How Developers Iterate on Machine Learning Workflows

A Survey of the Applied Machine Learning Literature

Doris Xin, Litian Ma, Shuchen Song, Aditya Parameswaran

University of Illinois, Urbana-Champaign (UIUC)

{dorx0,litianm2,ssong18,adityagp}@illinois.edu

ABSTRACT

Machine learning workflow development is anecdotally regarded to be an iterative process of trial-and-error with humans-in-the-loop. However, we are not aware of statistics-based evidence corroborating this popular belief. A statistical characterization of iteration can serve as a benchmark for machine learning workflow development in practice, and can aid the development of human-in-the-loop machine learning systems. To this end, we conduct a small-scale survey of the applied machine learning literature from five distinct application domains. We use statistics collected from the papers to estimate the role of iteration within machine learning workflow development, and report preliminary trends and insights from our investigation, as a starting point towards this benchmark. Based on our findings, we finally describe desiderata for effective and versatile human-in-the-loop machine learning systems that can cater to users in diverse domains.

1 INTRODUCTION

Development of machine learning (ML) applications is governed by an iterative process: starting with an initial workflow, developers iteratively modify their workflow, based on previous results, to improve performance. They may add or modify data sources, features, hyperparameters, and training algorithms, among others. These iterations of trial-and-error are necessary due to data variability, algorithmic complexity, and overall unpredictability of ML. A *detailed, statistical characterization of how developers iteratively modify ML workflows can serve as a benchmark for human-in-the-loop ML systems.* At present, due to the lack of such studies, we are forced to resort to anecdotal evidence to identify usage patterns and motivate design decisions.

To this end, *we conduct a statistical study of iteration by surveying the applied ML literature across five application domains.* The statistics collected in this study provide the first quantitative evidence of how developers iterate on ML workflows, beyond anecdotal ones. Moreover, the insights and trends discovered from our survey provide concrete guidelines on desired human-in-the-loop ML system properties, while the models and statistics provide a starting point for the development of benchmarks for standardized and automatic evaluation of human-in-the-loop ML systems.

Statistical studies of end-to-end ML workflow development pose several challenges. First, it is difficult to gather data that captures the entire process, and not just the final snapshot. One approach, for example, may involve examining code repositories over time to determine what has changed—one downside of this approach is that developers may not commit intermediate iterations, leading to less transparency for the overall process. Moreover, this approach will require understanding code, and mapping code fragments to classes of iterative modifications, both of which are extremely challenging

to do. Second, we need to ensure that our study captures a diverse set of application domains. Surveys [1, 3, 9, 12] often end up focusing on industry-relevant application areas (e-commerce, recommendations), and data-types (language, vision). Since our eventual goal is to develop a benchmark for general-purpose human-in-the-loop ML systems, this limited view may hinder our ability to adequately support all application domains. Third, once the data is collected, we need to devise methods to analyze the data and collect statistics related to iteration. Finally, we need to turn the raw statistics into models that capture iteration and relate trends and insights discovered from these models to ML system design.

Our study includes an analysis of 105 applied machine learning papers sampled from multiple conferences in 2016 and across five application domains, including social sciences, natural sciences, web application, computer vision, and natural language processing. We collect statistics from each paper that capture iterative development and use these statistics to infer common practices in each application domain surveyed. We describe the statistics collected, how they are used to estimate iteration counts, and discuss the limitations of our approach in the next section. To ensure the quality of our statistics, we take consensus over results collected by multiple surveyors, and open-source the final aggregated data for further studies by interested readers, as well as development of formal benchmarks. We conduct data analysis on our survey results to highlight key insights unearthed by our survey and propose system requirements suggested by our analysis.

Related Work. To the best of our knowledge, our survey is the first effort in conducting a statistical study of machine learning model development from empirical evidence. However, the pursuit of understanding iterative ML development is not singularly ours. Several surveys have been conducted in recent years to profile industry and academic ML users [1, 3, 9, 12]. These surveys differ from ours in that they were self-reported responses from a select set of industry and academic users. Findings from self-reporting surveys are known to suffer from response bias [13]. Many articles discuss general trends and design patterns in ML workflows [2, 6, 7], while a number of articles focus on providing guidance and taxonomies for novice users to perform iteration better [14, 15, 18]. Other works such as [4] and [11] study general trends and needs in data science using NLP techniques to study a large corpus en masse. Vartak et al. [16] describe a system-building vision for iterative human-in-the-loop ML. Kery et al. [8] specifically study the versioning aspect of iterative development, whereas Koesten et al. [10] analyze in-depth surveys to understand the typical workflow for data scientists.

The rest of paper is organized as follows: In Section 2, we describe the data, the statistics collected from the data, and the methods to
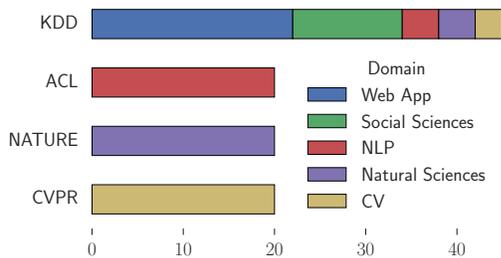
Figure 1: Paper count per domain by conference.

study iteration using the statistics. In Section 3, we report interesting results and insights discovered from our survey and propose concrete system requirements to support human-in-the-loop ML based on the survey analysis.

2 DATA & METHODOLOGY

In this section we describe the dataset and the methods used to collect the statistics that enable analyses of iteration in publications.

2.1 Corpus

We surveyed 105 papers published in 2016 on applied data science. To ensure relevance, we selected four venues that specifically publish applied machine learning studies: KDD Applied Data Science Track, Association for Computational Linguistics (ACL), Computer Vision and Pattern Recognition (CVPR), and Nature Biotechnology (NB). We randomly sample 20 papers from ACL, CVPR, and NB each, and 45 papers from KDD. These papers span applications in social sciences (SocS), web applications (WWW), natural sciences (NS), natural language processing (NLP), and computer vision (CV). Paper topics were determined using the ACM Computing Classification System (CCS) [1]. Keywords in each paper are matched with entries in the CCS tree, and each paper is assigned as its domain the most appropriate high level entry containing its keywords. Figure 1 illustrates the domain composition of the conferences surveyed. While ACL, CVPR, and Nature specialize in a single domain, KDD embraces many domains, with a focus on web applications and social science.

Limitations. Our approach is limited in its ability to accurately model iterations due to several characteristics of the corpus:

1) While the corpus spans multiple domains, the number of paper in each domain is small, which can lead to spurious trends.
2) Papers provide an incomplete picture of the overall iterative process. Machine learning papers are results-driven and focus more on modeling than data pre-processing by convention. Due to space constraints, authors often omit a large number of iterative steps and report only on the small subset that led to the final results.
3) Papers often present results side by side instead of the order they were obtained, making it difficult to determine the exact transitions between the variants studied in the iterative process.

We attempt to overcome some of these limitations by

- Having multiple surveyors and aggregating the results to reduce the change of spurious results, to be elaborated in Section 2.3;

- Devising estimators that do no rely on information about the order of operations, to be elaborated in Section 2.4.

2.2 Brief Overview of ML Workflows

ML workflows commonly consist of three major components:

Data Pre-processing (DPR). This stage contains all the data manipulation operations, such as data cleaning and feature extraction, used to turn raw data into a format compatible with ML algorithms.

Learning/Inference (L/I). Once the data is transformed into a learnable representation, such as feature vectors, learning takes place, using the transformed data to derive an ML model via optimization. Inference refers to the processing by which the learned model is used to make predictions on unseen data, and is often performed after learning.

Post Processing (PPR). Post processing is the all-encompassing term for operations following learning and inference. Bruha et al. [5] classifies PPR operations in to four categories: 1) rule-based knowledge filtering, 2) and knowledge integration, 3) interpretation and explanation, 4) evaluation. While 1) and 2) involve transformations of the L/I output, 3) and 4) are about the analysis of the L/I output. Mentions of 1) and 2) are sparse in our corpus and thus excluded from our study.

In the context of ML application development, an *iteration* involves creating a version of the workflow, either from scratch or by copying/modifying a previous version, and executing this version end to end to obtain some results. Program termination marks the end of an iteration, and any results that are not written to disk during execution can only be obtained by modifying the workflow to explicitly save the results and rerunning the workflow.

2.3 Statistics Collection

Our goal in this survey is to collect statistics on how users iterate on ML workflows. However, iterations are often not explicitly reported in publications. To overcome this challenge, we design a set of statistics that allow us to infer the iterative process leading to the results reported in each paper. We introduce the statistics for each individual component of the ML workflow below.

DPR. As mentioned above, DPR encompasses all operations involved in transforming raw data into learnable representations, such as feature engineering, data cleaning, and feature value normalization. We record $\mathcal{D}$, the set of distinct DPR operation types found in each paper and collect $n_{\mathcal{D}} = |\mathcal{D}|$. Mentions of DPR operations are usually found in the data and methods sections in the paper.

L/I. Workflow modifications concerning L/I fall into one of three categories: 1) hyperparameter tuning for a model (e.g., increasing learning rate, changing the architecture of a neural net) and 2) switching between model classes (e.g., from decision tree to SVM). For each paper, we record $\mathcal{M}$, the set of all model classes and $\mathcal{P}$, the set of distinct hyperparameters tuned across all model classes, and collect $n_{\mathcal{M}} = |\mathcal{M}|$ and $n_{\mathcal{P}} = |\mathcal{P}|$. Evidence for these statistics is usually found in the algorithms section, as well as result tables and figures.

PPR. Of the four types of PPR operations enumerated above, evaluation and interpretation/explanation are the most commonly reported in papers, often presented in tables or figures. For each

paper, we record $\mathcal{E}$, the set of evaluation metrics used, and collect $n_{\mathcal{E}} = |\mathcal{E}|$. In addition, we collect n_{table} and n_{figure}, the number of tables and figures containing results and case studies, respectively.

We refer to $\mathcal{D}, \mathcal{M}, \mathcal{P}, \mathcal{E}$ collectively as *entity sets* in the rest of the paper [2].

To ensure the quality of the statistics collected, we had three graduate students in data mining, henceforth referred to as *surveyors*, perform the survey independently on the same corpus. We reference the results collected by each surveyor with a subscript, e.g., $\mathcal{M}_1$ is the set of model classes recorded by surveyor 1. To increase the likelihood of consensus, we first had the surveyors discuss and agree on a seed set for each entity set, e.g., $\mathcal{E}$ = {Accuracy, RMSE, NDCG}. Surveyors were then asked to remove from and add to this set as they see fit for each paper. Let n'_x be the aggregated value of the statistic n_x. We aggregate the three sets of results as follows:

- For an entity set S (e.g., $\mathcal{M}$, the set of model classes), let $S_a = S_1 \cup S_2 \cup S_3$. We filter S_a to obtain $S' \subseteq S_a$ such that $s \in S'$ is identified by at least two surveyors. That is, a paper is considered to contain an operation only if it is identified to be in the paper by at least two surveyors independently. We define n'_S for the corresponding statistic as $|S'|$.
- For n_{table} and n_{figure}, we define $n'_{table/figure}$ to be the average of the values obtained by the three surveyors.

2.4 Estimating Iterations using Statistics

The information collected above indicate versions of the workflow studied but not the iterative modifications themselves. To infer the number of iterations using the statistics collected above, we make the following assumptions:

- Each iteration involves a single change. While it is possible for multiple changes to be tested in a single iteration, it is unlikely the case since the interactions can obfuscate the contribution of individual changes.
- Each element in an entity set is tested exactly once. For the authors to report on a variant, there must have been at least one version of the workflow containing that variant. Although it is likely for a variant to be revisited in multiple iterations in the actual research process, papers, by convention, provide little information on this aspect. Due to this lack of evidence, we take the conservative approach by taking the minimum value.

Let t_{DPR}, t_{LI}, t_{PPR} be the number of iterations containing changes to the DPR, L/I, and PPR components of the workflow, respectively. Using the two assumptions above, we estimate t_{DPR}, t_{LI}, and t_{PPR} as follows:

- $\hat{t}_{DPR} = n'_{\mathcal{D}}$
- $\hat{t}_{LI} = (n'_{\mathcal{M}} - 1) + (n'_{\mathcal{P}} - 1)$
- $\hat{t}_{PPR} = \min\left(n'_{\mathcal{E}}, n'_{table} + n'_{figure}\right)$

For $\hat{t}_{DPR}$, we assume that the authors start with the raw data and incrementally add more data pre-processing operations in each iteration. We subtract one from $n'_{\mathcal{M}}$ and $n'_{\mathcal{P}}$ in $\hat{t}_{LI}$ to account for the fact that the initial version of the workflow must contain a model, a set of hyperparameters, and an optimization algorithm.

The estimator $\hat{t}_{PPR}$ assumes that in a PPR iteration, the authors can either gather all information on a single metric or generate an entire figure/table.

3 RESULTS AND INSIGHTS

In this section we share interesting trends about ML workflow development discovered from our survey.
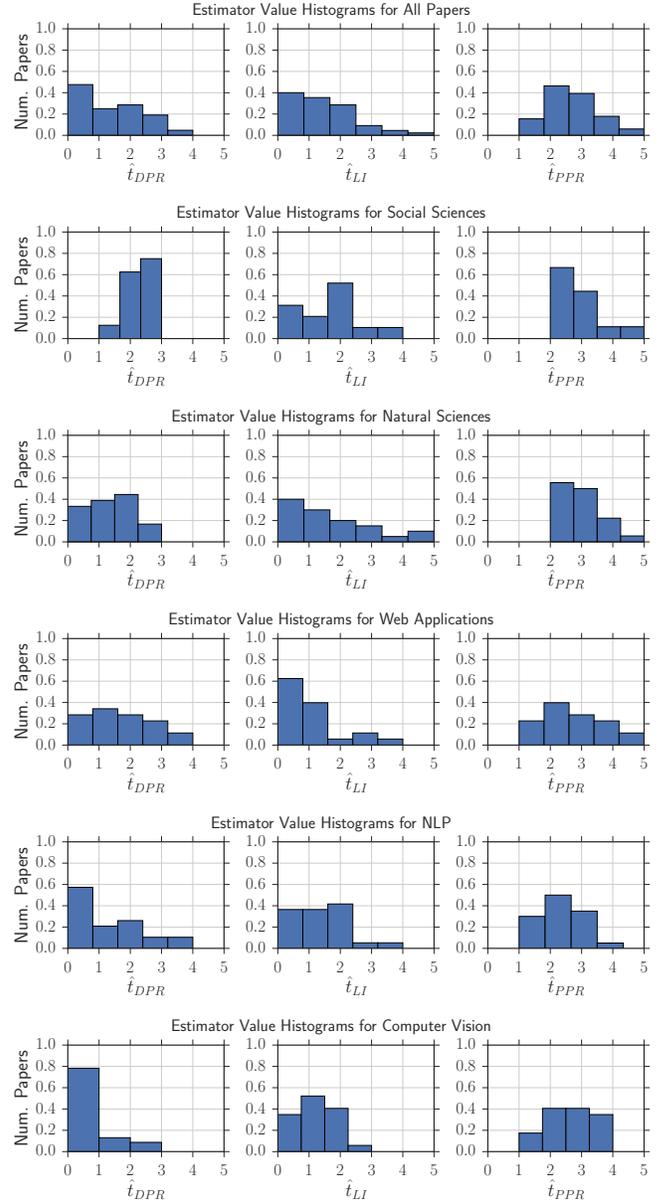
3.1 Iteration Count

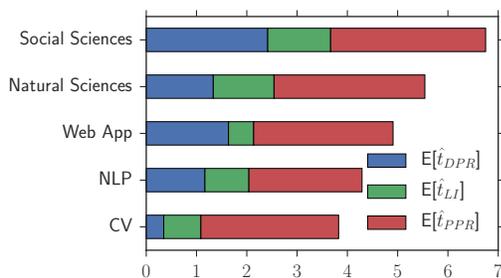**Figure 2: Distribution of number of iterations by workflow component.**

Figure 3: Mean iteration count by domains.

Figure 2 shows the histograms for the three iteration estimators $\hat{t}_{DPR}, \hat{t}_{LI}, \hat{t}_{PPR}$ across the entire corpus (top row) and by domain (rows 2-6). A bin in every histogram represents an integral value for the estimators, and bin heights equal the fraction of papers with the bin value as their estimates. The mean values for the estimators by domains are shown in the stacked bar chart in Figure 3, where the total bar length is equal to the average number of iterations in each domain. From these two figures, we see that 1) most papers use ≥ 1 evaluation methods, evident from the fact that histograms in the third column in Figure 2 are skewed towards $\hat{t}_{PPR} \geq 2$; 2) PPR is the most common iteration type across all domains, evident from the length of the $E[\hat{t}_{PPR}]$ bars in Figure 3; and 3) on average, more DPR iterations are reported than L/I iterations in every domain except computer vision, as illustrated by the relative lengths of the $E[\hat{t}_{DPR}]$ and $E[\hat{t}_{LI}]$ bars in Figure 3.

When grouped by domains, we see that the distributions for certain domains deviate a great deal from the overall trends in Figure 2. Domains dominated by deep neural nets (DNNs), which are designed to replace manual feature engineering for higher order features, tend to skew towards fewer DPR and more L/I iterations, such as NLP and CV. Additionally, there are only a few highly processed datasets studied in all NLP and CV papers, further reducing the need for data pre-processing in these domains. On the other hand, social and natural sciences exhibit the opposite trend in the histograms in Figure 2, biasing towards more DPR iterations. This is largely due to the fact that both domains rely heavily on domain knowledge to guide ML and strongly prefer explainable models. In addition, a large amount of data is required to enable training of DNNs. The scale of data is often much smaller for SocS and NS than NLP and CV, thus preventing effective application of DNNs and requiring more manual features.

3.2 Data Pre-processing by Domain

Table 1 shows the most popular DPR operations in each application domain, ordered top to bottom by popularity, with abbreviations expanded in the caption. While the table reaffirms common knowledge such as feature normalization is important, Table 1 also shows two striking results: 1) joining multiple data sources is common in four of the five domains surveyed; 2) $\frac{1}{3}$ of the papers contain fine-grained features defined using domain knowledge across all domains. Result 1) suggest that unlike classroom and data competition settings in which the input data resides conveniently in a single file, data in real-world ML applications is aggregated from multiple sources (e.g., user database and event logs). Result 2) contradicts the

common belief that ML applications have collectively progressed beyond handcrafted features thanks to the advent of deep learning (DL). In addition to the incompatibilities with DL in some domains mentioned in Section 3.1, the efficacy of features designed using domain knowledge versus using DL to search for the same features without domain knowledge is possibly another contributing factor.

3.3 Learning/Inference by Domain

Table 2 lists the most popular model classes for each application domain, with abbreviations expanded in the caption. We have already discussed the disparity between the popularity of DL in CV/NLP and other domains in Section 3.1. Most traditional approaches such as GLM, SVM, and Random Forest are still in favor with most domains, since the large additional computation cost for DL often fails to justify the incremental model performance gain. Matrix factorization, which is highly amenable to parallelization, is popular in web applications for supporting recommendation engines. Interestingly, SVM is the most popular method in natural sciences by a large margin (100% more popular than the second most popular option), possibly due to its ability to support higher order functions through kernels. NS applications experimenting with DL are mostly computer vision related.

Table 3 shows the most popular model tuning operations by domains. The top two operations, learning rate and batch size, are both concerned with the training convergence rate, suggesting that training time is an important factor in all domains. Cross validation and regularization are both mechanisms to control model complexity and overfitting to observed data. Lower complexity models usually result in faster inference time and better ability to generalize to more unseen data.

3.4 Post Processing by Domain

Of the evaluation methods listed in Table 4, P/R, accuracy, correlation, and DCG are summary evaluations of model performance while case study, feature contribution, human evaluation, and visualization are fine-grained methods towards insights to improve upon the current model. While the former group can be used automatically such as in grid search, the latter group is aimed purely for human understanding.

3.5 System Desiderata

The results in Section 3 suggest a number of properties that a versatile and effective human-in-the-loop ML system should possess:

- **Iteration.** Developers iterate on their workflows in every application domain and test out changes to all components of the workflow. Understanding the most frequent changes helps us develop systems that anticipate and respond rapidly to iterative changes.
- **Fine-grained feature engineering.** Handcrafted features designed using domain knowledge is still an indispensable part of the workflow development systems in all domains and should therefore be adequately supported instead of dismissed as an outdated practice.
- **Efficient joins.** Data is often pooled from multiple sources, thus requiring systems to support efficient joins in the data pre-processing component.

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 Factorization (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%)	Acc. (28.6%)	Acc. (20.8%)	P/R (29.2%)	Vis. (33.3%)
Acc. (20.0%)	P/R (18.6%)	P/R (20.8%)	Acc. (27.1%)	Acc. (29.8%)
Feat. Contrib. (17.1%)	Vis. (15.7%)	Case (13.2%)	Case (14.6%)	P/R (17.5%)
Vis. (14.3%)	Correlation (11.4%)	DCG (9.4%)	Human Eval. (8.3%)	Case (12.3%)

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

- **Explainable models.** Many domains have yet to embrace deep learning due to their needs for explainable models. The system should provide ample support to help developer interpret model behaviors.
- **Fast model training.** The fact that the most tuned model parameters are related to training time suggests that developers are in need of systems that have fast model training, but also low latency for the end-to-end workflow execution in general.
- **Fine-grained results analysis.** Fine-grained and summary evaluation methods are equally popular across all domains. Thus, model management systems should provide support for not only summary metrics but also more detailed model characteristics.

We are in the process of developing a system, titled HELIX [17], that is aimed at accelerating iterations in human-in-the-loop ML workflow development, using many of the properties listed above as guiding principles.

4 CONCLUSION AND FUTURE WORK

We conduct a statistical study on the iterative development process for ML applications in multiple domains. Our approach involves collecting carefully designed statistics from applied machine learning literature in order to reconstruct the iterative process that led to the results reported. We present our survey findings across domains and discuss desired ML system properties as suggested by the trends discovered from our survey data. The statistics and estimators described in our work can be further developed into a benchmark for systems specifically designed to address human-in-the-loop ML needs.

REFERENCES

[1] 2017. Machine Learning: The New Proving Ground for Competitive Advantage. (2017). https://s3.amazonaws.com/files.technologyreview.com/whitepapers/MITTR_GoogleforWork_Survey.pdf

[2] 2017. Machine learning: the power and promise of computers that learn by example. (2017). https://royalsociety.org/~/media/policy/projects/machine-learning/publications/machine-learning-report.pdf

[3] 2017. The State of Data Science and Machine Learning. (2017). https://www.kaggle.com/surveys/2017

[4] Ashraf Abdul, Jo Vermeulen, Danding Wang, Brian Y Lim, and Mohan Kankanhalli. 2018. Trends and Trajectories for Explainable, Accountable and Intelligible Systems: An HCI Research Agenda. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 582.

[5] Ivan Bruha and A Famili. 2000. Postprocessing in machine learning and data mining. *ACM SIGKDD Explorations Newsletter* 2, 2, 110–114.

[6] Pedro Domingos. 2012. A few useful things to know about machine learning. *Commun. ACM* 55, 10 (2012), 78–87.

[7] M. I. Jordan and T. M. Mitchell. 2015. Machine learning: Trends, perspectives, and prospects. *Science* 349 (2015), 255–260. http://science.sciencemag.org/content/349/6245/255

[8] Mary Beth Kery, Amber Horvath, and Brad Myers. 2017. Variolite: Supporting Exploratory Programming by Data Scientists. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 1265–1276.

[9] John King and Roger Magoulas. 2016. Data science salary survey: tools, trends, what pays (and what doesn't) for data professionals. (2016).

[10] Laura M Koesten, Emilia Kacprzak, Jenifer FA Tennison, and Elena Simperl. 2017. The Trials and Tribulations of Working with Structured Data:-a Study on Information Seeking Behaviour. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 1277–1289.

[11] Yong Liu, Jorge Goncalves, Denzil Ferreira, Bei Xiao, Simo Hosio, and Vassilis Kostakos. 2014. CHI 1994-2013: mapping two decades of intellectual progress through co-word analysis. In *proceedings of the SIGCHI conference on human factors in computing systems*. ACM, 3553–3562.

[12] M Arthur Munson. 2012. A study on the importance of and time spent on different modeling steps. *ACM SIGKDD Explorations Newsletter* 13, 2 (2012), 65–71.

[13] Anton J Nederhof. 1985. Methods of coping with social desirability bias: A review. *European journal of social psychology* 15, 3 (1985), 263–280.

[14] Junfei Qiu, Qihui Wu, Guoru Ding, Yuhua Xu, and Shuo Feng. 2016. A survey of machine learning for big data processing. *EURASIP Journal on Advances in Signal Processing* 2016, 1 (2016), 67.

[15] Carlton E. Sapp. 2017. Preparing and Architecting for Machine Learning. (2017).

[16] Manasi Vartak, Pablo Ortiz, Kathryn Siegel, Harihar Subramanyam, Samuel Madden, and Matei Zaharia. 2015. Supporting fast iteration in model building. In *NIPS Workshop LearningSys*.

[17] Doris Xin et al. 2018. Helix: Holistic Optimization for Accelerating Iterative Machine Learning. *Technical Report http://data-people.cs.illinois.edu/helix-tr.pdf* (2018).

[18] Martin Zinkevich. 2017. Rules of Machine Learning: Best Practices for ML Engineering. (2017).