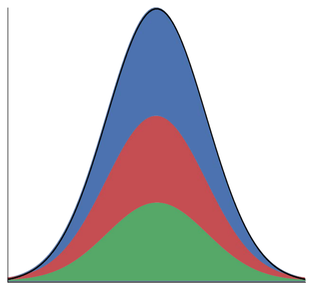
How Developers Iterate on Machine Learning Workflows

-- A Survey of the Applied Machine Learning Literature

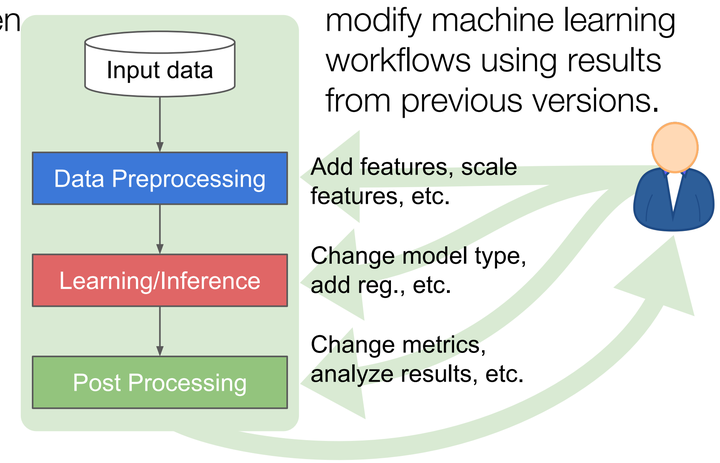
Doris Xin[1], Litian Ma[1], Shuchen Song[1], Rong Ma[2], Aditya Parameswaran[1]
[1] University of Illinois at Urbana-Champaign [2] Peking University

(1) Motivation

❑ How do developers split their time between different components of the workflow?



Num. Iterations



❑ Developers ***iteratively*** modify machine learning workflows using results from previous versions.

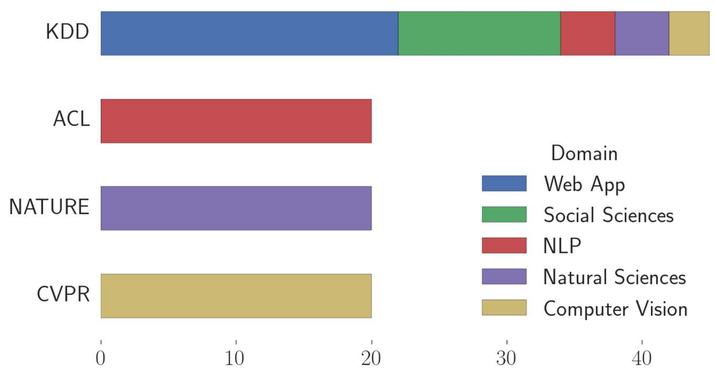
Current studies suffers from response bias.

❑ We conduct a *statistical* study of machine learning model development from *empirical evidence*.

❑ Study results reveal desiderata for ML systems.

(2.1) Data

❑ 105 applied data science papers published in 2016



Application domain of papers breakdown by conference

❑ Limitations of data
- Incomplete picture of iterations
- Results not presented in iteration order
- Small per domain corpus can lead to spurious results

❑ Remedies
- Multiple surveyors to reduce chance of spurious results
- Design iteration estimators that do not rely on order

(2.2) Estimating Iterations

	Data Prep.			ML Model Class			ML Tuning			Evaluation Metrics			
	norm.	impute	...	DT	SVM	...	Reg.	λ	...	AUC	...	# tables	# figs
	✔	✔			✔					✔		5	2
	✔				✔		✔			✔		5	2
	✔	✔			✔					✔		5	2
Aggregate	✔	✔			✔					✔		5	2
	$n_{\mathcal{D}} = 2$			$n_{\mathcal{M}} = 1$			$n_{\mathcal{P}} = 1$			$n_{\mathcal{E}} = 1$		$n_{table} = 5$	$n_{figure} = 2$

❑ Three surveyors independently identify operations mentioned in a paper.

❑ Take majority vote to aggregate the three results.

❑ Collect statistics on the aggregate annotation.

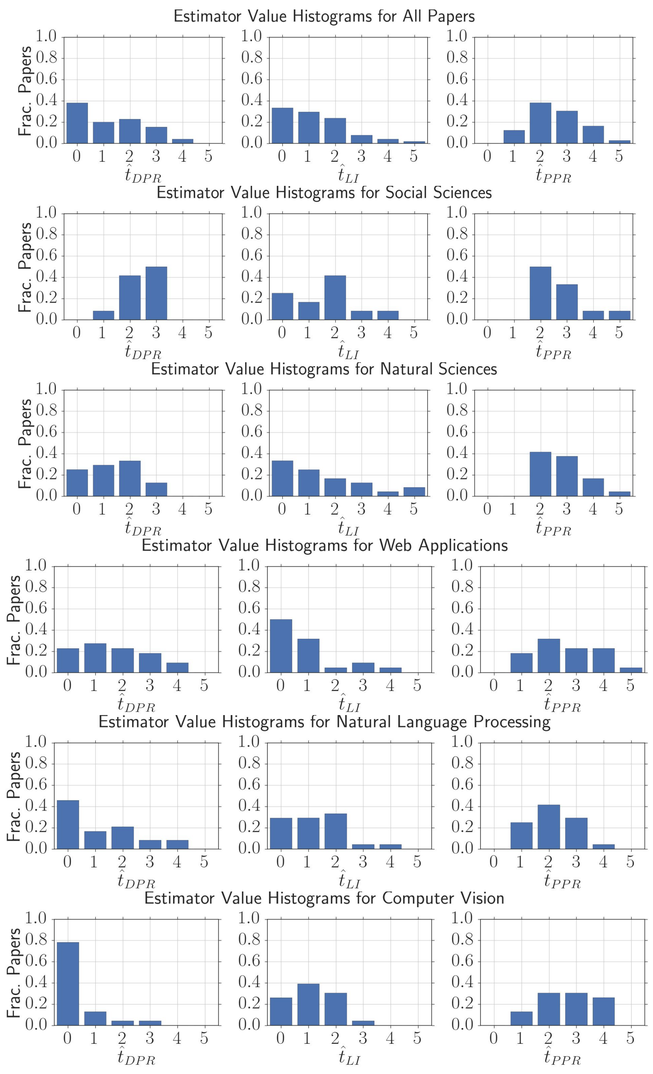
Estimator for # data prep. iterations $\hat{t}_{DPR} = n_{\mathcal{D}}$

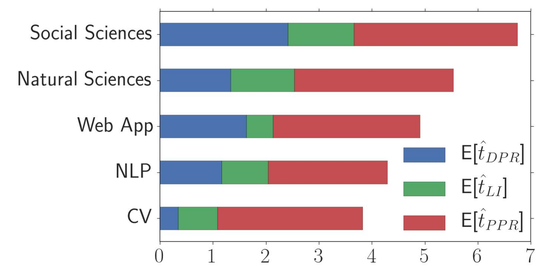
Estimator for # ML iterations $\hat{t}_{LI} = (n_{\mathcal{M}} - 1) + (n_{\mathcal{P}} - 1)$

Estimator for # post proc. iter. $\hat{t}_{PPR} = \min(n_{\mathcal{E}}, n_{table} + n_{figure})$

Open source dataset at https://github.com/helix-ml/AppliedMLSurvey

(3.1) Iteration Distributions





Highlights

❑ ***Evaluation*** iterations are the ***most common in all domains.***

❑ ***Natural and social sciences*** have ***more data prep***. iterations due to
- reliance on domain knowledge to guide ML and strong preference for explainable models.
- Datasets too small for DNNs

❑ ***Web apps*** collect data in organic forms, requiring ***a lot of cleaning***.

❑ ***Computer vision and NLP*** require ***fewer data prep.*** iterations due to
- DNNs require less feat. eng.
- Standardized datasets studied in all papers require less data prep.

(3.1) Common Operations

SocS	NS	WWW	NLP	CV
Join (31.0%)	Feature Def. (40.6%)	Feature Def. (36.1%)	Feature Def. (32.1%)	Feature Def. (37.5%)
Feature Def. (27.6%)	Univar. FS (18.8%)	Join (22.2%)	BOW (17.9%)	BOW (25.0%)
Normalize (17.2%)	Normalize (12.5%)	Normalize (13.9%)	Join (14.3%)	Interaction (25.0%)
Impute (6.9%)	PCA (9.4%)	Discretize (8.3%)	Normalize (10.7%)	Join (12.5%)

Table 1: Common DPR operations ordered top to bottom by popularity. Join = joining multiple data sources; Feat. def. = custom logic for fine-grained feature extraction; Univer. FS = univariate feature selection, using criteria such as support and correlation per feature; BOW = bag of words; PCA = principal component analysis, a common dimensionality reduction technique.

SocS	NS	WWW	NLP	CV
GLM (36.0%)	SVM (32.7%)	GLM (37.0%)	RNN (32.4%)	CNN (38.2%)
SVM (28.0%)	GLM (15.4%)	RF (11.1%)	GLM (14.7%)	SVM (17.6%)
RF (20.0%)	RF (13.5%)	SVM (11.1%)	SVM (11.8%)	RNN (17.6%)
Decision Tree (12.0%)	DNN (13.5%)	Matrix Factoriz. (11.1%)	CNN (8.8%)	RF (5.9%)

Table 2: Common model classes ordered top to bottom by popularity per domain. GLM = generalized linear models (e.g., logistic regression); RF = random forest; SVM = support vector machine; R/CNN = recursive/convolutional neural networks.

SocS	NS	WWW	NLP	CV
Regularize (40.0%)	CV (31.8%)	Regularize (41.2%)	LR (39.4%)	LR (46.2%)
CV (30.0%)	LR (22.7%)	LR (23.5%)	Batch Size (24.2%)	Batch Size (30.8%)
LR (10.0%)	DNN Arch. (18.2%)	Batch Size (11.8%)	DNN Arch. (18.2%)	DNN Arch. (11.5%)
Batch Size (10.0%)	Kernel (9.1%)	CV (11.8%)	Kernel (6.1%)	Regularize (11.5%)

Table 3: Most popular model tuning operations by domain. CV = cross validation; LR = learning rate; DNN arch. = DNN architecture modification; Kernel specifically applies to SVM.

SocS	NS	WWW	NLP	CV
P/R (25.7%)	Accuracy (28.6%)	Accuracy (20.8%)	P/R (29.2%)	Visualization (33.3%)
Accuracy (20.0%)	P/R (18.6%)	P/R (20.8%)	Accuracy (27.1%)	Accuracy (29.8%)
Feat. Contrib. (17.1%)	Visualization (15.7%)	Case Studies (13.2%)	Case Studies (14.6%)	P/R (17.5%)
Visualization (14.3%)	Correlation (11.4%)	DCG (9.4%)	Human Eval. (8.3%)	Case Studies (12.3%)

Table 4: Most popular evaluation methods by domain. P/R = precision/recall; Feat. Contrib. = feature contribution to model performance; NCG = discounted cumulative gain, popular in ranking tasks; Case studies = case studies of individual results.

Highlights

❑ Data prep.
- Human def. Features (long live feat. eng.!)
- Data joins

❑ Model class
- GLMs
- SVMs
- ***DNN only in NLP/CV***

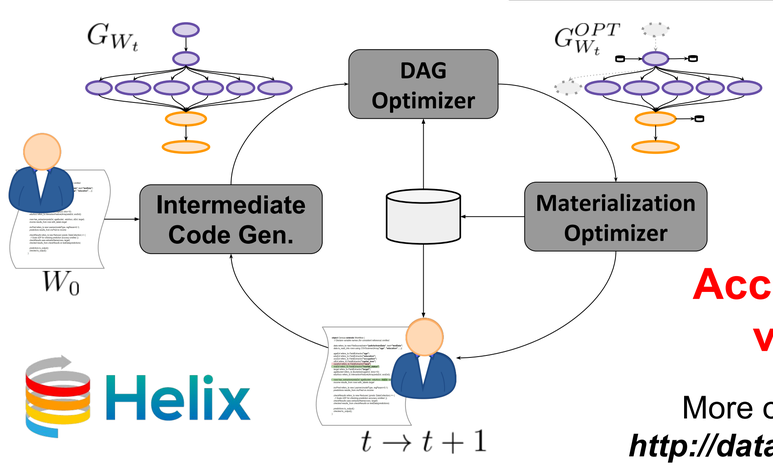
❑ Model Tuning
- Learning rate } fast training
- Batch size

❑ Evaluation
- ***Coarse grained***: P/R, Accuracy
- ***Fine-grained***: Case studies, Vis.

Prioritize system support for common operations

Helix

https://helix-ml.github.io/



❑ Address user needs discovered in our survey

❑ Selectively materialize intermediate results for reuse in future iterations

Accelerate Iterative Execution via Intermediates Reuse

More on Helix in the technical report @ ***http://data-people.cs.illinois.edu/helix-tr.pdf***