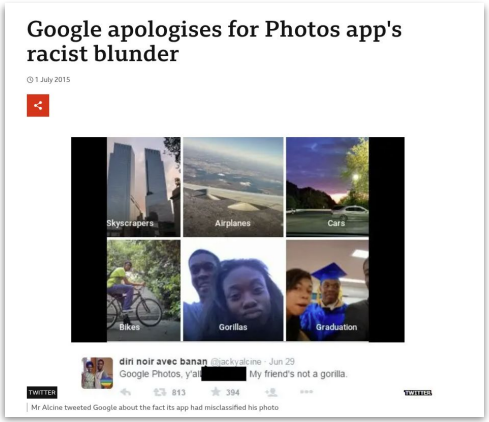
Navigating Data Errors in Machine Learning Pipelines: Identify, Debug, and Learn

Bojan Karlaš (Harvard University), Babak Salimi (UC San Diego), Sebastian Schelter (BIFOLD & TU Berlin)

navigating-data-errors.github.io

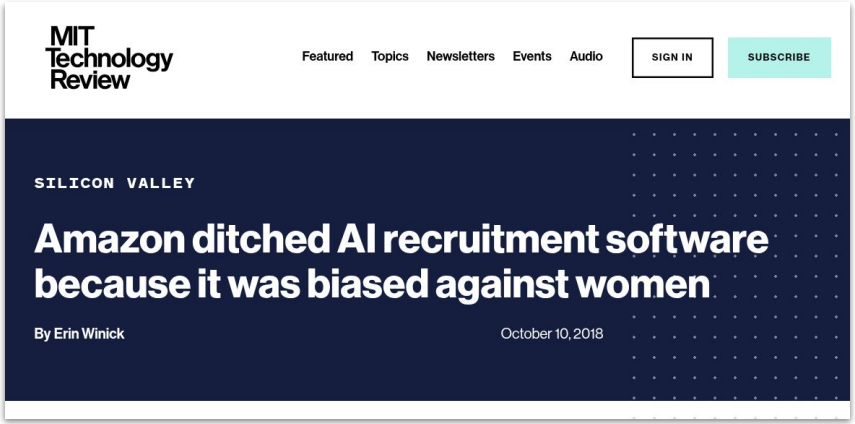
Background: ML apps often behave in unintended ways

Wrong



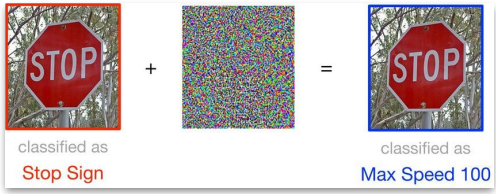
Source: BBC

Biased



Source: MIT Technology Review

Unstable

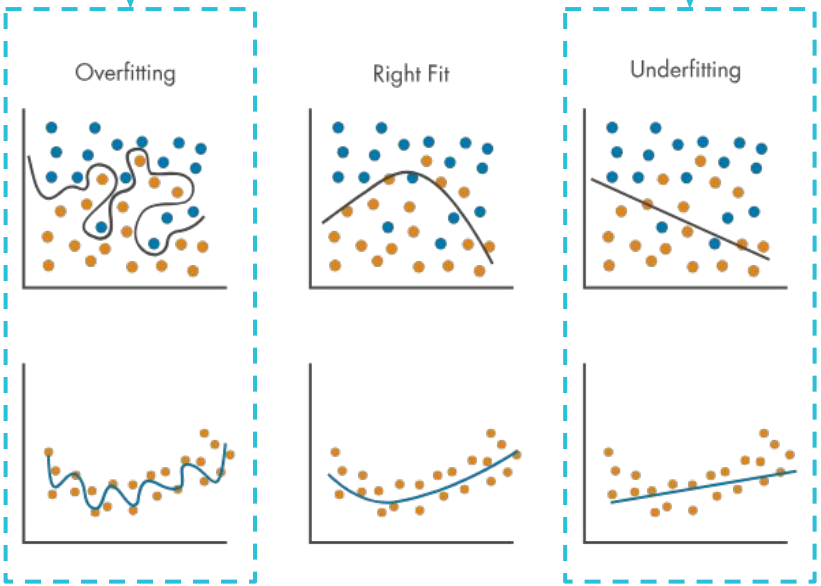


Source: Xiong et al. ACM Comput. Surv. 2023.

Source: The Guardian

Primary approach: Focus on improving the model

Example Approach:
Increase regularization by applying dropout

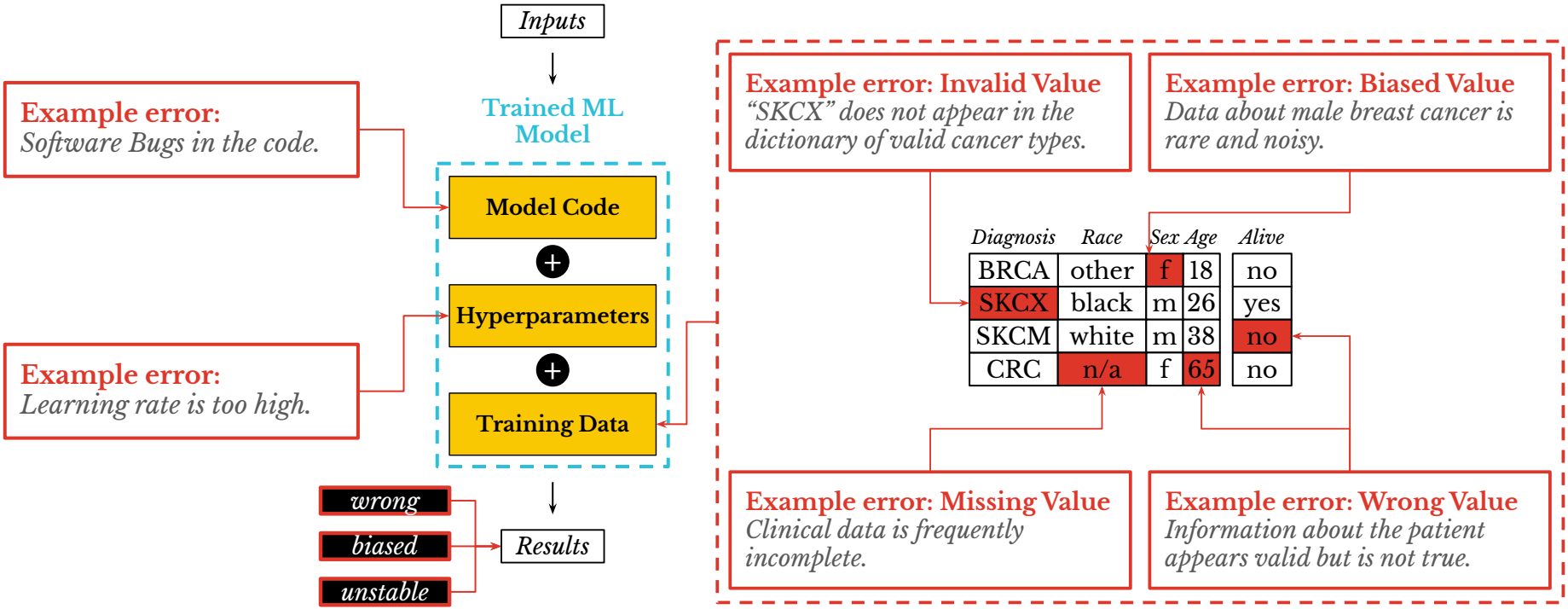


Example Approach:
Improve model capacity by increasing the number of parameters

Source: MathWorks

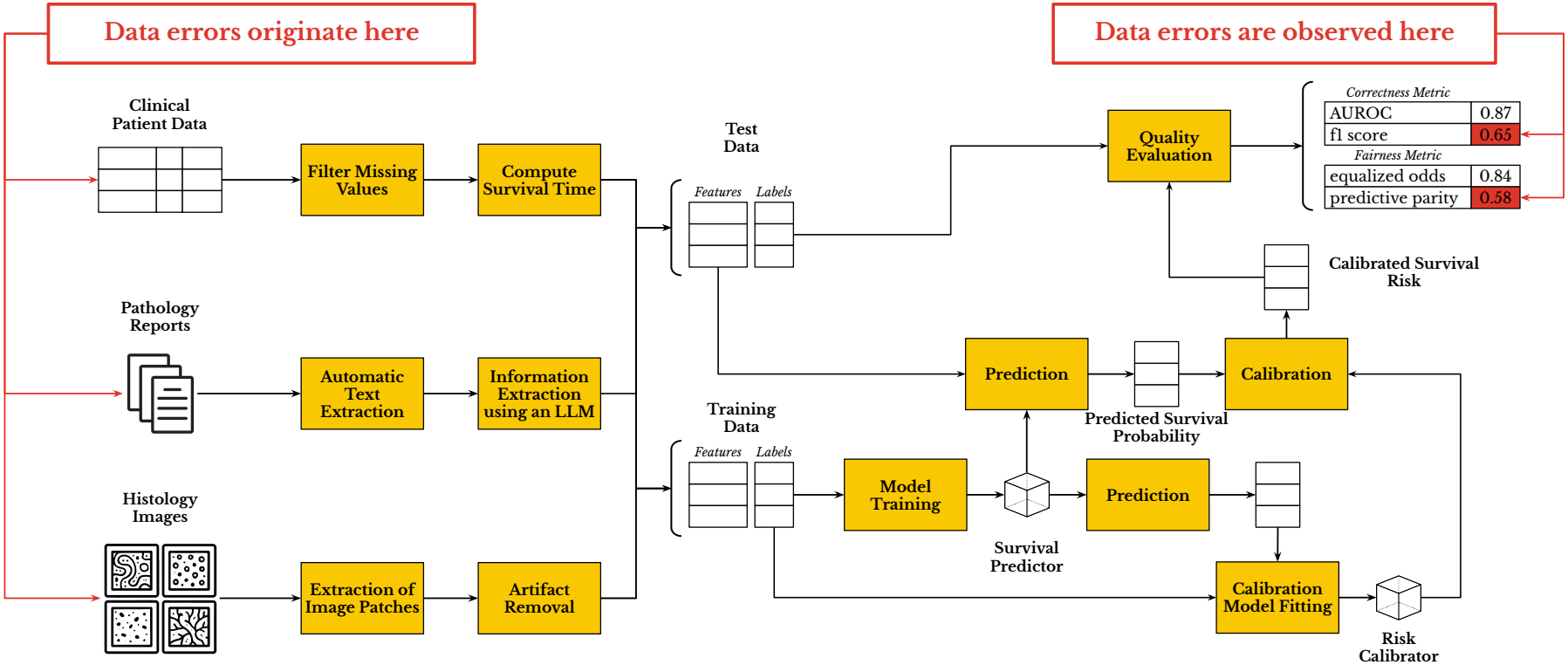
Problem: *This is only one piece of the puzzle!*

Observation 1: Data is a crucial piece of the puzzle



Challenge 1: *Can we identify the most important data errors?*

Observation 2: ML apps are built by complex pipelines



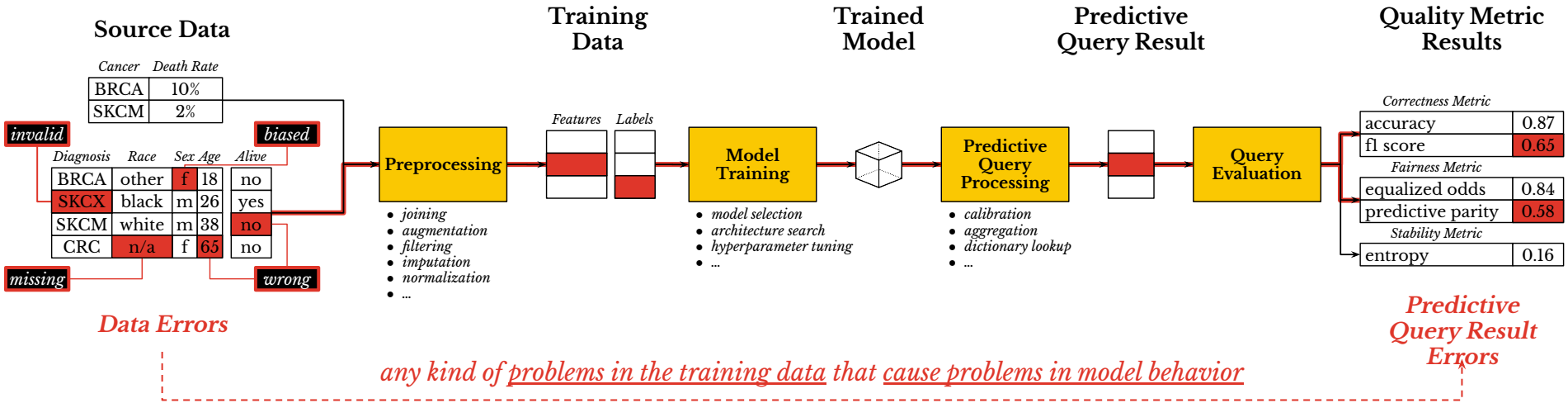
Challenge 2: *Can we trace data errors as they pass through the pipeline?*

Observation 3: Not all data errors are meant to be fixed

For each data error, we can choose to perform one of the following actions:	**Discard**  *Remove the faulty data from the training set.*	**Repair**  *Perform manual quality control which might include repeating the data acquisition process.*	**Ignore**  *Let the faulty data remain in the training set.*
Benefits:	Easy to Perform	Data Quality Improves	No Labor Required
Shortcomings:	Loss of Useful Data	Often Labor-intensive	Risk Hurting Model Quality
Optimal trade-off:	Discard or Repair the Portion of Data that will Bring the Highest Model Quality Increase		

Challenge 3: *Can we ensure reliable model performance after (partial) data repairs?*

Tutorial Overview: Data Errors in ML pipelines



Part I: Data Importance for Data Error Detection

What are good approaches for identifying data errors?

Part II: Data Debugging in ML Pipelines

What are practical challenges when debugging complex ML pipelines?

Part III: Learning from Uncertain and Incomplete Data

When we cannot repair all errors, can we still have reliable models?

Opportunities for the Data Management Community

(1) Data quality is an established discipline in data management, but most practitioners still rely on **manual effort**.

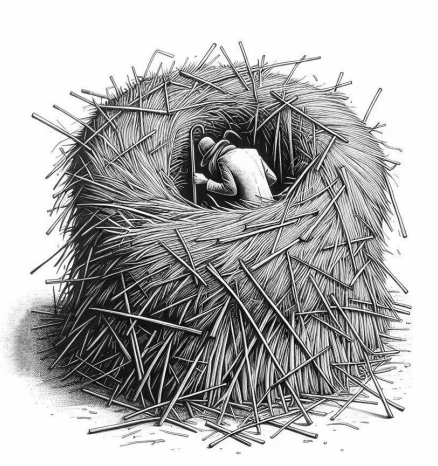
(2) ML pipelines are data processing pipelines. Models are learned data transformation operators. Many systems have been developed, but most practitioners still rely on **rudimentary scripts for crunching data**.

(3) Many promising methods for handling data errors suffer from **scalability issues**.

Main Goal: *Present the current state of the art and inspire novel research.*

Part I:
Data Importance for Data Error Detection

Bojan Karlaš



1) Introducing the Concept of Data Importance
2) Examples of Data Attribution Functions
3) Case Study of Shapley Value as a Measure of Importance
4) Applications of Data Importance

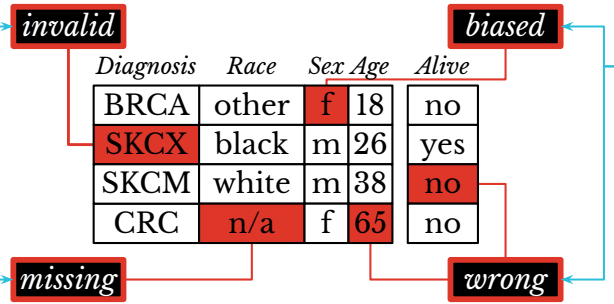
How can we identify data errors?

Trivial

Not So Trivial

Solution approach:
Apply a rule-based validation function that performs a dictionary lookup.

Solution approach:
Check if the value is marked as missing.



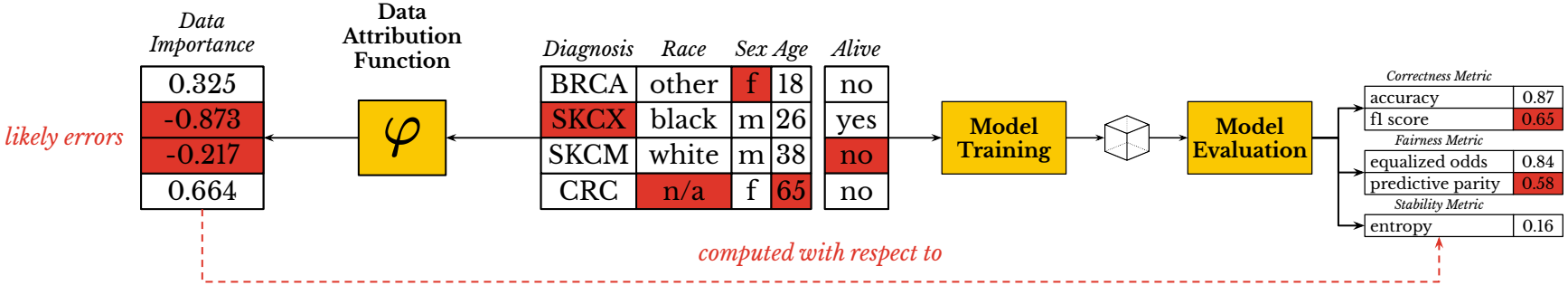
Solution approach:
Measure the impact of the value on model quality.

How do we measure this?
That is the main topic of this part of the tutorial.

Recall: *Data errors are any kind of problem in the training data that cause problems in model behavior.*

Challenge: *Can we define a unified way to think about identifying data errors?*

We can define a data attribution function



Recall: *Data errors are any kind of problem in the training data that cause problems in model behavior.*

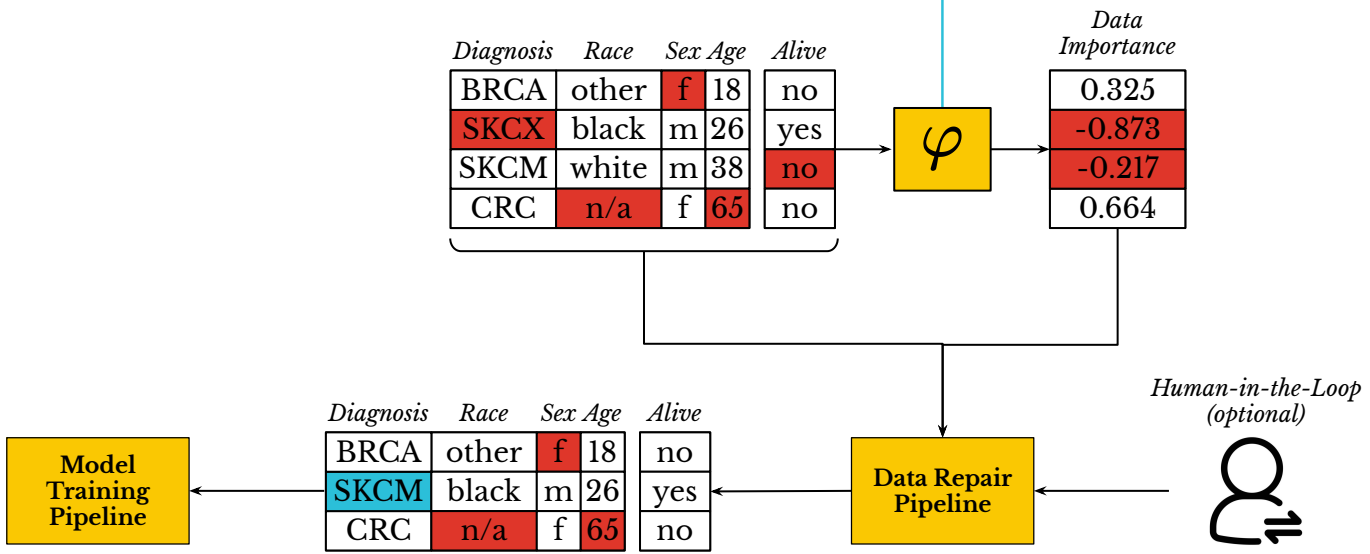
How do we use importance to detect data errors?

Attribution Function Example 1:

```
def compute_importance(value):
    return -1.0 if value == "n/a" else 1.0
```

Attribution Function Example 2:

```
VALID_CANCER_CODES = ...

def compute_importance(value):
    if value not in VALID_CANCER_CODES:
        return -1.0
    return 1.0
```



What makes a good attribution function?

Design Consideration 1

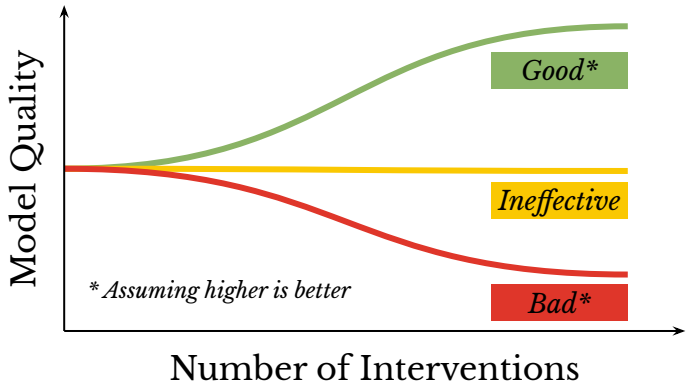
Which model quality metric do we care about improving?

Correctness Metric
- accuracy
- f1 score

Fairness Metric
- equalized odds
- predictive parity

Stability Metric
- entropy

Recall:
Data errors are any kind of problem in the training data that cause problems in model behavior.



Design Consideration 2

What kind of intervention do we intend to apply?

- *Discard*
- *Repair*
- *Something Else*

Challenge: *How do we define an effective attribution function?*

1) Introducing the Concept of Data Importance

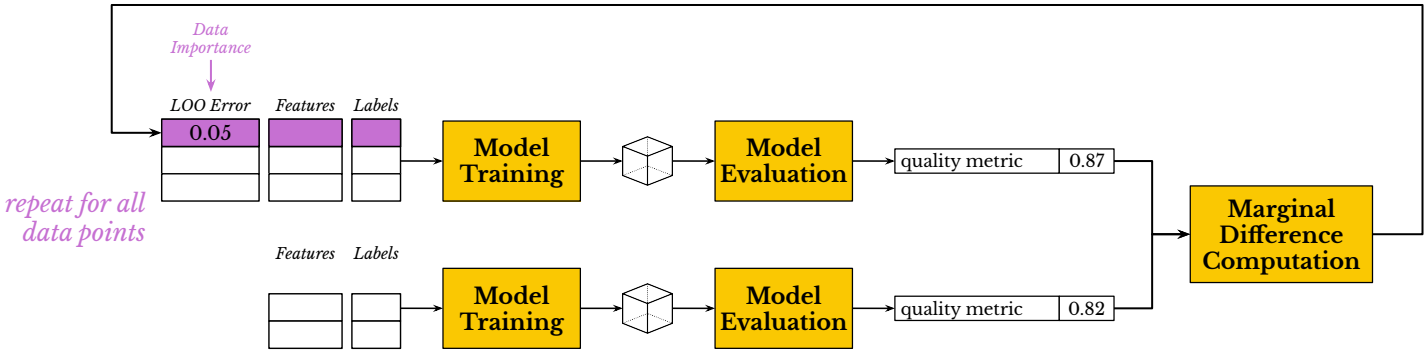
2) Examples of Data Attribution Functions

3) Case Study of Shapley Value as a Measure of Importance

4) Applications of Data Importance

Leave-one-Out Error

[Approach: **Marginal Contribution**]



Insights:

- Removing important data points affects model quality.

Approach:

- Remove a data point from the training set, train and evaluate the model again
- Interpret the difference in model quality as data importance.

Benefits:

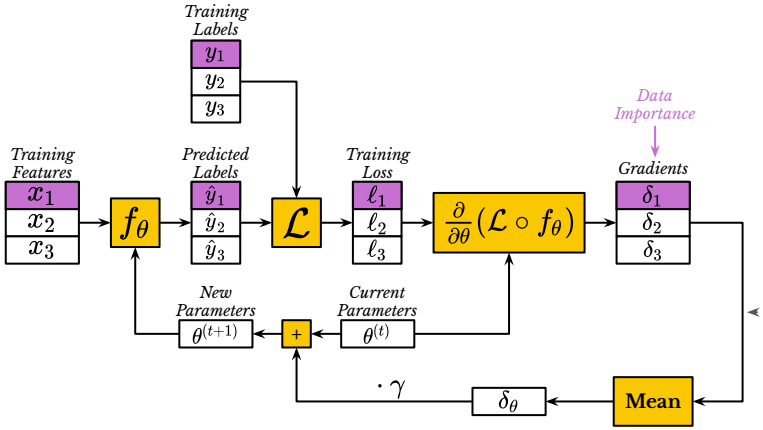
- Very simple to implement.

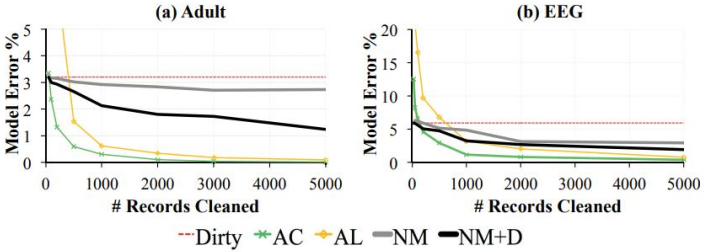
Shortcomings:

- Requires re-training the model once for each data point.
- Treats data points independently.

Error Gradient

[Approach: **Gradient**]





Insights:

- Data points vary in their contribution to the gradients that update the model.

Approach:

- Importance is proportional to the magnitude of the gradient.

Benefits:

- Simple to compute.

Shortcomings:

- Treats data points independently.

ActiveClean: Interactive Data Cleaning For Statistical Modeling

Sanjay Krishnan, Jiannan Wang [†], Eugene Wu [‡], Michael J. Franklin, Ken Goldberg
UC Berkeley, [†]Simon Fraser University, [‡]Columbia University
{sanjaykrishnan, franklin, goldberg}@berkeley.edu jiannanw@sfu.ca ewu@cs.columbia.edu

ABSTRACT

Analysts often clean dirty data iteratively–cleaning some data, exploring the analysis, and then cleaning more data based on the results. We explore the iterative cleaning process in the context of statistical model training, which is an increasingly popular form of data analytics. We propose ActiveClean, which allows for progressive and iterative cleaning in statistical modeling problems while preserving convergence guarantees. ActiveClean supports an important class of models called convex loss models (e.g., linear regression and SVMs), and prioritizes cleaning those records likely to affect the results. We evaluate ActiveClean on five real-world datasets UCI Adult, UCI EEG, MNIST, IMDB, and Dollars For Docs with both real and synthetic errors. The results show that our proposed optimizations can improve model accuracy by up-to 2.5x for the same amount of data cleaned. Furthermore for a fixed cleaning budget and on all real dirty datasets, ActiveClean returns more accurate models than uniform sampling and Active Learning.

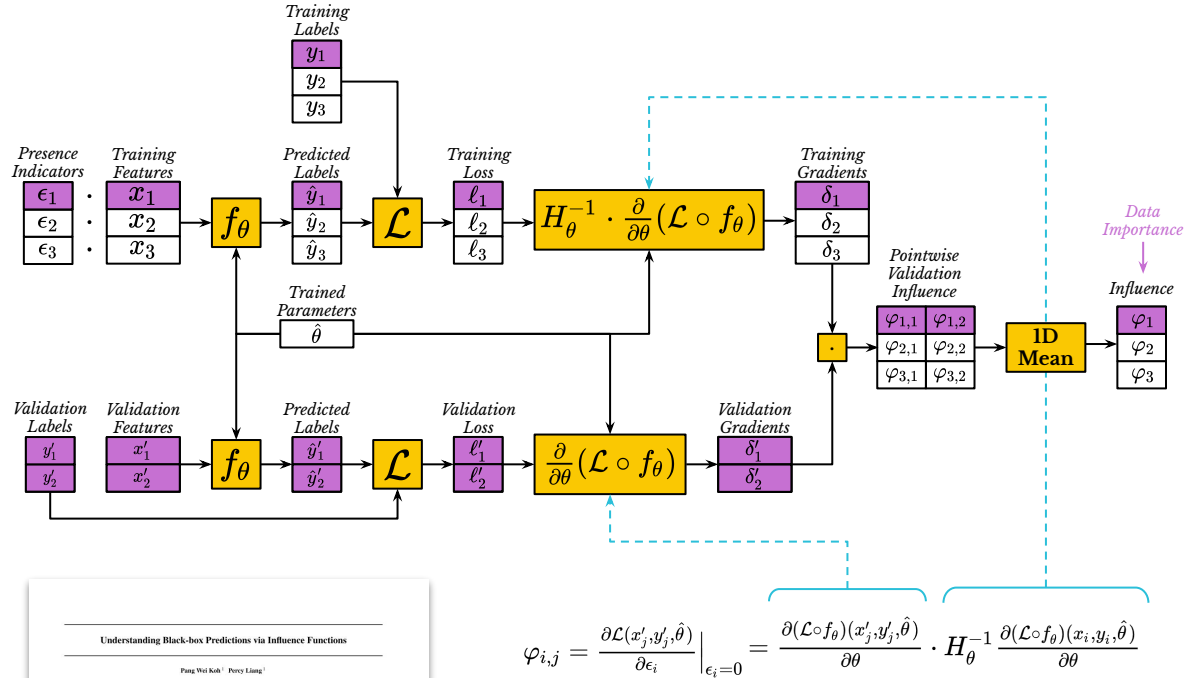
Figure 1: (a) Systematic corruption in one variable can lead to a shifted model. The dirty examples are labeled 1-5 and the cleaned examples are labeled 1'-5'. (b) Mixed dirty and clean data results in a less accurate model than no cleaning. (c) Small samples of only clean data can result in similarly loses.

pre-processing step, and instead, repeatedly alternate between cleaning and analysis. It is common to use the preliminary analysis on

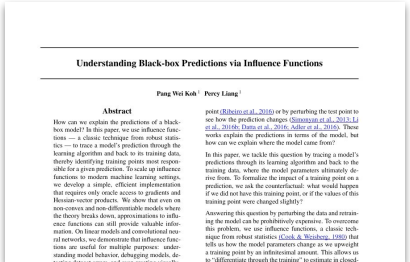
[Krishnan VLDB'16]
Krishnan, Sanjay, et al. "Activeclean: Interactive data cleaning for statistical modeling." Proceedings of the VLDB Endowment 9.12 (2016): 948-959. [Paper][Website]

Influence Function

[Approach: **Marginal Contribution, Gradient**]



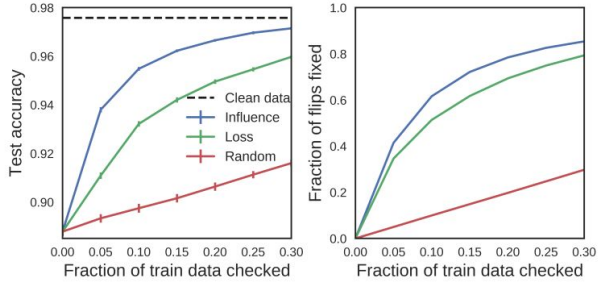
$$\varphi_{i,j} = \frac{\partial \mathcal{L}(x'_j, y'_j, \hat{\theta})}{\partial \epsilon_i}\Big|_{\epsilon_i = 0} = \frac{\partial (\mathcal{L} \circ f_\theta)(x'_j, y'_j, \hat{\theta})}{\partial \theta} \cdot H_\theta^{-1} \frac{\partial (\mathcal{L} \circ f_\theta)(x_i, y_i, \hat{\theta})}{\partial \theta}$$


Understanding Black-box Predictions via Influence Functions

Pang Wei Koh Percy Liang

Abstract

How can we explain the predictions of a black-box model? In this paper, we use influence functions — a classic technique from robust statistics — to trace a model's prediction through the learning algorithm and back to its training data, thereby identifying training points most responsible for a given prediction. To scale up influence functions to modern machine learning settings, we develop a simple, efficient implementation that requires only oracle access to gradients and Hessian-vector products. We show that even on non-convex and non-differentiable models where the theory breaks down, approximations to influence functions can still provide valuable information. On linear models and convolutional neural networks, we demonstrate that influence functions are useful for multiple purposes: understanding model behavior, debugging models, detecting dataset errors, and creating visually-indistinguishable training-set attacks.



Insights:

- The marginal contribution of a single data point can be approximated with gradients.

Approach:

- Introduce presence indicator variables ϵ for each data point and compute the gradient w.r.t. ϵ.

Benefits:

- Easily applicable to arbitrarily complex (twice) differentiable machine learning models.

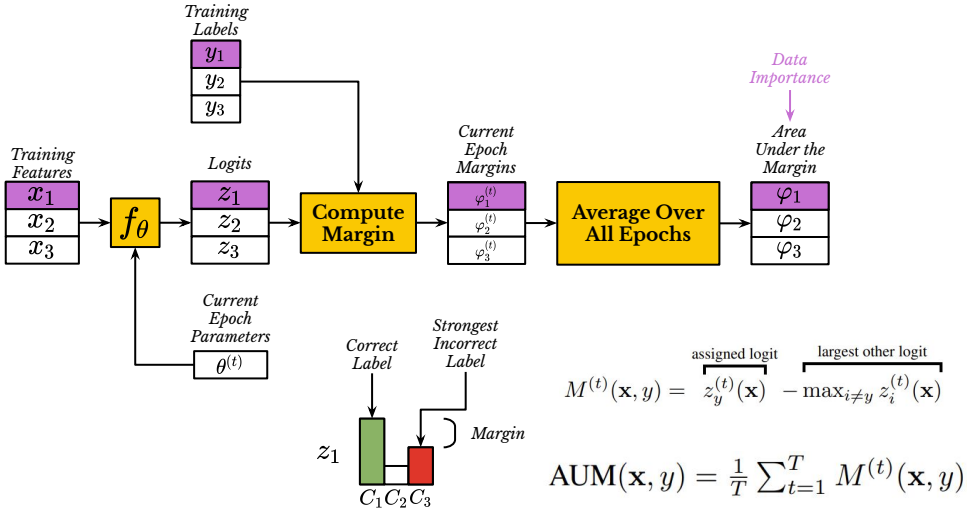
Shortcomings:

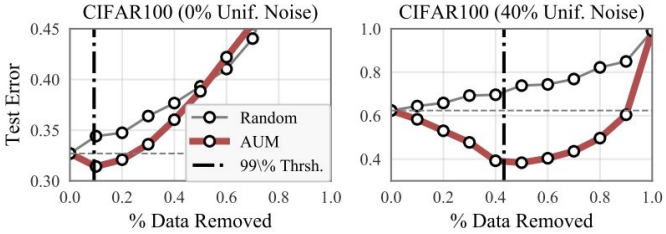
- Treats data points independently.

[Koh ICML '17]
Koh, Pang Wei, and Percy Liang. "Understanding black-box predictions via influence functions." International conference on machine learning. PMLR, 2017. [Paper][Code]

Area Under the Margin

[Approach: **Uncertainty Analysis**]





Insights:

- If similar samples have the same label, the model will learn to activate only the correct logit.
- In the presence of mislabeled samples, the model will learn to activate alternative logits.

Approach:

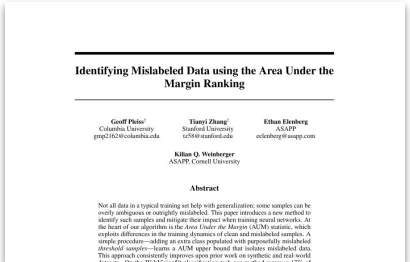
- The importance of a data point is proportional to its margin averaged across all training epochs.

Benefits:

- Very simple to implement in a wide array of models.
- Does not rely on a separate clean dataset.

Shortcomings:

- Focuses only on label noise.



Identifying Mislabeled Data using the Area Under the Margin Ranking

Geoff Pleiss, Columbia University, gmp2162@columbia.edu
Tianyi Zhang, Stanford University, tz58@stanford.edu
Ethan Elenberg, ASAPP, eelenberg@asapp.com
Kilian Q. Weinberger, ASAPP, Cornell University

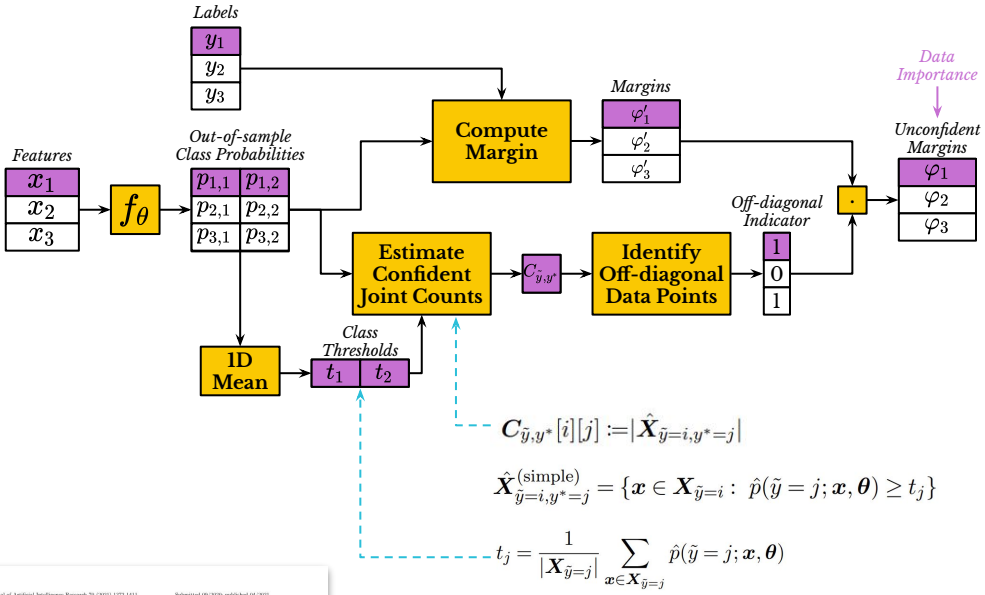
Abstract

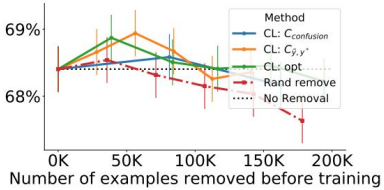
Not all data in a typical training set help with generalization; some samples can be overly ambiguous or outrightly mislabeled. This paper introduces a new method to identify such samples and mitigate their impact when training neural networks. At the heart of our algorithm is the *Area Under the Margin* (AUM) statistic, which exploits differences in the training dynamics of clean and mislabeled samples. A simple procedure—adding an extra class populated with purposefully mislabeled *threshold samples*—learns a AUM upper bound that isolates mislabeled data. This approach consistently improves upon prior work on synthetic and real-world

[Pleiss NeurIPS '20]
Pleiss, Geoff, et al. "Identifying mislabeled data using the area under the margin ranking." Advances in Neural Information Processing Systems 33 (2020): 17044-17056. [Paper][Blog][Code]

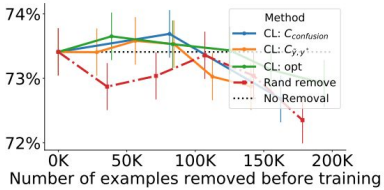
Unconfident Margins

[Approach: **Uncertainty Analysis**]





(a) ResNet18 Validation Accuracy



(b) ResNet50 Validation Accuracy

Insights:

- Given a data point, if a model assigns a higher than average probability to some specific class, it is likely because most similar data points have the same class label. This is likely to be the true label of that data point.

Approach:

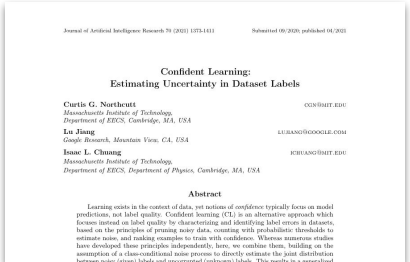
- Identify likely mislabeled data points and assign negative importance using the margin. Remaining data points get zero importance.

Benefits:

- Very simple to implement in a wide array of models.
- Does not rely on a separate clean dataset.

Shortcomings:

- Focuses only on label noise.
- Relies on having an adequately powerful model.



Confident Learning:
Estimating Uncertainty in Dataset Labels

Curtis G. Northcutt CGN@MIT.EDU
Massachusetts Institute of Technology, Department of EECS, Cambridge, MA, USA
Lu Jiang LUJIANG@GOOGLE.COM
Google Research, Mountain View, CA, USA
Isaac L. Chuang ICHUANG@MIT.EDU
Massachusetts Institute of Technology, Department of EECS, Department of Physics, Cambridge, MA, USA

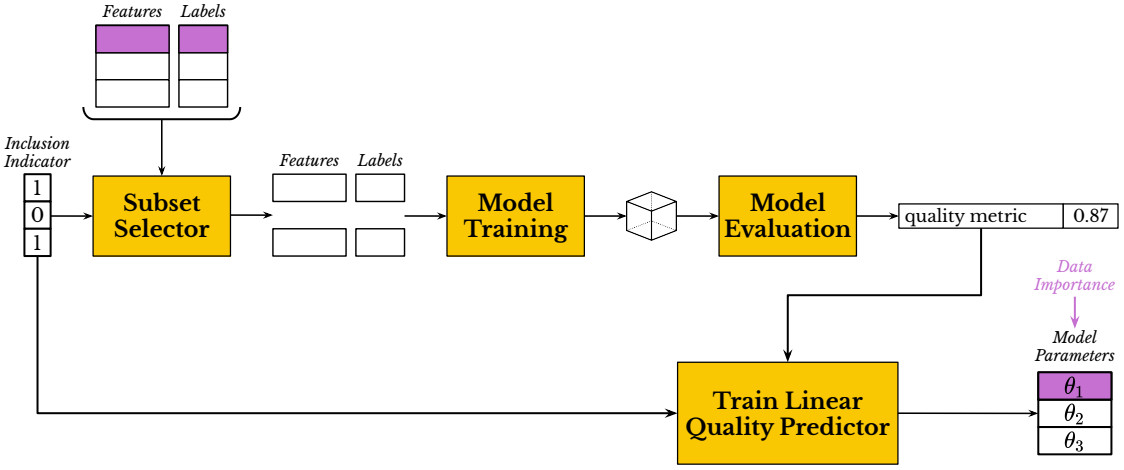
Abstract

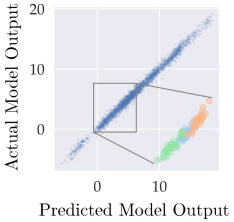
Learning exists in the context of data, yet notions of *confidence* typically focus on model predictions, not label quality. Confident learning (CL) is an alternative approach which focuses instead on label quality by characterizing and identifying label errors in datasets, based on the principles of pruning noisy data, counting with probabilistic thresholds to estimate noise, and ranking examples to train with confidence. Whereas numerous studies have developed these principles independently, here, we combine them, building on the assumption of a class-conditional noise process to directly estimate the joint distribution

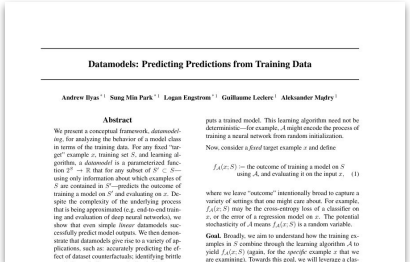
[Northcutt JAIR '21]
Northcutt, Curtis, Lu Jiang, and Isaac Chuang. "Confident learning: Estimating uncertainty in dataset labels." Journal of Artificial Intelligence Research 70 (2021): 1373-1411. [Paper][Blog][Code]

Model Training Outcome

[Approach: **Surrogate Data Model**]







Datamodels: Predicting Predictions from Training Data

Andrew Ilyas [1] Sung Min Park [1] Logan Engstrom [1] Guillaume Leclerc [1] Aleksander Mądry [1]

Abstract

We present a conceptual framework, *datamodeling*, for analyzing the behavior of a model class in terms of the training data. For any fixed "target" example x, training set S, and learning algorithm, a *datamodel* is a parameterized function $2^S \to \mathbb{R}$ that for any subset of $S' \subset S$—using only information about which examples of S are contained in S'—predicts the outcome of training a model on S' and evaluating on x. Despite the complexity of the underlying process that is being approximated (e.g. end-to-end training and evaluation of deep neural networks), we show that even simple *linear* datamodels successfully predict model outputs. We then demonstrate that datamodels give rise to a variety of applications, such as: accurately predicting the effect of dataset counterfactuals; identifying brittle

puts a trained model. This learning algorithm need not be deterministic—for example, A might encode the process of training a neural network from random initialization.

Now, consider a *fixed* target example x and define

$$f_A(x; S) := \text{the outcome of training a model on } S \text{ using } A, \text{ and evaluating it on the input } x, \quad (1)$$

where we leave "outcome" intentionally broad to capture a variety of settings that one might care about. For example, $f_A(x; S)$ may be the cross-entropy loss of a classifier on x, or the error of a regression model on x. The potential stochasticity of A means $f_A(x; S)$ is a random variable.

Goal. Broadly, we aim to understand how the training examples in S combine through the learning algorithm A to yield $f_A(x; S)$ (again, for the *specific* example x that we are examining). Towards this goal, we will leverage a clas-

[Ilyas ICML '22]
Ilyas, Andrew, et al. "Datamodels: Predicting Predictions from Training Data." Proceedings of the 39th International Conference on Machine Learning. 2022. [Paper][Blog][Code]

Insights:

- A linear model can be good at predicting the quality of a model trained on an arbitrary subset of the training data and tested on a single test example.

Approach:

- Train a linear quality predictor and interpret its parameters as data importance.

Benefits:

- Conceptually simple yet powerful framework for analyzing datasets.

Shortcomings:

- The original method requires retraining the model many times.

1) Introducing the Concept of Data Importance
2) Examples of Data Attribution Functions
3) **Case Study of Shapley Value as a Measure of Importance**
4) **Applications of Data Importance**

Improving Upon the Marginal Contribution Methods

Recall

Marginal contribution methods treat data points independently, ignoring any interactions that might exist.

Consequence

Let there be a data point that has high importance. If we make two copies of that data point, their individual marginal contribution to the dataset as a whole will be zero.

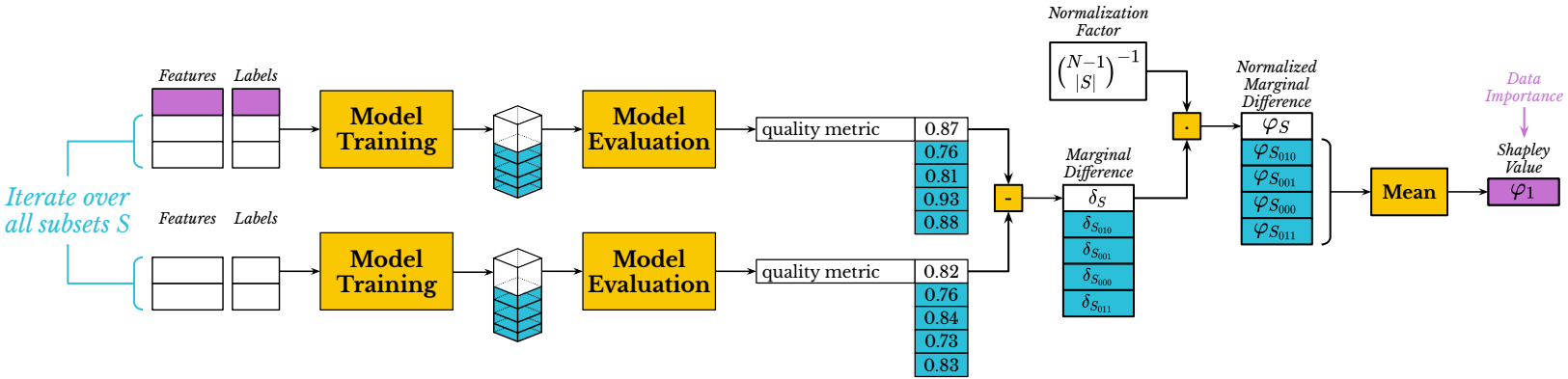
Approach

We should measure marginal contribution over all subsets.

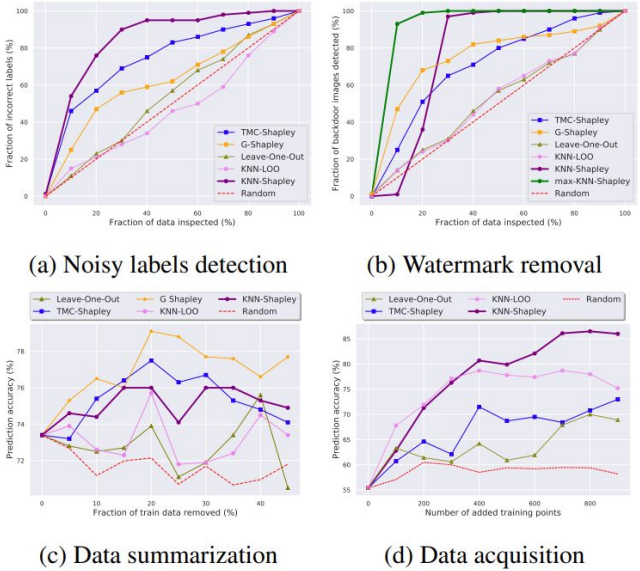
Shapley value

A standard method from game theory for distributing surplus among a coalition of players.

$$\varphi_i = \frac{1}{N} \sum_{S \subseteq X \setminus \{i\}} \binom{N-1}{|S|}^{-1} \left(u(S \cup \{i\}) - u(S) \right)$$



Effectiveness at Data Debugging

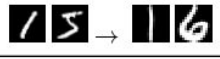
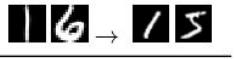


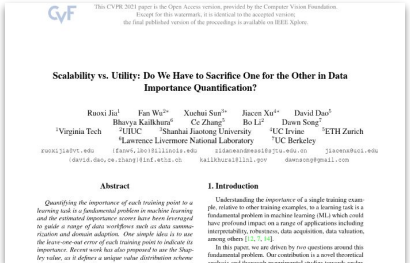
(a) Noisy labels detection (b) Watermark removal

(c) Data summarization (d) Data acquisition

Figure 2: The experiment result of (a) noisy label detection on fashion-MNIST dataset; (b) instance-based watermark removal on MNIST dataset; (c) data summarization on UCI Adult Census dataset [15]; (d) data acquisition on MNIST dataset with injected noise. In (a)-(b) the "random" line shows the results of random guess; while in (c)-(d), the "random" line corresponds to the empirical results of the random baseline introduced in Section 4.1.

Table 2: Domain adaptation between MNIST and USPS.

Method	MNIST → USPS 	USPS → MNIST 
KNN-Shapley	**31.70%** → **47.00%**	**23.35%** → **29.80%**
KNN-LOO	31.70% → 37.40%	23.35% → 24.50%
TMC-Shapley	31.70% → 44.90%	23.35% → 29.55%
LOO	31.70% → 29.40%	23.35% → 23.53%



Scalability vs. Utility: Do We Have to Sacrifice One for the Other in Data Importance Quantification?

Ruoxi Jia[1] Fan Wu[2*] Xuehui Sun[3*] Jiacen Xu[4*] David Dao[5]
Bhavya Kailkhura[6] Ce Zhang[7] Bo Li[2] Dawn Song[2]
[1]Virginia Tech [2]UIUC [3]Shanhai Jiaotong University [4]UC Irvine [5]ETH Zurich
[6]Lawrence Livermore National Laboratory [7]UC Berkeley

Abstract

1. Introduction

[Jia CVPR '21]
Jia, Ruoxi, et al. "Scalability vs. utility: Do we have to sacrifice one for the other in data importance quantification?." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2021. [Paper] [Code]

Benefits and Challenges

Beneficial Properties of the Shapley Value

Symmetry

If two data points have the same contribution to every subset, their value should be the same.

Efficiency

The sum of importances of all data points should equal the marginal contribution of the entire set over an empty set.

Linearity

If the utility function can be expressed as a sum of two other functions, then the importance of a data point using the combined function should equal the sum of importances computed using the individual functions.

Null Player

If a data point has a zero marginal contribution to every single subset, its importance should be zero.

Key Challenge

The number of subsets to enumerate is exponential, making it intractable to compute the exact Shapley value for an arbitrary model.

$$\varphi_i = \frac{1}{N} \sum_{S \subseteq X \setminus \{i\}} \binom{N-1}{|S|}^{-1} \left(u(S \cup \{i\}) - u(S) \right)$$

Approximation: Monte Carlo Sampling

Challenge

Computing Shapley values is intractable.

Insight

Since Shapley value can be seen as a statistic over exponentially many subsets, we can estimate it using Monte Carlo sampling.

Approach

Use the permutation-based definition of the Shapley value and sample permutations.

$$\varphi_i(v) = \frac{1}{n!} \sum_{R} \left[v(P_i^R \cup \{i\}) - v(P_i^R) \right]$$

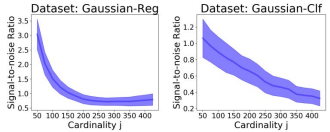
$$\phi_i = \mathbb{E}_{\pi \sim \Pi}[V(S_\pi^i \cup \{i\}) - V(S_\pi^i)]$$

Challenge

We need many Monte Carlo samples to produce good estimates.

Insight

When estimating the marginal contribution of a data point to a subset, we empirically observe that larger subsets incur a slower signal-to-noise ratio.

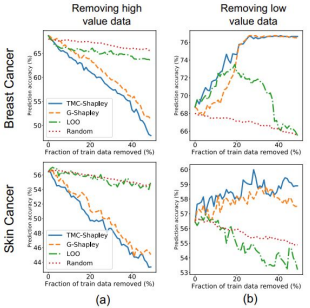


Approach

Leverage the importance sampling strategy and apply a larger weight to smaller subsets, based on the beta distribution.

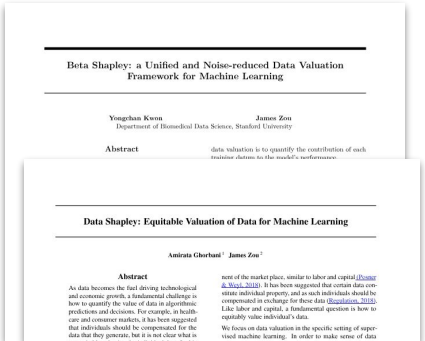
Benefits

Estimating the Shapley value becomes tractable and is shown to be effective at identifying important data points.



Shortcomings

Each Monte Carlo sample relies on retraining the model from scratch, which is expensive for large models.



Beta Shapley: a Unified and Noise-reduced Data Valuation Framework for Machine Learning

Yongchan Kwon James Zou
Department of Biomedical Data Science, Stanford University

Abstract

Data Shapley: Equitable Valuation of Data for Machine Learning

Amirata Ghorbani James Zou

Abstract

[Kwon AISTATS '22]
Kwon, Yongchan, and James Zou. "Beta Shapley: a Unified and Noise-reduced Data Valuation Framework for Machine Learning." International Conference on AI and Statistics. 2022. [Paper] [Code]

[Ghorbani ICML '19]
Ghorbani, Amirata, and James Zou. "Data shapley: Equitable valuation of data for machine learning." International conference on machine learning. PMLR, 2019. [Paper] [Code]

Approximation: K-Nearest Neighbor Surrogate Model

Challenge

To get good Shapley value estimates, we need to retrain the model many times.

Insight

The simple KNN classifier can make it easy to design efficient and exact algorithms.

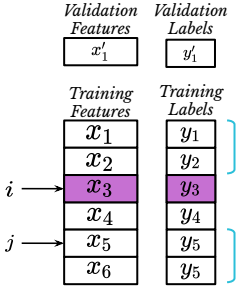
Approach

Use the KNN model as a proxy to develop an exact Shapley computation algorithm with polynomial time complexity.

Example Situation

- *We are computing the Shapley value of data point i*
- *Data is sorted by similarity to the validation data point*

Observation 1:
Since K=1, for any subset S, the top-1 data point will determine the model prediction.



Observation 3:
*If the subset **S** contains these data points, the data point **i** will not be the top-1.*

Observation 4:
*If data point **j** is going to become the top-1 after **i** is removed, all data points above it cannot be included in **S**, while the ones below may or may not be included in **S**.*

Starting point: Shapley value definition

$$\varphi_i = \frac{1}{N} \sum_{S \subseteq X \setminus \{i\}} \binom{N-1}{|S|}^{-1} \underbrace{\left(u(S \cup \{i\}) - u(S)\right)}$$

Observation 2:
If data point i is not in the top-1, this term will be zero.

Dynamic Programming

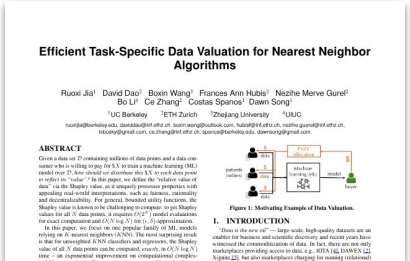
$$\varphi_i(t) = \frac{1}{N} \sum_{j=i+1}^{N} \sum_{a=1}^{n-j} \binom{N-1}{a}^{-1} \left(u(\{i\}) - u(\{j\})\right) \binom{N-j}{a}$$

Final Simplification

$$\varphi_i(t) = \frac{1}{N} \sum_{j=i+1}^{N} \left(u(\{i\}) - u(\{j\})\right) \binom{N-j}{j+1}$$

Result:
After sorting the data, we can compute exact Shapley values in a single pass. Final computational complexity is

$$\mathcal{O}(N \log N)$$


Efficient Task-Specific Data Valuation for Nearest Neighbor Algorithms

Ruoxi Jia[1] David Dao[2] Boxin Wang[1] Frances Ann Hubis[1] Nezihe Merve Gurel[2] Bo Li[3] Ce Zhang[2] Costas Spanos[1] Dawn Song[4]

[1]UC Berkeley [2]ETH Zurich [3]Zhejiang University [4]UIUC

ABSTRACT

1. INTRODUCTION

[Jia VLDB '19]
Jia, Ruoxi, et al. "Efficient task-specific data valuation for nearest neighbor algorithms." Proceedings of the VLDB Endowment 12.11 (2019): 1610-1623. [Paper] [Code]

Approximation: Taylor Expansion

Challenge

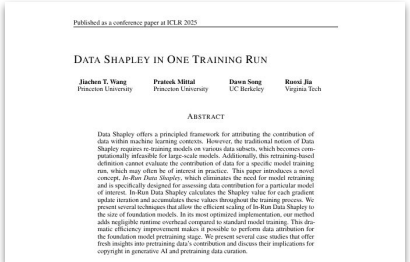
If we are using a large and complex model, retraining will be extremely slow (preventing Monte Carlo approaches), and the KNN approximation will be biased.

Insight

Models trained with stochastic gradient descent (SGD) compute the loss function many times, over many random subsets of the training dataset. Furthermore, the changes in the model quality metric that are small enough to be effectively approximated with Taylor expansion.

Approach

Redefine the utility function to measure the cumulative impact of a training data point on the validation loss across gradient update steps.



Published as a conference paper at ICLR 2025

DATA SHAPLEY IN ONE TRAINING RUN

Jiachen T. Wang, Princeton University
Prateek Mittal, Princeton University
Dawn Song, UC Berkeley
Ruoxi Jia, Virginia Tech

ABSTRACT

Data Shapley offers a principled framework for attributing the contribution of data within machine learning contexts. However, the traditional notion of Data Shapley requires re-training models on various data subsets, which becomes computationally infeasible for large-scale models. Additionally, this retraining-based definition cannot evaluate the contribution of data for a specific model training run, which may often be of interest in practice. This paper introduces a novel concept, *In-Run Data Shapley*, which eliminates the need for model retraining and is specifically designed for assessing data contribution for a particular model of interest. In-Run Data Shapley calculates the Shapley value for each gradient update iteration and accumulates these values throughout the training process. We present several techniques that allow the efficient scaling of In-Run Data Shapley to the size of foundation models. In its most optimized implementation, our method adds negligible runtime overhead compared to standard model training. This dramatic efficiency improvement makes it possible to perform data attribution for the foundation model pretraining stage. We present several case studies that offer fresh insights into pretraining data's contribution and discuss their implications for copyright in generative AI and pretraining data curation.

[Wang ICLR '25]
Wang, Jiachen T., et al. "Data Shapley in One Training Run." The Thirteenth International Conference on Learning Representations. [Paper] [Blog]

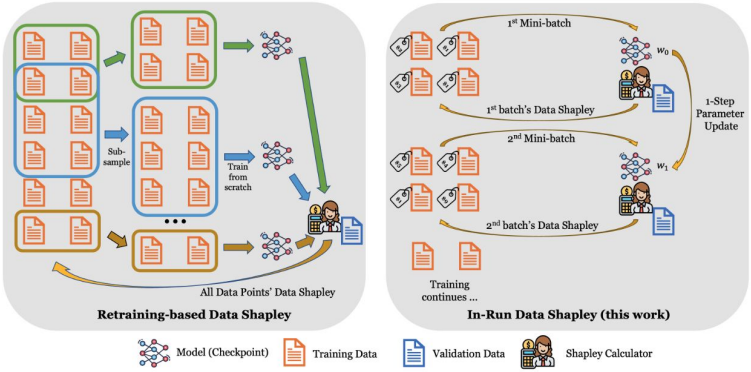
Redefined "local utility function" of subset S of a single SGD minibatch:

$$U^{(t)}(S; z^{(\text{val})}) := \underbrace{\ell(\widetilde{w}_{t+1}(S), z^{(\text{val})})}_{\text{Model updated only using data from } S} - \underbrace{\ell(w_t, z^{(\text{val})})}_{\text{Model at SGD step } t}$$

$$\widetilde{w}_{t+1}(S) := w_t - \eta_t \sum_{z \in S} \nabla \ell(w_t, z)$$

Redefined "global utility function" of subset S over the entire SGD run:

$$U(S) = \sum_{t=0}^{T-1} U^{(t)}(S)$$



1) Introducing the Concept of Data Importance
2) Examples of Data Attribution Functions
3) Case Study of Shapley Value as a Measure of Importance
4) **Applications of Data Importance**

Influence Function for Explaining Fairness Errors

Challenge

Data attribution gives us an ordered list of data points that impact model quality, but it does not explain what makes these data points impactful.

Insight

If we group important data points based on common predicates, we can derive more powerful conclusions about factors that cause models to underperform.

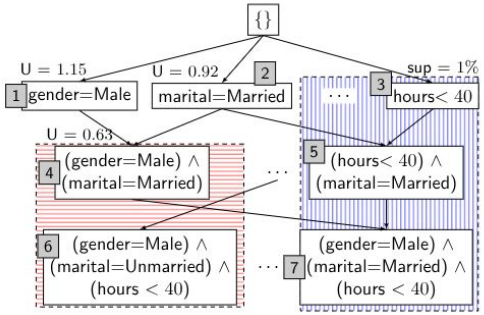
Approach

First, use influence functions to compute data importance with respect to fairness metrics. Second, use lattice-based search to identify combinations of predicates that define data subsets that are both small and impactful.

Data points ordered by importance

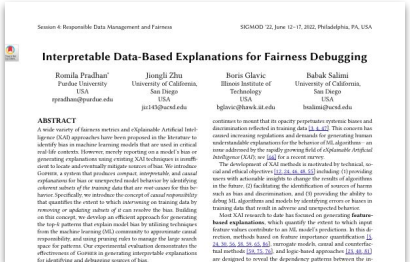
age	education	marital	...	gender	income
39	Bachelors	Never-married	...	Male	≤50K
53	11th	Never-married	...	Male	≤50K
28	Bachelors	Married-civ-spouse	...	Female	≤50K
37	Masters	Married-civ-spouse	...	Female	≤50K

Lattice-based search identifies predicates that select the most impactful training data subsets



Combinations of predicates that explain model behavior

1. Gender = Female ∧ Relationship = Not married ∧ Education = Associate-voc
2. Gender = Male ∧ Relationship = Spouse ∧ Hours < 40 ∧ Workclass = Federal-gov
3. Gender = Male ∧ Education = Prof-school



Session 4: Responsible Data Management and Fairness SIGMOD '22, June 12–17, 2022, Philadelphia, PA, USA

Interpretable Data-Based Explanations for Fairness Debugging

Romila Pradhan*, Purdue University, USA, rpradhan@purdue.edu
Jiongli Zhu, University of California, San Diego, USA, jiz143@ucsd.edu
Boris Glavic, Illinois Institute of Technology, USA, bglavic@iit.edu
Babak Salimi, University of California, San Diego, USA, bsalimi@ucsd.edu

ABSTRACT

A wide variety of fairness metrics and eXplainable Artificial Intelligence (XAI) approaches have been proposed in the literature to identify bias in machine-learning models that are used in critical real-life contexts. However, merely reporting on a model's bias or generating explanations using existing XAI techniques is insufficient to locate and eventually mitigate sources of bias. We introduce Gopher, a system that produces compact, interpretable, and causal explanations for bias or unexpected model behavior by identifying coherent subsets of the training data that are root-causes for this behavior. Specifically, we introduce the concept of causal responsibility that quantifies the extent to which intervening on training data by removing or updating subsets of it can resolve the bias. Building on this concept, we develop an efficient approach for generating the top-k patterns that explain model bias by utilizing techniques from the machine learning (ML) community to approximate causal responsibility, and using pruning rules to manage the large search space for patterns. Our experimental evaluation demonstrates the effectiveness of Gopher in generating interpretable explanations for identifying and debugging sources of bias.

continues to mount that its opacity perpetuates systemic biases and discrimination reflected in training data [2, 4, 47]. This concern has caused increasing regulations and demands for generating human understandable explanations for the behavior of ML algorithms– an issue addressed by the rapidly growing field of eXplainable Artificial Intelligence (XAI) see [26] for a recent survey.

The development of XAI methods is motivated by technical, social and ethical objectives [12, 18, 38, 48, 52] including: (1) providing users with actionable insights to change the results of algorithms in the future, (2) facilitating the identification of sources of harms such as bias and discrimination, and (3) providing the ability to debug ML algorithms and models by identifying errors or biases in training data that result in adverse and unexpected behavior.

Most XAI research to date has focused on generating **feature-based explanations**, which quantify the extent to which input feature values contribute to an ML model's predictions. In this direction, methods based on feature importance quantification [5, 24, 30, 36, 38, 39, 45, 81], surrogate models, causal and counterfactual methods [54, 75, 76], and logic-based approaches [13, 46, 61] are designed to reveal the dependency patterns between the us-

[Zhu SIGMOD '22]
Pradhan, Romila, et al. "Interpretable data-based explanations for fairness debugging." Proceedings of the 2022 international conference on management of data. 2022. [Paper]

Debugging the LLM Retrieval Corpus

Challenge

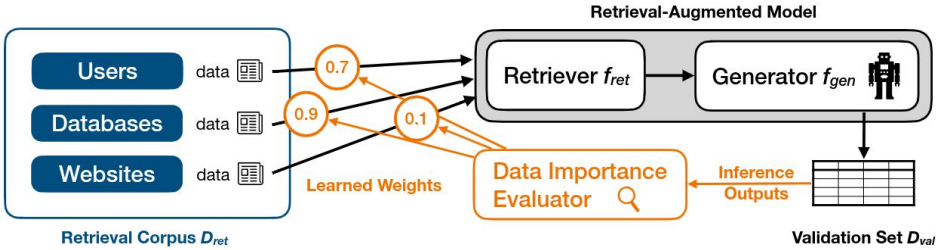
Retrieval augmented generation (RAG) is a widely used technique for providing pre-trained large language models (LLMs) with task-specific context. Data errors in the retrieval corpus have a negative impact on model quality.

Insight

The role of a retrieval corpus to an LLM is similar to the role of a training dataset to a classical ML model.

Approach

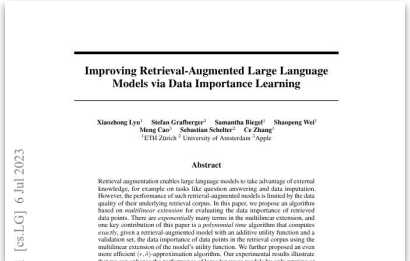
Define a data attribution function that will compute the importance of data points in the retrieval corpus. Use this to identify and debug data errors.



$$U(f_{gen}, f_{ret}, \mathcal{D}_{val}, \mathcal{D}_{ret}) := \sum_{x_i \subseteq \mathcal{D}_{val}} U\left(f_{gen}(x_i, f_{ret}(x_i, \mathcal{D}_{ret}))\right)$$

$$\tilde{U}(w_1, \cdots, w_M) := \sum_{\mathcal{S} \subseteq \mathcal{D}_{ret}} U(\mathcal{S}) \underbrace{\prod_{d_i \in \mathcal{S}} w_i \prod_{d_i \notin \mathcal{S}} (1 - w_i)}_{P[\mathcal{S}]}$$

Dataset	GPT-JT (6B)	GPT-JT (6B) w/ retrival				GPT-3.5 (175B)
		Vanilla	+LOO	+Reweight	+Prune	
Buy	0.102	0.789	0.808	**0.815**	0.813	0.764
Restaurant	0.030	0.746	0.756	0.760	**0.761**	0.463


Improving Retrieval-Augmented Large Language Models via Data Importance Learning

Xiaozhong Lyu[1] Stefan Grafberger[2] Samantha Biegel[1] Shaopeng Wei[1] Meng Cao[3] Sebastian Schelter[2] Ce Zhang[1]
[1]ETH Zürich [2] University of Amsterdam [3]Apple

Abstract

Retrieval augmentation enables large language models to take advantage of external knowledge, for example on tasks like question answering and data imputation. However, the performance of such retrieval-augmented models is limited by the data quality of their underlying retrieval corpus. In this paper, we propose an algorithm based on *multilinear extension* for evaluating the data importance of retrieved data points. There are *exponentially* many terms in the multilinear extension, and one key contribution of this paper is a *polynomial time* algorithm that computes *exactly*, given a retrieval-augmented model with an additive utility function and a validation set, the data importance of data points in the retrieval corpus using the multilinear extension of the model's utility function. We further proposed an even more efficient (ϵ, δ)-approximation algorithm. Our experimental results illustrate

[cs.LG] 6 Jul 2023

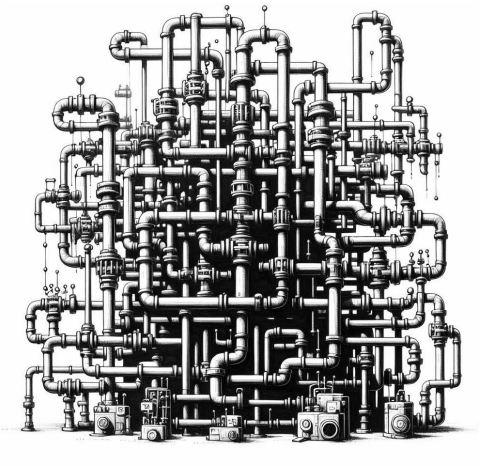
[Lyu arXiv '23]
Lyu, Xiaozhong, et al. "Improving retrieval-augmented large language models via data importance learning." arXiv preprint arXiv:2307.03027 (2023). [Paper] [Code]

Key Takeaways of Part I

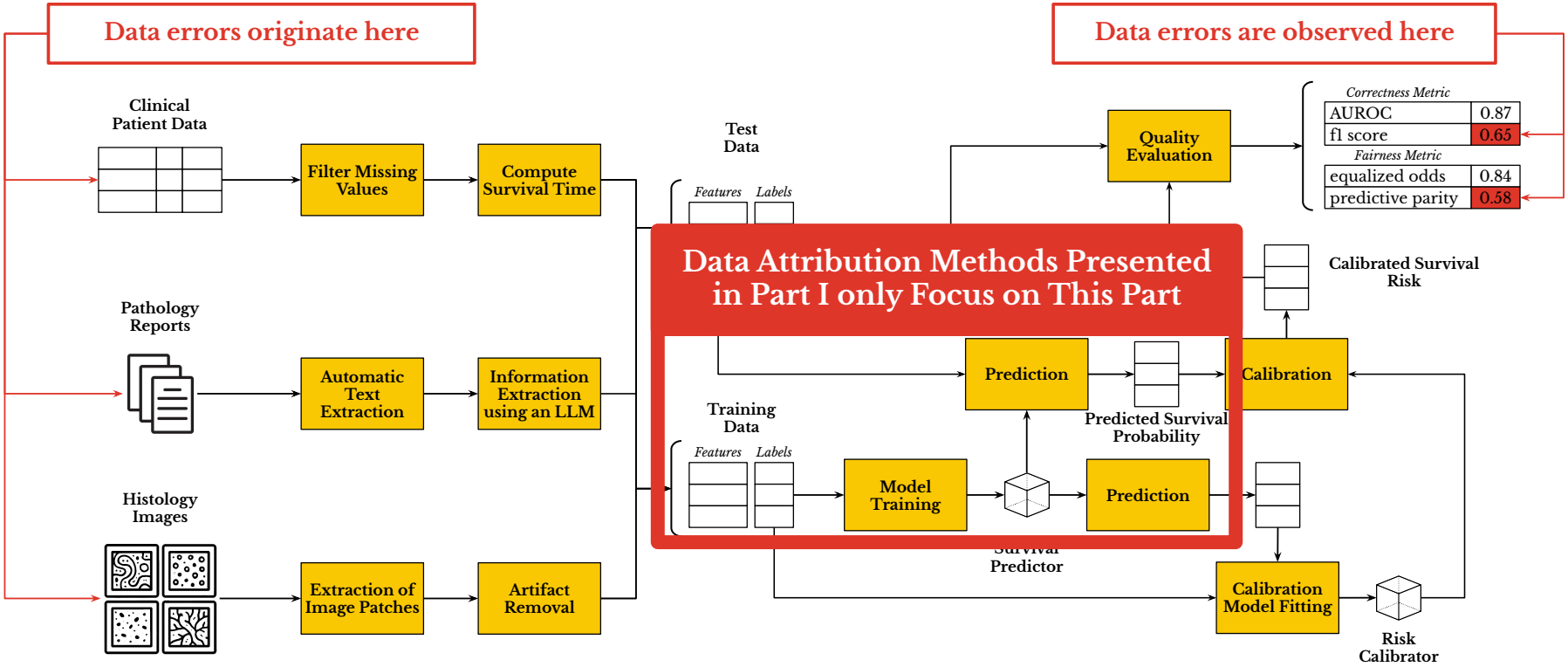
- Data attribution is a useful powerful framework for approaching the problem of data error detection.
- There are many existing data attribution methods with various strengths and shortcomings.
- The most powerful methods face scalability issues that have been tackled by existing research with many opportunities for future improvements.

Part II:
Data Debugging in ML Pipelines

Sebastian Schelter



Gap between Attribution Methods and ML Pipelines



Challenge: *How should we debug ML pipelines?*

1) Gap between Attribution Methods and ML Pipelines

2) **Libraries and Systems for ML Pipelines**

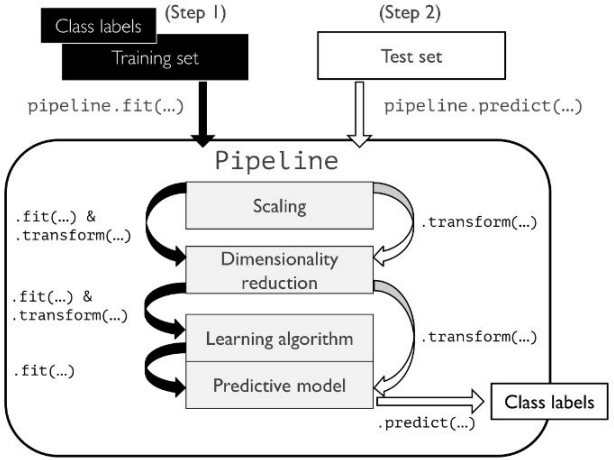
3) **Characteristics of Real World ML Pipelines**

4) **Methods for Debugging ML Pipelines**

Scikit-Learn

Highlights

- *Among the most popular data science Python libraries*
- *Has implementations of many machine learning models, as well as data processing operators*
- *Introduced the* ***estimator/transformer abstraction*** *for composing complex, nested pipelines*
    - ***Transformer****: tuple-at-a time transformation*
    - ***Estimator****: create a data-specific transformer via a global aggregation over the data*



Source: https://vitalflux.com/sklearn-machine-learning-pipeline-python-example/

Scikit-learn: Machine Learning in Python

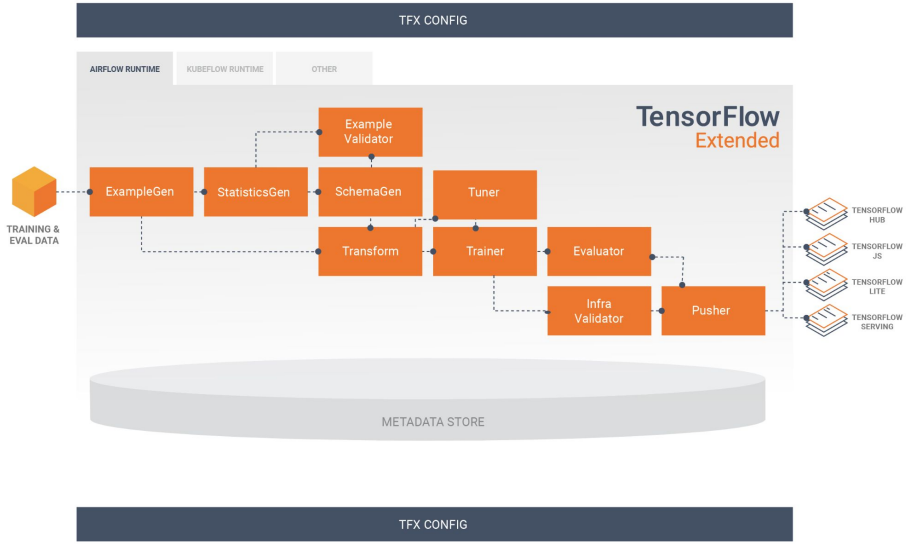
[Pedregosa JMLR '11]
Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." the Journal of machine Learning research 12 (2011): 2825-2830. [Paper] [Website] [Code]

Tensorflow Extended (TFX)

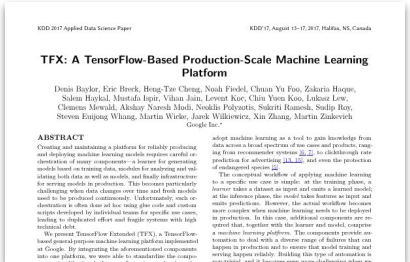
TensorFlow Extended

Highlights

- *End-to-end platform for production ML pipelines*
- *Built on TensorFlow and optimized for scalability, strong* ***emphasis on model validation and monitoring***
- *Includes reusable components for pipelines, inspired by estimator/transformer paradigm*
- ***Apache Beam for dataflow operations, Tensorflow for numerical operations***



Source: https://www.tensorflow.org/tfx/guide



KDD 2017 Applied Data Science Paper — KDD'17, August 13–17, 2017, Halifax, NS, Canada

TFX: A TensorFlow-Based Production-Scale Machine Learning Platform

Denis Baylor, Eric Breck, Heng-Tze Cheng, Noah Fiedel, Chuan Yu Foo, Zakaria Haque, Salem Haykal, Mustafa Ispir, Vihan Jain, Levent Koc, Chiu Yuen Koo, Lukasz Lew, Clemens Mewald, Akshay Naresh Modi, Neoklis Polyzotis, Sukriti Ramesh, Sudip Roy, Steven Euijong Whang, Martin Wicke, Jarek Wilkiewicz, Xin Zhang, Martin Zinkevich
Google Inc.*

ABSTRACT

Creating and maintaining a platform for reliably producing and deploying machine learning models requires careful orchestration of many components – a learner for generating models based on training data, modules for analyzing and validating both data as well as models, and finally infrastructure for serving models in production. This becomes particularly challenging when data changes over time and fresh models need to be produced continuously. Unfortunately, such orchestration is often done ad hoc using glue code and custom scripts developed by individual teams for specific use cases, leading to duplicated effort and fragile systems with high technical debt.

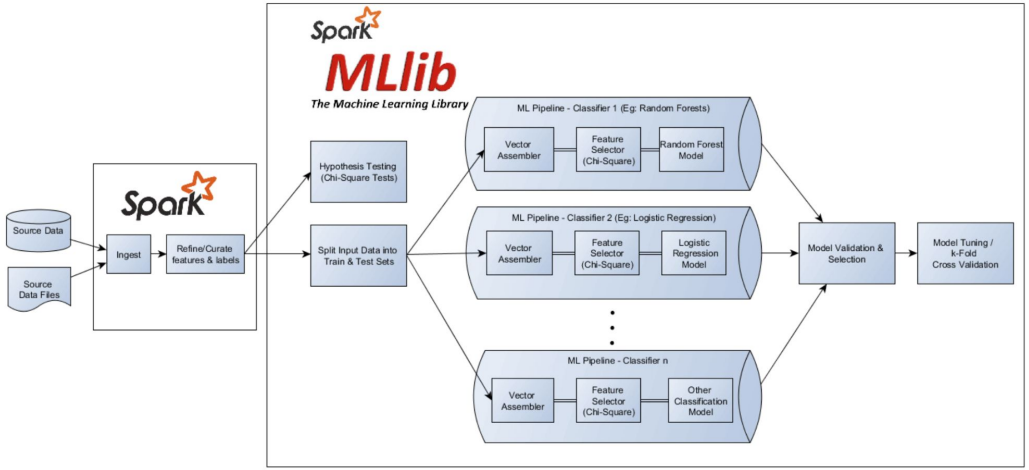
We present TensorFlow Extended (TFX), a TensorFlow-based general-purpose machine learning platform implemented at Google. By integrating the aforementioned components into one platform, we were able to standardize the components, simplify the orchestration, and significantly reduce the overhead of keeping individual systems in sync. In this paper, we describe the context in which TFX was developed, the architecture of TFX, and the libraries and components that comprise it. We also share our lessons learned developing and operating TFX. We hope that describing our approach will help other machine learning practitioners streamline their workflows and reduce the fragility and technical debt of their platforms.

[Baylor SIGKDD '17]
Baylor, Denis, et al. "Tfx: A tensorflow-based production-scale machine learning platform." Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining. 2017. [Paper] [Website] [Code]

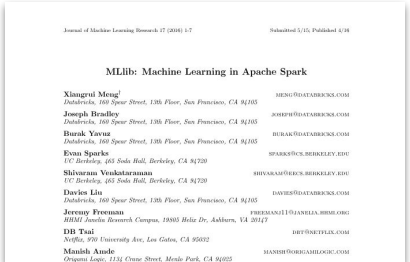
Spark MLlib

Highlights

- *Built on top of Apache Spark*
- *Includes implementations for classification, regression, clustering, collaborative filtering, and dimensionality reduction*
- ***Works natively with Spark DataFrames, SQL, and streaming data***
- *Adoption of estimator/transformer paradigm from scikit-learn*



Source: https://www.qubole.com/developers/spark-getting-started-guide/workflow



MLlib: Machine Learning in Apache Spark

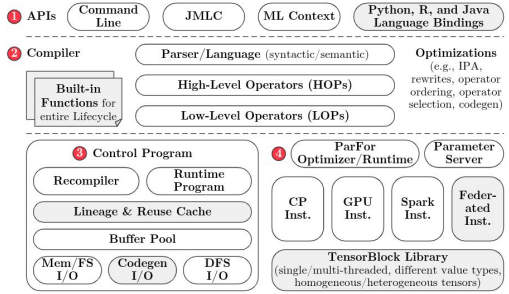
[Meng JMLR '16]
Meng, Xiangrui, et al. "Mllib: Machine learning in apache spark." Journal of Machine Learning Research 17.34 (2016): 1-7. [Paper] [Website] [Code]

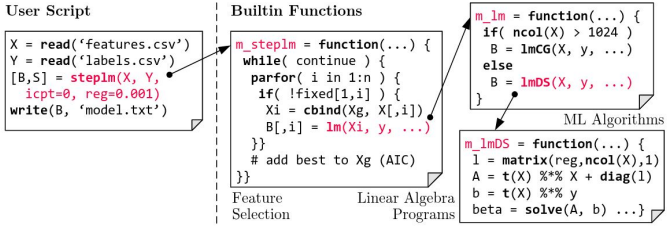
Apache SystemDS

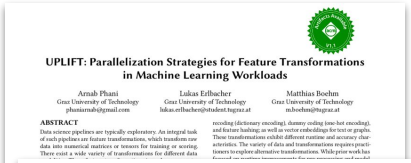
Apache SystemDS ™

Highlights

- *Designed for scalable and efficient execution on both single-node and distributed environments*
- *Offers a high-level scripting language for expressing ML algorithms and workflows with a declarative R-like language*
- *Performs* ***cost-based optimization*** *and* ***automatic operator selection*** *for efficient execution across different hardware endpoints*
- *Optimised feature encoders based on estimator/transformer paradigm*

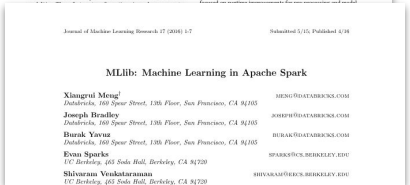






UPLIFT: Parallelization Strategies for Feature Transformations in Machine Learning Workloads

Arnab Phani, Lukas Erlbacher, Matthias Boehm
Graz University of Technology



MLlib: Machine Learning in Apache Spark

Xiangrui Meng, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman
Databricks, 160 Spear Street, 13th Floor, San Francisco, CA 94105

[Boehm CIDR '20]
Boehm, Matthias, et al. "SystemDS: A Declarative Machine Learning System for the End-to-End Data Science Lifecycle." 10th Conference on Innovative Data Systems Research. 2020. [Paper] [Website] [Code]

[Phani VLDB '20]
Phani, Arnab, et al. "UPLIFT: parallelization strategies for feature transformations in machine learning workloads." Proceedings of the VLDB Endowment, Volume 15, Issue 11, 2020. [Paper]

ML Pipelines in the Cloud

Netflix Metaflow

[Website] [Documentation]

Highlights

- *Notebook based development environment*
- *Storing and tracking of code, data and models*
- *Scaling from local execution to the cloud*

Amazon SageMaker Pipelines

[Website] [Documentation]

Highlights

- *Define, automate, and manage end-to-end ML workflows*
- *Automatically tracks pipeline artifacts*
- *Leverages AWS Cloud infrastructure*

Azure Machine Learning Pipelines

[Website] [Documentation]

Highlights

- *Orchestration of ML workflows with reusable, modular pipeline components*
- *Versioning, monitoring, and CI/CD integration*

Vertex AI Pipelines

[Website] [Documentation]

Highlights

- *Connects with Vertex AI services*
- *Tracks pipeline steps, metadata, and artifacts*
- *Orchestrates ML workflows on Google Cloud*

Observations

- No universal way to express ML pipelines, design often prioritises flexibility and ease-of-use
- Many pipelines combine relational / dataflow operators with ML-specific operators based on estimator/transformer abstraction
- Pipelines often executed via multiple runtimes
- Lack of algebraic operator semantics
- Lack of fine-grained data provenance

1) Gap between Attribution Methods and ML Pipelines
2) Libraries and Systems for ML Pipelines
3) **Characteristics of Real World ML Pipelines**
4) **Methods for Debugging ML Pipelines**

Study of Pipelines at Google

Highlights

- *Study of **3000 production pipelines with over 450K models** trained over a 4 month period*
- *About half the pipelines studied used data- and model-validation operators*
- *Input data typically has up to 100 features, but can have over 10K in extreme cases*
- *53% of features were categorical, often with very large domains (averaging over 10M unique values)*
- ***Training accounts for only 20% of the total runtime cost, over 30% is for model validation and 20% for data ingestion***
- *About 1/4 of model training runs results in model deployment*
- ***Deep learning models account for 60% of pipelines***

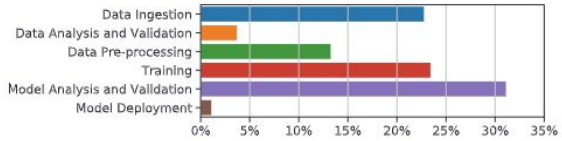
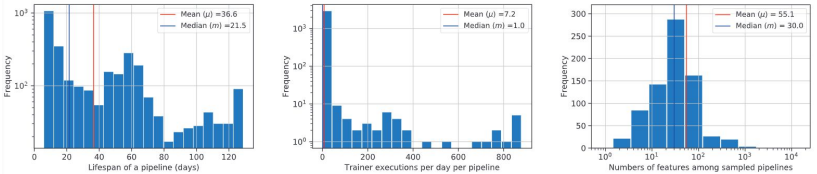


Figure 7: Compute cost of different operators.



(a) Distribution of pipeline span. **(b) Distribution of trained models per day.** **(c) Distribution of the number of features.**

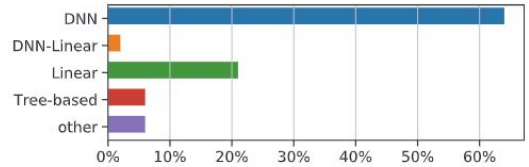
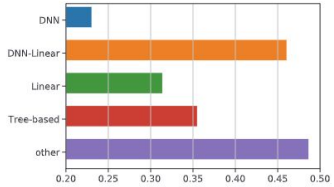
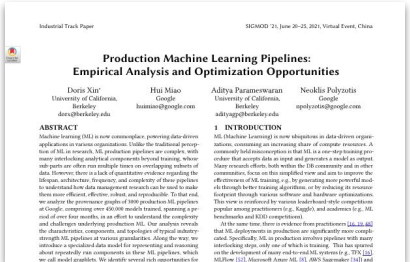


Figure 5: Percentage of Trainer runs with each model type



(f) Model type vs. likelihood of pushes.



Industrial Track Paper SIGMOD '21, June 20–25, 2021, Virtual Event, China

Production Machine Learning Pipelines: Empirical Analysis and Optimization Opportunities

Doris Xin' University of California, Berkeley dorx@berkeley.edu
Hui Miao Google huimiao@google.com
Aditya Parameswaran University of California, Berkeley adityagp@berkeley.edu
Neoklis Polyzotis Google npolyzotis@google.com

ABSTRACT

Machine learning (ML) is now commonplace, powering data-driven applications in various organizations. Unlike the traditional perception of ML in research, ML production pipelines are complex, with many interlocking analytical components beyond training, whose sub-parts are often run multiple times on overlapping subsets of data. However, there is a lack of quantitative evidence regarding the lifespan, architecture, frequency, and complexity of these pipelines to understand how data management research can be used to make them more efficient, effective, robust, and reproducible. To that end, we analyze the provenance graphs of 3000 production ML pipelines at Google, comprising over 450,000 models trained, spanning a period of over four months, in an effort to understand the complexity and challenges underlying production ML. Our analysis reveals the characteristics, components, and topologies of typical industry-strength ML pipelines at various granularities. Along the way, we introduce a specialized data model for representing and reasoning about repeatedly run components in these ML pipelines, which we call model graphlets. We identify several rich opportunities for

1 INTRODUCTION

ML (Machine Learning) is now ubiquitous in data-driven organizations, consuming an increasing share of computer resources. A commonly held misconception is that ML is a one-step training procedure that accepts data as input and generates a model as output. Many research efforts, both within the DB community and in other communities, focus on this simplified view and aim to improve the effectiveness of ML training, e.g., by generating more powerful models through better training algorithms, or by reducing its resource-footprint through various software and hardware optimizations. This view is reinforced by various leaderboard-style competitions popular among practitioners (e.g., Kaggle), and academics (e.g., ML benchmarks and KDD competitions).

At the same time, there is evidence from practitioners [16, 39, 46] that ML deployments in production are significantly more complicated. Specifically, ML in production involves pipelines with many interlocking steps, only one of which is training. This has spurred on the development of many end-to-end ML systems (e.g., TFX [10], MLFlow [52], Microsoft Azure ML [8], AWS Sagemaker [34] and

[Xin SIGMOD '21]
Xin, Doris, et al. "Production machine learning pipelines: Empirical analysis and optimization opportunities." Proceedings of the 2021 international conference on management of data. 2021. [Paper]

Study of Pipelines at Microsoft

Highlights

- *Study of over **8M public Jupyter notebooks on GitHub** (from 2017, 2019, and 2020), and **2M enterprise pipelines developed with ML.NET***
- ***Python is emerging as the de-facto standard language for data science** (81% of notebooks in 2017 and 91% in 2020)*
- *Around 80% cells were linear (no conditional statements) and 76% were completely linear (no conditionals, classes, or functions)*
- ***Libraries like numpy, matplotlib, pandas, and scikit-learn are used very frequently** (e.g., numpy in >60% of notebooks)*
- *Few highly used libraries have significant coverage (e.g., top-10 cover ~40% of notebooks, top-100 cover ~75%), **but there is a long tail***
- *Explicit ML pipelines (defined with sklearn.pipeline) are gaining traction but there are still **5 times more implicit pipelines** in GitHub notebooks*
- *There is a **large number of distinct operators**, and **a significant portion are user-defined** (especially in ML.NET and implicit GitHub pipelines)*

Dimension	Metric	GH17	GH19	GH20
Notebooks	Total	1.23M	4.6M	8.7M
	Deduped	66.0%	65.5%	65.7%
	Linear	26.4%	29.1%	30.3%
	Completely Linear	21.2%	23.3%	24.6%
Languages	Python	81.7%	91.7%	91.1%
	Other	18.3%	8.3%	8.9%
Cells	Total	34.6M	143.1M	261.2M
Code Cells	Total	64.5%	66.4%	66.9%
	Deduped	41.0%	38.6%	38.5%
	Linear	72.1%	80.2%	79.3%
	Completely Linear	68.3%	76.1%	75.6%
Users	Total	100K	400K	697K

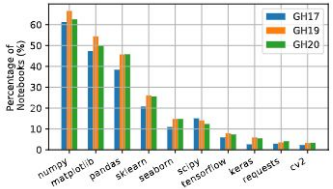


Figure 2: Top-10 used libraries.

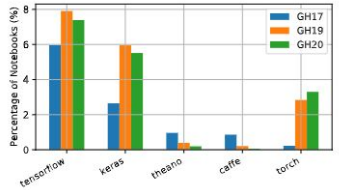
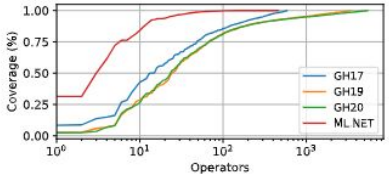
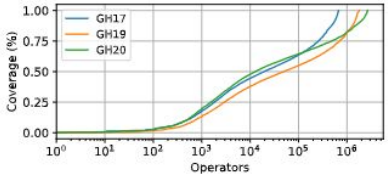
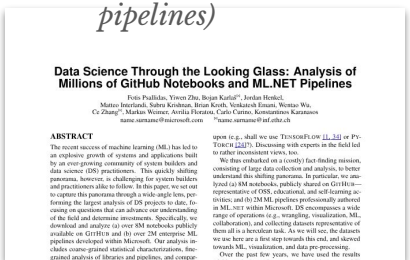


Figure 3: DL libraries usage percentages.

		GH17	GH19	GH20	ML.NET
#Pipelines	Implicit	164K	415K	1.4M	N/A
	Explicit	10K	129K	252K	29.7M
#Distinct Ops	Implicit	668K	1.8M	2.6M	N/A
	Explicit	584	3.4K	5.5K	23.5K







Data Science Through the Looking Glass: Analysis of Millions of GitHub Notebooks and ML.NET Pipelines

Fotis Psallidas, Yiwen Zhu, Bojan Karlaš[*], Jordan Henkel, Matteo Interlandi, Subru Krishnan, Brian Kroth, Venkatesh Emani, Wentao Wu, Ce Zhang[*], Markus Weimer, Avrilia Floratou, Carlo Curino, Konstantinos Karanasos

ABSTRACT

The recent success of machine learning (ML) has led to an explosive growth of systems and applications built by an ever-growing community of system builders and data science (DS) practitioners. This quickly shifting panorama, however, is challenging for system builders and practitioners alike to follow. In this paper, we set out to capture this panorama through a wide-angle lens, performing the largest analysis of DS projects to date, focusing on questions that can advance our understanding of the field and determine investments. Specifically, we download and analyze (a) over 8M notebooks publicly available on GitHub and (b) over 2M enterprise ML pipelines developed within Microsoft. Our analysis includes coarse-grained statistical characterizations, fine-grained analysis of libraries and pipelines, and conjur-

upon (e.g., shall we use TensorFlow [1, 34] or PyTorch [187]). Discussing with experts in the field led to rather inconsistent views, too.

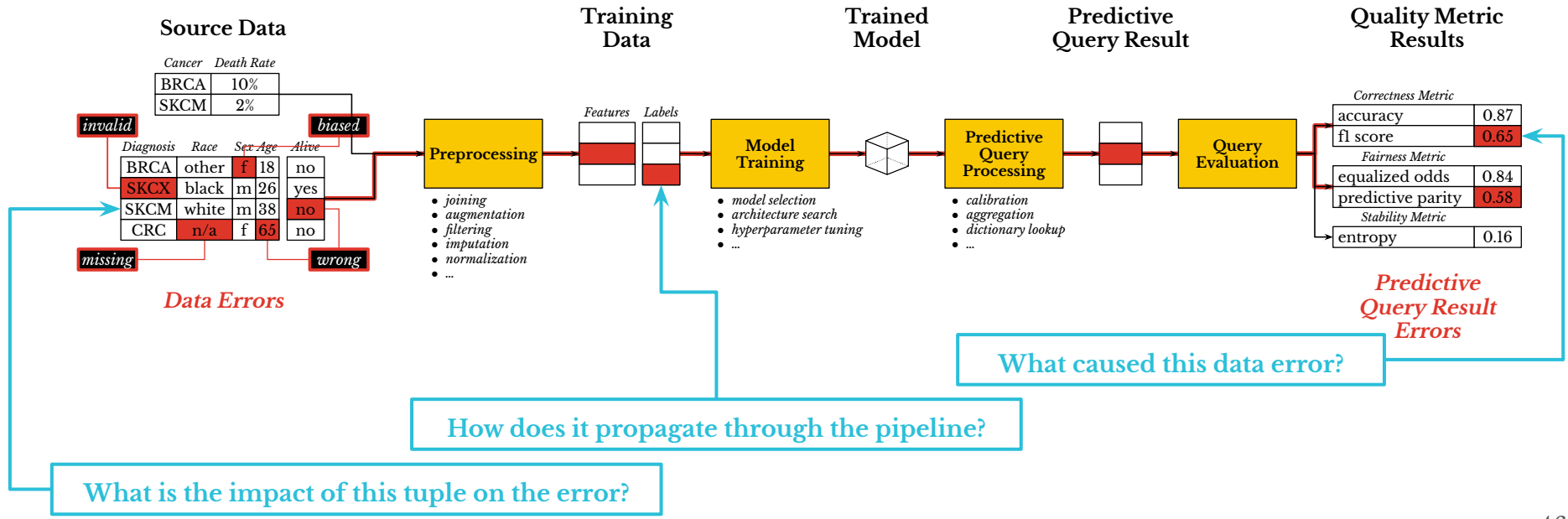
We thus embarked on a (costly) fact-finding mission, consisting of large data collection and analysis, to better understand this shifting panorama. In particular, we analyzed (a) 8M notebooks, publicly shared on GitHub—representative of OSS, educational, and self-learning activities; and (b) 2M ML pipelines professionally authored in ML.NET within Microsoft. DS encompasses a wide range of operations (e.g., wrangling, visualization, ML, collaboration), and collecting datasets representative of them all is a herculean task. As we will see, the datasets we use here are a first step towards this end, and skewed towards ML, visualization, and data pre-processing.

Over the past few years, we have used the results

[Psallidas SIGMOD Record '22]
Psallidas, Fotis, et al. "Data science through the looking glass: Analysis of millions of github notebooks and ml. net pipelines." ACM SIGMOD Record 51.2 (2022): 30-37. [Paper]

1) Gap between Attribution Methods and ML Pipelines
2) Libraries and Systems for ML Pipelines
3) Characteristics of Real World ML Pipelines
4) **Methods for Debugging ML Pipelines**

How should we reason about pipelines?



Modeling ML Pipelines with "Logical Query Plans"

Challenge

Understanding of the semantics of operations and the flow of data required *to reason about ML pipelines*

Insight

Many common pipeline abstractions offer ***declarative operations****, enables the extraction and definition of "logical query plans"*

Approach

Instrument functions *of Python data science libraries to* ***extract query plan****, enable* ***annotation propagation*** *through operators. Apply rule-based approaches to determine if an error has occurred (e.g. if a bias against a sensitive group has been introduced).*

Potential issues in preprocessing pipeline:

1. Join might change proportions of groups in data
2. Column 'age_group' projected out, but required for fairness
3. Selection might change proportions of groups in data
4. Imputation might change proportions of groups in data
5. 'race' as a feature might be illegal!
6. Embedding vectors may not be available for rare names!

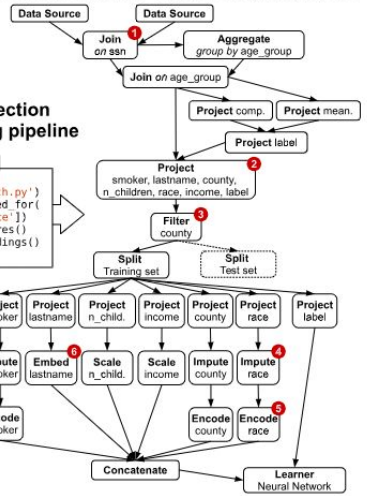
Python script for preprocessing, written exclusively with native pandas and sklearn constructs

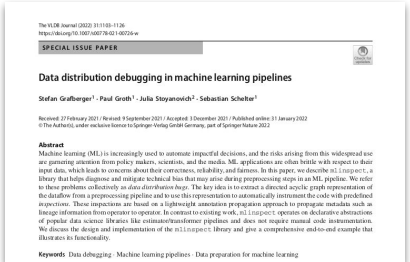
```
# load input data sources, join to single table
patients = pandas.read_csv(…)
histories = pandas.read_csv(…)
data = pandas.merge([patients, histories], on=['ssn'])

# compute mean complications per age group, append as column
complications = data.groupby('age_group')
 .agg(mean_complications=('complications','mean'))
data = data.merge(complications, on=['age_group'])

# Target variable: people with frequent complications
data['label'] = data['complications'] >
  1.2 * data['mean_complications']

# Project data to subset of attributes, filter by counties
data = data[['smoker', 'last_name', 'county',
             'num_children', 'race', 'income', 'label']]
data = data[data['county'].isin(counties_of_interest)]

# Define a nested feature encoding pipeline for the data
impute_and_encode = sklearn.Pipeline([
  (sklearn.SimpleImputer(strategy='most_frequent')),
  (sklearn.OneHotEncoder())])
featurisation = sklearn.ColumnTransformer(transformers=[
  (impute_and_encode, ['smoker', 'county', 'race']),
  (Word2VecTransformer(), 'last_name')
  (sklearn.StandardScaler(), ['num_children', 'income']])

# Define the training pipeline for the model
neural_net = sklearn.KerasClassifier(build_fn=create_model())
pipeline = sklearn.Pipeline([
  ('features', featurisation),
  ('learning_algorithm', neural_net)])

# Train-test split, model training and evaluation
train_data, test_data = train_test_split(data)
model = pipeline.fit(train_data, train_data.label)
print(model.score(test_data, test_data.label))
```

Declarative inspection of preprocessing pipeline

```
mlinspect
PipelineInspector
.on_pipeline('health.py')
.no_bias_introduced_for(
['age_group', 'race'])
.no_illegal_features()
.no_missing_embeddings()
.verify()
```

Corresponding dataflow DAG for instrumentation, extracted by *mlinspect*





The VLDB Journal (2022) 31:1103–1126
https://doi.org/10.1007/s00778-021-00726-w

SPECIAL ISSUE PAPER

Data distribution debugging in machine learning pipelines

Stefan Grafberger[1] · Paul Groth[1] · Julia Stoyanovich[2] · Sebastian Schelter[3]

Received: 27 February 2021 / Revised: 9 September 2021 / Accepted: 3 December 2021 / Published online: 31 January 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
Machine learning (ML) is increasingly used to automate impactful decisions, and the risks arising from this widespread use are garnering attention from policy makers, scientists, and the media. ML applications are often brittle with respect to their input data, which leads to concerns about their correctness, reliability, and fairness. In this paper, we describe `mlinspect`, a library that helps diagnose and mitigate technical bias that may arise during preprocessing steps in an ML pipeline. We refer to these problems collectively as *data distribution bugs*. The key idea is to extract a directed acyclic graph representation of the dataflow from a preprocessing pipeline and to use this representation to automatically instrument the code with predefined *inspections*. These inspections are based on a lightweight annotation propagation approach to propagate metadata such as lineage information from operator to operator. In contrast to existing work, `mlinspect` operates on declarative abstractions of popular data science libraries like estimator/transformer pipelines and does not require manual code instrumentation. We discuss the design and implementation of the `mlinspect` library and give a comprehensive end-to-end example that illustrates its functionality.

Keywords Data debugging · Machine learning pipelines · Data preparation for machine learning

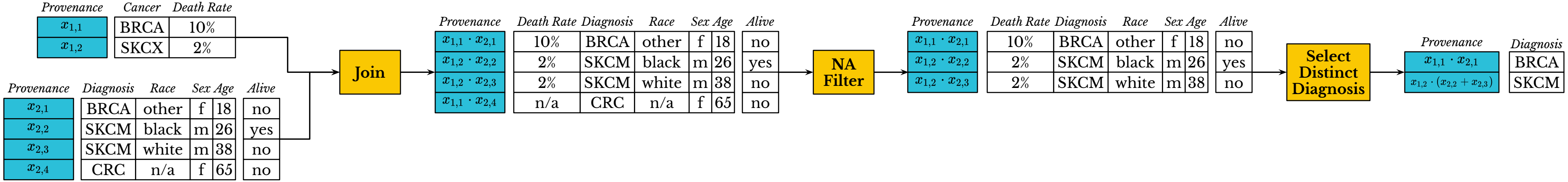
[Grafberger VLDBJ '22]
Grafberger, Stefan, et al. "Data distribution debugging in machine learning pipelines." The VLDB Journal 31.5 (2022): 1103-1126. [Paper] [Code]

Leveraging the Provenance Semiring Framework

Highlights

- *Theoretical framework* ***analyzing the relationship between input and output tuples of relational queries***
- *It allows us to d**etermine the presence of an output tuple as a function of the presence of an input tuples***
- ***Easy to adapt for ML pipelines** once logical query plan is known*

Application to an Example Pipeline



Provenance Semirings

Todd J. Green tjgreen@cis.upenn.edu — Grigoris Karvounarakis gkarvoun@cis.upenn.edu — Val Tannen val@cis.upenn.edu

Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104, USA

ABSTRACT

We show that relational algebra calculations for incomplete databases, probabilistic databases, bag semantics and why-provenance are particular cases of the same general algorithms involving semirings. This further suggests a comprehensive provenance representation that uses semirings of polynomials. We extend these considerations to datalog and semirings of formal power series. We give algorithms for datalog provenance calculation as well as datalog evaluation for incomplete and probabilistic databases. Finally, we show that for some semirings containment of conjunctive queries is the same as for standard set semantics.

Categories and Subject Descriptors

H.2.1 [**Database Management**]: Data Models

General Terms

We observe that in all four cases, the calculations with annotations are strikingly similar. This suggests looking for an algebraic structure on annotations that captures the above as particular cases. We propose using commutative semirings for this purpose. In fact, we can show that the laws of commutative semirings are *forced* by certain expected identities in $\mathcal{RA}$. Having identified commutative semirings as the right algebraic structure, we argue that a symbolic representation of semiring calculations is just what is needed to record, document, and track $\mathcal{RA}$ querying from input to output for applications which require rich *provenance* information. It is a standard philosophy in algebra that such symbolic representations form *the most general* such structure. In the case of commutative semirings, just as for rings, the symbolic representation is that of polynomials. We therefore propose to use polynomials to capture provenance. Next we look to extend our approach to recursive datalog queries. To achieve this we combine semirings with fixed point theory.

[Green SIGMOD '07]
Green, Todd J., Grigoris Karvounarakis, and Val Tannen. "Provenance semirings." Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems. 2007. [Paper]

Debugging Preprocessing Pipelines with Datascope

[Attribution Function: Shapley Value]

Challenge

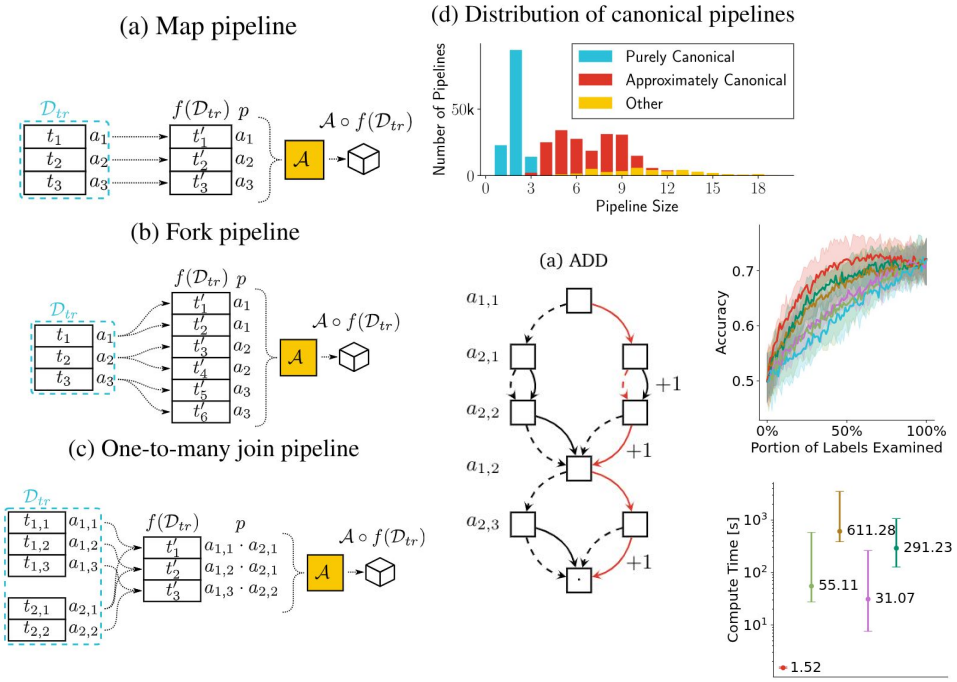
Computing the Shapley value using the KNN proxy method assumes that the presence of a single source data point maps directly to a single data point fed to the model. Hence, the results are not directly applicable to arbitrary pipelines.

Insight

We can use the provenance framework to analyze pipelines and develop PTIME algorithms for computing the Shapley value. We notice that there are ***three canonical types of pipelines*** *that are both representative of real-world pipelines, and lend themselves to efficient Shapley value computation.*

Approach

Compile provenance polynomials to Additive Decision Diagrams and use them to compute Shapley values in PTIME.



[Karlaš ICLR '24]
Karlaš, Bojan, et al. "Data Debugging with Shapley Importance over Machine Learning Pipelines." The Twelfth International Conference on Learning Representations. 2024. [Paper] [Website] [Code]

Debugging Predictive Queries with Rain

[Attribution Function: Influence]

Challenge

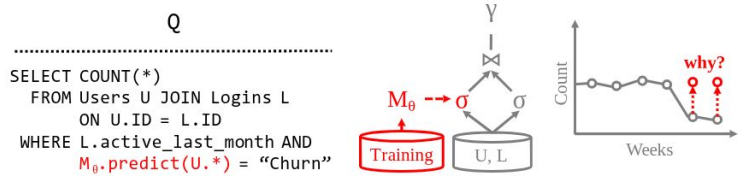
The existing influence-based attribution methods assume that the model predictions are directly used for computing model quality. However, model inference is often part of a larger predictive query.

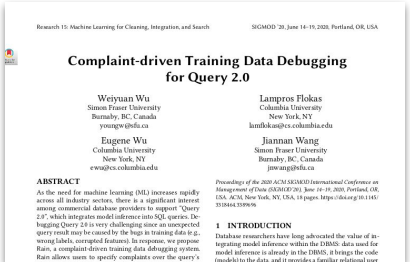
Insight

Using provenance polynomials to ***track lineage starting from training tuples all the way to predictive query outputs*** *allows us to make the entire expression differentiable.*

Approach

User complaints on query outputs (e.g. what-if-queries) are used to identify errors. Make the ***entire query differentiable*** *using provenance polynomials and run the influence framework to identify errors in the training dataset.*





Research 15: Machine Learning for Cleaning, Integration, and Search SIGMOD '20, June 14–19, 2020, Portland, OR, USA

Complaint-driven Training Data Debugging for Query 2.0

Weiyuan Wu
Simon Fraser University
Burnaby, BC, Canada
youngw@sfu.ca

Lampros Flokas
Columbia University
New York, NY
lamflokas@cs.columbia.edu

Eugene Wu
Columbia University
New York, NY
ewu@cs.columbia.edu

Jiannan Wang
Simon Fraser University
Burnaby, BC, Canada
jnwang@sfu.ca

ABSTRACT

As the need for machine learning (ML) increases rapidly across all industry sectors, there is a significant interest among commercial database providers to support "Query 2.0", which integrates model inference into SQL queries. Debugging Query 2.0 is very challenging since an unexpected query result may be caused by the bugs in training data (e.g., wrong labels, corrupted features). In response, we propose Rain, a complaint-driven training data debugging system. Rain allows users to specify complaints over the query's

Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA. ACM, New York, NY, USA, 18 pages. https://doi.org/10.1145/3318464.3389696

1 INTRODUCTION

Database researchers have long advocated the value of integrating model inference within the DBMS: data used for model inference is already in the DBMS, it brings the code (models) to the data, and it provides a familiar relational user

[Wu SIGMOD '20]
Wu, Weiyuan, et al. "Complaint-driven training data debugging for query 2.0." Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020. [Paper]

ArgusEyes - Continuous Integration for ML Pipelines

Challenge

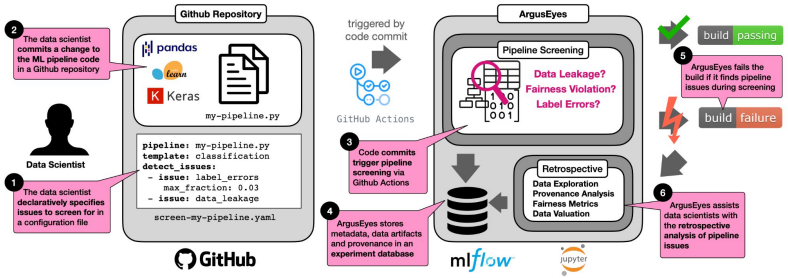
ML systems lack sophisticated testing infrastructure developed for classical software engineering. Many data-related problems only become apparent in production.

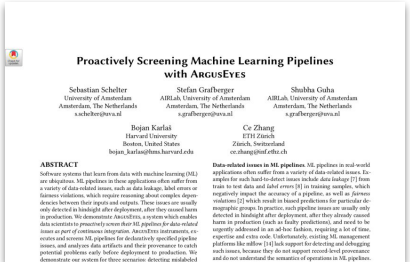
Insight

Logical query plans for ML pipelines combined with data debugging techniques enable ML-specific CI infrastructure.

Approach

Instrument, execute and screen ML pipelines for declaratively specified pipeline issues, and analyze data artifacts and their provenance to catch potential problems early before deployment to production.





Proactively Screening Machine Learning Pipelines with ArgusEyes

Sebastian Schelter, University of Amsterdam, Amsterdam, The Netherlands, s.schelter@uva.nl
Stefan Grafberger, AIRLab, University of Amsterdam, Amsterdam, The Netherlands, s.grafberger@uva.nl
Shubha Guha, AIRLab, University of Amsterdam, Amsterdam, The Netherlands, s.grafberger@uva.nl
Bojan Karlaš, Harvard University, Boston, United States, bojan_karlas@hms.harvard.edu
Ce Zhang, ETH Zürich, Zürich, Switzerland, ce.zhang@inf.ethz.ch

ABSTRACT

Software systems that learn from data with machine learning (ML) are ubiquitous. ML pipelines in these applications often suffer from a variety of data-related issues, such as data leakage, label errors or fairness violations, which require reasoning about complex dependencies between their inputs and outputs. These issues are usually only detected in hindsight after deployment, after they caused harm in production. We demonstrate ArgusEyes, a system which enables data scientists to *proactively screen their ML pipelines for data-related issues as part of continuous integration*. ArgusEyes instruments, executes and screens ML pipelines for declaratively specified pipeline issues, and analyzes data artifacts and their provenance to catch potential problems early before deployment to production. We demonstrate our system for three scenarios: detecting mislabeled

Data-related issues in ML pipelines. ML pipelines in real-world applications often suffer from a variety of data-related issues. Examples for such hard-to-detect issues include *data leakage* [7] from train to test data and *label errors* [8] in training samples, which negatively impact the accuracy of a pipeline, as well as *fairness violations* [2] which result in biased predictions for particular demographic groups. In practice, such pipeline issues are usually only detected in hindsight after deployment, after they already caused harm in production (such as faulty predictions), and need to be urgently addressed in an ad-hoc fashion, requiring a lot of time, expertise and extra code. Unfortunately, existing ML management platforms like mlflow [14] lack support for detecting and debugging such issues, because they do not support record-level provenance and do not understand the semantics of operations in ML pipelines.

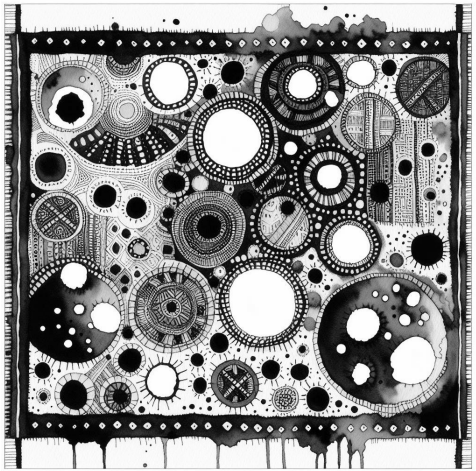
[Schelter SIGMOD Demo '23]
Schelter, et al.: "Proactively Screening Machine Learning Pipelines with ArgusEyes." Proceedings of the 2023 ACM SIGMOD International Conference on Management of Data (demo). 2023. [Paper]

Key Takeaways of Part II

- Attribution methods presented in Part I assume models are trained with source data.
- ML pipelines are complex and present many opportunities for methods development.
- Logical query plans combined with data provenance offer a powerful framework for analyzing ML pipelines.

Part III:
Learning from Uncertain and Incomplete Data

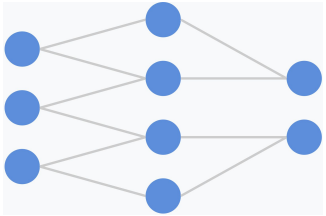
Babak Salimi



The Standard ML Pipeline

Input Data ➡ **Data Cleaning** ➡ **Model** ➡ **Inference**

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	NULL	60K	…	8K
3	35	NULL	…	[10K, 12K]
…	…	…	…	…



Loan Denied: **High** Risk

Loan Approved: **Low** Risk

Alice

Bob

⚠️ **Common Assumption:** once we "clean" the data, the pipeline consumes accurate and unbiased inputs.

❌ **Reality:** cleaning/pre-processing yields one reconstruction, driven by heuristic choices & domain assumptions → it can embed hidden bias and hide genuine uncertainty.

➡️ **Key insight for Part III:** even after best-effort cleaning, *real-world data remains incomplete and uncertain*. Our models—and the theory behind them—must make that uncertainty explicit rather than ignore it.

Why “Fixing” Data Errors Is Impossible in Principle

Missing values (📉 🏥 / 💰)
Irrecoverable uncertainty: any imputation is just a guess; the true value is unobservable.
Unverifiable assumption: “missing at random,” parametric model of the data, etc.

[Pearl & Mohan, AAAI 2014], [Mohan, Pearl & Tian, NeurIPS 2013]

Measurement / annotation bias (🗣️ sentiment, 🏥 diagnoses)
Systematic distortion: recorded values can be consistently wrong.
Unverifiable assumption: symmetric, independent label-noise model.

[Pearl, UAI 2010], [Zhang & Yu, IJCAI 2015]

Why “Fixing” Data Errors Is Impossible in Principle

Selection bias & missing counterfactuals (⚠️ rejected-loan applicants, excluded patients)
Unknown outcomes: whole sub-populations are never seen.
Finite-sample limits: re-weighting needs the true selection mechanism—which we can’t test.

[Bareinboim, Tian & Pearl, AAAI 2014] [Cortes et al., ALT2008], [Heckman, Econometrica 1979]

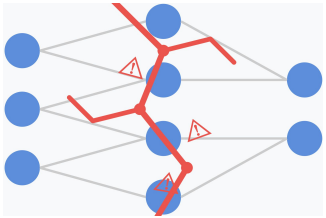
Schema / integration mismatch (⚠️ inconsistent units, 🚫 fuzzy entity resolution)
Ambiguous merges: no ground-truth correspondences.
Pre-processing bias: heuristics distort original distributions; matching is probabilistic.

[Dong, Halevy & Madhavan, VLDB 2009], [Getoor & Machanavajjhala, ACM 2012]

Challenges with Traditional Data Pipelines

Input Data ➡ **Data Cleaning** ➡ **Model** ➡ **Inference**

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	NULL	60K	…	8K
3	35	NULL	…	[10K, 12K]
…	…	…	…	…



Loan Denied: **High** Risk

Loan Approved: **Low** Risk

Alice

Bob

📉 **Generalization Failure** – Models trained on "repaired" data collapse under real-world shifts.

❌ **High-Stakes Mis-decisions** – Hidden bias drives flawed credit, medical, and justice outcomes.

⚠️ **Broken Uncertainty** – Bayesian & conformal intervals lose calibration when data are incomplete.

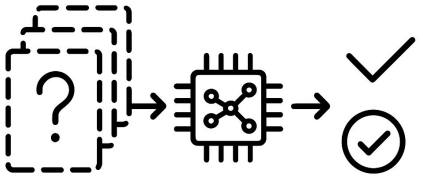
Learning from Incomplete Databases

Perfect cleaning is a myth. Even with best-effort repairs, many plausible datasets remain

Hidden uncertainty ⇒ hidden risk. A model trained on one arbitrary repair can look accurate yet flip decisions on another equally valid repair.

Needed: an explicit uncertainty framework.

- capture what is *unknown* in the data,
- propagate that uncertainty through training,
- surface it at inference time.



Practical pay-off.

- **Robustness check:** see when all admissible models agree (safe to act).
- **Guardrail:** abstain or seek more data when predictions diverge.
  Targeted cleaning: focus effort on the cells that actually shrink uncertainty.

Incomplete Databases

Formalism from databases & AI to handle **uncertainty** by modeling **all plausible data interpretations**. *(Rooted in modal logic & philosophy)*

Dataset with Quality Issues

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	NULL	60K	…	8K
3	35	NULL	…	[10K, 12K]
…	…	…	…	…

Q : What is the total income?

Possible Worlds Semantics

Inference:

- **All repairs agree → Certain answer**
  Range ≤ τ → Robust interval (e.g., [5 k – 6 k])
- **Range > τ → Uncertain** → warn / seek more cleaning

Dataset with Quality Issues

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	NULL	60K	…	8K
3	35	NULL	…	[10K, 12K]
…	…	…	…	…

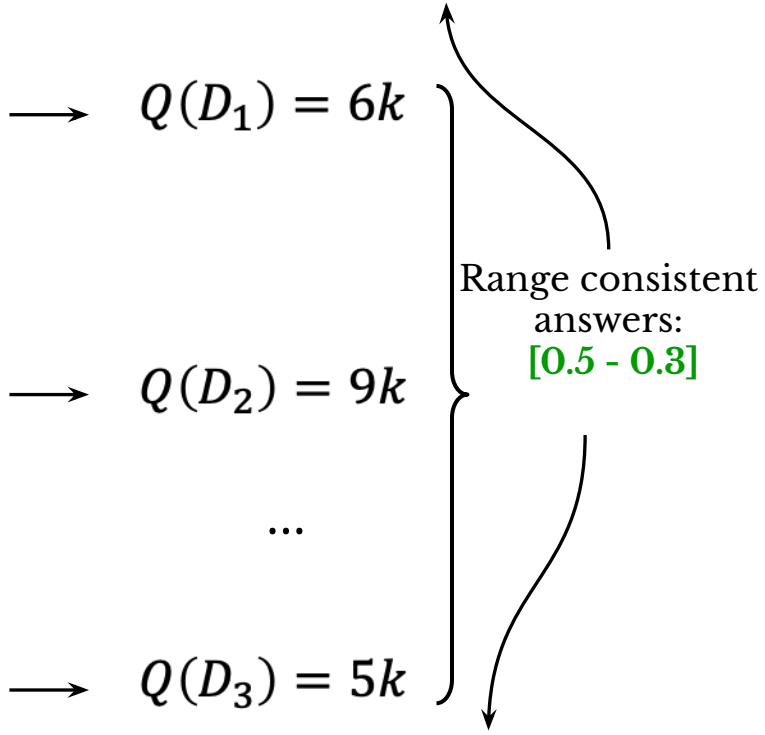
Q : What is the total income?

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	30	60K	…	8K
3	35	55K	…	7K
…	…	…	…	…

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	35	60K	…	8K
3	35	60K	…	8K
…	…	…	…	…

…

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	35	60K	…	8K
3	35	60K	…	8K
…	…	…	…	…



Representing Uncertainty in Databases

C-Tables/M-Tables: Compactly represent multiple possible worlds using variables and conditions.

[Imieliński & Lipski, JACM 1984], [Sundarmurthy et al., ICDT 2017]

Probabilistic Databases: Assign probabilities to possible worlds, quantifying their likelihood.

[Suciu, Olteanu, Ré & Koch, Book 2022]

Answering queries across possible worlds is computationally expensive, often NP-hard or exponential.

ML from Possible Repairs

Dataset with Quality Issues

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	NULL	60K	…	8K
3	35	NULL	…	[10K, 12K]
…	…	…	…	…

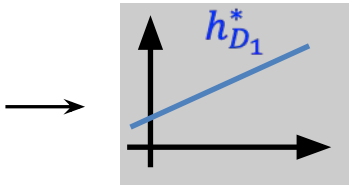
Machine-learning analogue of **Consistent Query Answering**: swap the SQL query **Q** for a training routine **T**—e.g., gradient descent, decision-tree induction, SVM fitting.

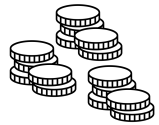
Inference

- All models ($h^*_{D_i}$)concur → ***Certain*** **prediction** (e.g., payout = 3 K)
- disagree → ***Range*** **prediction** (e.g., payout ∈ [2 K , 4 K])

NP-HARD

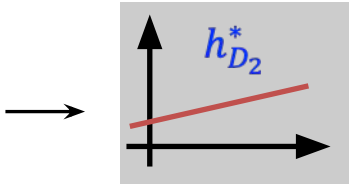
ID	Age	Income	…	Loan
1	25	50K	…	5K
2	30	60K	…	8K
3	35	55K	…	7K
…	…	…	…	…





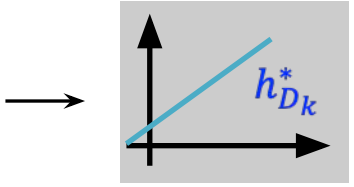
3K

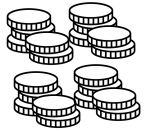
ID	Age	Income	…	Loan
1	25	50K	…	5K
2	35	60K	…	8K
3	35	60K	…	8K
…	…	…	…	…



2K

ID	Age	Income	…	Loan
1	25	50K	…	5K
2	35	60K	…	8K
3	35	60K	…	8K
…	…	…	…	…





4K

KNN Classifiers over Incomplete Information

[Approach: "Certain-kNN" → returns a label only when it is guaranteed across all completions of the missing values]

Insights:

- Missing attributes can flip k-NN labels; intersecting votes across **all** imputations yields a *guaranteed* label.

Approach:

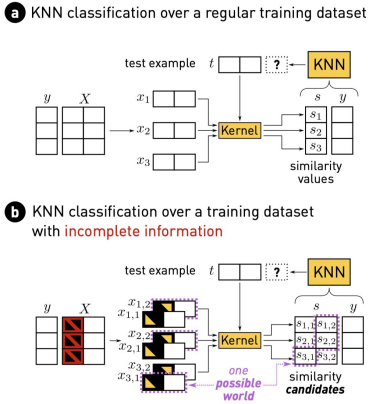
- Model each incomplete record as a value set (hyper-rectangle).
- Two polynomial-time tests (**SS**, **MM**) decide if a test point is "certain" without enumerating possible worlds.

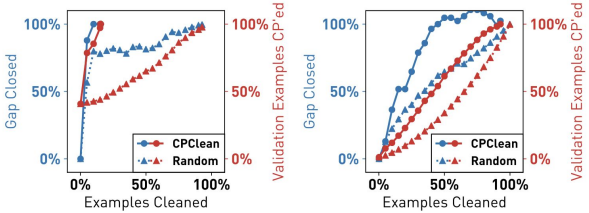
Benefits:

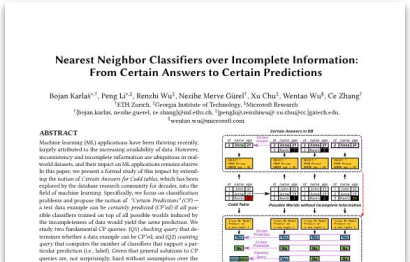
- **100 % precision on "certain" points – i.e., points whose prediction is certain across every imputation.**
- **CPClean add-on** ranks the missing cells whose repair would turn "uncertain" points into certain ones, guiding targeted data cleaning.

Shortcomings:

- Guarantees apply only to **numeric-feature k-NN**






Nearest Neighbor Classifiers over Incomplete Information: From Certain Answers to Certain Predictions

Bojan Karlaš, Peng Li, Renzhi Wu, Nezihe Merve Gürel, Xu Chu, Wentao Wu, Ce Zhang

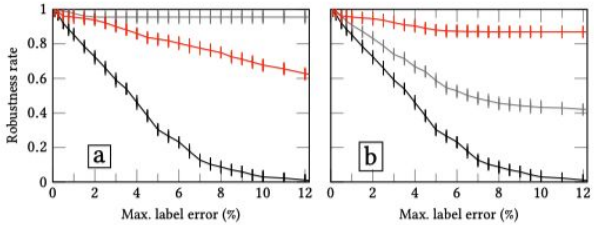
ABSTRACT

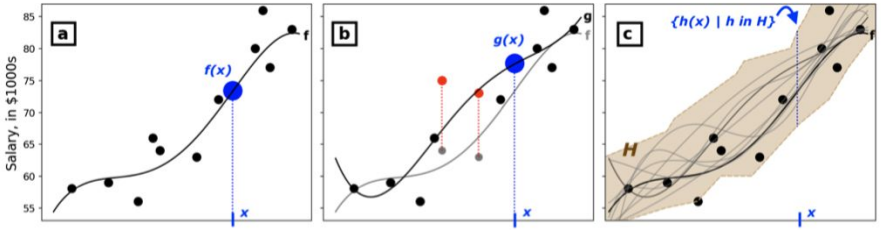
Machine learning (ML) applications have been thriving recently, largely attributed to the increasing availability of data. However, inconsistency and incomplete information are ubiquitous in real-world datasets, and their impact on ML applications remains elusive. In this paper, we present a formal study of this impact by extending the notion of *Certain Answers for Codd tables*, which has been explored by the database research community for decades, into the field of machine learning. Specifically, we focus on classification problems and propose the notion of *"Certain Predictions"* (CP)—a test data example can be *certainly predicted* if CP'ed by all possible classifiers trained on top of all possible worlds induced by the incompleteness of data would yield the same prediction. We study two fundamental CP queries: (Q1) *checking query* that determines whether a data example can be CP'ed, and (Q2) *counting query* that computes the number of classifiers that support a particular prediction (i.e., label). Given that general solutions to CP queries are, not surprisingly, hard without assumption over the

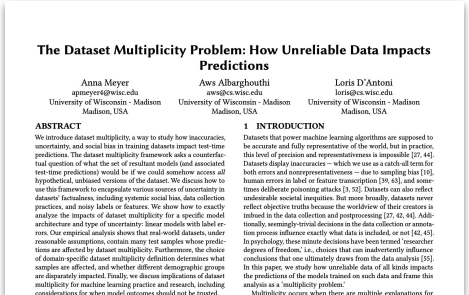
[Karlaš VLDB '20]
Karlaš, Bojan, et al. "Nearest neighbor classifiers over incomplete information: from certain answers to certain predictions." Proceedings of the VLDB Endowment 14.3 (2020): 255-267. [Paper]

The Dataset Multiplicity Problem

[Approach: bound model risk across every dataset consistent with the errors]







The Dataset Multiplicity Problem: How Unreliable Data Impacts Predictions

Anna Meyer, apmeyer4@wisc.edu, University of Wisconsin - Madison, Madison, USA
Aws Albarghouthi, aws@cs.wisc.edu, University of Wisconsin - Madison, Madison, USA
Loris D'Antoni, loris@cs.wisc.edu, University of Wisconsin - Madison, Madison, USA

ABSTRACT

We introduce dataset multiplicity, a way to study how inaccuracies, uncertainty, and social bias in training datasets impact test-time predictions. The dataset multiplicity framework asks a counterfactual question of what the set of resultant models (and associated test-time predictions) would be if we could somehow access *all* hypothetical, unbiased versions of the dataset. We discuss how to use this framework to encapsulate various sources of uncertainty in datasets' factualness, including systemic social bias, data collection practices, and noisy labels or features. We show how to exactly analyze the impacts of dataset multiplicity for a specific model architecture and type of uncertainty: linear models with label errors. Our empirical analysis shows that real-world datasets, under reasonable assumptions, contain many test samples whose predictions are affected by dataset multiplicity. Furthermore, the choice of domain-specific dataset multiplicity definition determines what samples are affected, and whether different demographic groups are disparately impacted. Finally, we discuss implications of dataset multiplicity for machine learning practice and research, including considerations for when model outcomes should not be trusted.

1 INTRODUCTION

Datasets that power machine learning algorithms are supposed to be accurate and fully representative of the world, but in practice, this level of precision and representativeness is impossible [27, 44]. Datasets display inaccuracies – which we use as a catch-all term for both errors and nonrepresentativeness – due to sampling bias [16], human errors in label or feature transcription [39, 43], and sometimes deliberate poisoning attacks [3, 52]. Datasets can also reflect undesirable societal inequities. But more broadly, datasets never reflect objective truths because the worldview of their creators is imbued in the data collection and postprocessing [27, 42, 44]. Additionally, seemingly-trivial decisions in the data collection or annotation process influence exactly what data is included, or not [42, 45]. In psychology, these minute decisions have been termed 'researcher degrees of freedom,' i.e., choices that can inadvertently influence conclusions that one ultimately draws from the data analysis [55]. In this paper, we study how unreliable data of all kinds impacts the predictions of the models trained on such data and frame this analysis as a 'multiplicity problem.'

Multiplicity occurs when there are multiple explanations for

Insights:

- Introduces a risk interval: the tightest possible lower/upper bound on test error that any admissible dataset can induce for a fixed linear model.

Approach:

- Derive closed-form formulas for the worst- and best-case hinge / logistic loss of any linear classifier under those rules, avoiding enumeration.

Benefits:

- Gives practitioners a numeric certificate of how much reported accuracy can deteriorate.

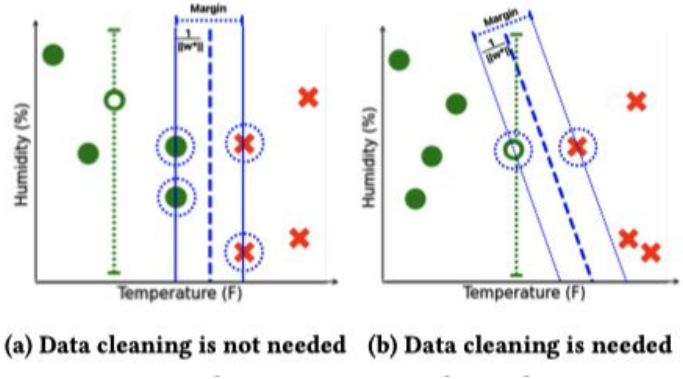
Shortcomings:

- Theory currently limited to linear models and label-noise rules; deep nets need looser convex relaxations.

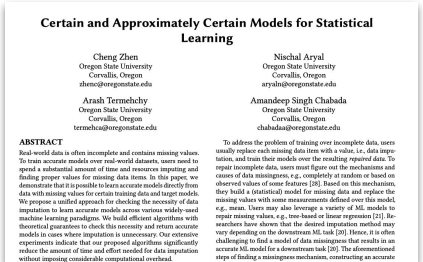
[Meyer FAccT'23]
Meyer, A. P.; Albarghouthi, A.; D'Antoni, L. "The Dataset Multiplicity Problem: How Unreliable Data Impacts Predictions. [Paper]

Certain & Approximately Certain Models for Statistical Learning

[Approach: Fast "certainty test" that lets you skip imputation whenever the missing cells don't affect the optimum]



(a) Data cleaning is not needed **(b) Data cleaning is needed**



Certain and Approximately Certain Models for Statistical Learning

Cheng Zhen
Oregon State University
Corvallis, Oregon
zhenc@oregonstate.edu

Nischal Aryal
Oregon State University
Corvallis, Oregon
aryalni@oregonstate.edu

Arash Termehchy
Oregon State University
Corvallis, Oregon
termehca@oregonstate.edu

Amandeep Singh Chabada
Oregon State University
Corvallis, Oregon
chabadaa@oregonstate.edu

ABSTRACT

Real-world data is often incomplete and contains missing values. To train accurate models over real-world datasets, users need to spend a substantial amount of time and resources imputing and finding proper values for missing data items. In this paper, we demonstrate that it is possible to learn accurate models directly from data with missing values for certain training data and target models. We propose a unified approach for checking the necessity of data imputation to learn accurate models across various widely-used machine learning paradigms. We build efficient algorithms with theoretical guarantees to check this necessity and return accurate models in cases where imputation is unnecessary. Our extensive experiments indicate that our proposed algorithms significantly reduce the amount of time and effort needed for data imputation without imposing considerable computational overhead.

To address the problem of training over incomplete data, users usually replace each missing data item with a value, i.e., data imputation, and train their models over the resulting *repaired data*. To repair incomplete data, users must figure out the mechanisms and causes of data missingness, e.g., completely at random or based on observed values of some features [28]. Based on this mechanism, they build a (statistical) model for missing data and replace the missing values with some measurements defined over this model, e.g., mean. Users may also leverage a variety of ML models to repair missing values, e.g., tree-based or linear regression [21]. Researchers have shown that the desired imputation method may vary depending on the downstream ML task [20]. Hence, it is often challenging to find a model of data missingness that results in an accurate ML model for a downstream task [20]. The aforementioned steps of finding a missingness mechanism, constructing an accurate

Insights:

- **Not every example with missing values requires cleaning.**
- If the missing cells lie in directions that do **not** change the model's optimum, we can train directly on the incomplete data—with full guarantee.

Approach:

- Provide **fast algebraic tests** (no world enumeration) that decide certainty for linear regression, linear SVM, and two kernel SVMs. When tests pass → output the **certain model** (exactly optimal).
- When tests fail → compute an **ε-certain model** whose loss is within ε of the global optimum.

Benefits:

- **Skips imputation** for datasets that pass the test, saving cleaning effort and avoiding imputation bias.
- Same code works across several common model families.

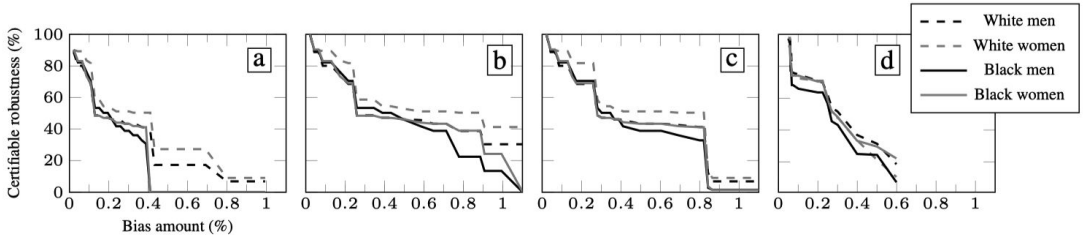
Shortcomings:

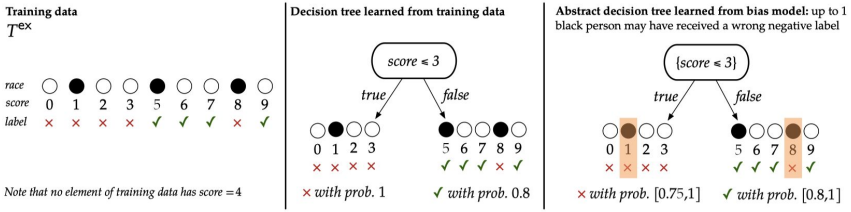
- Certainty rarely holds under heavy missingness.
  Guarantees limited to the studied linear & kernel models; deep nets need other methods.

[Zhen SIGMOD'24]
Zhen, C. et al. "Certain and Approximately Certain Models for Statistical Learning. [Paper]

Certifying Robustness to Programmable Data Bias in Decision Trees

[Approach — ProgBiasCert: encode "tree + bias program" in SMT to prove the label never flips]





Insights:

- Treat data bias as a **user-written program** (e.g., *age ± 2*, *race swap*, *income × 0.9–1.1*).
- A tree is *robust* if its prediction is invariant under **all** transformations allowed by that program.

Approach:

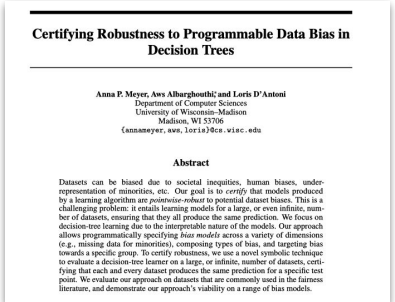
- Translate each path of the decision tree **and** the bias constraints into a single SMT formula.

Benefits:

- Exact guarantees—no sampling; works with real & categorical features and generates independently checkable proofs

Shortcomings:

- Does not yet handle ensembles or probabilistic bias distributions.



Certifying Robustness to Programmable Data Bias in Decision Trees

Anna P. Meyer, Aws Albarghouthi, and Loris D'Antoni
Department of Computer Sciences
University of Wisconsin–Madison
Madison, WI 53706
{annameyer, aws, loris}@cs.wisc.edu

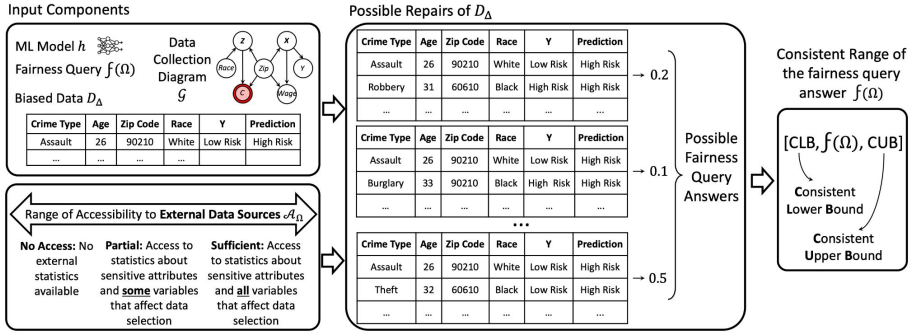
Abstract

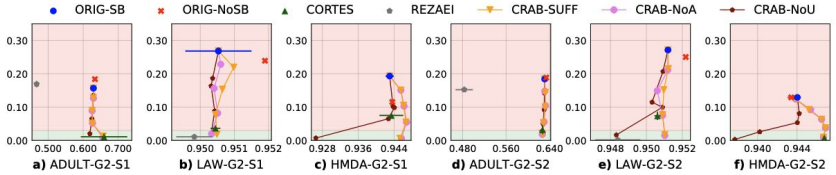
Datasets can be biased due to societal inequities, human biases, underrepresentation of minorities, etc. Our goal is to *certify* that models produced by a learning algorithm are *pointwise-robust* to potential dataset biases. This is a challenging problem: it entails learning models for a large, or even infinite, number of datasets, ensuring that they all produce the same prediction. We focus on decision-tree learning due to the interpretable nature of the models. Our approach allows programmatically specifying *bias models* across a variety of dimensions (e.g., missing data for minorities), composing types of bias, and targeting bias towards a specific group. To certify robustness, we use a novel symbolic technique to evaluate a decision-tree learner on a large, or infinite, number of datasets, certifying that each and every dataset produces the same prediction for a specific test point. We evaluate our approach on datasets that are commonly used in the fairness literature, and demonstrate our approach's viability on a range of bias models.

[Meyer NeurIPS'21]
Zhen, C.; Aryal, N.; Termehchy, A.; Chabada, A. S. "Certifying Robustness to Programmable Data Bias in Decision Trees." [Paper]

Consistent Range Approximation for Fair Predictive Modeling

[Approach: Fair-aware prediction ranges: bound each score so it stays fair under every repair of noisy / missing sensitive attributes





Insights:

- With selection bias we **don't know** the target-population fairness.
- Treat fairness evaluation as a **query over incomplete data**; answer with a *range* that is guaranteed to contain the truth.

Approach:

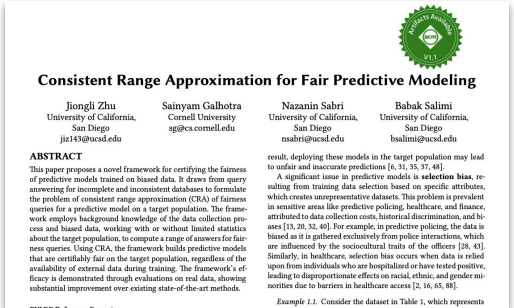
- Derive a closed-form range for fairness aggregates.
- Train a classifier that minimises risk while keeping the worst-case value inside the acceptable fairness range.

Benefits:

- Certifies fairness without unbiased samples; needs only the biased data + background knowledge.

Shortcomings:

- Relies on correct causal diagram; ranges may be wide if knowledge is weak.



Consistent Range Approximation for Fair Predictive Modeling

Jiongli Zhu, University of California, San Diego, jiz143@ucsd.edu
Sainyam Galhotra, Cornell University, sg@cs.cornell.edu
Nazanin Sabri, University of California, San Diego, nsabri@ucsd.edu
Babak Salimi, University of California, San Diego, bsalimi@ucsd.edu

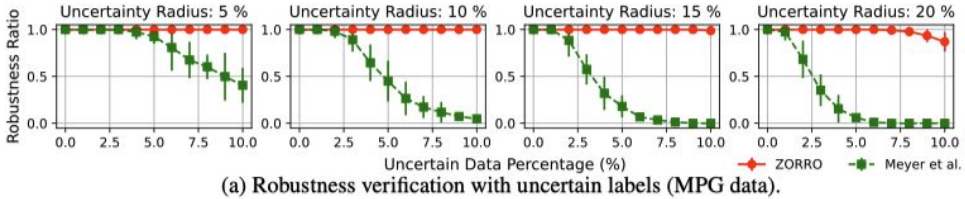
ABSTRACT

This paper proposes a novel framework for certifying the fairness of predictive models trained on biased data. It draws from query answering for incomplete and inconsistent databases to formulate the problem of consistent range approximation (CRA) of fairness queries for a predictive model on a target population. The framework employs background knowledge of the data collection process and biased data, working with or without limited statistics about the target population, to compute a range of answers for fairness queries. Using CRA, the framework builds predictive models that are certifiably fair on the target population, regardless of the availability of external data during training. The framework's efficacy is demonstrated through evaluations on real data, showing substantial improvement over existing state-of-the-art methods.

[Zhu VLDB '23]
Consistent Range Approximation for Fair Predictive Modeling. [Paper]

Learning from Uncertain Data: From Possible Worlds to Possible Models

[Approach: Abstract interpretation + zonotopes: train once on a single convex polytope that encodes every possible repair

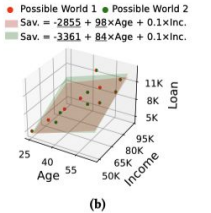


(a) Robustness verification with uncertain labels (MPG data).

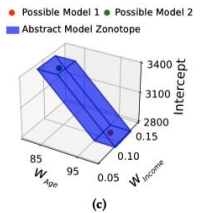
Training Data

Age	Income	Savings
25	50K	5K
NULL	60K	6K
35	NULL	7K
NULL	NULL	[8K,9K]
45	90K	12K
50	NULL	[10K,12K]
55	75K	9K
60	85K	10K
65	80K	13K

(a)



(b)



(c)

Possible Model 1 Predictions

Age	Income	Savings
20	90K	8.1K
70	50K	9K

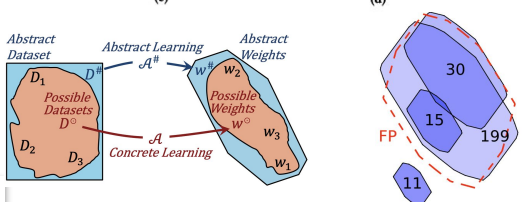
Possible Model 2 Predictions

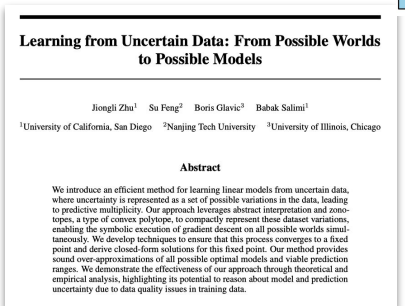
Age	Income	Savings
20	90K	7.3K
70	50K	11.5K

ZORRO Prediction Ranges

Age	Income	Savings
20	90K	[7K,8.2K]
70	50K	[8.5K,12K]

(d)





Learning from Uncertain Data: From Possible Worlds to Possible Models

Jiongli Zhu[1] Su Feng[2] Boris Glavic[3] Babak Salimi[1]

[1]University of California, San Diego [2]Nanjing Tech University [3]University of Illinois, Chicago

Abstract

We introduce an efficient method for learning linear models from uncertain data, where uncertainty is represented as a set of possible variations in the data, leading to predictive multiplicity. Our approach leverages abstract interpretation and zonotopes, a type of convex polytope, to compactly represent these dataset variations, enabling the symbolic execution of gradient descent on all possible worlds simultaneously. We develop techniques to ensure that this process converges to a fixed point and derive closed-form solutions for this fixed point. Our method provides sound over-approximations of all possible optimal models and viable prediction ranges. We demonstrate the effectiveness of our approach through theoretical and empirical analysis, highlighting its potential to reason about model and prediction uncertainty due to data quality issues in training data.

Insights:

- Zonotope = all repairs in a compact affine form.
- Training on the zonotope gives one weight-box that subsumes every per-repair model.

Approach:

- Map each uncertain record to an affine form; the full dataset becomes **one zonotope**.
  Run gradient descent **symbolically**.
  Output is a convex box of model weights; any concrete repair yields weights inside this box.

Benefits:

- **Guaranteed intervals** for weights & predictions—true model always inside.

Shortcomings:

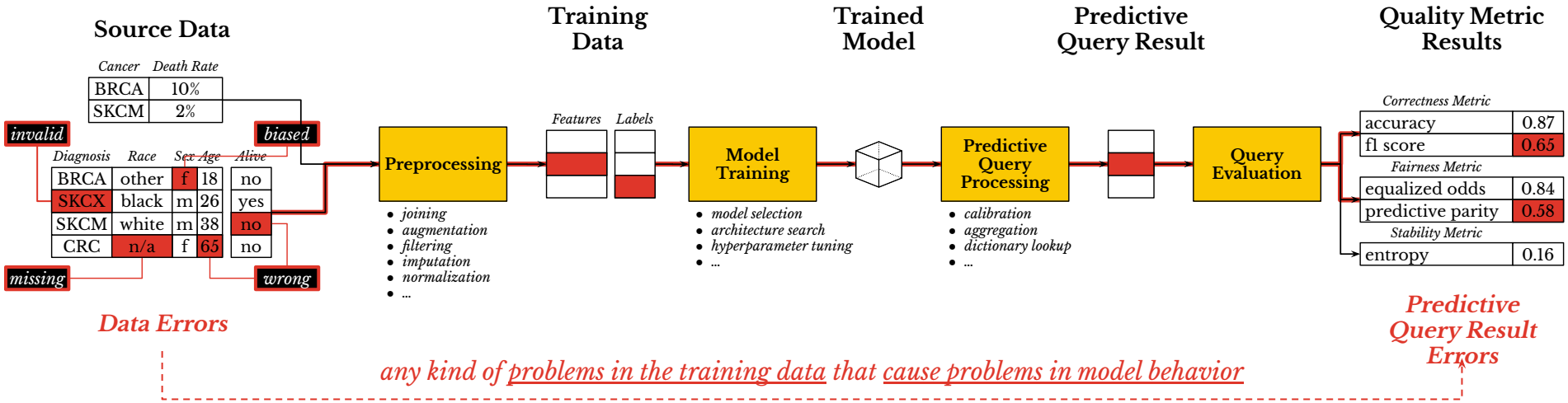
- Supports **linear** models only.

[Zhu NeurIPS'24]
Zhu, J.; Feng, S.; Glavic, B.; Salimi, B. "Learning from Uncertain Data: From Possible Worlds to Possible Models. [Paper]

Key Takeaways of Part III

- Residual data uncertainty is inevitable. Cleaning produces at best one plausible version; we must reason over the space of possibilities.
- Guarantee ↔ coverage trade-off. Certainty methods (Certain-kNN, CRA, ProgBiasCert) give perfect precision or fairness—but may abstain widely.
- Targeted cleaning beats blanket imputation. Algorithms like CPClean and OTClean identify the few cells whose repair actually widens certified coverage.
- Model-side defences matter. Dataset Multiplicity, Certain/Approx-Certain Models, and Zorro show how to train / audit over the whole uncertainty set—returning intervals, ensembles, or risk bounds.
- Certification > best-guess. When stakes are high, prefer guaranteed ranges or proofs of robustness to a single point prediction from a guessed-clean dataset.
- Open frontiers: extend guarantees to deep nets & categorical features, tighten bounds under heavy missingness, and scale zonotope / SMT methods to larger models.

Conclusion: How should we navigate data errors?



Error Detection: Compute Data Importance	ML Pipeline Debugging: Leverage Data Provenance	Learning from Uncertain Data: Apply Possible Worlds Semantics

Thank you!