Veridical data science

Bin Yu[a,b,c,d,1] **and Karl Kumbier**[a]

[a]Statistics Department, University of California, Berkeley, CA 94720; [b]Electrical Engineering and Computer Sciences Department, University of California, Berkeley, CA 94720; [c]Chan Zuckerberg Biohub, San Francisco, CA 94158; and [d]Lawrence Berkeley National Laboratory, Environmental Genomics and Systems Biology Division, Berkeley, CA 94720

This contribution is part of the special series of Inaugural Articles by members of the National Academy of Sciences elected in 2014.

Contributed by Bin Yu, November 25, 2019 (sent for review January 25, 2019; reviewed by Jun S. Liu, David Madigan, and Larry Wasserman)

Building and expanding on principles of statistics, machine learning, and scientific inquiry, we propose the predictability, computability, and stability (PCS) framework for veridical data science. Our framework, composed of both a workflow and documentation, aims to provide responsible, reliable, reproducible, and transparent results across the data science life cycle. The PCS workflow uses predictability as a reality check and considers the importance of computation in data collection/storage and algorithm design. It augments predictability and computability with an overarching stability principle. Stability expands on statistical uncertainty considerations to assess how human judgment calls impact data results through data and model/algorithm perturbations. As part of the PCS workflow, we develop PCS inference procedures, namely PCS perturbation intervals and PCS hypothesis testing, to investigate the stability of data results relative to problem formulation, data cleaning, modeling decisions, and interpretations. We illustrate PCS inference through neuroscience and genomics projects of our own and others. Moreover, we demonstrate its favorable performance over existing methods in terms of receiver operating characteristic (ROC) curves in high-dimensional, sparse linear model simulations, including a wide range of misspecified models. Finally, we propose PCS documentation based on R Markdown or Jupyter Notebook, with publicly available, reproducible codes and narratives to back up human choices made throughout an analysis. The PCS workflow and documentation are demonstrated in a genomics case study available on Zenodo.

stability | prediction | computation | data science

Data science is a field of evidence seeking that combines data with domain information to generate new knowledge. The data science life cycle (DSLC) begins with a domain question or problem and proceeds through collecting, managing, processing (cleaning), exploring, modeling, and interpreting[†] data results to guide new actions (Fig. 1). Given the transdisciplinary nature of this process, data science requires human involvement from those who collectively understand both the domain and tools used to collect, process, and model data. These individuals make implicit and explicit judgment calls throughout the DSLC. The limited transparency in reporting such judgment calls has blurred the evidence for many analyses, resulting in more false discoveries than might otherwise occur (2, 3). This fundamental issue necessitates veridical data science, that is, principled inquiry to extract reliable and reproducible information from data, with an enriched technical language to communicate and evaluate empirical evidence in the context of human decisions and domain knowledge. Three core principles, predictability, computability, and stability (PCS), provide the foundation for such a data-driven language and a unified data analysis framework. They serve as minimum requirements for veridical data science[‡].

Many ideas embedded in PCS have been widely used across various areas of data science. Predictability plays a central role in science through Popperian falsifiability (4). If a model does not accurately predict new observations, it can be rejected or updated. Predictability has been adopted by the machine-learning community as a goal of its own right and more generally to evaluate the quality of a model or data result (5). While statistics have always considered prediction, machine learning emphasized its importance for empirical rigor. This was in large part powered by computational advances that made it possible to compare models through cross-validation (CV), developed by statisticians Stone (6) and Allen (7).

The role of computation extends beyond prediction, setting limitations on how data can be collected, stored, and analyzed. Computability has played an integral role in computer science tracing back to Alan Turing's seminal work on the computability of sequences (8). Analyses of computational complexity have since been used to evaluate the tractability of machine-learning algorithms (9). Kolmogorov built on Turing's work through the notion of Kolmogorov complexity, which describes the minimum computational resources required to represent an object (10, 11). Since Turing machine-based notions of computabiltiy are not computable in practice, we treat computability as an issue of algorithm efficiency and scalability. This narrow definition of computability addresses computational considerations

Significance

Predictability, computability, and stability (PCS) are three core principles of data science. They embed the scientific principles of prediction and replication in data-driven decision making while recognizing the central role of computation. Based on these principles, we propose the PCS framework, including workflow and documentation (in R Markdown or Jupyter Notebook). The PCS framework aims at responsible, reliable, reproducible, and transparent analysis across fields of science, social science, engineering, business, and government. It can be used as a recommendation system for scientific hypothesis generation and experimental design. In particular, we propose (basic) PCS inference for reliability measures on data results, extending statistical inference to a much broader scope as current data science practice entails.

Author contributions: B.Y. and K.K. designed research, performed research, analyzed data, and wrote the paper.

Reviewers: J.S.L., Harvard University; D.M., Columbia University; and L.W., Carnegie Mellon University.

The authors declare no competing interest.

Data deposition: All data discussed in this paper are available to readers at Zenodo, https://zenodo.org/record/3522419#.XjWiJhNKjRY.

See QnAs on page 3893.

[1]To whom correspondence may be addressed. Email: binyu@stat.berkeley.edu.

This article contains supporting information online at https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.1901326117/-/DCSupplemental.

First published February 13, 2020.

[†]For a precise definition of interpretable machine learning, we refer to our recent paper (1).

[‡]Veridical data science is the broad aim of our proposed framework (veridical meaning "truthful" or "coinciding with reality"). This paper has been on arXiv since January 2019 under the old title: Three principles of data science: predictability, computability, stability (PCS).

at the modeling stage of the DSLC but does not deal with data collection, storage, or cleaning.

Stability[§] is a common-sense principle and a prerequisite for knowledge. It is related to the notion of scientific reproducibility, which Fisher (12) and Popper (4) argued is a necessary condition for establishing scientific results. While replicability across laboratories has long been an important consideration in science, computational reproducibility has come to play an important role in data science as well. For example, ref. 13 discusses reproducible research in the context of computational harmonic analysis. More broadly, ref. 14) advocates for "preproducibility" to explicitly detail all steps along the DSLC and ensure sufficient information for quality control. Stability at the modeling stage of the DSLC has been advocated in ref. 15 as a minimum requirement for reproducibility and interpretability. Modeling stage stability unifies numerous previous works, including Jackknife, subsampling, bootstrap sampling, robust statistics, semiparametric statistics, and Bayesian sensitivity analysis (ref. 15 and references therein). These methods have been enabled in practice through computational advances and allow researchers to investigate the reproducibility of data results. Econometric models with partial identification (ref. 16 and references therein) and fundamental theoretical results in statistics, such as the central limit theorem (CLT), can also be viewed as stability considerations.

In this paper, we unify and expand on these ideas through the PCS framework. Our framework, which consists of the PCS workflow and transparent PCS documentation, uses predictability as a reality check, computability to ensure results are tractable, and stability to test the reproducibility of results relative to human decisions (*PCS Principles in the DSLC*). The PCS workflow may be broadly interpreted as the practice of incorporating these three principles into each step of the DSLC (Fig. 1). We detail this workflow at the modeling stage through basic PCS inference (*PCS Inference through Perturbation Analysis*), which can be extended to other steps in the DSLC (*Stability Assumptions Initiate the DSLC*) through data and model perturbations. PCS documentation justifies decisions made throughout the DSLC in R MarkDown or a Jupyter (iPython) Notebook using narratives, code, and visualizations (*PCS Documentation*). We draw connections between causal inference and the PCS framework, demonstrating the utility of the latter for generating scientific hypotheses (*PCS Recommendation System for Scientific Hypothesis Generation*). We conclude by discussing areas for further work, including additional vetting of the framework and theoretical analyses on connections between the three principles. A case study of our framework based on the authors' work studying gene regulation in *Drosophila* is documented on Zenodo.

PCS Principles in the DSLC

Given a domain problem and data, the purpose of the DSLC is to generate knowledge, conclusions, and actions (Fig. 1). The PCS framework aims at veridical data science through the three fundamental principles. Below we discuss the roles of these three principles, organizing our discussion with respect to the steps in the DSLC.

Stability Assumptions Initiate the DSLC. The ultimate goal of the DSLC is to generate knowledge that is useful for future actions, be it a biological experiment, a business decision, or government policy. Stability is a useful concept to address whether another researcher making alternative, appropriate[¶] decisions would obtain similar conclusions. At the modeling stage, stability has previously been advocated in ref. 15. In this context, stability refers to acceptable consistency of a data result relative to appropriate perturbations of the data or model. For example, jackknife (17–19), bootstrap (20), and cross-validation (6, 7) may be considered appropriate perturbations if the data are deemed approximately independent and identically distributed (i.i.d.) based on domain knowledge and an understanding of the data collection process.

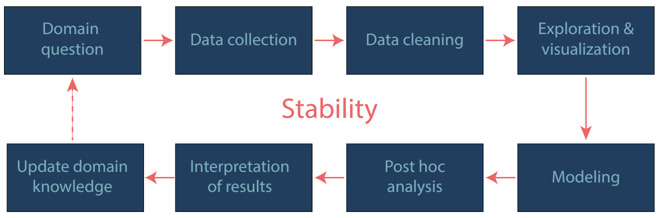

Fig. 1. The data science life cycle. The PCS workflow considers predictability, computability, and stability at every step, with a strong emphasis on stability.

Human judgment calls prior to modeling also impact data results. The validity of an analysis relies on implicit stability assumptions that allow data to be treated as an informative representation of some natural phenomena. When these assumptions do not hold, conclusions rarely generalize to new settings unless empirically proved by future data. This makes stability assessments essential to guard against costly future actions and false discoveries, particularly in science, business, and public policy, where data results are used to guide large-scale actions, and in medicine, where human lives are at stake. Below we outline stability considerations that impact the DSLC prior to modeling.

Question or problem formulation. The DSLC begins with a domain problem or a question, which could be hypothesis driven or discovery based. For instance, a biologist may want to discover biomolecules that regulate a gene's expression. In the DSLC this question must be translated into a question regarding the output of a model or analysis of data that can be measured/collected. There are often multiple translations of a domain problem into a data science problem. For example, the biologist described above could measure factors binding regulatory regions of the DNA that are associated with the gene of interest. Alternatively, the biologist could study how the gene covaries with regulatory factors across time and space. From a modeling perspective, the biologist could identify important features in a random forest or through logistic regression. Stability relative to question or problem formulation implies that the domain conclusions are qualitatively consistent across these different translations.

Data collection. To answer a domain question, domain experts and data scientists collect data based on prior knowledge and available resources. When these data are used to guide future decisions, researchers implicitly assume that the data are relevant to a future time. In other words, they assume that conditions affecting data collection are stable, at least relative to some aspects of the data. For instance, if multiple laboratories collect data to answer a domain question, protocols must be comparable across experiments and laboratories if they expect to obtain consistent results. These stability considerations are closely related

[§]We differentiate between the notions of stability and robustness as used in statistics. The latter has traditionally been used to investigate performance of statistical methods across a range of distributions, while the former captures a much broader range of perturbations throughout the DSLC as discussed in this paper. At a high level, stability is about robustness.

[¶]We use the term appropriate to mean well justified from domain knowledge and an understanding of the data-generating process. The term "reasonable" has also been used with this definition (15).

to external validity in medical research, which characterizes similarities between subjects in a study and subjects that researchers hope to generalize results to. We discuss this idea more in *Predictability as Reality Check*.

Data cleaning and preprocessing. Statistics and machine-learning models or algorithms help data scientists answer domain questions. Using models or algorithms requires cleaning (preprocessing) raw data into a suitable format, be it a categorical demographic feature or continuous measurements of biomarker concentrations. For instance, when data come from multiple laboratories, biologists must decide how to normalize individual measurements (for example, see ref. 21). When data scientists preprocess data, they are implicitly assuming that their choices do not unintentionally bias essential information in the raw data. In other words, they assume that the knowledge derived from data is stable with respect to their processing choices. If such an assumption cannot be justified, they should use multiple appropriate processing methods and interpret results that are stable across these methods. Others have advocated evaluating results across alternatively processed datasets under the name "multiverse analysis" (22). Although the stability principle was developed independently of this work, it naturally leads to a multiverse-style analysis.

Exploratory data analysis. Both before the modeling stage and in post hoc analyses, data scientists engage in exploratory data analysis (EDA) to identify interesting relationships in the data and interpret results. When visualizations or summaries are used to communicate these analyses, it is implicitly assumed that the relationships or results are stable with respect to any decisions made by the data scientist. For example, if the biologist believes that clusters in a heatmap are biologically meaningful, the biologist should expect to observe the same clusters with respect to any appropriate choice of distance metric, data perturbation, or clustering method.

Predictability as Reality Check[#].

After data collection, cleaning/preprocessing, and EDA, models or algorithms[||] are often used to identify more complex relationships in data. Many essential components of the modeling stage rely on the language of mathematics, both in technical papers and in code. A seemingly obvious but often ignored question is why conclusions presented in the language of mathematics depict external reality and to what extent we should trust mathematical conclusions to impact this external reality.[††]

This concern has been articulated and addressed by many others in terms of prediction. For instance, Philip Dawid (23) drew connections between statistical inference and prediction under the name "prequential statistics," highlighting the importance of forecasts in statistical analyses. David Freedman (24) argued that when a model's predictions are not tested against reality, conclusions drawn from the model are unreliable. Seymour Geisser (25) advocated that statistical analyses should focus on prediction rather than parametric inference, particularly in cases where the statistical model is an inappropriate description of reality. Leo Breiman et al. (5) championed the essential role of prediction in developing realistic models that yield sound scientific conclusions. It can even be argued that the goal of most domain problems is prediction at the metalevel. That is, the primary value of learning relationships in data is often to predict some aspect of future reality.

[#]Predictability is a form of empirical validation. Other reality checks may be performed beyond prediction (e.g., checking whether a model recovers known phenomena).

[||]Different model or algorithm choices could correspond to different translations of a domain problem.

[††]The PCS documentation in *PCS Documentation* helps users assess whether this connection is reliable.

Formulating prediction. We describe a general framework for prediction with data $D=(\mathbf{x},y)$, where $\mathbf{x}\in\mathcal{X}$ represents input features and $y\in\mathcal{Y}$ the prediction target. Prediction targets $y\in\mathcal{Y}$ may be observed responses (e.g., supervised learning) or extracted from data (e.g., unsupervised learning). Predictive accuracy is a simple, quantitative metric to evaluate how well a model represents relationships in D. It is well defined relative to a prediction function, testing data, and an evaluation metric. We detail each of these elements below.

Prediction function. The prediction function

$$h:\mathcal{X}\to\mathcal{Y} \qquad [1]$$

represents relationships between the observed features and the prediction target. For instance, in the case of supervised learning h may be a linear predictor or decision tree. In this setting, y is typically an observed response, such as a class label. In the case of unsupervised learning, h could map from input features to cluster centroids.

To compare multiple prediction functions, we consider

$$\{h^{(\lambda)}:\lambda\in\Lambda\}, \qquad [2]$$

where Λ denotes a collection of models/algorithms. For example, Λ may define different tuning parameters in lasso (26) or random forest (27). For deep neural networks, Λ could describe different network architectures. For algorithms with a randomized component, such as *k*-means or stochastic gradient descent, Λ can represent repeated runs. More broadly, Λ may describe a set of competing algorithms such as linear models, random forests, and neural networks, each corresponding to a different problem translation. We discuss model perturbations in more detail in *Algorithm or model perturbation*.

Testing (held-out) data. We distinguish between training data that are used to fit a collection of prediction functions and testing data that are used to evaluate the accuracy of fitted prediction functions[‡‡]. At a minimum, one should evaluate predictive accuracy on a held-out test set generated at the same time and under the same conditions as the training data (e.g., by randomly sampling a subset of observations). This type of assessment addresses questions of internal validity, which describe the strength of a relationship in a given sample. It is also often important to understand how a model will perform in future conditions that differ from those that generated the training data. For instance, a biologist may want to apply a model to new cell lines. A social scientist might use a model trained on residents from one city to predict the behavior of residents in another city. As an extreme example, one may want to use transfer learning to apply part of one's model to an entirely new prediction problem. Testing data gathered under different conditions from the training data directly addresses questions of external validity, which describe how well a result will generalize to future observations. Domain knowledge and/or empirical validation are essential to assess the appropriateness of different prediction settings. These decisions should be reported in the proposed PCS documentation (*PCS Documentation*).

Prediction evaluation metric. The prediction evaluation metric

$$\ell:\mathcal{H}\times\mathcal{X}\times\mathcal{Y}\to\mathbb{R}_+ \qquad [3]$$

quantifies the accuracy of a prediction function $h\in\mathcal{H}$ by measuring the similarity between $h(\mathbf{x})$ and y. We adopt the convention that larger values of $\ell(h,\mathbf{x},y)$ correspond to worse predictive accuracy. ℓ should be selected to reflect domain-specific considerations, such as the types of errors that are more costly. In fact,

[‡‡]In some settings, a third set of data is used to tune model parameters.

there is an entire area of research devoted to evaluating probabilistic forecasts through "scoring rules" (ref. 28 and references therein). In some cases, it may be appropriate to consider multiple prediction evaluation metrics and focus on models that are accurate with respect to all.

Prediction requires human input to formulate, including the preferred structure of a model/algorithm and what it means for a model to be suitably accurate. For example, the biologist studying gene regulation may believe that the simple rules learned by decision trees are an appealing representation of interactions that exhibit thresholding behavior (29). If the biologist is interested in a particular cell type, the biologist may evaluate prediction accuracy on test data measuring only these environments. If the responses are binary with a large proportion of class-0 responses, the biologist may choose ℓ to handle the class imbalance. All of these decisions should be documented and argued for (*PCS Documentation*) so that other researchers can assess the strength of conclusions based on transparent evidence. The accompanying PCS documentation provides a detailed example.

Cross-validation. As alluded to earlier, CV has become a powerful approach to select regularization parameters when data are approximately i.i.d. (6, 7). CV divides data into blocks of observations, trains a model on all but one block, and evaluates the prediction error over each held-out block. In other words, CV incorporates the scientific principle of replication by evaluating whether a model accurately predicts the responses of pseudoreplicates. CV works more effectively as a tool to select regularization parameters than as an estimate of prediction error, where it can incur high variability due to the often positive dependencies between the estimated prediction errors in the summation of the CV error (30). Just as peer reviewers make judgment calls on whether a laboratory's experimental conditions are suitable to replicate scientific results, data scientists must determine whether a removed block represents a justifiable pseudoreplicate, which requires information from the data collection process and domain knowledge.

Computability. In a broad sense, computability is the gatekeeper of data science. If data cannot be generated, stored, managed, and analyzed efficiently and scalably, there is no data science. Modern science relies heavily on information technology as part of the DSLC. Each step, from raw data collection and cleaning to model building and evaluation, relies on computing technology and falls under computability in a broad sense. In a narrow sense, computability refers to the computational feasibility of algorithms or model building.

Here we use computability in the narrow sense, which is closely associated with the rise of machine learning over the last three decades. Just as scientific instruments and technologies determine what processes can be effectively measured, computing resources and technologies determine the types of analyses that can be carried out. In particular, computability is necessary to carry out predictability and stability analyses within the PCS framework. Computational constraints can also serve as a device for regularization. For example, stochastic gradient descent is widely used for optimization in machine-learning problems (31). Both the stochasticity and early stopping of a stochastic gradient algorithm play the role of implicit regularization.

Computational considerations and algorithmic analyses have long been an important part of statistics and machine learning. Even before digital computing, calculus played a computational role in statistics through Taylor expansions applied to different models. In machine learning, computational analyses consider the number of operations and required storage space in terms of observations n, features p, and tuning (hyper)parameters. When the computational cost of addressing a domain problem or question exceeds available computational resources, a result is not computable. For instance, the biologist interested in gene regulation may want to model interaction effects in a supervised learning setting. However, there are $O(p^s)$ possible order-s interactions among p regulatory factors. For even a moderate number of factors, exhaustively searching for high-order interactions is not computable. In such settings, data scientists must restrict modeling decisions to draw conclusions. Thus it is important to document why certain restrictions were deemed appropriate and the impact they may have on conclusions (*PCS Documentation*).

Increases in computing power also provide an unprecedented opportunity to enhance analytical insights into complex natural phenomena. We can now store and process massive datasets, which can be used to simulate large-scale processes. Simulations provide concrete, quantitative representations of natural phenomena relative to known input parameters, which can be perturbed to assess the stability of data results. As a result, simulation experiments inspired by observed data and domain knowledge are powerful tools to understand how results may behave in real-world settings. They represent a best effort to emulate complex processes, where the reliability of results is not always clear. Pairing such simulation studies with empirical evidence makes the DSLC more transparent for peers and users to review, aiding in the objectivity of science.

Stability at the Modeling Stage. Computational advances have fueled our ability to analyze the stability of data results in practice. At the modeling stage, stability measures how a data result changes when the data and/or model are perturbed (15). Stability extends the concept of sampling variability in statistics, which is a measure of instability relative to other data that could be generated from the same distribution. Statistical uncertainty assessments implicitly assume stability in the form of a distribution that generated the data. This assumption highlights the importance of other datasets that could be observed under similar conditions (e.g., by another person in the laboratory or another laboratory at another time).

The concept of a model ("true") distribution[§§] is a construct. When randomization is explicitly carried out, the model distribution can be viewed as a physical construct. Otherwise, it is a mental construct that must be justified through domain knowledge, an understanding of the data-generating process, and downstream utility. Statistical inference procedures use distributions to draw conclusions about the real world. The relevance of such conclusions requires empirical support for the postulated model distribution, especially when it is a mental construct. In data science and statistical problems, practitioners often do not make much of an attempt to justify this mental construct. At the same time, they take the uncertainty conclusions very seriously. This flawed practice is likely related to the high rate of false discoveries (2, 3). It is a major impediment to progress of science and to data-driven knowledge extraction in general.

While the stability principle encapsulates uncertainty quantification when the model distribution construct is well supported, it is intended to cover a much broader range of perturbations, such as problem formulation (e.g., different problem translations), preprocessing, EDA, randomized algorithms, and choice of models/algorithms. Although seldom carried out in practice, evaluating stability across the entire DSLC is necessary to ensure that results are reliable and reproducible. For example, the biologist studying gene regulation must choose both how to normalize raw data and what algorithm(s) will be used in analysis. When

[§§]We use the term "model" distribution to avoid confusion over whether it is well justified.

there is no principled approach to make these decisions, the knowledge data scientists can extract from analyses is limited to conclusions that are stable across appropriate choices (22, 32, 33). This ensures that other scientists studying the same data will reach similar conclusions, despite slight variation in their independent choices.

Formulating stability at the modeling stage. Stability at the modeling stage is defined with respect to a stability target, an appropriate perturbation to the data and/or algorithm/model, and a stability metric to measure the change in target upon perturbation. We describe each of these in detail below.

Stability target. The stability target

$$\mathcal{T}(D, \lambda) \qquad [4]$$

corresponds to the data result or estimand of interest. It depends on input data D and a specific model/algorithm λ used to analyze the data. For simplicity, we will sometimes suppress the dependence on D and λ in our notation. As an example, $\mathcal{T}$ can represent responses predicted by $h^{(\lambda)}$. Other examples of $\mathcal{T}$ include features selected by lasso with penalty parameter λ or saliency maps derived from a convolutional neural network (CNN) with architecture λ.

Data and model/algorithm perturbations. To evaluate the stability of a data result, we measure the change in target $\mathcal{T}$ that results from a perturbation to the input data or learning algorithm. More precisely, we define a collection of data perturbations $\mathbf{D}$ and model/algorithm perturbations Λ and compute the stability target distribution

$$\{\mathcal{T}(D, \lambda) : D \in \mathbf{D}, \lambda \in \Lambda\}. \qquad [5]$$

For example, appropriate data perturbations include bootstrap sampling when observations are approximately i.i.d., block bootstrap for weakly dependent time series, generative models that are supported by domain knowledge (*Data perturbation*), and probabilistic models that are justified from an understanding of the data-generating process or explicit randomization. When different prediction functions are deemed equally appropriate based on domain knowledge, each may represent an appropriate model perturbation (*Algorithm or model perturbation*).

It can be argued that the subjectivity surrounding appropriate perturbations makes it difficult to evaluate results within the PCS framework. Indeed, perturbation choices are both subjective human judgment calls and critical considerations of PCS. The degree to which a data result can be trusted depends on the justification for a perturbation. This is true if the perturbation comes from a probabilistic model, as in traditional statistical inference, or some broader set of perturbations, as in PCS. The goal of PCS is to use and explicitly document perturbations that are best suited to assess stability in complex, high-dimensional data rather than relying on probabilistic models alone, which have little objective meaning when the model is not justified. To ensure that results can be evaluated, the case for an appropriate perturbation must be made in the publication and in the PCS documentation (*PCS Documentation*). These transparent narratives allow readers to scrutinize and discuss perturbations to determine which one should be applied for a particular field and/or type of data, encouraging objectivity.

Stability evaluation metric. The stability evaluation metric $s(\mathcal{T}; \mathbf{D}, \Lambda)$ summarizes the stability target distribution in Eq. **5**. For example, if $\mathcal{T}$ indicates features selected by a model trained on data D, we may report the proportion of times each feature is selected across data perturbations $D \in \mathbf{D}$. If $\mathcal{T}$ corresponds to saliency maps derived from different CNN architectures $\lambda \in \Lambda$, we may report each pixel's range of salience across Λ. When the stability evaluation metric combines targets across model/algorithm perturbations, it is important that these different targets are scaled appropriately to ensure comparability.

A stability analysis that reveals the target $\mathcal{T}$ is unstable (relative to a meaningful threshold for a domain problem) may suggest an alternative analysis or target. This raises issues of multiplicity and/or overfitting if the same data are used to evaluate new stability targets. Held-out test data offer one way to mitigate these concerns. That is, training data can be used to identify a collection of targets that are suitably stable. These targets can then be evaluated on the test data. More broadly, the process of refining analyses and stability targets can be viewed as part of the iterative approach to data analysis and knowledge generation described by ref. 34. Before defining a new target or analysis, it may be necessary to collect new data to help ensure reproducibility and external validity.

Data perturbation. The goal of data perturbation under the stability principle is to mimic a process that could have been used to produce model input data but was not. This includes human decisions, such as preprocessing and data cleaning, as well as data-generating mechanisms. When we focus on the change in target under possible realizations of the data from a well-supported probabilistic model, we arrive at well-justified sampling variability considerations in statistics. Hence data perturbation under the stability principle includes, but is much broader than, the concept of sampling variability. It formally recognizes many other important considerations in the DSLC beyond sampling variability. Furthermore, it provides a framework to assess trust in estimates of $\mathcal{T}$ when a probabilistic model is not well justified and hence sampling interpretations are not applicable.

Data perturbations can also be used to reduce variability in the estimated target, which corresponds to a data result of interest. Random forests incorporate subsampling data perturbations (of both the data units and predictors) to produce predictions with better generalization error (27). Generative adversarial networks (GANs) use synthetic adversarial examples to retrain deep neural networks and produce predictions that are more robust to such adversarial data points (35). Bayesian models based on conjugate priors lead to marginal distributions that can be derived by adding observations to the original data. Thus they can be viewed as a form of data perturbation that implicitly introduces synthetic data through the prior. Empirically supported generative models, including partial differential equations (PDEs), can be used to explicitly introduce synthetic data. As with Bayesian priors, synthetic data perturbations from generative models can be used to encourages stability of data results relative to prior knowledge, such as mechanistic rules based on domain knowledge (for examples see ref. 36).

Algorithm or model perturbation. The goal of algorithm or model perturbation is to understand how alternative analyses of the same data affect the target estimate. A classic example of model perturbation is from robust statistics, where one searches for a robust estimator of the mean of a location family by considering alternative models with heavier tails than the Gaussian model. Another example of model perturbation is sensitivity analysis in Bayesian modeling (37, 38). Many of the model conditions used in causal inference are in fact stability concepts that assume away confounding factors by asserting that different conditional distributions are the same (39, 40).

Modern algorithms often have a random component, such as random projections or random initial values in gradient descent and stochastic gradient descent. These random components provide natural model perturbations that can be used to assess the stability of $\mathcal{T}$. In addition to the random components of a single algorithm, multiple models/algorithms can be used to evaluate stability of the target. This is useful when there are many appropriate choices of model/algorithm and no established criteria or established domain knowledge to select among them. The stability principle calls for interpreting only the targets of interest that are stable across these choices of algorithms or models (32).

As with data perturbations, model perturbations can help reduce variability or instability in the target. For instance, ref. 41 selects lasso coefficients that are stable across different regularization parameters. Dropout in neural networks is a form of algorithm perturbation that leverages stability to improve generalizability (42). Our previous work (33) stabilizes random forests to interpret decision rules in tree ensembles (33, 43), which are perturbed using random feature selection (model perturbation) and bootstrap (data perturbation).

Dual Roles of Generative Models in PCS. Generative models include both probabilistic models and PDEs with initial or boundary conditions. These models play dual roles in the PCS framework. On one hand, they can concisely summarize past data and prior knowledge. On the other hand, they can be used to generate synthetic observations that offer a form of data perturbation.

When a generative model is used to summarize data, a common target of interest is the model's parameters. Generative models with known parameters may be used for prediction or to advance understanding through the mechanistic rules they represent. Such models correspond to infinite data, although finite under computational constraints. Generative models with unknown parameters can be used to motivate surrogate loss functions through maximum-likelihood and Bayesian modeling methods. Mechanistic interpretations of such models should not be used to draw scientific conclusions. They are simply useful starting points to optimize algorithms that must be subjected to empirical validation.

Generative models that approximate the data-generating process, a human judgment call argued for in the PCS documentation, can be used as a form of data perturbation. Here synthetic data augment the observed data and serve the purpose of domain-inspired regularization. The amount of synthetic data to combine with the observed data reflects our degree of belief in the models and is an interesting area for future exploration. Using synthetic data for domain-inspired regularization allows the same algorithmic and computing platforms to be applied to the combined data. This style of analysis is reminiscent of AdaBoost, which uses the current data and model to modify the data used in the next iteration without changing the base learner (44).

Connections among the PCS Principles. Although we have discussed the three principles of PCS individually, they share important connections. Computational considerations can dictate tractable predictive models/algorithms, particularly for large, high-dimensional datasets. These computability issues are often addressed in practice through scalable optimization methods such as gradient descent (GD) or stochastic gradient descent (SGD). Evaluating predictability on held-out data is a form of stability analysis where the training/test sample split represents a data perturbation. Other perturbations used to assess stability require multiple runs of similar analyses. Parallel computation is well suited for these perturbations.

PCS Inference through Perturbation Analysis

When data results are used to guide future decisions or actions, it is important to assess the quality of the target estimate. For instance, suppose a model predicts that an investment will generate a 10% return over 1 y. Intuitively, this prediction suggests that "similar" investments return 10% on average. Whether or not a particular investment will realize a return close to 10% depends on whether returns for similar investments ranged from -20% to 40% or from 8% to 12%. In other words, the variability of a prediction conveys important information about how much one should trust it.

In traditional statistics, confidence measures describe the uncertainty of an estimate due to sampling variability under a well-justified probabilistic model. However, decisions made throughout the DSLC add another layer of uncertainty that may bias data results. This issue has been previously acknowledged in the modeling stage by the authors of ref. 45, who derive "hacking intervals" to assess the range of a summary statistic optimized over a possible set of data and algorithm perturbations. In the PCS framework, we propose perturbation intervals or perturbation regions in general, to quantify the stability of target estimates relative to different perturbations, including data cleaning and problem translations. Perturbation intervals are conceptually similar to confidence intervals. The primary difference is that they are explicitly connected to perturbations, justified in PCS documentation (*PCS Documentation*) and evaluated by independent reviewers and domain experts.

As an example, perturbation intervals for a target parameter from a single method based on bootstrap sampling specialize to traditional confidence intervals based on the bootstrap. More broadly, perturbation intervals quantify the variability of a target parameter value across the entire DSLC. For instance, a data scientist may consider multiple preprocessing, subsampling, and modeling strategies to predict investment returns. The resulting perturbation intervals describe the range of returns across worlds represented by each perturbation. Their reliability lies squarely on whether the set of perturbations captures the full spectrum of appropriate choices that could be made throughout the DSLC, which should be evaluated by domain experts and independent reviewers. This highlights the importance of perturbations that could plausibly generate the observed data, represent the range of uncertainty surrounding an analysis to the best degree possible, and are transparently documented for others to evaluate (*PCS Documentation*).

As a starting point, we focus on a basic form of PCS inference that generalizes traditional statistical inference. Our approach to inference allows for a range of data and algorithm/model perturbations, making it flexible in its ability to represent uncertainty throughout the DSLC.

PCS Perturbation Intervals. The reliability of perturbation intervals lies on the appropriateness of each perturbation. Consequently, perturbation choices should be seriously deliberated, clearly communicated, and evaluated by objective reviewers. Here we propose a framework for PCS inference based on a single problem translation and target estimand, leaving the case of multiple translations/estimands to future work.[¶¶]

1) Problem formulation: Translate the domain question into a data science problem. Define a prediction target y, appropriate data $\mathbf{D}$ and/or model Λ perturbations, prediction function(s) $\{h^{(\lambda)} : \lambda \in \Lambda\}$, training/test split, prediction evaluation metric ℓ, stability metric s, and stability target $\mathcal{T}(D, \lambda)$. Document why these choices are appropriate in the context of the domain question.
2) Prediction screening: For a threshold τ, screen out models that do not fit the data (via prediction accuracy):

$$\Lambda^* = \{\lambda \in \Lambda : \ell(h^{(\lambda)}, \mathbf{x}, y) < \tau\}. \quad [6]$$

[¶¶]The PCS perturbation intervals cover different problem translations through Λ and are clearly extendable to include perturbations in the preprocessing step through $\mathbf{D}$.

Examples of appropriate threshold include domain accepted baselines, the top k performing models, or models whose accuracy is suitably similar to the most accurate model. If the goal of an analysis is prediction, testing data should be held out until reporting the final prediction accuracy of a model in step 4. In such cases, Eq. **6** can be evaluated using a surrogate sample-splitting approach (e.g., CV). If the goal of an analysis extends beyond prediction (e.g., to feature selection), Eq. **6** may be evaluated on held-out test data.

3) Target value perturbation distributions: For each of the survived models Λ^* from step 2, compute the stability target under each data perturbation **D**. This results in a joint distribution of the target over data and model perturbations as in Eq. **5**. For a collection of perturbations, requiring stability of $\mathcal{T}$ across all perturbations is more conservative in terms of type I error than requiring stability for any single perturbation. However, different domain questions require control over different types of error. How and when to combine results across perturbations is thus a human judgment call that should be transparently justified and documented.
4) Perturbation result reporting: Summarize the target value perturbation distribution using the stability metric s. For instance, if $\mathcal{T}$ is one dimensional, we could summarize its perturbation distribution using the 10th and 90th percentiles or a visualization. If $\mathcal{T}$ is multidimensional, we could report a low-dimensional projection of the perturbation distribution. When perturbation results combine targets across models/algorithms, they may need to be rescaled to ensure comparability. When perturbation intervals are reported separately for model/algorithm perturbation, predictive accuracy evaluated in step 2 may be used as a measure of trust to rank each interval.

At a high level, the PCS inference uses perturbation intervals to identify the stable part of accurate models. If perturbation results reveal instability among accurate models, PCS inference can be used to interpret aspects that are shared (i.e., stable) across these models. In this setting, PCS can be viewed as an implicit application of Occam's razor. That is, it draws conclusions from the stable portion of predictive models to simplify data results, making them more reliable and easier to interpret. If perturbation intervals reveal that complex models are both stable and accurate, PCS inference provides justification for the added complexity.

PCS Hypothesis Testing. Hypothesis testing from traditional statistics is commonly used in decision making for science and business alike. The heart of Fisherian testing (46) lies in calculating the *P* value, which represents the probability of an event more extreme than in the observed data under a null hypothesis or distribution. Smaller *P* values correspond to stronger evidence against the null hypothesis or (ideally) the scientific theory embedded in the null hypothesis. For example, we may want to determine whether a particular gene is differentially expressed between patients with breast cancer and a control group. Given i.i.d. random samples from each population, we could address this question in the classical hypothesis-testing framework using a *t* test. The *P* value describes the probability of seeing a difference in means more extreme than observed if the genes are not differentially expressed.

While hypothesis testing is valid philosophically, many of the assumptions that it relies on are unrealistic in practice. For instance, unmeasured confounding variables can bias estimates of causal effects. These issues are particularly relevant in the social sciences, where randomized trials are difficult or impossible to conduct. Resource constraints can limit how data are collected, resulting in samples that do not reflect the population of interest, distorting the probabilistic interpretations of traditional statistical inference. Moreover, hypothesis testing assumes empirical validity of probabilistic data-generating models.[##] When randomization is not carried out explicitly, a particular null distribution must be justified from domain knowledge of the data-generating mechanism. Such issues are seldom taken seriously in practice, resulting in settings where the null distribution is far from the observed data. As a result, *P* values as small as 10^{-5} or 10^{-8} are now common to report, despite the fact that there are rarely enough data to reliably calculate these values, especially when multiple hypotheses (e.g., thousands of genes) are evaluated. When results are so far off on the tail of the null distribution, there is no empirical evidence as to why the tail should follow a particular parametric distribution. Moreover, hypothesis testing as practiced today often relies on analytical approximations or Monte Carlo methods, where issues arise for such small probability estimates. In fact, there is a specialized area of importance sampling to deal with simulating small probabilities (48, 49), but these ideas have not been widely adopted in practice.

PCS hypothesis testing builds on perturbation intervals to address these practical issues and the cognitively misleading nature of small *P* values. It uses the null hypothesis to define constrained perturbations that represent a plausible data-generating process, which in the best case corresponds to an existing scientific theory. This includes probabilistic models, when they are well founded, as well as other data and/or algorithm perturbations. For instance, generative models based on PDEs can be used to simulate data according to established physical laws. Alternatively, a subset of data can be selected as controls (for example, see ref. 50). By allowing for a broad class of perturbations, PCS hypothesis testing allows us to compare observed data with data that respect some simple structure known to represent important characteristics of the domain question. Of course, the appropriateness of a perturbation is a human judgment call that should be clearly communicated in PCS documentation and debated by researchers. Much like scientists deliberate over appropriate controls in an experiment, data scientists should debate the appropriate perturbations in a PCS analysis.

Formalizing PCS hypothesis testing. Formally, we consider settings with observable input features $\mathbf{x} \in \mathcal{X}$, prediction target $y \in \mathcal{Y}$, prediction functions $\{h^{(\lambda)} : \lambda \in \Lambda\}$, and a null hypothesis that qualitatively describes some aspect of the domain question. PCS hypothesis testing translates the null hypothesis into a constrained perturbation and generates data

$$D_0 = \{\mathbf{x}_0, y_0\} \qquad [7]$$

according to this perturbation.[†††] The particular choice of constrained perturbation should be explicitly documented and justified by domain knowledge. We use the constrained perturbation to construct and compare perturbation intervals for both D_0 and D and evaluate whether the observed data are consistent with the hypothesis embedded in D_0.

PCS Inference in Neuroscience and Biology. The work in ref. 51 considers the null hypothesis that population-level structure in single-neuron data is the expected byproduct of primary features (e.g., correlations across time). This can be viewed as a form of PCS inference. The authors use a maximum-entropy

[##] Under conditions, Freedman and Lane (47) showed that some tests can be approximated by permutation tests when data are not generated from a probabilistic model, but these results are not broadly applicable.

[†††] A null hypothesis may correspond to multiple data or model/algorithm perturbations. We focus on a single data perturbation here for simplicity.

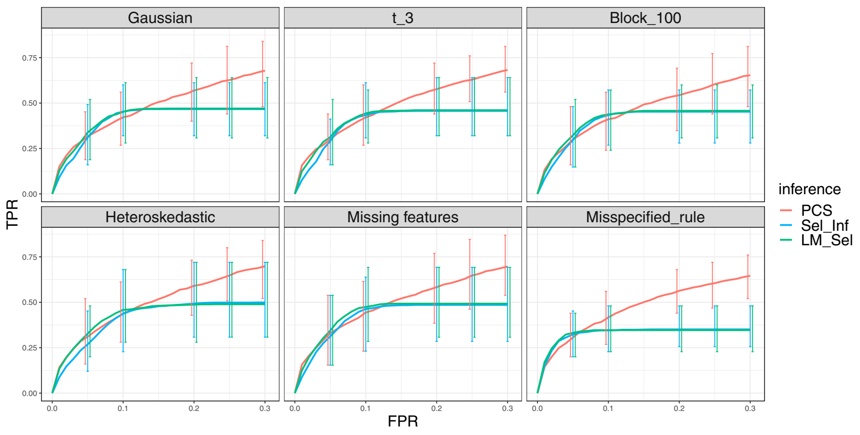

Fig. 2. ROC curves showing true positive rate (TPR) and false positive rate (FPR) for feature selection in a linear model setting with $n = 250$ observations. Each plot corresponds to a different generative model. Color corresponds to method of inference: red, PCS; blue, selective inference; green, linear model asymptotic normality. Error bars give the 10th and 90th percentiles over replicates.

approach, whose constraint is represented by the number of moments, to generate data that share primary features with the observed data but are otherwise random, and compare population-level findings between the observed and simulated data. In the accompanying PCS documentation, we consider the null hypothesis that genomic interactions appear with equal frequency among different classes of genomic elements. We use a sample-splitting strategy which treats inactive elements (class-0 observations) as a baseline to determine whether interactions appear with "unusual" frequency. Once again, these comparisons rely on human judgment to determine when results are sufficiently different. These choices depend on the domain context and how the problem has been translated. They should be transparently communicated by the researcher in the PCS documentation.

PCS Inference Simulation Studies in Sparse Linear Models. We tested PCS inference in an extensive set of data-inspired simulation experiments in the sparse linear model setting that has been widely studied by the statistics community over the past two decades (*SI Appendix*). In total, we considered six distinct generative models intended to reflect some of the issues that arise in practice. We compared our proposed PCS inference procedure with selective inference and asymptotic normality results using receiver operating characteristic (ROC) curves. These provide a useful criterion to assess false positives and true positives, which are both important considerations in settings where resources dictate how many findings can be evaluated in follow-up analyses/experiments. Across all models, PCS inference compares favorably to both selective inference and asymptotic normality results (Fig. 2). However, we note that the principal advantage of PCS inference is that it can be easily generalized to more complex settings faced by data scientists today as in the two examples described above.

PCS Documentation

The PCS framework includes an accompanying R Markdown or Jupyter (iPython) Notebook, which seamlessly integrates narratives, codes, and analyses. These narratives are necessary to describe the domain problem and support assumptions and choices made by the data scientist regarding computational platform, data cleaning and preprocessing, data visualization, model/algorithm, prediction metric, prediction evaluation, stability target, data and algorithm/model perturbations, stability metric, and data conclusions in the context of the domain problem. These narratives should be based on referenced prior knowledge and an understanding of the data collection process, including design principles or rationales. The narratives in the PCS documentation help bridge or connect the two parallel universes of reality and models/algorithms that exist in the mathematical world (Fig. 3). In addition to narratives justifying human judgment calls (possibly with data evidence), PCS documentation should include all codes used to generate data results with links to sources of data and metadata.

We propose the following steps in a notebook[‡‡‡]:

1) Domain problem formulation (narrative). Clearly state the real-world question and describe prior work related to this question. Indicate how this question can be answered in the context of a model or analysis.
2) Data collection and storage (narrative). Describe how the data were generated, including experimental design principles, and reasons why the data are relevant to answer the domain question. Describe where data are stored and how they can be accessed by others.
3) Data cleaning and preprocessing (narrative, code, visualization). Describe and justify steps to convert raw data into data for analysis. Ask whether more than one preprocessing method should be used and examine their impacts on the final data results.
4) Exploratory data analysis (narrative, code, visualization). Describe any preliminary analyses that influenced modeling decisions or conclusions along with code and visualizations to support these decisions.
5) Modeling and post hoc analysis (narrative, code, visualization). Carry out PCS inference in the context of the domain question. Specify appropriate model and data perturbations. If necessary, specify null hypotheses and associated perturbations.
6) Interpretation of results (narrative and visualization). Translate the data results to draw conclusions and/or make recommendations in the context of the domain problem.

This documentation gives the reader as much information as possible to make informed judgments regarding the evidence and process for drawing a data conclusion in the DSLC. A case study of the PCS framework in the genomics problem discussed earlier is documented on Zenodo.

[‡‡‡]This list is reminiscent of the list in the "data wisdom for data science" blog that B.Y. wrote at http://www.odbms.org/2015/04/data-wisdom-for-data-science/.

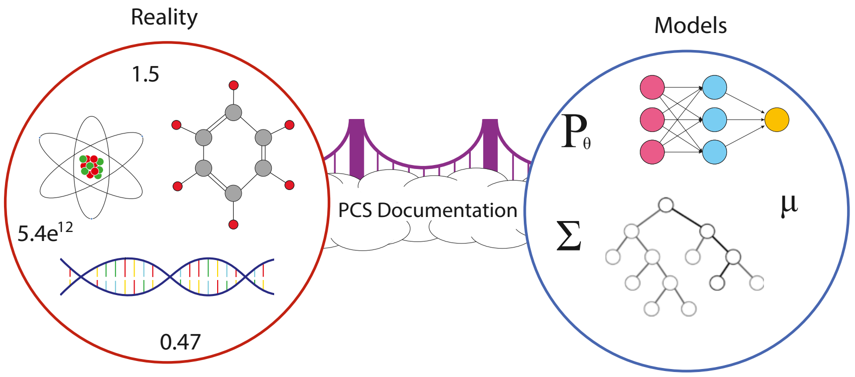

Fig. 3. Assumptions made throughout the DSLC allow researchers to use models such as decision trees, neural networks, or probability distributions as an approximation of reality, which may include physical, chemical, or biological laws. Narratives provided in PCS documentation can help justify assumptions to connect these two worlds.

PCS Recommendation System for Scientific Hypothesis Generation

In general, causality implies predictability and stability over many experimental conditions, but not vice versa. The causal inference community has long acknowledged connections between stability and estimates of causal effects. For instance, many researchers have studied paradoxes surrounding associations that lead to unstable estimates of causal effects (52–54). Estimates in the Neyman–Rubin potential outcomes framework rely on a stable treatment across observational units (55, 56). Sensitivity analyses test the stability of a causal effect relative to unmeasured confounding (57, 58). Stability, particularly with respect to predictions across experimental interventions, has even been proposed as a criterion to establish certain causal relationships under the name "invariance" (39, 59–62).

PCS inference builds on these ideas, using stability and predictability to rank target estimates for further studies, including follow-up experiments. In our recent works on DeepTune (32), iterative random forests (iRF) (33), and signed iterative random forests (siRF) (43), we use PCS inference to make recommendations as inputs to downstream human decisions. For example, PCS inference suggested potential relationships between neurons in the visual cortex and visual stimuli as well as third- and fourth-order interactions among biomolecules that are candidates for regulating gene expression. Predictability and stability do not replace physical experiments to prove or disprove causality. However, we hope computationally tractable analyses that demonstrate high predictability and stability suggest hypotheses or intervention experiments that have higher yields than otherwise. This hope is supported by the fact that 80% of the second-order interactions identified by iRF (33) had been verified in the literature through physical experiments.

Conclusion

In this paper, we unified PCS into a framework for veridical data science, composed of both a workflow and documentation. The PCS framework aims to provide responsible, reliable, reproducible, and transparent results across the DSLC. It is a step toward systematic and unbiased inquiry in data science, similar to strong inference (63). Prediction serves a reality check, evaluating how well a model/algorithm represents the natural phenomena that generated the data. Computability concerns with respect to algorithm efficiency determine the tractability of the DSLC and point to the importance of data-inspired simulations. Stability relative to data and model perturbations was advocated in ref. 15 as a minimum requirement for data results' reproducibility and interpretability.

We made important conceptual progress on stability by extending it to the entire DSLC, including problem formulation, data collection, data cleaning, and EDA. In addition, we developed PCS inference to evaluate the variability of data results with respect to a broad range of perturbations encountered in modern data science. Specifically, we proposed PCS perturbation intervals to evaluate the reliability of results and hypothesis testing to draw comparisons with simple structure in the data. We demonstrated that PCS inference performs favorably in a feature selection problem through data-inspired sparse linear model simulations and in a genomics case study. To communicate human judgment calls in the DSLC, we proposed PCS documentation, which integrates narratives justifying choices with reproducible codes and visualizations. This documentation makes data-driven decisions as transparent as possible for others to evaluate.

In summary, we have offered a conceptual and practical framework to guide the DSLC, but many open problems remain. Basic PCS inference needs to be extended to multitranslations of the same domain question and vetted in practice well beyond the case studies in this paper and in our previous works, especially by other researchers. Additional case studies will help unpack subjective human judgment calls in the context of specific domain problems. The knowledge gained from these studies can be shared and critiqued through transparent documentation. Based on feedback from practice, theoretical studies of PCS procedures in the modeling stage are also called for to gain further insights under empirically vetted, stylized models. Finally, although there have been some theoretical studies on the connections between the three principles (refs. 64 and 65 and references therein), much more work is necessary (66).

Data Availability. All data discussed in this paper are available to readers.

ACKNOWLEDGMENTS. We thank Reza Abbasi-Asl, Yuansi Chen, Ben Brown, Sumanta Basu, Adam Bloniarz, Peter Bickel, Jas Sekhon, and Söreon Künzel for stimulating discussions on related topics in the past many years. We also thank Richard Burk, Chris Holmes, Giles Hooker, Augie Kong, Avi Feller, Peter Bühlmann, Terry Speed, Erwin Frise, Andrew Gelman, Wei-cheng Kuo, and Rebecca Barter for their helpful comments on earlier versions of this paper. We thank Tian Zheng, David Madigan, and the Yu group for discussions that led to the final title of this paper, which was suggested by Tian Zheng. Finally, we thank the reviewers for their extensive and helpful feedback. Partial support is gratefully acknowledged from Army Research Office Grant W911NF1710005, Office of Naval Research Grant N00014-16-1-2664, NSF Grants DMS-1613002 and IIS 1741340, and the Center for Science of Information, a US NSF Science and Technology Center, under Grant CCF-0939370. B.Y. is a Chan Zuckerberg Biohub investigator. We note that an earlier version of this paper appeared as a chapter in the PhD dissertation of K.K.

1. W. J. Murdoch, C. Singh, K. Kumbier, R. Abbasi-Asl, B. Yu, Definitions, methods, and applications in interpretable machine learning. *Proc. Natl. Acad. Sci. U.S.A.* **116**, 22071–22080 (2019).
2. P. B. Stark, A. Saltelli, Cargo-cult statistics and scientific crisis. *Significance* **15**, 40–43 (2018).
3. J. P. A. Ioannidis, Why most published research findings are false. *PLoS Med.* **2**, e124 (2005).
4. K. R. Popper, *The Logic of Scientific Discovery* (University Press, 1959).
5. L. Breiman *et al.*, Statistical modeling: The two cultures (with comments and a rejoinder by the author). *Stat. Sci.* **16**, 199–231 (2001).
6. M. Stone, Cross-validatory choice and assessment of statistical predictions. *J. R. Stat. Soc. B* **36**, 111–133 (1974).
7. D. M. Allen, The relationship between variable selection and data augmentation and a method for prediction. *Technometrics* **16**, 125–127 (1974).
8. A. M. Turing, On computable numbers, with an application to the entscheidungs-problem. *Proc. Lond. Math. Soc.* **2**, 230–265 (1937).
9. J. Hartmanis, R. E. Stearns, On the computational complexity of algorithms. *Trans. Am. Math. Soc.* **117**, 285–306 (1965).
10. M. Li, P. Vitányi, "An introduction to Kolmogorov complexity and its applications" in *Texts in Computer Science*, D. Gries, F. B. Schneider, Eds. (Springer, New York, NY, 2008), vol. 9.
11. A. N. Kolmogorov, On tables of random numbers. *Sankhya Indian J. Stat. Ser. A* **25**, 369–376 (1963).
12. R. A. Fisher, *The Design of Experiments* (Oliver & Boyd, Edinburgh, London, UK, 1937).
13. D. L. Donoho, A. Maleki, I. U. Rahman, M. Shahram, V. Stodden, Reproducible research in computational harmonic analysis. *Comput. Sci. Eng.* **11**, 8–18 (2009).
14. P. B. Stark, Before reproducibility must come preproducibility. *Nature* **557**, 613 (2018).
15. B. Yu, Stability. *Bernoulli* **19**, 1484–1500 (2013).
16. C. F. Manski, *Public Policy in an Uncertain World: Analysis and Decisions* (Harvard University Press, 2013).
17. M. H. Quenouille *et al.*, Problems in plane sampling. *Ann. Math. Stat.* **20**, 355–375 (1949).
18. M. H. Quenouille, Notes on bias in estimation. *Biometrika* **43**, 353–360 (1956).
19. J. Tukey, Bias and confidence in not quite large samples. *Ann. Math. Stat.* **29**, 614 (1958).
20. B. Efron, Bootstrap methods: Another look at the jackknife. *Ann. Statist.* **7**, 1–26 (1979).
21. B. M. Bolstad, R. A. Irizarry, M. Åstrand, T. P. Speed, A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* **19**, 185–193 (2003).
22. S. Steegen, F. Tuerlinckx, A. Gelman, W. Vanpaemel, Increasing transparency through a multiverse analysis. *Perspect. Psychol. Sci.* **11**, 702–712 (2016).
23. A. P. Dawid, Present position and potential developments: Some personal views: Statistical theory: The prequential approach. *J. R. Stat. Soc.* **147**, 278–292 (1984).
24. D. A. Freedman, Statistical models and shoe leather. *Sociol. Methodol.* **21**, 291–313 (1991).
25. S. Geisser, *Predictive Inference* (CRC Press, 1993), vol. 55.
26. R. Tibshirani, Regression shrinkage and selection via the lasso. *J. R. Stat. Soc. B* **58**, 267–288 (1996).
27. L. Breiman, Random forests. *Mach. Learn.* **45**, 5–32 (2001).
28. T. Gneiting, A. E. Raftery, Strictly proper scoring rules, prediction, and estimation. *J. Am. Stat. Assoc.* **102**, 359–378 (2007).
29. L. Wolpert, Positional information and the spatial pattern of cellular differentiation. *J. Theor. Biol.* **25**, 1–47 (1969).
30. T. Fushiki, Estimation of prediction error by using k-fold cross-validation. *Stat. Comput.* **21**, 137–146 (2011).
31. H. Robbins, S. Monro, A stochastic approximation method. *Ann. Math. Stat.* **22**, 400–407 (1951).
32. R. Abbasi-Asl *et al.*, The DeepTune framework for modeling and characterizing neurons in visual cortex area V4. bioRxiv:465534 (9 November 2018).
33. S. Basu, K. Kumbier, J. B. Brown, B. Yu, Iterative random forests to discover predictive and stable high-order interactions. *Proc. Natl. Acad. Sci. U.S.A.* **115**, 1943–1948 (2018).
34. G. E. P. Box, Science and statistics. *J. Am. Stat. Assoc.* **71**, 791–799 (1976).
35. I. Goodfellow *et al.*, "Generative adversarial nets" in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, K. Q. Weinberger, Eds. (MIT Press, 2014), pp. 2672–2680.
36. L. T. Biegler, O. Ghattas, M. Heinkenschloss, B. van Bloemen Waanders, "Large-scale PDE-constrained optimization: An introduction" in *Large-Scale PDE-Constrained Optimization*, L. T. Biegler, O. Ghattas, M. Heinkenschloss, B. van Bloemen Waanders, Eds. (Springer, 2003), pp. 3–13.
37. A. M. Skene, J. E. H. Shaw, T. D. Lee, Bayesian modelling and sensitivity analysis. *The Statistician* **35**, 281–288 (1986).
38. G. E. P. Box, Sampling and Bayes' inference in scientific modelling and robustness. *J. R. Stat. Soc.* **143**, 383–430 (1980).
39. J. Peters, P. Bühlmann, N. Meinshausen, Causal inference by using invariant prediction: Identification and confidence intervals. *J. R. Stat. Soc. B* **78**, 947–1012 (2016).
40. C. Heinze-Deml, J. Peters, N. Meinshausen, Invariant causal prediction for nonlinear models. *J. Causal Inference* **6**, 1–35 (2018).
41. N. Meinshausen, P. Bühlmann, Stability selection. *J. R. Stat. Soc. B* **72**, 417–473 (2010).
42. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**, 1929–1958 (2014).
43. K. Kumbier, S. Basu, J. B. Brown, S. Celniker, B. Yu, Refining interaction search through signed iterative random forests. arXiv:1810.07287 (8 October 2018).
44. Y. Freund, R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.* **55**, 119–139 (1997).
45. B. Coker, C. Rudin, G. King, A theory of statistical inference for ensuring the robustness of scientific results. arXiv:1804.08646 (23 April 2018).
46. R. A. Fisher, "Statistical methods for research workers" in *Breakthroughs in Statistics*, S. Kotz, N. L. Johnson, Eds. (Springer, 1992), pp. 66–70.
47. D. Freedman, D. Lane, A nonstochastic interpretation of reported significance levels. *J. Bus. Econ. Stat.* **1**, 292–298 (1983).
48. G. Rubino, B. Tuffin, *Rare Event Simulation Using Monte Carlo Methods* (John Wiley & Sons, 2009).
49. J. Bucklew, *Introduction to Rare Event Simulation* (Springer Science & Business Media, 2013).
50. M. J. Schuemie, P. B. Ryan, G. Hripcsak, D. Madigan, M. A. Suchard, Improving reproducibility by using high-throughput observational studies with empirical calibration. *Philos. Trans. Math. Phys. Eng. Sci.* **376**, 20170356 (2018).
51. G. F. Elsayed, J. P. Cunningham, Structure in neural population recordings: An expected byproduct of simpler phenomena? *Nat. Neurosci.* **20**, 1310–1318 (2017).
52. R. A. Fisher, Statistical tests of agreement between observation and hypothesis. *Economica* **8**, 139–147 (1923).
53. W. G. Cochran, The omission or addition of an independent variate in multiple linear regression. *J. R. Stat. Soc. Suppl.* **5**, 171–176 (1938).
54. P. J. Bickel, E. A. Hammel, J. W. O'Connell, Sex bias in graduate admissions: Data from Berkeley. *Science* **187**, 398–404 (1975).
55. J. Neyman, Sur les applications de la théorie des probabilités aux experiences agricoles: Essai des principes. *Roczniki Nauk Rolniczych* **10**, 1–51 (1923).
56. D. B. Rubin, Randomization analysis of experimental data: The Fisher randomization test comment. *J. Am. Stat. Assoc.* **75**, 591–593 (1980).
57. J. Cornfield *et al.*, Smoking and lung cancer: Recent evidence and a discussion of some questions. *J. Natl. Cancer Inst.* **22**, 173–203 (1959).
58. P. Ding, T. J. VanderWeele, Sensitivity analysis without assumptions. *Epidemiology* **27**, 368–377 (2016).
59. T. Haavelmo, The probability approach in econometrics. *Econometrica* **12**, iii–115 (1944).
60. N. Cartwright, Two theorems on invariance and causality. *Philos. Sci.* **70**, 203–224 (2003).
61. B. Schölkopf *et al.*, On causal and anticausal learning. arXiv:1206.6471 (27 June 2012).
62. J. Pearl, *Causality* (Cambridge University Press, 2009).
63. J. R. Platt, Strong inference. *Science* **146**, 347–353 (1964).
64. M. Hardt, B. Recht, Y. Singer, Train faster, generalize better: Stability of stochastic gradient descent. arXiv:1509.01240 (3 September 2015).
65. Y. Chen, C. Jin, B. Yu, Stability and convergence trade-off of iterative optimization algorithms. arXiv:1804.01619 (4 April 2018).
66. K. Kumbier, "Domain-inspired machine learning for hypothesis extraction in biological data," PhD thesis, University of California, Berkeley, CA (2019).
67. B. Yu, Data wisdom for data science. Operational Database Management Systems. http://www.odbms.org/2015/04/data-wisdom-for-data-science/. Deposited 13 April 2015.