretability and explanations. Some of these efforts include: "A Survey of Methods for Explaining Black Box Models" by Riccardo Guidotti et al., "The Mythos of Model Interpretability" by Zachary Lipton, "Interpretable Machine Learn-

32 Patrick Hall, Wen Phan, and SriSatish Ambati, "Ideas on Interpreting Machine Learning," O'Reilly.com (Ideas), March 15, 2017, *https://oreil.ly/2H4aIC8*.

33 Riccardo Guidotti et al., "A Survey of Methods for Explaining Black Box Models," *ACM Computing Surveys (CSUR)* 51, no. 5 (2018): 93. *https://arxiv.org/pdf/1802.01933.pdf*.

34 Zachary C. Lipton, "The Mythos of Model Interpretability," arXiv:1606.03490, 2016, *https://arxiv.org/pdf/1606.03490.pdf*.

ing" by Christoph Molnar, "Interpretable Machine Learning: Definitions, Methods, and Applications" by W. James Murdoch et al., and "Challenges for Transparency" by Adrian Weller. Interested readers are encouraged to dive into these more technical, detailed, and nuanced analyses too!

Understanding and Trust

Some interpretability techniques are more geared toward fostering understanding, some help engender trust, and some enhance both. Trust and understanding are different, but not orthogonal, phenomena. Both are also important goals for any machine learning project. Understanding through transparency is necessary for human learning from machine learning, for appeal of automated decisions, and for regulatory compliance. The discussed techniques enhance understanding by either providing transparency and specific insights into the mechanisms of the algorithms and the functions they create or by providing detailed information for the answers they provide. Trust grows from the tangible accuracy, fairness, and security of machine learning systems. The techniques that follow enhance trust by enabling users to observe or ensure the fairness, stability, and dependability of machine learning algorithms, the functions they create, and the answers they generate.

A Scale for Interpretability

The complexity of a machine learning model is often related to its interpretability. Generally, the more complex and unconstrained the model, the more difficult it is to interpret and explain. The number of weights or rules in a model or its *Vapnik–Chervonenkis dimension*, a more formal measure, are good ways to quantify a model's complexity. However, analyzing the functional form of a model is particularly useful for commercial applications such as credit scoring. The following list describes the functional forms of models and discusses their degree of interpretability in various use cases.

---

35 Christoph Molnar, *Interpretable Machine Learning* (christophm.github.io: 2019), *https://oreil.ly/2YI5ruC*.

36 W. James Murdoch et al., "Interpretable Machine Learning: Definitions, Methods, and Applications," arXiv:1901.04592, 2019, *https://arxiv.org/pdf/1901.04592.pdf*.

37 Adrian Weller, "Challenges for Transparency," arXiv:1708.01870, 2017, *https://arxiv.org/pdf/1708.01870.pdf*.

High interpretability: linear, monotonic functions

Functions created by traditional regression algorithms are probably the most interpretable class of models. We refer to these models here as "linear and monotonic," meaning that for a change in any given input variable (or sometimes combination or function of an input variable), the output of the response function changes at a defined rate, in only one direction, and at a magnitude represented by a readily available coefficient. Monotonicity also enables intuitive and even automatic reasoning about predictions. For instance, if a credit lender rejects your credit card application, it can easily tell you why because its probability-of-default model often assumes your credit score, your account balances, and the length of your credit history are monotonically related to your ability to pay your credit card bill. When these explanations are created automatically, they are typically called *adverse action notices* or *reason codes*. Linear, monotonic functions play another important role in machine learning interpretability. Besides being highly interpretable themselves, linear and monotonic functions are also used in explanatory techniques, including the popular LIME approach.

Medium interpretability: nonlinear, monotonic functions

Although most machine-learned response functions are nonlinear, some can be constrained to be monotonic with respect to any given independent variable. Although there is no single coefficient that represents the change in the response function output induced by a change in a single input variable, nonlinear and monotonic functions do always change in one direction as a single input variable changes. They usually allow for the generation of plots that describe their behavior and both reason codes and variable importance measures. Nonlinear, monotonic response functions are therefore fairly interpretable and potentially suitable for use in regulated applications.

(Of course, there are linear, nonmonotonic machine-learned response functions that can, for instance, be created by the multivariate adaptive regression splines (MARS) approach. These functions could be of interest for your machine learning project and they likely share the medium interpretability characteristics of nonlinear, monotonic functions.)

Low interpretability: nonlinear, nonmonotonic functions

Most machine learning algorithms create nonlinear, nonmonotonic response functions. This class of functions is the most difficult to interpret, as they can change in a positive and negative direction and at a varying rate for any change in an input variable. Typically, the only standard interpretability measures these functions provide are relative variable importance measures. You should use a combination of several techniques, presented in the sections that follow, to interpret, explain, debug, and test these extremely complex models. You should also consider the accuracy, fairness, and security problems associated with black-box machine learning before deploying a nonlinear, nonmonotonic model for any application with high stakes or that affects humans.

Global and Local Interpretability

It's often important to understand and test your trained model on a global scale, and also to zoom into local regions of your data or your predictions and derive local information. Global measures help us understand the inputs and their entire modeled relationship with the prediction target, but global interpretations can be highly approximate in some cases. Local information helps us understand our model or predictions for a single row of data or a group of similar rows. Because small parts of a machine-learned response function are more likely to be linear, monotonic, or otherwise well-behaved, local information can be more accurate than global information. It's also very likely that the best analysis of a machine learning model will come from combining the results of global and local interpretation techniques. In subsequent sections we will use the following descriptors to classify the scope of an interpretable machine learning approach:

Global interpretability

Some machine learning interpretability techniques facilitate global measurement of machine learning algorithms, their results, or the machine-learned relationships between the prediction target(s) and the input variables across entire partitions of data.

Local interpretability

Local interpretations promote understanding of small regions of the machine-learned relationship between the prediction target(s) and the input variables, such as clusters of input records and their corresponding predictions, or deciles of predictions and their corresponding input rows, or even single rows of data.

Model-Agnostic and Model-Specific Interpretability

Another important way to classify model interpretability techniques is to determine whether they are *model agnostic*, meaning they can be applied to different types of machine learning algorithms, or *model specific*, meaning techniques that are applicable only for a single type or class of algorithm. For instance, the LIME technique is model agnostic and can be used to interpret nearly any set of machine learning inputs and machine learning predictions. On the other hand, the technique known as *Tree SHAP* is model specific and can be applied only to decision tree models. Although model-agnostic interpretability techniques are convenient, and in some ways ideal, they often rely on surrogate models or other approximations that can degrade the accuracy of the information they provide. Model-specific interpretation techniques tend to use the model to be interpreted directly, leading to potentially more accurate measurements.

Common Interpretability Techniques

Many credible techniques for training interpretable models and gaining insights into model behavior and mechanisms have existed for years. Many others have been put forward in a recent flurry of research. This section of the report discusses many such techniques in terms of the proposed machine learning interpretability taxonomy. The section begins by discussing data visualization approaches because having a strong understanding of a dataset is a first step toward validating, explaining, and trusting models. We then present white-box modeling techniques, or models with directly interpretable inner workings, followed by techniques that can generate explanations for the most complex types of predictive models such as model visualizations, reason codes, and global variable importance measures. We conclude the section by discussing approaches for testing and debugging machine learning models for fairness, stability, and trustworthiness. The techniques introduced in this sec-

tion will get you well on your way to using interpretable models and debugging, explanation, and fairness techniques.

Seeing and Understanding Your Data

Seeing and understanding data is important for interpretable machine learning because models represent data, and understanding the contents of that data helps set reasonable expectations for model behavior and output. Unfortunately, most real datasets are difficult to see and understand because they have many variables and many rows. Even though plotting many dimensions is technically possible, doing so often detracts from, instead of enhances, human understanding of complex datasets. Of course, there are many, many ways to visualize datasets. We chose the techniques highlighted in Tables 1-1 and 1-2 and in Figure 1-1 because they help illustrate many important aspects of a dataset in just two dimensions.

Table 1. A description of 2D projection data visualization approaches

Technique: 2D projections
Description: Projecting rows of a dataset from a usually high-dimensional original space into a more visually understandable lower-dimensional space, ideally two or three dimensions. Some techniques to achieve this include principal components analysis (PCA), multidimensional scaling (MDS), t-distributed stochastic neighbor embedding (t-SNE), and autoencoder networks.
Suggested usage: The key idea is to represent the rows of a dataset in a meaningful low-dimensional space. Datasets containing images, text, or even business data with many variables can be difficult to visualize as a whole. These projection techniques enable high-dimensional datasets to be projected into representative low-dimensional spaces and visualized using the trusty old scatter plot technique. A high-quality projection visualized in a scatter plot should exhibit key structural elements of a dataset, such as clusters, hierarchy, sparsity, and outliers. 2D projections are often used in fraud or anomaly detection to find outlying entities, like people, transactions, or computers, or unusual clusters of entities.
References: Laurens van der Maaten and Geoffrey Hinton, "Visualizing Data Using t-SNE," *Journal of Machine Learning Research*, 9 (2008): 2579-2605. *https://oreil.ly/2KlAaOtf*. T. F. Cox, *Multidimensional Scaling* (London: Chapman and Hall, 2001). Hastie et al., *The Elements of Statistical Learning*, Second Edition. G. E. Hinton and R. R. Salakhutdinov. "Reducing the Dimensionality of Data with Neural Networks." *Science*, 13, July 28, 2006. *https://oreil.ly/2yZbCvi*.
OSS: h2o.ai (note the H2O Aggregator sampling routine) R (various packages) scikit-learn (various functions)

Global or local scope: Global and local. Can be used globally to see a coarser view of the entire dataset, or provide granular views of local portions of the dataset by panning, zooming, and drill-down.	
Best-suited complexity: Any. 2D projections can help us understand very complex relationships in datasets and models.	**Model specific or model agnostic:** Model agnostic; visualizing complex datasets with many variables.
Trust and understanding: Projections add a degree of trust if they are used to confirm machine learning modeling results. For instance, if known hierarchies, classes, or clusters exist in training or test datasets and these structures are visible in 2D projections, it is possible to confirm that a machine learning model is labeling these structures correctly. A secondary check is to confirm that similar attributes of structures are projected relatively near one another and different attributes of structures are projected relatively far from one another. Consider a model used to classify or cluster marketing segments. It is reasonable to expect a machine learning model to label older, richer customers differently than younger, less affluent customers, and moreover to expect that these different groups should be relatively disjoint and compact in a projection, and relatively far from one another.	

Table 2. A description of the correlation network graph data visualization approach

Technique: Correlation network graphs	
Description: A correlation network graph is a 2D representation of the relationships (correlation) in a dataset. The authors create correlation graphs in which the nodes of the graph are the variables in a dataset and the edge weights (thickness) between the nodes are defined by the absolute values of their pairwise Pearson correlation. For visual simplicity, absolute weights below a certain threshold are not displayed, the node size is determined by a node's number of connections (node degree), node color is determined by a graph community calculation, and node position is defined by a graph force field algorithm. The correlation graph allows us to see groups of correlated variables, identify irrelevant variables, and discover or verify important, complex relationships that machine learning models should incorporate, all in two dimensions.	
Suggested usage: Correlation network graphs are especially powerful in text mining or topic modeling to see the relationships between entities and ideas. Traditional network graphs—a similar approach—are also popular for finding relationships between customers or products in transactional data and for use in fraud detection to find unusual interactions between entities like people or computers.	
OSS: Gephi corr_graph	
Global or local scope: Global and local. Can be used globally to see a coarser view of the entire dataset, or provide granular views of local portions of the dataset by panning, zooming, and drilling down.	
Best-suited complexity: Any, but becomes difficult to understand with more than several thousand variables.	**Model specific or model agnostic:** Model agnostic; visualizing complex datasets with many variables.

Trust and understanding: Correlation network graphs promote understanding by displaying important and complex relationships in a dataset. They can enhance trust in a model if variables with thick connections to the target are important variables in the model, and we would expect a model to learn that unconnected variables are not very important. Also, common sense relationships displayed in the correlation graph should be reflected in a trustworthy model.

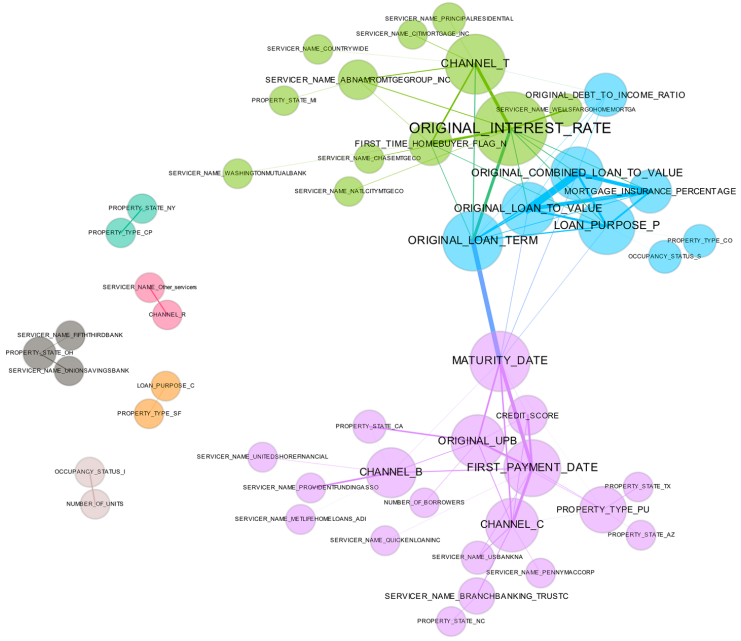

Figure 1. A correlation network graph is helpful for enhancing trust and understanding in machine learning models because it displays important, complex relationships between variables in a dataset as edges and nodes in an undirected graph. (Figure courtesy of H2O.ai.)

Techniques for Creating White-Box Models

When starting a machine learning endeavor, it's a best practice to determine to what degree your model could impact human beings or be used for other high-stakes decisions. In these high-stakes cases, maximum transparency safeguards against fairness and security issues. Also, with newer white-box modeling methods like XNN, monotonic GBM, and scalable Bayesian rule lists, interpretability comes at a minimal accuracy penalty, if any at all. Starting with an interpretable model will likely make subsequent debugging, explanation, and fairness auditing tasks easier too. The techniques

in Tables 1-3 through 1-8 will enable you to create highly transparent models, potentially well-suited for regulated industry or other vital applications in which interpretability is of extreme importance.

Table 3. A description of the decision tree white-box modeling approach

Technique: Decision trees
Description: Decision trees create a model that predicts the value of a target variable based on several input variables. Decision trees are directed graphs in which each interior node corresponds to an input variable. There are edges to child nodes for values of the input variable that creates the highest target purity in each child. Each terminal node or leaf node represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf. These paths can be visualized or explained with simple if-then rules. In short, decision trees are data-derived flowcharts.
Suggested usage: Decision trees are great for training simple, transparent models on IID data—data where a unique customer, patient, product, or other entity is represented in each row. They are beneficial when the goal is to understand relationships between the input and target variable with "Boolean-like" logic. Decision trees can also be displayed graphically in a way that is easy for nonexperts to interpret.
References: L. Breiman et al., *Classification and Regression Trees* (Boca Raton, FL: CRC Press, 1984). Hastie et al., *The Elements of Statistical Learning*, Second Edition.
OSS: rpart scikit-learn (various functions)
Global or local scope: Global.
Best-suited complexity: Low to medium. Decision trees can be complex nonlinear, nonmonotonic functions, but their accuracy is sometimes lower than more sophisticated models for complex problems and large trees can be difficult to interpret.
Model specific or model agnostic: Model specific.
Trust and understanding: Increases trust and understanding because input to target mappings follows a decision structure that can be easily visualized, interpreted, and compared to domain knowledge and reasonable expectations.

Table 4. A description of the XNN modeling approach and artificial neural network (ANN) explanations

Technique: XNN and ANN explanations	
Description: XNN, a new type of constrained artificial neural network, and new model-specific explanation techniques have recently made ANNs much more interpretable and explainable. Many of the breakthroughs in ANN explanation stem from derivatives of the trained ANN with respect to input variables. These derivatives disaggregate the trained ANN response function prediction into input variable contributions. Calculating these derivatives is much easier than it used to be due to the proliferation of deep learning toolkits such as Tensorflow.	
Suggested usage: While most users will be familiar with the widespread use of ANNs in pattern recognition, they are also used for more traditional data mining applications such as fraud detection, and even for regulated applications such as credit scoring. Moreover, ANNs can now be used as accurate and explainable surrogate models, potentially increasing the fidelity of both global and local surrogate model techniques.	
References: M. Ancona et al., "Towards Better Understanding of Gradient-Based Attribution Methods for Deep Neural Networks," ICLR 2018. *https://oreil.ly/2H6v1yz*. Joel Vaughan et al. "Explainable Neural Networks Based on Additive Index Models," arXiv: 1806.01933, 2018. *https://arxiv.org/pdf/1806.01933.pdf*.	
OSS: DeepLift Integrated-Gradients shap Skater	
Global or local scope: XNNs are globally interpretable. Local ANN explanation techniques can be applied to XNNs or nonconstrained ANNs.	
Best-suited complexity: Any. XNNs can be used to directly model nonlinear, nonmonotonic phenomena but today they often require manual variable selection. ANN explanation techniques can be used for very complex models.	**Model specific or model agnostic:** As directly interpretable models, XNNs rely on model-specific mechanisms. Used as surrogate models, XNNs are model agnostic. ANN explanation techniques are generally model specific.
Trust and understanding: XNN techniques are typically used to make ANN models themselves more understandable or as surrogate models to make other nonlinear models more understandable. ANN explanation techniques make ANNs more understandable.	

Table 5. A description of the monotonic gradient boosting machine (GBM) white-box modeling approach

Technique: Monotonic GBMs	
Description: Monotonicity constraints can turn difficult-to-interpret nonlinear, nonmonotonic models into interpretable, nonlinear, monotonic models. One application of this can be achieved with monotonicity constraints in GBMs by enforcing a uniform splitting strategy in constituent decision trees, where binary splits of a variable in one direction always increase the average value of the dependent variable in the resultant child node, and binary splits of the variable in the other direction always decrease the average value of the dependent variable in the other resultant child node.	
Suggested usage: Potentially appropriate for most traditional data mining and predictive modeling tasks, even in regulated industries, and potentially for consistent adverse action notice or reason code generation (which is often considered a gold standard of model explainability).	
Reference: XGBoost Documentation	
OSS: h2o.ai XGBoost Interpretable Machine Learning with Python	
Global or local scope: Global.	**Best-suited complexity:** Medium to high. Monotonic GBMs create nonlinear, monotonic response functions.
Model specific or model agnostic: As implementations of monotonicity constraints vary for different types of models in practice, they are a model-specific interpretation technique.	**Trust and understanding:** Understanding is increased by enforcing straightforward relationships between input variables and the prediction target. Trust is increased when monotonic relationships, reason codes, and detected interactions are parsimonious with domain expertise or reasonable expectations.

Table 6. A description of alternative regression white-box modeling approaches

Technique: Logistic, elastic net, and quantile regression and generalized additive models (GAMs)
Description: These techniques use contemporary methods to augment traditional, linear modeling methods. Linear model interpretation techniques are highly sophisticated and typically model specific, and the inferential features and capabilities of linear models are rarely found in other classes of models. These types of models usually produce linear, monotonic response functions with globally interpretable results like those of traditional linear models but often with a boost in predictive accuracy.
Suggested usage: Interpretability for regulated industries; these techniques are meant for practitioners who just can't use complex machine learning algorithms to build predictive models because of interpretability concerns or who seek the most interpretable possible modeling results.

References: Hastie et al., *The Elements of Statistical Learning*, Second Edition. R. Koenker, *Quantile Regression* (Cambridge, UK: Cambridge University Press, 2005).
OSS: gam ga2m (explainable boosting machine) glmnet h2o.ai quantreg scikit-learn (various functions)

Global or local scope: Alternative regression techniques often produce globally interpretable linear, monotonic functions that can be interpreted using coefficient values or other traditional regression measures and statistics.	**Best-suited complexity:** Low to medium. Alternative regression functions are generally linear, monotonic functions. However, GAM approaches can create complex nonlinear response functions.

Model specific or model agnostic: Model specific.
Trust and understanding: Understanding is enabled by the lessened assumption burden, the ability to select variables without potentially problematic multiple statistical significance tests, the ability to incorporate important but correlated predictors, the ability to fit nonlinear phenomena, and the ability to fit different quantiles of the data's conditional distribution. Basically, these techniques are trusted linear models but used in new, different, and typically more robust ways.

Table 7. A description of rule-based white-box modeling approaches

Technique: Rule-based models
Description: A rule-based model is a type of model that is composed of many simple Boolean statements that can be built by using expert knowledge or learning from real data.
Suggested usage: Useful in predictive modeling and fraud and anomaly detection when interpretability is a priority and simple explanations for relationships between inputs and targets are desired, but a linear model is not necessary. Often used in transactional data to find simple, frequently occurring pairs or triplets of items or entities.
Reference: Pang-Ning Tan, Michael Steinbach, and Vipin Kumar, *An Introduction to Data Mining*, First Edition (Minneapolis: University of Minnesota Press, 2006), 327-414.
OSS: RuleFit arules FP-growth Scalable Bayesian Rule Lists Skater

Global or local scope: Rule-based models can be both globally and locally interpretable.	**Best-suited complexity:** Low to medium. Most rule-based models are easy to follow for users because they obey Boolean logic ("if, then"). Rules can model extremely complex nonlinear, nonmonotonic phenomena, but rule lists can become very long in these cases.

Model specific or model agnostic: Model specific; can be highly interpretable if rules are restricted to simple combinations of input variable values.	**Trust and understanding:** Rule-based models increase understanding by creating straightforward, Boolean rules that can be understood easily by users. Rule-based models increase trust when the generated rules match domain knowledge or reasonable expectations.

Table 8. A description of the SLIM white-box modeling approach

Technique: SLIMs	
Description: SLIMs create predictive models that require users to only add, subtract, or multiply values associated with a handful of input variables to generate accurate predictions.	
Suggested usage: SLIMs are perfect for high-stakes situations in which interpretability and simplicity are critical, similar to diagnosing newborn infant health using the well-known Agpar scale.	
Reference: Berk Ustun and Cynthia Rudin, "Supersparse Linear Integer Models for Optimized Medical Scoring Systems," *Machine Learning* 102, no. 3 (2016): 349–391. *https://oreil.ly/31CyzjV*.	
Software: slim-python	
Global or local scope: Global.	**Best-suited complexity:** Low. SLIMs are simple, linear models.
Model specific or model agnostic: Model specific; interpretability for SLIMs is intrinsically linked to their linear nature and model-specific optimization routines.	**Trust and understanding:** SLIMs enhance understanding by breaking complex scenarios into simple rules for handling system inputs. They increase trust when their predictions are accurate and their rules reflect human domain knowledge or reasonable expectations.

Techniques for Enhancing Interpretability in Complex Machine Learning Models

The techniques in this section can be paired with interpretable models to create visual explanations, generate reason codes, or, potentially, create adverse action notices. Many of these techniques can be used to generate explanations for models of arbitrary complexity. So...no more black boxes!

Seeing model mechanisms with model visualizations

Model visualization techniques provide graphical insights into the prediction behavior of nearly any machine learning model and help debug the prediction mistakes they might make. ALE plots, decision tree surrogate models, ICE plots, partial dependence plots, and

residual plots are presented in Tables 1-9 to 1-13 and in Figures 1-2 and 1-3. ALE plots make up for some well-known shortcomings of partial dependence plots, surrogate models are simple models of more complex models, and decision tree surrogate models (Figure 1-2) create an approximate overall flowchart of a complex model's decision-making processes. ICE plots and partial dependence plots (Figure 1-3) provide a local and global view, respectively, into how a model's predictions change based on certain input variables. Residual analysis provides a mechanism to visualize any model's prediction errors while also highlighting anomalous data and outliers that might have undue influence on a model's predictions, and in just two dimensions too.

Interestingly, decision tree surrogates, partial dependence plots, and ICE plots can be used together to highlight potential interactions in machine learning models. Figures 1-2 and 1-3 were generated from the same GBM model of a simulated function with known interactions between input variables num1 and num4 and between inputs num8 and num9. Notice how the ICE curves diverge from partial dependence for the values ~-1 < num9 < ~1 in Figure 1-3. Compare this to the surrogate decision tree in Figure 1-2 for roughly the same values of num9 to see how the known interactions are represented.

Table 9. A description of the ALE model visualization technique

Technique: ALE plots
Description: ALE plots show us the overall behavior of a machine-learned response function with respect to an input variable without having to worry too much about correlations or interactions that could affect the trustworthiness of partial dependence plots. Like partial dependence plots, ALE plots show the shape—i.e., nonlinearity, nonmonotonicity—of the relationship between predictions and input variable values, even for very complex models.
Suggested usage: ALE plots are especially valuable when strong correlations or interactions exist in the training data, situations where partial dependence is known to fail. ALE plots can also be used to verify monotonicity of response functions under monotonicity constraints and can be used to check and confirm partial dependence plots. Note that most implementations are in R.
Reference: Daniel Apley, "Visualizing the Effects of Predictor Variables in Black Box Supervised Learning Models," arXiv:1612.08468, 2016, *https://arxiv.org/pdf/1612.08468.pdf*.
OSS: ALEPlot DALEX iml

Global or local scope: ALE plots are global in terms of the rows of a dataset but local in terms of the input variables.	**Best-suited complexity:** Any. Can be used to describe almost any function, including complex nonlinear, nonmonotonic functions.
Model specific or model agnostic: Model agnostic.	
Trust and understanding: ALE plots enhance understanding by showing the nonlinearity or non-monotonicity of the learned response between an input variable and a dependent variable in complex models. They can enhance trust when displayed relationships conform to domain knowledge.	

Table 10. A description of the decision tree surrogate model visualization technique

Technique: Decision tree surrogates
Description: A decision tree surrogate model is a simple model that is used to explain a complex model. Decision tree surrogate models are usually created by training a decision tree on the original inputs and predictions of a complex model. Variable importance, trends, and interactions displayed in the surrogate model are then assumed to be indicative of the internal mechanisms of the complex model. There are few, possibly no, theoretical guarantees that the simple surrogate model is highly representative of the more complex model.
Suggested usage: Use decision tree surrogate models to create approximate flowcharts of a more complex model's decision-making processes. Variables that are higher or used more frequently in the surrogate tree should be more important. Variables that are above and below one another can have strong interactions. Use surrogate trees with ICE and partial dependence to find and confirm interactions, as shown in Figures 1-2 and 1-3.
References: Mark W. Craven and Jude W. Shavlik, "Extracting Tree-Structured Representations of Trained Networks," *Advances in Neural Information Processing Systems* (1996): 24-30. *http://bit.ly/2FU4DK0*. Bastani et al., "Interpreting Blackbox Models via Model Extraction."
OSS: iml Skater Interpretable Machine Learning with Python
Global or local scope: Generally global. However, there is nothing to preclude using decision tree surrogate models in the LIME framework to explain more local regions of a complex model's predictions.
Best-suited complexity: Any. Surrogate models can help explain machine learning models of medium-to-high complexity, including nonlinear, monotonic or nonmonotonic models, but if the surrogate itself becomes large it can be difficult to interpret.
Model specific or model agnostic: Model agnostic.
Trust and understanding: Decision tree surrogate models enhance trust when their variable importance, trends, and interactions are aligned with human domain knowledge and reasonable expectations of modeled phenomena. Decision tree surrogate models enhance understanding because they provide insight into the internal mechanisms of complex models.

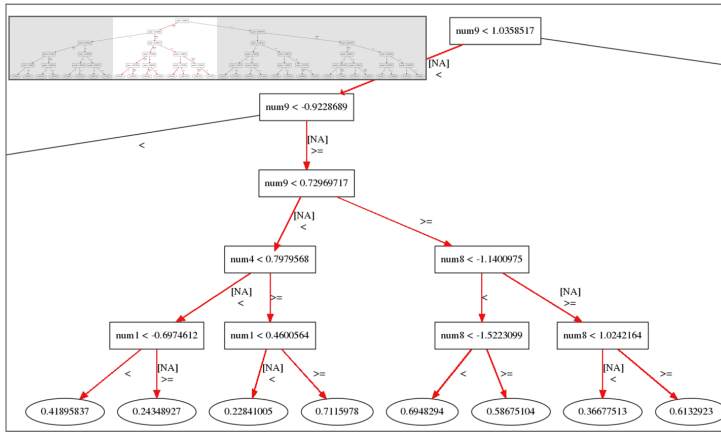

Figure 2. A visualization of a decision tree surrogate model as an approximate overall flowchart of the decision policies learned by a more complex machine learning model. In this simulated example, strong interactions exist between the input variables num1 and num4 and between num8 and num9. The call-out box (top left) emphasizes that the highlighted branches are part of a larger decision tree surrogate model. The highlighted branches allow for comparison to Figure 1-3. (Figure courtesy of Patrick Hall and H2O.ai.)

Table 11. A description of the ICE plot model visualization technique

Technique: ICE plots
Description: ICE plots are a newer, local, and less well-known adaptation of partial dependence plots. They depict how a model behaves for a single row of data (i.e., per observation). ICE pairs nicely with partial dependence in the same plot to provide local information to augment the global information provided by partial dependence. When ICE curves diverge from partial dependence curves as in Figure 1-3, this may indicate strong interactions between input variables.
Suggested usage: ICE plots can be used to create local, per-observation explanations using the same ideas as partial dependence plots. ICE can be used to verify monotonicity constraints and to detect when partial dependence fails in the presence of strong interactions or correlation among input variables.
Reference: Alex Goldstein et al., "Peeking Inside the Black Box: Visualizing Statistical Learning with Plots of Individual Conditional Expectation," *Journal of Computational and Graphical Statistics* 24, no. 1(2015): 44-65. *https://arxiv.org/abs/1309.6392*.
OSS: ICEbox iml PyCEbox Interpretable Machine Learning with Python

Global or local scope: Local.
Best-suited complexity: Any. Can be used to describe nearly any function, including nonlinear, nonmonotonic functions.
Model specific or model agnostic: Model agnostic.
Trust and understanding: ICE plots enhance understanding by showing the nonlinearity, nonmonotonicity, and two-way interactions between input variables and a target variable in complex models, per observation. They can also enhance trust when displayed relationships conform to domain knowledge.

Table 12. A description of the partial dependence plot model visualization technique

Technique: Partial dependence plots	
Description: Partial dependence plots show us the average manner in which machine-learned response functions change based on the values of one or two input variables of interest, while averaging out the effects of all other input variables.	
Suggested usage: Partial dependence plots show the nonlinearity, nonmonotonicity, and two-way interactions in very complex models and can be used to verify monotonicity of response functions under monotonicity constraints. They pair nicely with ICE plots, and ICE plots can reveal inaccuracies in partial dependence due to the presence of strong interactions as in Figure 1-3, where ICE and partial dependence curves diverge. Also, pairing partial dependence and ICE with a histogram of the variable of interest gives good insight into whether any plotted prediction is trustworthy and supported by training data. Use partial dependence with either ICE or use ALE plots instead of partial dependence alone if you suspect your dataset contains correlated or interacting variables.	
Reference: Hastie et al., *The Elements of Statistical Learning*, Second Edition.	
OSS: DALEX h2o.ai iml pdp PDPBox scikit-learn (various functions) Skater Interpretable Machine Learning with Python	
Global or local scope: Partial dependence plots are global in terms of the rows of a dataset but local in terms of the input variables.	**Best-suited complexity:** Any. Can be used to describe almost any function, including complex nonlinear, nonmonotonic functions.
Model specific or model agnostic: Model agnostic.	
Trust and understanding: Partial dependence plots enhance understanding by showing the nonlinearity, nonmonotonicity, and two-way interactions between input variables and a dependent variable in complex models. They can also enhance trust when displayed relationships conform to domain knowledge expectations.	

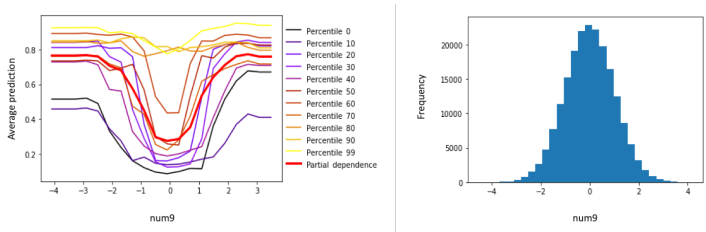

Figure 3. A model visualization in which partial dependence is displayed with ICE for the input variable num9 (left). In this simulated example, strong interactions exist between the input variables num1 and num4 and between num8 and num9. Notice how ICE curves diverge from partial dependence for ~-1 < num9 < 1. Compare this to Figure 1-2 to see how decision tree surrogates, ICE, and partial dependence can be used together to find and confirm modeled interactions. Also note that in the histogram for num9 (right) data values for num9 less than -3 and greater than 3 are very rare. Predictions for such values could be untrustworthy due to the lack of available training data. (Figure courtesy of Patrick Hall and H2O.ai.)

Table 13. A description of the residual plot model visualization technique

Technique: Residual plots	
Description: Residuals refer to the difference between the actual value of a target variable and the predicted value of a target variable for every row in a dataset. Residuals can be plotted in 2D to analyze complex predictive models.	
Suggested usage: Debugging for any machine learning model. Plotting the residual values against the predicted values is a time-honored model assessment technique and a great way to find outliers and see all of your modeling results in two dimensions.	
OSS: DALEX themis-ml Interpretable Machine Learning with Python	
Global or local scope: Global when used to assess the goodness-of-fit for a model over an entire dataset. Local when used to diagnose how a model treats a single row or small group of rows.	**Best-suited complexity:** Any. Can be used to assess machine learning models of varying complexity, including linear, nonlinear, and nonmonotonic functions.
Model specific or model agnostic: Model agnostic.	**Trust and understanding:** Residual analysis can promote understanding by guiding users toward problematic predictions and enabling users to debug such problems. It can enhance trust when residuals are appropriately distributed and other fit statistics (i.e., R^2, AUC, etc.) are in the appropriate ranges.

Variable importance

Variable importance is one of the most central aspects of explaining machine learning models. There are many methods for calculating variable importance and the methods tell us how much an input variable contributed to the predictions of a model, either globally or locally. While a handful of more established global variable importance metrics do not arise from the aggregation of local measures, averaging (or otherwise aggregating) local measures into global measures has become popular recently. So, we'll start by discussing newer methods for local variable importance below and then move onto global methods.

Deriving local variable importance for reason codes. Determining which input variables impacted a specific prediction is crucial to explanation. Local variable importance, reason codes, turn-down codes, and adverse action notices are several of the ways we make this determination. Local variable importance refers to the raw values that show how much a variable contributed to a prediction and the latter phrases mostly come from credit scoring. Reason codes are plain-text explanations of a model prediction in terms of a model's input variables. Turn-down codes and adverse action notices refer to another step of postprocessing where local variable importance and reason codes are matched to legal reasons a loan or employment application can be denied. We'll stick with the more general phrases, local variable importance and reason codes, in this section. These should provide the raw data and information needed to meet the higher bar of adverse action notices in many cases.

Aside from enabling appeal, reason codes are so important for machine learning interpretability in applied settings because they tell practitioners why a model makes a decision in terms of the model's input variables, and they can help practitioners understand if high weight is being given to potentially problematic inputs including gender, age, marital status, or disability status. Of course, generating reason codes for linear models is nothing new to banks, credit bureaus, and other entities. The techniques described in Tables 1-14 through 1-18 are interesting as you can apply them to generate reason codes for potentially more accurate machine learning models.

Depending on your application you may have different needs and expectations from your local variable importance and reason code techniques. The best technique today appears to be Shapley local

variable importance. Though an exact method with strong theoretical guarantees, it's a little time-consuming to calculate, even for tree-based models. (For other types of models, it can be infeasible.) Some other techniques are approximate, but work on nearly any model, such as LIME and leave-one-covariate-out (LOCO) variable importance. LIME also has the nice ability to generate "sparse" explanations, or explanations that only use the most important variables. The anchors technique has that same benefit, can be a bit more accurate, and generates rules, instead of numeric values, about the most important variables for a prediction. Variants of LIME, LOCO, and treeinterpreter have the advantage of being able to generate explanations extremely quickly for real-time explanations, but will likely not be as accurate as Shapley explanations.

We'll give you three pieces of advice before moving onto the techniques themselves:

1. Use Shapley if you can (and maybe even consider designing your project around using Shapley with tree-based models).
2. Don't hesitate to mix reason code techniques with interpretable models to get the best of both worlds like we did in Figure 1-4.
3. If your machine learning system will affect humans, please remember it will make wrong decisions, and explanations are needed by the human subjects of those wrong decisions to appeal your system's erroneous predictions.

Pairing a globally interpretable model—a single decision tree—with a local variable importance method—Shapley values—shows in Figure 1-4 the entire directed graph of the model's decision policies as well as the exact numeric contribution of each input variable to any prediction.

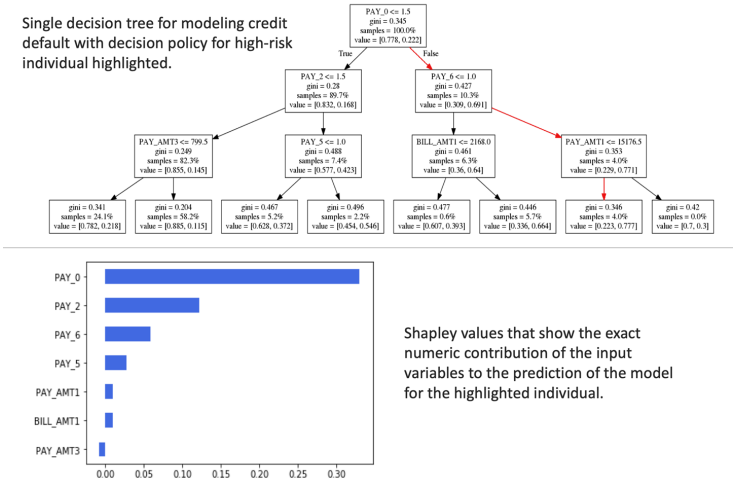

Figure 4. Here, the decision policy of a high risk of default individual is highlighted and the local Shapley variable importance values are presented for that same individual. (Figure courtesy of Patrick Hall and H2O.ai.)

Table 14. A description of the anchors local variable importance or reason code technique

Technique: Anchors
Description: A newer approach from the inventors of LIME that generates high-precision sets of plain-language rules to describe a machine learning model prediction in terms of the model's input variable values.
Suggested usage: Anchors is currently most applicable to classification problems in both traditional data mining and pattern-recognition domains. Anchors can be higher precision than LIME and generates rules about the most important variables for a prediction, so it can be a potential replacement for Shapley values for models that don't yet support the efficient calculation of Shapley values.
Reference: Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin, "Anchors: High-Precision Model-Agnostic Explanations," The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), April 25, 2018, *https://oreil.ly/2OWwzSb*.
OSS: anchor
Global or local scope: Local.
Best-suited complexity: Low to medium. Anchors can create explanations for very complex functions, but the rule set needed to describe the prediction can become large.
Model specific or model agnostic: Model agnostic.

Trust and understanding: Anchor explanations increase understanding by creating explanations for each prediction in a dataset. They enhance trust when the important variables for specific records conform to human domain knowledge and reasonable expectations.

Table 15. A description of the LOCO local variable importance or reason code technique

Technique: LOCO variable importance	
Description: LOCO, or even LOFO, variously stands for leave-one-{"column" or "covariate" or "feature"}-out. LOCO creates local interpretations for each row in a training or unlabeled score set by scoring the row of data once and then again for each input variable (e.g., column, covariate, feature) in the row. In each additional scoring run, one input variable is set to missing, zero, its mean value, or another appropriate value for leaving it out of the prediction. The input variable with the largest absolute impact on the prediction for that row is taken to be the most important variable for that row's prediction. Variables can also be ranked by their impact on the prediction on a per-row basis.	
Suggested usage: You can use LOCO to build reason codes for each row of data on which nearly any complex model makes a prediction. LOCO can deteriorate in accuracy when complex nonlinear dependencies exist in a model. Shapley explanations might be a better technique in this case, but LOCO is model agnostic and has speed advantages over Shapley both in training and scoring new data.	
Reference: Jing Lei et al., "Distribution-Free Predictive Inference for Regression," arXiv:1604.04173, 2016, *https://arxiv.org/pdf/1604.04173v1.pdf*.	
OSS: conformal Interpretable Machine Learning with Python	
Global or local scope: Local but can be aggregated to create global explanations.	**Best-suited complexity:** Any. LOCO measures are most useful for nonlinear, nonmonotonic response functions but can be applied to many types of machine-learned response functions.
Model specific or model agnostic: Model agnostic.	
Trust and understanding: LOCO measures increase understanding because they tell us the most influential variables in a model for a particular observation and their relative rank. LOCO measures increase trust if they are in line with human domain knowledge and reasonable expectations.	

Table 16. A description of the LIME local variable importance or reason code technique

Technique: LIME
Description: Typically uses local linear surrogate models to explain regions in a complex machine-learned response function around an observation of interest.

Suggested usage: Local linear model parameters can be used to describe the average behavior of a complex machine-learned response function around an observation of interest and to construct reason codes. LIME is approximate, but has the distinct advantage of being able to generate sparse, or simplified, explanations using only the most important local variables. Appropriate for pattern recognition applications as well. The original LIME implementation may sometimes be inappropriate for generating explanations in real-time on unseen data.
Reference: Ribeiro et al., "'Why Should I Trust You?' Explaining the Predictions of Any Classifier."
OSS: eli5 iml lime (Python) lime (R) Skater Interpretable Machine Learning with Python
Best-suited complexity: Low to medium. Suited for response functions of high complexity but can fail in regions of extreme nonlinearity or high-degree interactions.
Global or local scope: Local.
Model specific or model agnostic: Model agnostic.
Trust and understanding: LIME increases transparency by revealing important input variables and their linear trends. LIME bolsters trust when the important variables and their linear trends around specific records conform to human domain knowledge and reasonable expectations.

Table 17. A description of the treeinterpreter local variable importance or reason code technique

Technique: Treeinterpreter
Description: For each variable used in a model, treeinterpreter decomposes some decision tree, random forest, and GBM predictions into bias (overall training data average) and component terms. Treeinterpreter simply outputs a list of the bias and individual variable contributions globally and for each record.
Suggested usage: You can use treeinterpreter to interpret some complex tree-based models, and to create reason codes for each prediction. If you would like to use treeinterpreter, make sure your modeling library is fully supported by treeinterpreter. In some cases, treeinterpreter may not be locally accurate (local contributions do not sum to the model prediction) and treeinterpreter does not consider how contributions of many variables affect one another as carefully as the Shapley approach. However, treeinterpreter can generate explanations quickly. Also most treeinterpreter techniques appear as Python packages.
Reference: Ando Saabas, "Random Forest Interpretation with scikit-learn," Diving into Data [blog], August 12, 2015, *https://oreil.ly/33CtCK2*.
OSS: eli5 treeinterpreter

Global or local scope: Local but can be aggregated to create global explanations.	**Best-suited complexity:** Any. Treeinterpreter is meant to explain the usually nonlinear, nonmonotonic response functions created by certain decision tree, random forest, and GBM algorithms.
Model specific or model agnostic: Treeinterpreter is model specific to algorithms based on decision trees.	
Trust and understanding: Treeinterpreter increases understanding by displaying ranked contributions of input variables to the predictions of decision tree models. Treeinterpreter enhances trust when displayed variable contributions conform to human domain knowledge or reasonable expectations.	

Table 18. A description of the Shapley local variable importance or reason code technique

Technique: Shapley explanations	
Description: Shapley explanations are a Nobel-laureate technique with credible theoretical support from economics and game theory. Shapley explanations unify approaches such as LIME, LOCO, and treeinterpreter to derive consistent local variable contributions to black-box model predictions. Shapley also creates consistent, accurate global variable importance measures.	
Suggested usage: Shapley explanations are accurate, local contributions of input variables and can be rank-ordered to generate reason codes. Shapley explanations have long-standing theoretical support, which might make them more suitable for use in regulated industries, but they can be time consuming to calculate, especially outside of decision trees in H2O.ai, LightGBM, and XGBoost where Shapley is supported in low-level code and uses the efficient Tree SHAP approach.	
Reference: Lundberg and Lee, "A Unified Approach to Interpreting Model Predictions."	
OSS: h2o.ai iml LightGBM shap ShapleyR shapper XGBoost Interpretable Machine Learning with Python	
Global or local scope: Local but can be aggregated to create global explanations.	**Best-suited complexity:** Low to medium. This method applies to any machine learning model, including nonlinear and nonmonotonic models, but can be extremely slow for large numbers of variables or deep trees.

Model specific or model agnostic: Can be both. Uses a variant of LIME for model-agnostic explanations. Takes advantage of tree structures for decision tree models and is recommended for tree-based models.	**Trust and understanding:** Shapley explanations enhance understanding by creating accurate explanations for each observation in a dataset. They bolster trust when the important variables for specific records conform to human domain knowledge and reasonable expectations.

Global variable importance measures. Unlike local variable importance or reason code methods, global variable importance methods quantify the global contribution of each input variable to the predictions of a complex machine learning model over an entire dataset, not just for one individual or row of data. Global variable importance metrics can be calculated many ways, by the LOCO method, by shuffling variable values and investigating the difference in model scores, by the Shapley method, or by many other model-specific methods. Variable importance measures are typically seen in tree-based models but are also reported for other models. A simple heuristic rule for variable importance in a decision tree is related to the depth and frequency at which a variable is split in a tree, where variables used higher in the tree and more frequently in the tree are more important. For artificial neural networks, variable importance measures are typically associated with the aggregated, absolute magnitude of model parameters for a given variable of interest.

For some nonlinear, nonmonotonic response functions, global variable importance measures are the only commonly available, efficient, quantitative measure of the machine-learned relationships between input variables and the prediction target in a model. Variable importance measures sometimes give insight into the average direction that a variable pushes the response function, and sometimes they don't. At their most basic, they simply state the magnitude of a variable's relationship with the response as compared to other variables used in the model. This is hardly ever a bad thing to know, and since most global variable importance measures are older approaches, they are often expected by model validation teams. Like local variable importance, if you have the patience or ability to calculate Shapley values, they are likely the most accurate variable importance metric. Most others are approximate or inconsistent, but they are certainly still useful. In fact, comparing the differences between the imperfect results of several global variable importance techniques can help you reason about the overall drivers of your model's behavior (see Table 1-19).

Table 19. A description of global variable importance techniques

Technique: Global variable importance
Suggested usage: Understanding an input variable's global contribution to model predictions. Practitioners should be aware that unsophisticated measures of variable importance can be biased toward larger-scale variables or variables with a high number of categories. Global variable importance measures are typically not appropriate for creating reason codes.
References: Hastie et al., *The Elements of Statistical Learning*. Jerome Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," IMS 1999 Reitz Lecture, April 19, 2001, *https://oreil.ly/2yZr3Du*. Leo Breiman, "Random Forests," *Machine Learning* 45, no. 1 (2001): 5–32. *https://oreil.ly/30c4Jml*. Lundberg and Lee, "A Unified Approach to Interpreting Model Predictions."
OSS: DALEX h2o.ai iml lofo-importance LightGBM shap ShapleyR shapper Skater vip XGBoost R (various packages) scikit-learn (various functions) Interpretable Machine Learning with Python
Global or local scope: Global.
Best-suited complexity: Any. Variable importance measures are most useful for nonlinear, nonmonotonic response functions but can be applied to many types of machine-learned response functions.
Model specific or model agnostic: Both. Some global variable importance techniques are typically model specific, but the LOCO, permutation, and Shapley approaches are model agnostic.
Trust and understanding: Variable importance measures increase understanding because they tell us the most influential variables in a model and their relative rank. Variable importance measures increase trust if they are in line with human domain knowledge and reasonable expectations.

Fairness

As we discussed earlier, fairness is yet another important facet of interpretability, and a necessity for any machine learning project whose outcome will affect humans. Traditional checks for fairness, often called disparate impact analysis, typically include assessing model predictions and errors across sensitive demographic seg-

ments of ethnicity or gender. Today the study of fairness in machine learning is widening and progressing rapidly, including the development of techniques to remove bias from training data and from model predictions, and also models that learn to make fair predictions. A few of the many new techniques are presented in Tables 1-20 to 1-23. To stay up to date on new developments for fairness techniques, keep an eye on the public and free *Fairness and Machine Learning* book and https://fairmlbook.org.

Table 20. A description of the disparate impact testing fairness techniques

Technique: Disparate impact testing	
Description: A set of simple tests that show differences in model predictions and errors across demographic segments.	
Suggested usage: Use for any machine learning system that will affect humans to test for biases involving gender, ethnicity, marital status, disability status, or any other segment of possible concern. If disparate impact is discovered, use a remediation strategy (Tables 1-21 to 1-23), or select an alternative model with less disparate impact. Model selection by minimal disparate impact is probably the most conservative remediation approach, and may be most appropriate for practitioners in regulated industries.	
Reference: Feldman et al., "Certifying and Removing Disparate Impact."	
OSS: aequitas AIF360 Themis themis-ml Interpretable Machine Learning with Python	
Global or local scope: Global because fairness is measured across demographic groups, not for individuals.	**Best-suited complexity:** Low to medium. May fail to detect local instances of discrimination in very complex models.
Model specific or model agnostic: Model agnostic.	**Trust and understanding:** Mostly trust as disparate impact testing can certify the fairness of a model, but typically does not reveal the causes of any discovered bias.

Table 21. A description of the reweighing fairness preprocessing technique

Technique: Reweighing
Description: Preprocesses data by reweighing the individuals in each demographic group differently to ensure fairness before model training.
Suggested usage: Use when bias is discovered during disparate impact testing; best suited for classification.

Reference: Faisal Kamiran and Toon Calders, "Data Preprocessing Techniques for Classification Without Discrimination," *Knowledge and Information Systems* 33, no. 1 (2012): 1–33. *https://oreil.ly/2Z3me6W*.	
OSS: AIF360	
Global or local scope: Global because fairness is measured across demographic groups, not for individuals.	**Best-suited complexity:** Low to medium. May fail to remediate local instances of discrimination in very complex models.
Model specific or model agnostic: Model agnostic, but mostly meant for classification models.	**Trust and understanding:** Mostly trust, because the process simply decreases disparate impact in model results by reweighing the training dataset.

Table 22. A description of the adversarial debiasing fair modeling technique

Technique: Adversarial debiasing	
Description: Trains a model with minimal disparate impact using a main model and an adversarial model. The main model learns to predict the outcome of interest, while minimizing the ability of the adversarial model to predict demographic groups based on the main model predictions.	
Suggested usage: When disparate impact is detected, use adversarial debiasing to directly train a model with minimal disparate impact without modifying your training data or predictions.	
Reference: Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell, "Mitigating Unwanted Biases with Adversarial Learning," arXiv:1801.07593, 2018, *https://oreil.ly/2H4rvVm*.	
OSS: AIF360	
Global or local scope: Potentially both because the adversary may be complex enough to discover individual instances of discrimination.	**Best-suited complexity:** Any. Can train nonlinear, nonmonotonic models with minimal disparate impact.
Model specific or model agnostic: Model agnostic.	**Trust and understanding:** Mostly trust, because the process simply decreases disparate impact in a model.

Table 23. A description of the reject option-based classification postprocessing fairness technique

Technique: Reject option-based classification
Description: Postprocesses modeling results to decrease disparate impact; switches positive and negative labels in unprivileged groups for individuals who are close to the decision boundary of a classifier to decrease discrimination.
Suggested usage: Use when bias is discovered during disparate impact testing; best suited for classification.
Reference: Faisal Kamiran, Asim Karim, and Xiangliang Zhang, "Decision Theory for Discrimination-Aware Classification," IEEE 12th International Conference on Data Mining (2012): 924-929. *https://oreil.ly/2Z7MNId*.

OSS: AIF360 themis-ml	
Global or local scope: Global because fairness is measured across demographic groups, not for individuals.	**Best-suited complexity:** Low. Best suited for linear classifiers.
Model specific or model agnostic: Model agnostic, but best suited for linear classifiers (i.e., logistic regression and naive Bayes).	**Trust and understanding:** Mostly trust, because the process simply decreases disparate impact in model results by changing some results.

Sensitivity Analysis and Model Debugging

Sensitivity analysis investigates whether model behavior and outputs remain acceptable when data is intentionally perturbed or other changes are simulated in data. Beyond traditional assessment practices, sensitivity analysis of predictions is perhaps the most important validation and debugging technique for machine learning models. In practice, many linear model validation techniques focus on the numerical instability of regression parameters due to correlation between input variables or between input variables and the target variable. It can be prudent for those switching from linear modeling techniques to machine learning techniques to focus less on numerical instability of model parameters and to focus more on the potential instability of model predictions.

One of the main thrusts of linear model validation is sniffing out correlation in the training data that could lead to model parameter instability and low-quality predictions on new data. The regularization built into most machine learning algorithms makes their parameters and rules more accurate in the presence of correlated inputs, but as discussed repeatedly, machine learning algorithms can produce very complex nonlinear, nonmonotonic response functions that can produce wildly varying predictions for only minor changes in input variable values. Because of this, in the context of machine learning, directly testing a model's predictions on simulated, unseen data, such as recession conditions, is likely a better use of time than searching through static training data for hidden correlations.

Single rows of data with the ability to swing model predictions are often called adversarial examples. Adversarial examples are extremely valuable from a security and model debugging perspective. If we can easily find adversarial examples that cause model pre-

dictions to flip from positive to negative outcomes (or vice versa) this means a malicious actor can game your model. This security vulnerability needs to be fixed by training a more stable model or by monitoring for adversarial examples in real time. Adversarial examples are also helpful for finding basic accuracy and software problems in your machine learning model. For instance, test your model's predictions on negative incomes or ages, use character values instead of numeric values for certain variables, or try input variable values 10% to 20% larger in magnitude than would ever be expected to be encountered in new data. If you can't think of any interesting situations or corner cases, simply try a *random data attack*: score many random adversaries with your machine learning model and analyze the resulting predictions (Table 1-24). You will likely be surprised by what you find.

Table 24. A description of sensitivity analysis and adversarial examples

Technique: Sensitivity analysis and adversarial examples
Suggested usage: Testing machine learning model predictions for accuracy, fairness, security, and stability using simulated datasets, or single rows of data, known as adversarial examples. *If you are using a machine learning model, you should probably be conducting sensitivity analysis.*
OSS: cleverhans foolbox What-If Tool Interpretable Machine Learning with Python
Global or local scope: Sensitivity analysis can be a global interpretation technique when many input rows to a model are perturbed, scored, and checked for problems, or when global interpretation techniques are used with the analysis, such as using a single, global surrogate model to ensure major interactions remain stable when data is lightly and purposely corrupted. Sensitivity analysis can be a local technique when an adversarial example is generated, scored, and checked or when local interpretation techniques are used with adversarial examples (e.g., using LIME to determine if the important variables in a credit allocation decision remain stable for a given customer after perturbing their data values). In fact, nearly any technique in this section can be used in the context of sensitivity analysis to determine whether visualizations, models, explanations, or fairness metrics remain stable globally or locally when data is perturbed in interesting ways.
Best-suited complexity: Any. Sensitivity analysis can help explain the predictions of nearly any type of response function, but it is probably most appropriate for nonlinear response functions and response functions that model high-degree variable interactions. For both cases, small changes in input variable values can result in large changes in a predicted response.
Model specific or model agnostic: Model agnostic.

Trust and understanding: Sensitivity analysis enhances understanding because it shows a model's likely behavior and output in important situations, and how a model's behavior and output may change over time. Sensitivity analysis enhances trust when a model's behavior and outputs remain stable when data is subtly and intentionally corrupted. It also increases trust if models adhere to human domain knowledge and expectations when interesting situations are simulated, or as data changes over time.

Updating Your Workflow

Now that you've read about these machine learning interpretability techniques, you may be wondering how to fit them into your professional workflow. Figure 1-5 shows one way to augment the standard data mining workflow with steps for increasing accuracy, interpretability, privacy, security, transparency, and trustworthiness using the classes of techniques introduced in this section.

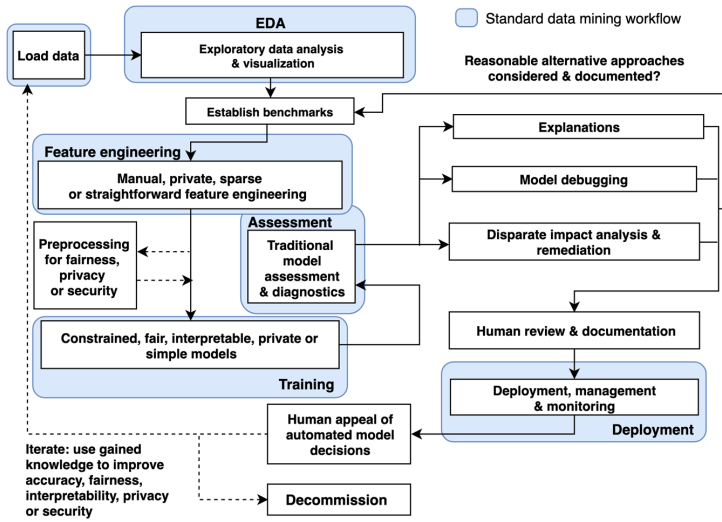

Figure 5. A proposed holistic training and deployment workflow for human-centered or other high-stakes machine learning applications. (Figure courtesy of Patrick Hall and H2O.ai. For more details on this workflow, please see: https://github.com/jphall663/hc_ml.)

We suggest using the introduced techniques in these workflow steps. For instance, you may visualize and explore your data using projections and network graphs, preprocess your data using fairness reweighing, and train a monotonic GBM model. You might then explain the monotonic GBM with a combination of techniques such

as decision tree surrogates, partial dependence and ICE plots, and Shapley explanations. Then you could conduct disparate impact testing to ensure fairness in your predictions and debug your model with sensitivity analysis. Such a combination represents a current best guess for a viable human-centered, or other high-stakes application, machine learning workflow.

Limitations and Precautions

By this point we hope that you can see the tremendous intellectual, social, and commercial potential for interpretable machine learning models and debugging, explanation, and fairness techniques. And though we've tried to present a balanced, practical portrait of the technologies, it's important to call out some specific limitations and precautions. Like many technologies, machine learning explanations can be abused, particularly when used as a faulty safeguard for harmful black boxes (e.g., *fairwashing*) to make a biased model appear fair. Explanations can also be used for malicious hacking, to steal predictive models, to steal sensitive training data, and to plan other more sophisticated attacks.[39] In addition to fairwashing and hacking, there are at least three other general concerns to be aware of: the idea that explanations alone do not equal trust, the multiplicity of good models, and the limitations of surrogate models.

Explanations Alone Foster Understanding and Appeal, Not Trust

While they are likely necessary for trust in many cases, explanations are certainly not sufficient for trust in all cases. Explanation, as a general concept, is related more directly to understanding and transparency than to trust (as shown by the Merriam-Webster defi-

38 Ulrich Aïvodji et al., "Fairwashing: The Risk of Rationalization," arXiv:1901.09749, 2019, *https://arxiv.org/pdf/1901.09749.pdf*.

39 Reza Shokri et al., "Membership Inference Attacks Against Machine Learning Models," IEEE Symposium on Security and Privacy (SP), 2017, *https://oreil.ly/2Z22LHI*.

40 Florian Tramèr et al., "Stealing Machine Learning Models via Prediction APIs," in 25th {USENIX} Security Symposium ({USENIX} Security 16) (2016): 601–618. *https://oreil.ly/2z1TDnC*.

41 Patrick Hall, "Guidelines for the Responsible and Human-Centered Use of Explainable Machine Learning," arXiv:1906.03533, 2019, *https://arxiv.org/pdf/1906.03533.pdf*.

nition, which doesn't mention *trust*) Simply put, you can understand and explain a model without trusting it. You can also trust a model and not be able to understand or explain it. Consider the following example scenarios:

- **Explanation and understanding without trust:** In Figure 1-6, global Shapley explanations and residual analysis identify a pathology in an unconstrained GBM model trained to predict credit card default. The GBM overemphasizes the input variable PAY_0, or a customer's most recent repayment status. Due to overemphasis of PAY_0, the GBM usually can't predict on-time payment if recent payments are delayed (PAY_0 > 1), causing large negative residuals. The GBM also usually can't predict default if recent payments are made on time (PAY_0 ≤ 1), causing large positive residuals. In this example scenario, a machine learning model is explainable, but not trustworthy.
- **Trust without explanation and understanding:** Years before reliable explanation techniques were widely acknowledged and available, black-box predictive models, such as autoencoder and MLP neural networks, were used for fraud detection in the financial services industry. When these models performed well, they were trusted. However, they were not explainable or well understood by contemporary standards.

If trust in models is your goal, then explanations alone are not sufficient. However, in an ideal scenario, we would all use explanation techniques with a wide variety of other methods to increase accuracy, fairness, interpretability, privacy, security, and trust in machine learning models.

42 Krishna M. Gopinathan et al., "Fraud Detection Using Predictive Modeling," October 6, 1998. US Patent 5,819,226. *https://oreil.ly/2Z3FLIn*.

43 "Reduce Losses from Fraudulent Transactions," SAS [company site]. *https://oreil.ly/2KwolMk*.

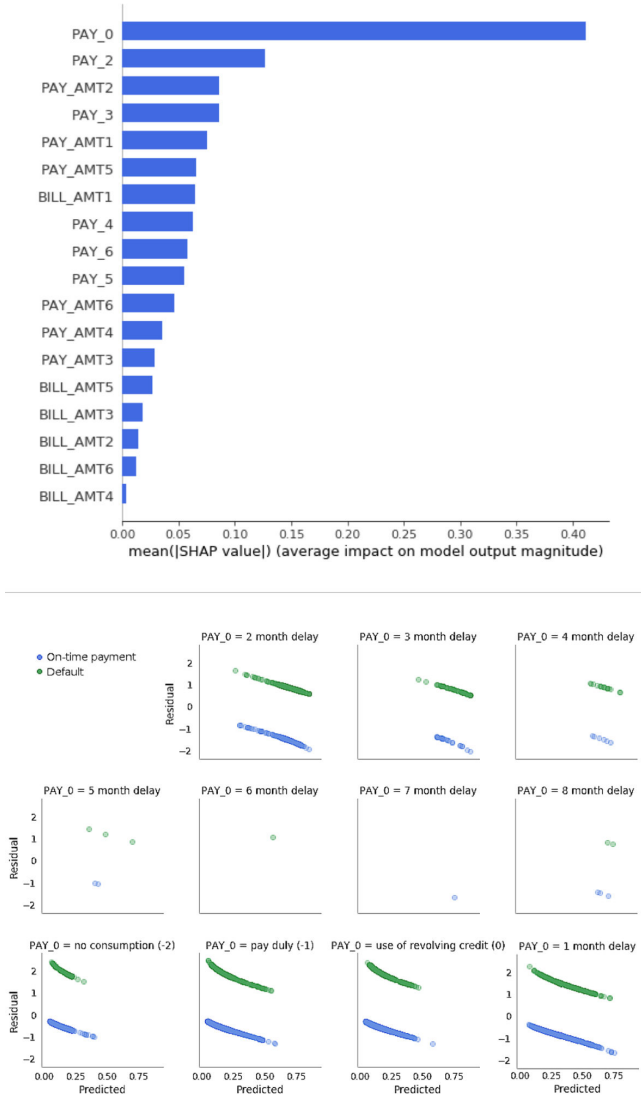

Figure 6. Explanatory and model debugging techniques show us why a machine learning model isn't trustworthy: this machine learning model is too dependent on one input variable, PAY_0. The model makes many errors because of its overemphasis of that variable. (Figure courtesy of Patrick Hall and H2O.ai.)

The Multiplicity of Good Models

For the same set of input variables and prediction targets, complex machine learning algorithms can produce multiple accurate models with very similar, but not the same, internal architectures. It is important to remember that details of explanations and fairness can change across multiple accurate, similar models trained on the same data! This difficult mathematical problem goes by several names. In his seminal 2001 paper, Professor Leo Breiman of UC Berkeley called this problem *the multiplicity of good models*. Some in credit scoring refer to this same phenomenon as *model locality*. Whatever you call it, this means almost every debugging, explanation, or fairness exercise on a complex machine learning model assumes we are choosing to debug, explain, or audit for fairness just one of many, many similar models. Let's discuss why this happens and what can be done to address it.

Figures 1-7 and 1-8 are cartoon illustrations of the surfaces defined by error functions for two fictitious predictive models. In Figure 1-7 the error function is representative of a traditional linear model's error function. The surface created by the error function in Figure 1-7 is convex. It has a clear global minimum in three dimensions, meaning that given two input variables, such as a customer's income and a customer's interest rate, the most accurate model trained to predict loan defaults (or any other outcome) would almost always give the same weight to each input in the prediction, and the location of the minimum of the error function and the weights for the inputs would be unlikely to change very much if the model was retrained, even if the input data about a customer's income and interest rate changed a little bit. (The actual numeric values for the weights could be ascertained by tracing a straight line from the minimum of the error function pictured in Figure 1-7 to the interest rate axis [the X axis] and income axis [the Y axis].)

44 Leo Breiman, "Statistical Modeling: The Two Cultures (with Comments and a Rejoinder by the Author)," *Statistical Science* 16, no. 3 (2001): 199-231. *https://oreil.ly/303vbOJ*.

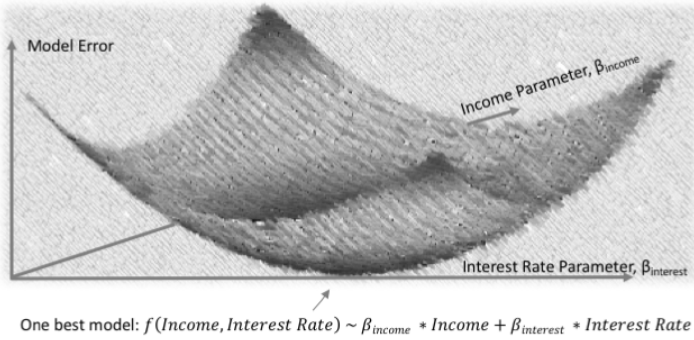

Figure 7. An illustration of the error surface of a traditional linear model. (Figure courtesy of H2O.ai.)

Because of the convex nature of the error surface for linear models, there is basically only one *best* model, given some relatively stable set of inputs and a prediction target. The model associated with the error surface displayed in Figure 1-7 would be said to have strong model locality. Even if we retrained the model on new or updated data, the weight of income versus interest rate is likely mostly stable in the pictured error function and its associated linear model. Explanations about how the function made decisions about loan defaults based on those two inputs would also probably be stable and so would results for disparate impact testing.

Figure 1-8 depicts a nonconvex error surface that is representative of the error function for a machine learning function with two inputs—for example, a customer's income and a customer's interest rate—and an output, such as the same customer's probability of defaulting on a loan. This nonconvex error surface with no obvious global minimum implies there are many different ways a complex machine learning algorithm could learn to weigh a customer's income and a customer's interest rate to make an accurate decision about when they might default. Each of these different weightings would create a different function for making loan default decisions, and each of these different functions would have different explanations and fairness characteristics! This would likely be especially obvious upon updating training data and trying to refit a similar machine learning model.

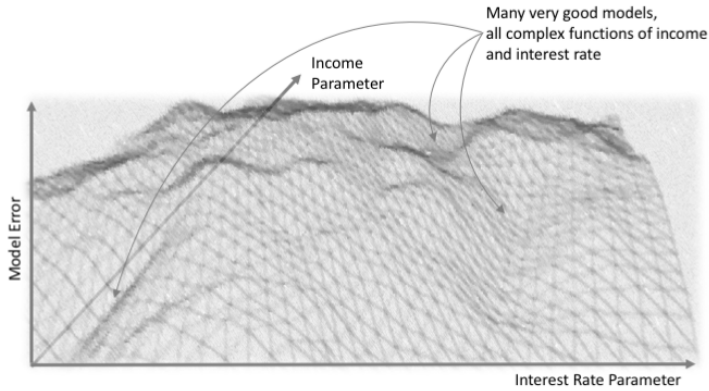

Figure 8. An illustration of the error surface of a machine learning model. (Figure courtesy of H2O.ai.)

While there is no remedy to the multiplicity of good models challenge in machine learning, Shapley values do often account for perturbations of numerous similar models and constrained interpretable models are probably your best bet for creating an accurate, stable, and representative model that won't change too much if your training data is updated. Also, the multiplicity of good models is not necessarily always a problem. You can use it to your advantage in some situations: of the many machine learning models you can train for a given dataset, it's possible that you can find one with the desired accuracy, explanation, and fairness characteristics.

Limitations of Surrogate Models

Surrogate models are important explanation and debugging tools. They can provide global and local insights into both model predictions and model residuals or errors. However, surrogate models are approximate. There are few theoretical guarantees that the surrogate model truly represents the more complex original model from which it has been extracted. Let's go over the rationale for surrogate models and then outline the nuances of working with surrogate models responsibly. Linear models, and other types of more straightforward models, like the function in Figure 1-9, create approximate models with neat, exact explanations. Surrogate models are meant to accomplish the converse (i.e., approximate explanations for more exact models). In the cartoon illustrations in Figures

1-9 and 1-10, we can see the benefit of accurate and responsible use of surrogate models. The explanation for the simple interpretable model in Figure 1-9 is very direct, but still doesn't really explain the modeled age versus purchasing behavior because the linear model just isn't fitting the data properly.

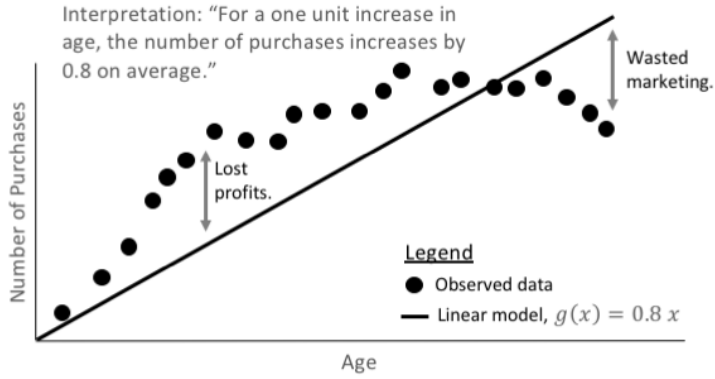

Figure 9. A linear model, g(x), predicts the average number of purchases given a customer's age. The predictions can be inaccurate but the associated explanations are straightforward and stable. (Figure courtesy of H2O.ai.)

Although the explanations for the more complex machine learning function in Figure 1-10 are approximate, they are at least as useful, if not more so, than the linear model explanations above because the underlying machine learning response function has learned more exact information about the relationship between age and purchases. Of course, surrogate models don't always work out like our cartoon illustrations. So, it's best to always measure the accuracy of your surrogate models and to always pair them with more direct explanation or debugging techniques.

When measuring the accuracy of surrogate models, always look at the R^2, or average squared error (ASE), between your surrogate model predictions and the complex response function you are trying to explain. Also use cross-validated error metrics for decision tree surrogate models. Single decision trees are known to be unstable. Make sure your surrogate decision trees have low and stable error across multiple folds of the dataset in which you would like to create explanations.

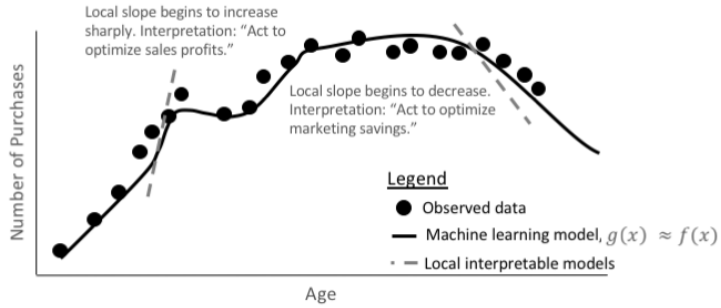

Figure 10. A machine learning model, g(x), predicts the number of purchases, given a customer's age, very accurately, nearly replicating the true, unknown signal-generating function, f(x). (Figure courtesy of H2O.ai.)

Additionally, there are many options to pair surrogate models with more direct techniques. In Figures 1-2 and 1-3 we've already shown how to pair decision tree surrogate models with direct plots of model predictions (i.e., ICE and partial dependence curves) to find and confirm interactions. You can pair Shapley values with LIME coefficients to see accurate point estimates of local variable importance and the local linear trends of the same variables. You can also pair decision tree surrogate models of residuals with direct adversarial perturbations to uncover specific error pathologies in your machine learning models, and you can probably think of several other ways to combine surrogate models and direct explanation, fairness, or debugging techniques. Just remember that if your surrogate model is not an accurate representation of the model you are trying to explain, or if surrogate model explanations don't match up with more direct explanation techniques, you probably should not use surrogate models for the interpretability task at hand.

Testing Interpretability and Fairness

The novelty and limitations of interpretable machine learning might call into question the trustworthiness of debugging, explanation, or fairness techniques themselves. Don't fret! You can test these techniques for accuracy too. Originally, researchers proposed testing machine learning model explanations by their capacity to enable humans to correctly determine the outcome of a model prediction

based on input values. Given that human evaluation studies are likely impractical for some commercial data science or machine learning groups, and that we're not yet aware of any formal testing methods for debugging or fairness techniques today, several potential approaches for testing debugging, explanations, and fairness techniques themselves are proposed here:

Simulated data

You can use simulated data with known characteristics to test debugging, explanation, and fairness techniques. For instance, models trained on totally random data with no relationship between a number of input variables and a prediction target should not give strong weight to any input variable, nor generate compelling local explanations or reason codes. Nor should they exhibit obvious fairness problems. Once this baseline has been established, you can use simulated data with a known signal generating function to test that explanations accurately represent that known function. You can simulate data with known global correlations and local dependencies between demographic variables, or other proxy variables, and a prediction target and ensure your fairness techniques find these known group and individual fairness issues. You can also switch labels for classification decisions or inject noise into predicted values for regression models and check that model debugging techniques find the simulated errors. Of course, this kind of empirical testing doesn't guarantee theoretical soundness, but it can certainly help build the case for using debugging, explanation, and fairness techniques for your next machine learning endeavor.

Explanation and fairness metric stability with increased prediction accuracy

Another workable strategy for building trust in explanation and fairness techniques may be to build from an established, previously existing model with known explanations and acceptable fairness characteristics. Essentially, you can perform tests to see how accurate a model can become or how much its form can change before its predictions' reason codes veer away from known standards and its fairness metrics drift in undesirable

45 Doshi-Velez and Kim, "Towards a Rigorous Science of Interpretable Machine Learning."

ways. If previously known, accurate explanations or reason codes from a simpler linear model are available, you can use them as a reference for the accuracy of explanations from a related, but more complex and hopefully more accurate, model. The same principle likely applies for fairness metrics. Add new variables one by one or increase the complexity of the model form in steps, and at each step make sure fairness metrics remain close to the original, trusted model's measurements.

Debugging explanation and fairness techniques with sensitivity analysis and adversarial examples

If you agree that explanations and fairness metrics likely should not change unpredictably for minor or logical changes in input data, then you can use sensitivity analysis and adversarial examples to debug explanation and fairness techniques. You can set and test tolerable thresholds for allowable explanation or fairness value changes and then begin manually or automatically perturbing input data and monitoring for unacceptable swings in explanation or fairness values. If you don't observe any unnerving changes in these values, your explanation and fairness techniques are likely somewhat stable. If you do observe instability, try a different technique or dig in and follow the trail the debugging techniques started you down.

Machine Learning Interpretability in Action

To see how some of the interpretability techniques discussed in this report might look and feel in action, a public, open source repository has been provided.

This repository contains examples of white-box models, model visualizations, reason code generation, disparate impact testing, and sensitivity analysis applied to the well-known Taiwanese credit card customer dataset using the popular XGBoost and H2O libraries in Python.

46 D. Dua and C. Graff, UCI Machine Learning Repository, University of California Irvine, School of Information and Computer Science, 2019. *https://oreil.ly/33vI2LK*.

Looking Forward

FATML, XAI, and machine learning interpretability are new, rapidly changing, and expanding fields. Automated machine learning (autoML) is another important new trend in artificial intelligence. Several open source and proprietary software packages now build machine learning models automatically with minimal human intervention. These new autoML systems tend to be even more complex, and therefore potentially black box in nature, than today's somewhat human-oriented data science workflows. For the benefits of machine learning and autoML to take hold across a broad cross-section of industries, our cutting-edge autoML systems will also need to be understandable and trustworthy.

In general, the widespread acceptance of machine learning interpretability techniques will be one of the most important factors in the increasing adoption of machine learning and artificial intelligence in commercial applications and in our day-to-day lives. Hopefully, this report has convinced you that interpretable machine learning is technologically feasible. Now, let's put these approaches into practice, leave the ethical and technical concerns of black-box machine learning in the past, and move on to a future of FATML and XAI.

Acknowledgments

We are thankful to colleagues past and present whose comments, thoughts, and tips undoubtedly shaped our own thinking on the topic of interpretable machine learning.

In particular, from H2O.ai, we thank SriSatish Ambati for putting the resources of an amazing company behind our efforts. We thank Mark Chan, Doug Deloy, Mateusz Dymczyk, Martin Dvorak, Ladislav Ligart, and Zac Taschdjian for their tireless efforts to turn the ideas in this book into enterprise software. And we thank Pramit Choudhary, Michal and Megan Kurka, Kerry O'Shea, Josephine Wang, and Leland Wilkinson for additional ideas, guidance, and software that are reflected in this report.

We also thank our colleagues from the broader interpretability community for their expertise and input: Andrew Burt, Immuta; Mark Chan, JPMC; Przemyslaw Biecek, Warsaw University of Technology; Christoph Molnar, Ludwig-Maximilians-Universität München; Wen Phan, Domino Data Labs; Nick Schmidt, BLDS, LLC; and Sameer Singh, University of California, Irvine.

About the Authors

Patrick Hall is senior director for data science products at H2O.ai, where he focuses mainly on model interpretability. Patrick is also currently an adjunct professor in the Department of Decision Sciences at George Washington University, where he teaches graduate classes in data mining and machine learning. Prior to joining H2O.ai, Patrick held global customer-facing roles and research and development roles at SAS Institute.

Navdeep Gill is a senior data scientist and software engineer at H2O.ai, where he focuses mainly on machine learning interpretability. He previously focused on GPU accelerated machine learning, automated machine learning, and the core H2O-3 platform. Prior to joining H2O.ai, Navdeep worked at Cisco focusing on data science and software development. Before that he was a researcher and analyst in several neuroscience labs at California State University, East Bay; University of California, San Francisco; and Smith Kettlewell Eye Research Institute. Navdeep graduated from California State

University, East Bay with a MS in computational statistics, a BS in statistics, and a BA in Psychology (minor in mathematics).