Working Paper 23-029

Nailing Prediction: Experimental Evidence on the Value of Tools in Predictive Model Development

Daniel Yue
Paul Hamilton
Iavor Bojinov

Nailing Prediction: Experimental Evidence on the Value of Tools in Predictive Model Development

Daniel Yue
Harvard Business School

Paul Hamilton
Harvard Business School

Iavor Bojinov
Harvard Business School

Working Paper 23-029

Funding for this research was provided in part by Harvard Business School.

Nailing Prediction: Experimental Evidence on the Impact of Tools in Predictive Model Development

Daniel Yue Paul Hamilton Iavor Bojinov

April 28, 2023

Abstract

Predictive model development is understudied despite its centrality in modern artificial intelligence and machine learning business applications. Although prior discussions highlight advances in methods (along the dimensions of data, computing power, and algorithms) as the primary driver of model quality, the tools that implement those methods have been neglected. In a field experiment leveraging a predictive data science contest, we study the impact of tools by restricting access to software libraries for machine learning models. By only allowing access to these libraries in our control group, we find that teams with unrestricted access perform 30% better in log-loss error — a statistically and economically significant amount, equivalent to a 10-fold increase in the training data set size. We further find that teams with high general data-science skills are less affected by the intervention. In contrast, teams with high tool-specific skills significantly benefit from access to modeling libraries. Our findings are consistent with a mechanism we call 'Tools-as-Skill,' where tools automate and abstract some general data science skills but, in doing so, create the need for new tool-specific skills.

Keywords: Economics of AI, Predictive Model Development, Skills, Tools

1 Introduction

Recent breakthroughs in artificial intelligence[1] (henceforth predictive modeling) have changed how businesses operate. Most visibly, predictive models are at the heart of new technologies like the optimized search and recommendation systems developed by Google, Amazon, and Netflix. But, as illustrated in a rich case literature, predictive models have also enabled operating model innovations for businesses in traditionally non-digital industries like finance, manufacturing, and retail (Iansiti & Lakhani, 2020; Greenstein, Myers, & Mehta, 2022; Ferreira et al., 2016). However, these celebrated applications may be more of an exception than the rule. Recent surveys of business executives suggest that, while almost all firms are investing in some form of data analytics or AI capabilities, less than a quarter deploy these technologies in production (Ransbotham et al., 2020; New Vantage Partners, 2022). Why are some businesses better than others at developing predictive models at scale, especially given that the underlying methods are broadly known and accessible?

The literature on information technology (IT) productivity has shown that across an enormous range of applications[2], new technologies only realize their full value when they are accompanied by complementary, co-invented factors (Bresnahan et al., 1996). The recent literature especially emphasizes the role of *worker skills* in driving the adoption and effective deployment of new technologies like Hadoop – especially as transmitted through the channel of inter-firm hiring networks (Tambe, 2014; Tambe & Hitt, 2014; Wu et al., 2018). More specific to AI, recent literature has emphasize the role of *data* as a key complementary factor to algorithm development (Iansiti & Lakhani, 2020; Bessen et al., 2022; Gregory et al., 2021; Varian, 2018).

However, there is a gap in the literature in explaining a key concern in improving predictive model development: internal technological infrastructure, or *tools*[3]. Yet practitioner emphasis on the value of tools is widespread: almost every leading company invests significantly in their internal data platforms, including centralized data warehouses, CRMs, customer support systems, reporting dashboards, and more ‘internal’ tools that facilitate data science and other operations (Budibund, 2022; Spotify, 2021). Due to the importance of tools, consulting firms are routinely employed to reorganize a firm’s internal data and

[1]By artificial intelligence (AI), we refer to applications of machine learning (ML) or “Weak AI” rather than the broader AI problem of constructing a general intelligence machine (Iansiti & Lakhani, 2020).

[2]For example, researchers have shown that the adoption of IT only increases productivity and profits when accompanied by an appropriate shift in complementary changes to the organization, such as team-based work organization, distributed decision-making authority, or specific skills in the labor force (Bresnahan et al., 2002; Wu et al., 2019; Brynjolfsson et al., 2021)

[3]A key component of our argument will be to introduce a conceptual distinction between tools and methods. However, for completeness, defer introducing our definitions until Section 2.2.

reporting infrastructure (McKinsey, 2022; BCG, 2023), and integrating open-source tools is seen as a key first step in developing AI capabilities (Greenstein, Yue, et al., 2022). However, tools, even free open-sourced ones, are unevenly adopted because adoption requires firms to make costly infrastructure investments to integrate them into their technology stacks (BCG, 2021). Furthermore, some businesses restrict the use of these core tools for regulatory or security purposes (Amgen, 2022) or because they are simply unsure how to apply them to drive business value. Taken together, the case studies and anecdotal evidence suggest that tools may help explain why some businesses are better at developing predictive models at scale. Yet there is almost no prior quantitative evidence on the impact of tools on predictive model development or productivity, more generally.

To address this gap, we ran a field experiment. Specifically, we conducted a prediction competition based on a binary classification problem, described in Section 3. Our treatment restricted access to machine learning modeling functions in Python (such as those implemented by the popular package `sci-kit learn` (Pedregosa et al., 2011)); the control group had no such restrictions. Crucially, teams were blocked from using tools (software library implementations of methods) but not the methods themselves; therefore, our experiment isolates the causal effect of the tools rather than the methods (allowing methods to vary endogenously).

Our experimental results suggest two new reasons why firms may differ in the success of their predictive modeling efforts. First, firms may fail in predictive model development because of their restricted use of tools. Our main effect, presented in Section 4, shows that access to library implementations improves model accuracy by an average of 12 percentage points over baseline — corresponding to a 50 percent relative improvement over treatment relative to the baseline. This economically and statistically significant impact is equivalent to reducing the training data size by a factor of 10 (to 10% of its original size).

Second, firms may fail in predictive model development because their workforce lacks the necessary skills to leverage the tools. To show this, we collect data on the skill backgrounds of participating teams to study the mechanism behind our main effect. In our analysis, we distinguish between specific and general skills — where specific skills denote prior experience with related tools and methods, and general skills denote broad knowledge of statistics and computer science. Our results show that these two types of skills interact differently with the availability of tools: specific skills are complementary to software libraries, whereas general skills are substitutable. This is consistent with a mechanism we call 'Tools-as-Skill,' where tools automate and abstract some general data science skills but, in doing so, create the need for new tool-specific skills. We detail this mechanism and rule out alternatives in Section 5.

To the best of our knowledge, this is the first paper in the management literature to conceptualize or empirically study predictive model development. Beyond the novelty of our research question, conceptual framework, and experimental approach, our results contribute to the literature by showing how tools can lead to dramatic differences in predictive model development in firms. First, we show that access to appropriate tools dramatically affects the model quality— at a magnitude comparable to collecting a data set that is ten times as large. Second, we show that tools change the skills needed for a team to successfully develop predictive models. Although general data science skills are necessary to solve prediction problems, tools automate some of these general skills, enabling teams to focus on the specific skills needed to apply the tools. Thus, by investing and integrating the proper tools, firms can intentionally change the amount and type of skills needed for their data scientists to be effective.

Our paper proceeds as follows. We first contextualize our study by reviewing the literature on AI in firms and the drivers of predictive model development, introducing the testable hypotheses that guide our subsequent study. We then present our research design and core results. Third, we show that our main effect is best explained by a mechanism where tools (in the form of software libraries) acts as a substitute for general data science problem-solving skills. Fourth, we extend this model and our analysis to explain why the availability of tools causes shifts in teams' problem-solving approaches to the contest problem. Finally, we conclude by framing our contributions to the literature on IT Productivity and the Economics of AI, as well as highlighting key managerial implications of our results.

2 Literature Review

In this section, we first contextualize predictive model development by reviewing the literature on the economics of AI, and second introduce our core conceptual framework.

2.1 The Economics of AI and AI Production: Contextualizing Predictive Model Development

The recent explosion in interest in AI[4] ignited a large, multidisciplinary literature focused on the economic impact of AI. From a macroeconomic perspective, AI is synonymous with *automation* — the impact of AI can be thought of as the automation of specific tasks, and the subsequent creation of newer, higher-value tasks for humans to take on (Acemoglu & Restrepo, 2018) Yet this perspective assumes an automation production function, with scarce evidence microfounding how automation of older tasks or creation of new tasks works in general. From a more microeconomic perspective, the literature has made strong headway on demonstrating how ML can be applied to specific business problems (Ferreira et al., 2022; Senoner et al., 2022) and illustrating the broader systematic impact of AI technologies (Babina et al., 2021; Rock, 2021). However, this literature does not develop a perspective on the factors that may help firms better develop and deploy predictive models.

Both perspectives motivate the need to develop work focusing on *AI production* that explains variation in a firm's ability to successfully develop and apply machine learning technologies to create value. Such work would contribute to our understanding of the macroeconomics of AI by identifying AI production with the automation production function, the microeconomics of AI by explaining how AI drives firm performance and competitive advantage, and the operations of AI by providing guidance on how to develop, deploy, and integrate AI into a firm's operations. We conceive of AI production as a five-step process wherein firms have to:

1. Find valuable problems to solve; we call this **problem selection**. A key observation from this literature is that the most economically relevant applications of AI in business are based on solving prediction problems, especially supervised learning problems[5] (Agrawal et al., 2018; Iansiti & Lakhani, 2020; Greenstein, Myers, & Mehta, 2022; Snapdocs, 2022).

2. Solve the selected problem by developing a model; we call this **predictive model**

[4]Over the past five years, there has been exponential growth in the use of data science and AI/ML in businesses. The 2021 Stanford AI Index reported that since 2016, the number of AI job postings has doubled and that corporate investment in AI has tripled (Zhang et al., 2021). This matches the trend reported on by consulting and government organizations. In a 2021 global survey of AI adoption, McKinsey found that 56% of responding firms employed AI in at least one function, a 50% growth compared to the result from 2020 (McKinsey, 2021). Similarly, PwC reported in 2021 that a quarter of respondent firms reported widespread AI adoption, and fifty-two percent of respondents stated they had accelerated their plans for AI adoption due to the COVID-19 crisis (PricewaterhouseCoopers, 2022). The Bureau of Labor and Statistics projects a growth of 31.5% in data science employment from 2020 to 2030 (BLS, 2022).

[5]Supervised learning is a subset of AI/ML problems focused on predicting a label/outcome variable from covariates after training a model on pairings of labels and covariates.

development[6]. The literature here is underdeveloped, and is our core focus in this paper.

3. Model evaluation on real users through **experimentation**[7] (Bojinov & Gupta, 2022; Kohavi & Thomke, 2017; Thomke, 1998, 2020). The growing literature on business experimentation has primarily focused on the general use of techniques like "A/B testing" to measure the impact of an innovation and have shown that there is a measurable positive impact on business metrics and performance (Koning et al., 2022; Mao & Bojinov, 2021).

4. Integrate the solution into firm operations; we call this **solution deployment**. This literature covers issues related to technical constraints ((Huyen, 2022)), ethical constraints (Mattu et al., 2022; Hidalgo et al., 2021), and behavioral constraints (Hoffman et al., 2018; Doshi-Velez & Kim, 2017).

5. After deployment, companies need to monitor the model for drift, bias, or other performance depredations and unintended consequences (Luca et al., 2016). This literature has primarily been developed by practitioners and focuses on the technical implications (Oladele, 2021).

We summarize our framing of the literature on the Economics of AI and AI Production in Figure 1, which is elaborated on in our subsequent discussion. To the best of our knowledge, our paper is one of the first to directly frame AI production as an economic activity of interest or directly study predictive model development[8].

2.2 Predictive Model Development

Having contextualized predictive model development in the broader literature, we now develop our conceptual framework and testable hypotheses. Although the predictive model development process has not been directly studied from an academic perspective, a number of other works provide a starting point for identifying the factors that drive the process and are described in Table 1.

[6]Most applications of AI involve predictive modeling, and so we exclusively focus on predictive model development (Agrawal et al., 2018). We acknowledge that generative models may create value in business applications, but they falls outside of the scope of the categorization presented here.

[7]We distinguish between model evaluation on real users and test-holdout split. The latter is a standard technique for assessing the model's fit and is generally part of predictive model development—it also provides little information on the real-world impact of using the model.

[8]Cowgill et al. (2020) use a prediction contest to study how engineers' demographics and training data quality affect the occurrence of errors in predictive models, but do not explicitly consider the overall quality of the models being developed.

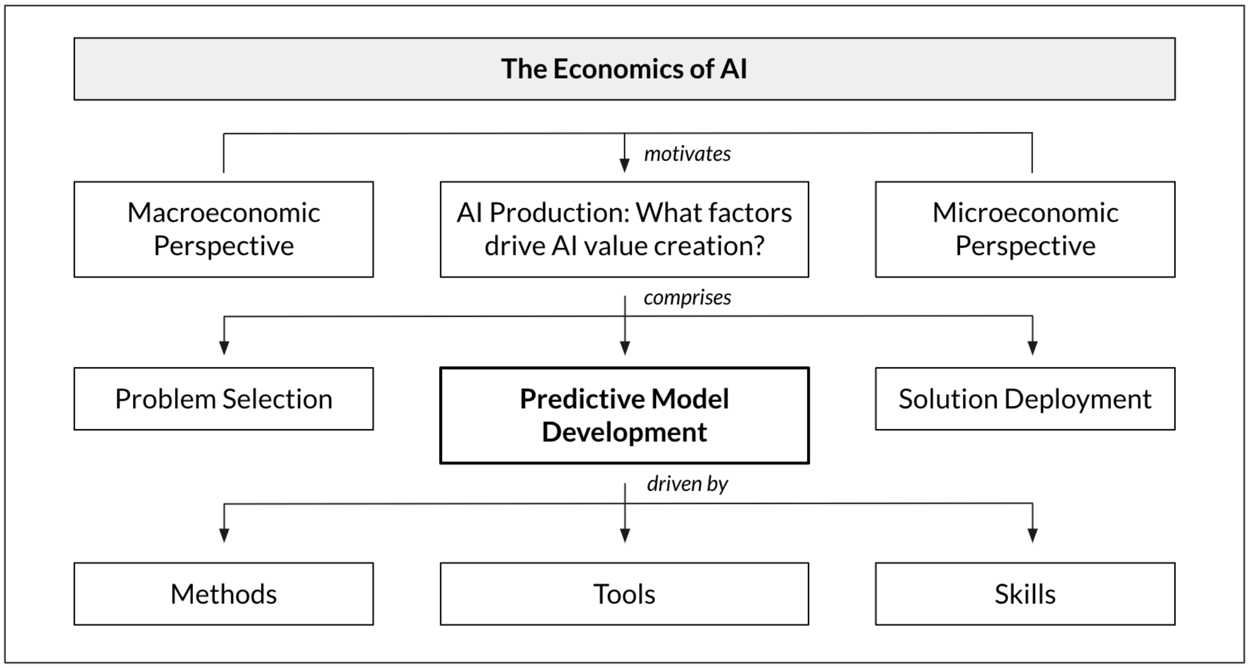

The Economics of AI
motivates
Macroeconomic Perspective
AI Production: What factors drive AI value creation?
Microeconomic Perspective
comprises
Problem Selection
Predictive Model Development
Solution Deployment
driven by
Methods
Tools
Skills

Figure 1: An outline of our literature review on the economics of AI, to help contextualize our paper. In this figure, each row outlines a more granular breakdown of the sub-components of the parent literature (node) above that row. Our paper frames the problem of *predictive model development* and describes the key factors that drive the process and their relationship. Understanding Predictive Model Development serves to deepen our understanding of the the drivers of *AI production* in firms more generally.

The starting point for Table 1 is a standard taxonomy of drivers of predictive model development (see, for example, Iansiti & Lakhani (2020)), listed along the columns. These include data (how an empirical reality is represented), compute (how computational resources are structured and employed), and algorithms (how algorithms and models are structured to capture empirical regularities in the data). While this taxonomy is helpful as a way to remember the necessary ML system components needed for developing algorithms, we find it less conceptually useful as an analytic tool to explain the core barriers to deploying AI within companies.

To develop a more useful conceptual tool, we apply the distinction of technology and associated worker skills, taken from the literature on IT productivity (Tambe, 2014; Tambe & Hitt, 2014; Wu et al., 2018), listing these concepts along the rows of Table 1. Our main innovation is to further separate technology into *methods* and *tools* and distinguish skills into *general* and *specific* skills. To illustrate that the methods-tools-skills framework is useful and orthogonal to the data-compute-models framework, we provide examples familiar to working data scientists at the intersection of each of these concepts. Further, as will be shown, by applying and extending this methods-tools-skills framework from IT productivity to the

	Data	**Compute**	**Models**
	How an empirical reality is represented.	*How computational resources are structured an employed*	*How models are structured so as to capture empirical regularities in the data.*
Methodology	Split Apply Combine One-hot encoding More Observations	Backpropagation / AutoDiff Accelerators (GPUs / TPUs) Infrastructure-as-a-Service	Random Forest BERT (Language Models) AutoML
Tools	R's `dplyr` Python's `pandas` PostgreSQL	PyTorch NVIDIA's CUDA Google Cloud Platform	*Sci-kit Learn* *Hugging Face Transformers* *AWS SageMaker AutoPilot*
Skill	Domain-Specific Skill	Computer-Specific Skill	*Modeling-Specific Skill*
	General Skill		

Table 1: The Drivers of Predictive Model Development. The tables provides examples of the various drivers of predictive model development, organized by data, compute, and models (columns) along the dimensions of methods, tools, and skills (rows).

problem of predictive model development, we are able to develop testable hypotheses that guide the development of our experiment.

2.2.1 Technology as Methods and Tools

Our first extension to the technology-skills framework is to distinguish technology into two component parts: *methods* and *tools*. In the context of predictive model development, we define *methods* as abstract, conceptual knowledge of how to solve a predictive problem, distinct from *tools* as specific implementations and integrations of known methods. For example, a data method refers to the existence of a data set that describes an empirical setting we wish to study; a computing method refers to the existence of efficient computational techniques that could be applied on the data to store, transform, and clean it at scale; an algorithmic method refers to the existence of models and algorithms that could take the stored data and generate suitable predictions.

One key finding from the literature on methods is that advances along each dimension, holding the others fixed, exhibits diminishing marginal returns. Consider the example of data set size: as a general rule from statistics, model accuracy typically improves with the inverse of the square root of the sample size (Iansiti, 2021). This has been demonstrated across a wide range of model architectures, including decision trees (Frey & Fisher, 1999), support vector machines (Juntu et al., 2010), convolutional neural networks (Cho et al., 2016), and architectures specialized to natural language processing tasks (Banko & Brill, 2001). Similarly, even though neural networks were developed decades ago (Rumelhart et al., 1986), breakthroughs in deep learning appeared only when there was sufficient computing power available to train models with a million times more parameters than what was previously possible (Sevilla & Villalobos, 2022).

A second finding is that methods generally apply across use cases and are becoming standardized. For example, consider the recent shift in computing norms, where companies have largely moved away from on-premise computing centers towards using commoditized cloud computing for hosting and training machine learning models (LinkedIn, 2022). Furthermore, most practical applications of machine learning revolve around a small set of *foundation models* (either ensemble or neural-net-based models), leading to further computing specialization through the creation of chips specifically designed for running the backpropagation algorithm (Bommasani et al., 2022).

Methods are important to predictive model development (as anyone developing these models necessarily interacts with each of these methods), and identifying them as the central factors of AI production is useful for management theory and practice. From a theoretical perspective, one implication is that differential access to these methods is regarded as a source of competitive advantage when evaluating firm AI strategies. For example, in 2012, Baidu, Google, Microsoft, and DeepMind famously competed in an auction to hire Geoff Hinton and his two Ph.D. students, resulting in an acquisition cost of $44MM paid by Google. The acquisition was motivated by the belief that deep-learning methods would provide a competitive advantage to the firm (Metz, 2021). More systematically, Bessen et al. (2022) surveyed AI startups and found a positive correlation between venture capital funding and access to proprietary training data. Additionally, the authors found a similar relationship between funding and using more sophisticated ML algorithms. From a practical perspective, some firms pursue strategies to develop new proprietary methodologies. For example, beyond acquisitions, Google, Meta, and Microsoft invest heavily in developing novel algorithmic methods (Hartmann & Henkel, 2020; CSET, 2022).

However, while the focus on methods is important, the *lack* of research examining other factors might lead us to conclude that differential access to methods is the *only* explanation of variation in AI production across firms. Theoretically, this limits our understanding of firm performance in the digital age. Practically, this may lead firms to conclude that there is only a limited number of options for improving model accuracy — and feel stuck if traditional approaches like increasing data set size or changing the modeling algorithm fail to deliver better performance. While there have been calls for attention to other levers to improve models, they tend to still focus on method changes such as building better data tools (DeepLearningAI, 2021). This is reasonable but not enough; we argue that other further factors beyond methods drive predictive model development and should get greater attention.

We propose that *tools*, defined as the hardware and software used to implement and integrate specific methods, are an important factor that impacts predictive model development. Examples of such tools include database management software like MS SQL Server, Cloud Computing services like Amazon Elastic Compute Cloud (EC2), and software libraries for building machine learning algorithms like `scikit-learn` or `pytorch`.

While these tools have not previously been proposed as drivers of predictive model development, we know that only a small set of AI tools are leveraged across a broad set of AI applications (LinkedIn, 2022; Bommasani et al., 2022). One reason for this follows directly from the aforementioned commoditization of methods: as the methods are standardized, the tools that implement them are also standardized. For example, since almost all NLP or Computer Vision models are converging on the use of deep neural networks, almost all AI researchers implement their models using either higher-level neural-network oriented libraries (like PyTorch or TensorFlow) or lower-level scientific computing libraries (like Jax), because of their superiority in automatically differentiating through arbitrary model architectures and in allocating computational resources to different types of accelerators (e.g. GPUs, TPUs). A second reason for the concentrated usage of a small set of tools is the prevalence of open-source technologies in the ML space. Because of the advantages of open-source in its transparency, community resources, and free use, almost all ML applications use open-source implementations of methods either directly or indirectly (through an ML product that provides a wrapper of open-source implementations). A final reason is that AI human capital is indirectly distributed across companies in the economy, so smaller firms must rely on the tools developed by larger firms to take advantage of new developments.

Beyond its ubiquity, we expect tools to have a positive impact on predictive model development because they codify and abstract away a tremendous amount of methodological and technical knowledge, lowering the cost of discovering and applying any given method. For example, while many data scientists may have learned the *theory* of maximum likelihood estimation or variational inference, few have actually written software that solves the optimization problems associated with those techniques. Our observation is that in many cases, if there is no library implementation of a new method, then most data scientists simply will not use it. This motivates the central hypothesis of our paper: that tools are at least as important as advances in methods in determining predictive model development.

Hypothesis 1 *Access to tools (such as a new software library) improves the quality of predictive models. Moreover, its effect will be on the same order of magnitude as that of increases to gains to other key factors like data set size.*

If tools are so vital, why have technical factors been overlooked in practitioner discussions of predictive model development? The core argument against the explanatory value of this factor is its availability: if everyone has access to the same tools, then there is no variation, so it cannot explain predictive model development outcomes. If there *are* differences in tools due to ignorance, then these differences should be short-lived and should disappear in the medium term as firms adjust to the new tools. For example, cloud computing is equally available on the market, and eventually, most firms will shift over to cloud-based computing architectures.

We disagree with this argument for three reasons. First, not all AI tools are available on the market or via open-source technology. The large technology companies compete over the top AI researchers not only to have access to novel methods but also to turn those methods into tools that can drive usage of those methods privately within the firm before they become publicly accessible; indeed, this was the origin of TensorFlow (Abadi et al., 2016). Second, even given the availability of tools, some firms restrict the usage of certain tools due to technical, business, or ethical reasons or simply because they are unaware of their existence. Finally, although many tools are open source, it takes special skills to use these tools. Firms must commit to integrating these tools into their technology stack and may have variations in their ability or willingness to do so. For all three of these reasons, we expect considerable variation in access to AI technology, motivating its study as a factor driving predictive model development.

2.2.2 Skills as General or Specific

Our second extension to the methods-tools-skills framework is to nuance the concept of skills with guidance from the literature on skill-biased technical change. By skills, we refer to the knowledge needed to execute aspects of predictive model development, including knowledge of specific software, familiarity with modeling tools, statistical knowledge, quantitative analysis skills, and domain expertise (Smaldone et al., 2022). The foundational elements of these skills are regularly taught in standard statistics and data science curricula across universities.

To motivate our theoretical analysis of skills as a driver of predictive model development, we note that our preceding argument regarding the effect of tools on predictive model development suggests two possible but opposing mechanisms for how tools and skills may interact. On the one hand, we argued that tools codify and abstract skills — suggesting that tools and skills are substitutable. On the other hand, special skills (including methodological and technological knowledge) are needed to effectively use the tools in the first place and may accelerate problem-solving with tools — suggesting that tools and skills are complements. Is skill a complement or substitute to tools in this setting?

The answer depends on the *type* of skill in question. For instance, the literature on skill-biased technical change (SBTC) (Chari & Hopenhayn, 1991) provides a useful theoretical distinction based on whether the skill can be codified. This is because tools automate low-level tasks, enabling greater focus on high-level problem-solving. This argument was originally developed when applying SBTC to the study of IT productivity; for example, Autor et al. (2003) proposed that computer capital serves as a substitute for workers performing common (general) tasks that follow explicit rules but complements workers performing non-routine problem-solving. Similarly, Bresnahan et al. (2002) summarize evidence on the positive correlation between IT use and problem-solving skill at the worker, firm, and industry level. Borrowing this conceptual framework, we call these higher-level skills needed to solve problems *tool-specific* (or just *specific*) skills — the skills needed to apply tools to focal problems. In contrast, we call the skills that are able to be codified in our setting *general-task* (or just *general*) skills.

How can we apply this concept of skill types to our setting of predictive model development? One key challenge is that, in the broader SBTC literature, general skills are those common to the broader population (e.g., data input, summation, reporting), and specific skills are those associated with knowledge of IT tools (namely, the ability to use computers) and problem-solving skills more generally. By contrast, in our predictive model development setting, the skills of interest are typically known only by those with knowledge of data science and are thus fairly rare across the broader population. Indeed, a naive application of the specific/general skill distinction to predictive model development might imply that all data science problem-solving is specific; that is, the data problem-solving process cannot be automated. Thus, we should conclude that skills are inherently complementary to tools.

However, a closer look at the typical data scientist's workflow reveals that there is a more insightful way to apply the concept. We observe that data scientists do not typically engage in the implementation details of models in their problem-solving process but rather reason conceptually about the problem and implement it by manipulating tools through their designed interfaces. Thus, we hypothesize that, in this setting, general skills are statistical knowledge (e.g., knowledge of probability theory, ability to reason about substantive data, ability to wrangle data) and computer science knowledge (e.g., efficiently using computational resources, writing bug-free code), as these are abstracted over by tools. Similarly, we hypothesize that specific skills are experience with or knowledge of the interfaces of specific tools (e.g., cloud computing and modeling libraries) used to solve predictive problems. With this application of general and specific skills, we can hypothesize an answer to our motivating question.

Hypothesis 2 *In predictive model development, statistical and computer science knowledge are general skills; that is, they substitute with access to tools that automate them. By contrast, experience with various tool interfaces is a specific skill; that is, it is complementary to those* *tools.*

To summarize our preceding discussion, we developed a framework of the factors driving predictive model development. In particular, we distinguished between *methods* and *tools* and argued that tools used to implement methodologies have been neglected in prior discussions of predictive model development. Further, we distinguish between *general* and *specific* skills, identified what these are in the context of predictive model development, and argued that they have different complementary behavior with respect to tools. While there may be more factors that drive predictive model development[9], we leave their consideration to future work and turn instead to an empirical examination of our hypotheses.

3 Research Design

3.1 The Setup

In the ideal experimental setting to estimate the impact of the drivers of predictive model development, data scientists would independently work on the same problem under exogenously imposed resource constraints and training, leading to variation in final model performance. To approximate this ideal experiment, we designed and ran a prediction contest, similar to public competitions run on Kaggle or TopCoder[10]. In this section, we focus on communicating the facts of our contest design, and delay describing the benefits of this design (in terms of measurement, external, and internal validity) until Section 3.4. Full details on the implementation of the contest are given in Appendix A.

For our competition, we chose a binary classification problem focused on predicting the operational status of water pumps in Tanzania based on data on their installation context[11,12]. Specifically, participants were tasked with predicting whether a pump's operational

[9]For example, work in IT productivity has shown that for organizations to derive value from IT products, they must undergo an extensive redesign of business processes to adapt to new technologies (Broadbent et al., 1999). However, our paper does not consider complementary business processes in this paper, although we acknowledge that they are likely important in this context.

[10]Data science competitions have been leveraged in prior literature to study questions about open innovation and contest design (Jeppesen & Lakhani, 2010; Boudreau & Lakhani, 2013; Lemus & Marshall, 2021). Our idea here is to use these competitions as a way to study predictive model development, rather than studying contests as a focal phenomenon in itself.

[11]We adapted this problem from a problem provided by DrivenData, a non-profit that coordinates data science competitions focused on problems with high social impact. Despite this problem being used for a public competition before our contest, none of our participants had seen the problem before. Furthermore, we took steps to nullify any benefit of prior experience on the original DrivenData problem by converting the problem from multivariate classification to binary and de-identifying the pump data by removing ids and injecting small amounts of statistical noise into possible identifying data like latitudes and longitudes.

[12]Beyond being a realistic problem, we chose this problem because it was inherently socially impactful.

status (a "label") given covariates including the date of a report, the pump's installation details (installation year, original funder, installing organization), the management (managing organization, payment structure), and location details (water source, region, lat/long, altitude, local population). As is typical in supervised prediction contests, participants were given a "training" data set of covariates and labels, as well as a "test" data set of covariates without labels. During a fixed competition timeframe, they developed algorithms to predict the operational status (i.e., the likelihood of needing repair) of the test data. At the end of the contest timeframe, the contest rankings were determined based on each team's prediction accuracy on the test data.

The contest was implemented as a remote 48-hour datathon[13] over a weekend in Winter 2022[14]. We recruited contest teams of one or two participants from leading USA universities. We marketed our contest through emails on the primary mailing lists managed by each university's statistics and data science departments, as well as specific club mailing lists when possible. We clearly stated the prior knowledge requirements for the contest, which helped to ensure that our final participant pool was qualified to work on the problem. Our experiment had participants from over 20+ distinct universities, including MIT, CMU, Harvard, Columbia, JHU, University of Washington, and many other leading data science schools in the USA. Students spanned undergraduate and postgraduate programs, as well as several recent graduates.

We incentivize participation by offering prizes for accurate models: \$1000 for first place, and monetary prizes of decreasing value for those finishing within the top-10 (see Table A1). Due to the need to generate treatment variation, we implemented two separate tracks of the contest (one for our test group and one for our control), awarding prizes for each track separately.

Contest participants began their work by filling out a pre-contest Qualtrics survey, after which they were given access to our contest Colab[15] notebook and details of the contest problem. We required participants to work entirely within the provided Colab notebook during the contest, allowing us to enforce rules compliance and collect detailed measures of code development throughout the contest. In addition, contest participants could submit intermediate predictions on our private online Kaggle contest and receive intermediate feedback based on 10% of the available data. We also collected data with a post-contest survey.

This made the contest problem intrinsically interesting for participants, leading to them putting more effort into the contest. Many contest participants commented through emails and our post-contest survey on how they enjoyed the specific choice of problem.

[13]"Datathon" is a typical term used to describe prediction contests of the type that we implement here.

[14]Our experimental design was reviewed and approved by an Institutional Review Board (IRB21-1421).

[15]Colab notebooks are Google's online version of a Jupyter Notebook.

See Figure 2 for a summary of the recruitment and contest participation pipeline.

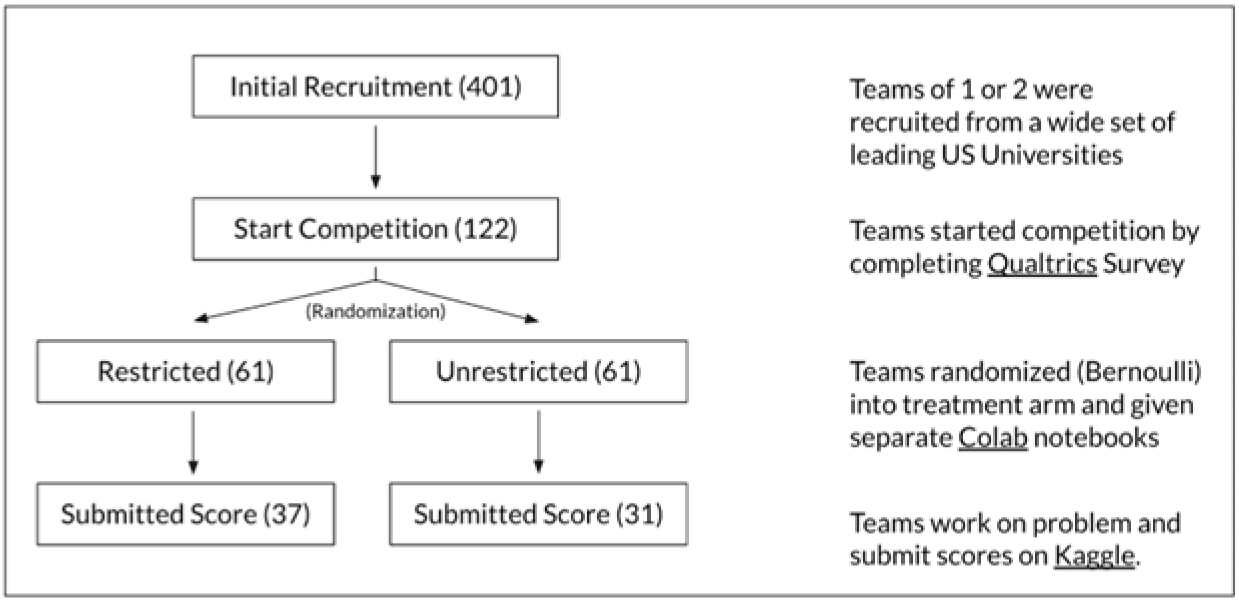

Initial Recruitment (401)
Start Competition (122)
(Randomization)
Restricted (61)
Unrestricted (61)
Submitted Score (37)
Submitted Score (31)
Teams of 1 or 2 were recruited from a wide set of leading US Universities
Teams started competition by completing Qualtrics Survey
Teams randomized (Bernoulli) into treatment arm and given separate Colab notebooks
Teams work on problem and submit scores on Kaggle.

Figure 2: The prediction contest recruitment and participation pipeline.

3.2 The Treatment

Our treatment restricted half of our participants from using machine learning modeling functions from software libraries[16]. Our rule specifically stated: "the list of banned modeling functions includes (but is not limited to): linear/quadratic discriminant analysis, support vector machines, k-nearest neighbors, naïve Bayes, CART/decision trees, random forest, gradient boosted trees, and neural networks." However, treatment participants could use any library model functions from the family of generalized linear models (GLM), techniques like regularization, and any functions in software libraries associated with non-modeling tasks, such as exploratory data analysis or feature engineering. Beyond standard GLM approaches, treatment participants were also free to implement algorithms manually, using more low-level scientific computing libraries like `scipy` (Virtanen et al., 2020) and `numpy` (Harris et al., 2020). We enforced compliance with this restriction through automated scanning of participant notebook code, looking for banned libraries and functions. We also did manual spot-checking for our contest winners. Participants in either track were aware that the other track existed but were unaware of the specific constraints faced by the other track.

The comprehensiveness of the treatment eliminates the possibility of teams substituting between specific tools implementing the same methods, allowing us to clearly interpret our

[16]By contrast, control participants were free to use any publicly available software library as they saw fit.

treatment effect as the impact of restricting access to advanced machine learning modeling tools. Our treatment also removes noise arising from idiosyncrasies in participant education or experience. A less comprehensive ban may have arbitrarily advantaged or disadvantaged participants based on the specific tools that they may be familiar with. For example, banning a specific implementation of neural networks would only disadvantage participants that are familiar with that tool but not those familiar with another implementation. Second, the comprehensiveness of our treatment eliminates the possibility of participants *learning* a new tool in response to having a particular tool banned. Thus, our treatment emphasizes skills participants already have versus skills they can learn on the fly – improving the interpretability of our skill measures. Lastly, our treatment suggests a clear alternative path forward to solve the problem: working more carefully with the provided data[17]. In that sense, treatment changes the relevance of the type of skills that participants have, increasing the relevance of general data-analysis skills and decreasing the importance of tool-specific skills. We explore these effects in our subsequent analysis.

Our design is specifically optimized to estimate the effect of access to software libraries on model quality. We do this because software libraries are a quintessential example of a tools — their existence often follows the creation of a methods but is independent of the methods's existence. Although we present estimates of the association of other factors (such as time available or team skills) with model quality, we leave the careful causal study of other factors driving model quality to future work.

3.3 Measurement

3.3.1 Data Sources

As discussed in Section 3.1, we collect several sources of data, including pre- and post-contest Qualtrics survey, contest Colab notebook, and submissions to our online Kaggle competition. Each of the data sources provides insight into a different aspect of the contest problem-solving process.

The Kaggle data is perhaps the most straightforward to understand: it provides us with an objective measure of performance. In particular, we collect the log-loss score[18] and time-stamp of each submission by participating teams, where teams can submit many times throughout the contest — this score forms the basis of our primary outcome variable.

[17]Although participants were free to manually implement algorithms, we did not observe participants do this in practice. This is perhaps illustrative of the value of tools over methodologies.

[18]Log-loss is a standard error metric used for binary classification models. See any stats reference for the formal definition of log-loss; for example, the Scikit-Learn documentation provides details on the specific way log-loss is calculated (Pedregosa et al., 2011).

We leverage our Qualtrics survey to ask questions related to participants' skills. For each participant, we collect standard educational information, including their school, program level (Undergrad / Master's / Ph.D.), graduation year, and major. We also collect information on *general data analysis* and *tool-specific* skill. For general data analysis, we measured the participants' skill working with data (e.g., through research experiences or SQL skills) and general programming ability (e.g., asking about prior software developer experience and coursework in operating systems). For tool-specific skill, we included measures of direct experience with tools and methods relevant to the contest: familiarity with tools like Scikit Learn and SciPy, as well as familiarity with techniques like regularization, feature engineering, or model building. To improve the usability of the survey, we ask participants to respond based on a three (None, One, Two or More) or four option (Never Used, Heard of, Used, Comfortable) scale, as we found this most closely matched pilot participants' understanding of their prior experiences. Finally, the time-stamp on the qualtrics survey corresponds to the moment that the participants started the contest. We use this information in our analysis of the effect of time on contest scores (see the following section). For full details on the survey, see Appendix B.

The Colab notebook provided detailed measures of team code development throughout the contest, which we used to explore the core mechanism behind our main results around tools. Specifically, Colab records a time-stamped "version" of teams' contest notebook every few minutes. The contest notebook was split into specific subsections related to the problem-solving process: Exploratory Data Analysis, Feature Engineering, Model Building, Model Evaluation, and Submission. Our chosen subsection labels match the way that prediction problem solving is taught in standard data science curricula and seem to fit quite naturally with the problem-solving approach taken during pilot tests of the contest. As a result, contest participants organized their code according to these sections as they worked; for example, participants would place visualizations of the raw data in the Exploratory Data Analysis section and put model estimates into the Model Building section. This effectively meant that participants labeled their work as they went, allowing us to reconstruct the time teams spent on different problem-solving phases and providing us with extremely granular data. In addition, because versions are created via frequent time intervals rather than participant control, this is a more granular measure of code development than prior measures used in the literature, such as from git repositories.

3.3.2 Primary Outcomes

Transformed Log-Loss Score (`Score`). We chose each team's best submission based on its associated log-loss score as evaluated on our (hidden to participants) holdout set. In order to increase the interpretability of this score measure, we transformed the score as follows. First, we calculated the "baseline" score obtained from naively predicting the average proportion of pumps that needed repairs for all test entries[19]. If a score was worse than the baseline, we replaced it with the baseline score[20]. The transformation ensures that variation in the results is driven by meaningful improvement in the algorithm, rather than an improvement on a meaningless margin of the problem. Second, we took the best score across both contest tracks and applied an affine transformation to all scores such that the best score has a value of 1.0, the benchmark score has a value of 0.0, and all other scores fall between 0.0 and 1.0. In this way, the score is interpretable as being "percent of possible improvement over baseline" while only linearly scaling the variation of the original log-loss measure.

Accuracy for Top 30% of Predictions (`Percent Accuracy`). Although the log-loss score is a standard objective function for binary classification models, it is harder to interpret improvement in log-loss in terms of its economic value. To complement log-loss, we thus compute an additional "percent accuracy" score, computed as follows. Recall that each team produced predictions for a fixed "test" sample (in our case, of size $N \approx 12000$). We take each team's predictions, order them by the model's predicted likelihood of needing repair, and filter to the top 30% of predictions (∼3600 predictions)[21]. We then compute the percent of those pumps that did need repair—according to the hidden true label—and call this *Percent Accuracy*. We conceptualize this variable as the percentage of pumps visited by a hypothetical repair engineer that actually needed repairs. We believe this process of computing percent accuracy mimics how classification models are developed and practically used by firms. Further, as we detail in Section 3.4.3, improvement in percent accuracy is proportional to improvement in business outcomes in many cases; therefore, changes in percent accuracy provide insights into the economic magnitude of our estimated effects.

3.3.3 Contest Measures

Treatment (`Unrestricted`). We define our treatment variable as having *unrestricted* access to software libraries during the contest. Note that this is the inverse of our "treatment" described above (which describes the specific restriction). We choose this definition so as to conceptually align our treatment with the other outcome variables (where 1 encodes *more* and 0 encodes *less*). When used in regressions, we center this variable (e.g. subtract the

[19]This is a standard benchmark for binary classification tasks, and represents the score one would receive if they had no additional information about each pump apart from the labels themselves.

[20]We did not exclude these below-baseline participants altogether because we expect that below-baseline performance is likely and meaningful in general business settings – in the sense that there are genuine prediction projects where teams do not make progress over baseline performance.

[21]Note that the choice of 30% is arbitrary, but that our results around *Percent Accuracy* are robust to changes to this choice.

mean of the variable) in order to improve the interpretability of our interaction effects (this allows us to measure interactions at the center rather than the 'edge' of our data). We follow this practice for all other dummy variables used in this analysis.
Number of Team Members (`Team Is Pair`). We encode the number of team members as a dummy variable: 1 if the team had two members, and 0 if the team was an individual.
Team Participation (`Participated`). For teams that were randomly allocated to treatment or control but did not end up submitting a score, we mark them as 0. For those that did submit a score, we mark them as 1. This measure is used to test for selective attrition and rule it out as an explanation of our results.
Time Spent On Contest (`Work Hours, Contest Hours`). We use the Colab Notebook data to form measures of the time each team spent working on the problem overall, as well as by subsection (e.g., Feature Engineering or Model Building). We also convert our version-level notebook data into team-level effort measures:

1. For each version, we compute the difference between that version and the prior version (a "notebook difference"). We then calculate the number of lines of code modified in each notebook section, as well as the overall time between notebook versions.

2. Each notebook difference is labeled as "active" if there was more than one change made. This allows us to exclude the time when the team was not actively working, but the Colab created a version. We mark notebook differences that are "inactive" as having zero overall time spent.

3. For each notebook difference, we attribute overall time to specific notebook subsections[22] by multiplying the overall time by the proportion of lines modified in a given section. For example, suppose we observed a notebook difference of 10 minutes where the team added two lines of Exploratory Data Analysis and three lines of Feature Engineering. In that case, we attribute $10(\frac{2}{5}) = 4$ minutes to the Exploratory Data Analysis and $10(\frac{3}{5}) = 6$ minutes to the Feature Engineering section.

4. We then sum across active notebook differences to create a measure of time spent by subsection. For interpretation purposes, we aggregate subsections into categories: feature engineering (comprising Exploratory Data Analysis and Feature Engineering) and modeling (comprising Model Building, Model Evaluation, and Submission).

5. Finally, for each team, we sum these two subsection measures to create a measure of the overall time spent by the team during the contest.

[22]We describe these subsections in greater detail above under Section 3.3.1"Data Sources".

Our proposed measure is more insightful than simply computing the difference between the start time of the contest and the timestamp of each notebook version because it captures *active work time*, not just elapsed time (where the participant may have been doing something other than working on the contest). We use this measure to develop tests to explore the mechanism behind our main effect.

3.3.4 Skill Measures

Individual Skill Measures (`Scikit-learn, Feature Engineering, SQL, Had Prior DS Job`). To operationalize our survey data, we convert each question's survey responses into binary (dummy) variables. For some variables like `Had Prior DS Job`, this was already the natural representation of the data. For responses with more than two ordinal values, we chose combine in order to maximize variance in the variable. For example, we asked participants their comfort level with the library Scikit-Learn. Participants could respond with one of four options: "Never heard of it", "Heard of it but unfamiliar", "Used / Done Before", "Comfortable." Because a clear majority (66%) were "Comfortable" with Scikit-Learn, we defined a new binary variable taking a value of "high" if the participant was "comfortable," and "low" otherwise. We followed this procedure for all of our variables from our survey. However, in our analysis, we choose highlight variables like comfort with feature engineering or SQL, and whether they had a data science job or internship before in order to illustrate the different *types* of skills that we measure through our survey (comfort with data science processes, programming languages, libraries, and work experience).
Skill Indices (`Total Skill, General Skill, Specific Skill`). Our interest in testing Hypothesis 2 means we need to combine our individual skill measures into an aggregate index. We define our skill index by aggregating our Qualtrics survey responses, as follows:

1. *Aggregate skills to team level.* For teams of two, we aggregate each participant skill measure into a team-level measure[23] because our primary outcome (`Score`) occurs at the team level. Our aggregation is based on a maximal approach; that is, we chose the max of the two team members' skill for each measure, where the max implies more relevance to the contest. For instance, if one had 'Heard of Scikit-Learn but unfamiliar', but the other was 'Comfortable' with it, then we marked the team as 'Comfortable'; and if a team had a STEM and Data Science major, we marked it as Data Science.

2. *Binarize individual skill measures.* Following the same procedure as taken with skill measures (above), we transformed questions with ordinal response into binary measures

[23]For teams of one, there is no ambiguity and we just directly use the one participant's skill as the team measure.

("high" or "low") so as to maximize variation. In practice, this meant turning the three-part measures into binary "yes" or "no", and the four-part measures into "comfortable" or "not comfortable". (See Appendix B for the details of the raw survey measures.)

3. *Remove statistically invalid measures.* We assessed whether each variable was statistically useful, according to two criteria. First, we eliminated variables with significant ex-post differences between our two treatment conditions (which only eliminates two variables). Then, we eliminated variables that had responses that had more than 75% of responses as the modal response. This is done to increase variation in our measure. For example, almost 80% of our participants had taken two or more statistics courses, so we discarded that variable from consideration due to low variation.

4. *Code variables by skill type.* We marked each variable as specific, general, irrelevant, or unknown, according to our definition provided in Section 2.2.2. Recall that we define *specific* skills as those related to knowledge of predictive model development process and specific tool-interfaces for solving those processes. We define *general* skills as those associated with more general statistical and computer science skills. *Irrelevant* skills are ones that may be general or specific, but do not have to do with the tools relevant to the problem we study in our experiment. *Unclear* means that the measure could be capture either/both specific and general skills, and are therefore hard to interpret given our theory. We did not include irrelevant or unclear variables in our aggregate skill measure due to their lack of theoretical interpretability. For example, we discarded our pytorch experience measure because it was a specific tool skill, but was unrelated to the specific problem at hand (pytorch is not generally useful on the tabular data that characterize our problem and was not used by any team).

5. *Aggregate into indices.* We sum across our specific and general skill dummies to form our `Specific Skill` and `General Skill` indices, and across both types of skills to form our `Total Skill` index. Our specific skill index aggregates whether the team majored in data science, whether they are comfortable with Scikit-learn, whether they are comfortable with SciPy, whether they are comfortable with linear regression, and whether they are comfortable with R. Our general skill measure aggregates prior employment as a data scientist, whether they were a STEM major, prior employment as a software developer, comfort with SQL, or significant prior coursework with Operating Systems.

6. *Convert to Binary Indicators.* We then convert our three indices into binary indicators by splitting it into high and low, choosing the median as the cutoff.

Variable	Count	Mean	Standard Deviation	Minimum	25th Quartile	Median	75th Quartile	Maximum
Score	68	0.55	0.33	0.00	0.32	0.61	0.81	1.00
Percent Accuracy	68	0.83	0.12	0.46	0.80	0.86	0.91	0.95
Unrestricted	68	0.46	0.50	0.00	0.00	0.00	1.00	1.00
Team is Pair	68	0.47	0.50	0.00	0.00	0.00	1.00	1.00
Work Hours	63	6.96	5.96	0.00	2.16	4.89	11.26	23.15
Contest Hours	68	42.54	9.64	4.72	43.71	46.77	47.58	47.93
Comfortable with sklearn	68	0.75	0.44	0.00	0.75	1.00	1.00	1.00
Comfortable with Feature Engineering	68	0.59	0.50	0.00	0.00	1.00	1.00	1.00
Comfortable with SQL	68	0.46	0.50	0.00	0.00	0.00	1.00	1.00
Had Prior Data Science Job	68	0.57	0.50	0.00	0.00	1.00	1.00	1.00
High Experience (Total)	68	0.49	0.50	0.00	0.00	0.00	1.00	1.00
High Experience (Specific)	68	0.57	0.50	0.00	0.00	1.00	1.00	1.00
High Experience (General)	68	0.40	0.49	0.00	0.00	0.00	1.00	1.00

Table 2: Summary Statistics on Participating Teams.

We summarize the results of this process in Appendix C, Table A2. Although we label our aggregate skill index measures based on the type of variables they include, we concede that they provide a one-dimensional, approximate measure of skill — an otherwise extremely high-dimensional concept. Nevertheless, we believe these provide a reasonable approximation of the different types of skills needed for predictive model development, and that variation in this measure captures meaningful variation in participant skills. Furthermore, while this procedure necessarily introduces a degree of arbitrariness in constructing the indices, we show that our key results based on this index measure are robust to alternative constructions in Appendix C.

We present team-level summary statistics for our primary measures in Table 2, ordered in the same sequence described above. All non-outcome variables are dummy variables encoded as 0 or 1, except for `Work Hours` and `Contest Hours`. 47% of our teams were individuals, while the rest were pairs. We include specific skill measures (such as `Scikit-learn` and `SQL`) to illustrate the data underlying our aggregated skill indices (`Total Skill, Specific Skill`).

3.4 Estimation

3.4.1 Estimating the Effect of Unrestricted Access to Libraries

We leverage ordinary least squares (OLS) regression at the team-level as our workhorse specification, although our results are robust to alternative analysis such as non-parametric and logistic regression. For attrition analysis, we estimate our model using the entire pool of participants assigned to treatment or control. For our main effect, to estimate the relationship between the quality of models produced by teams and the treatment of being restricted in the use of software libraries, we restrict our sample to teams that submitted scores. In either, the main model used is as follows:

$$Y_i = \alpha + \beta_1 \texttt{Unrestricted}_i + \beta_2 X_i + \epsilon_i \tag{1}$$

for outcome Y_i (`Participation, Score,` or `Percent Accuracy`), team i, and controls X_i such as our Skill Measures and Indices. To explore heterogeneity in our effect arising from participant skill, we add interaction terms associated with each of our skill indices:

$$Y_i = \alpha + \beta_1 \texttt{Unrestricted}_i + \beta_2 \texttt{High Skill}_i + \beta_3 \texttt{High Skill}_i \times \texttt{Unrestricted}_i + \epsilon_i \tag{2}$$

where $\texttt{High Skill}_i$ may be total, specific, or general skill. In our most complete specification, we include both general and specific skill, their interactions with treatment, and a three-way interaction between treatment, general, and specific skill. As noted above, when using dummy variables in these interactions, we use 'centered' versions of these variables to improve the interpretability of the interaction term.

3.4.2 Internal Validity of Estimates

The experimental setup allows us to interpret the difference between the two treatment conditions as the causal effect of software libraries on model building. Specifically, our design enables us to fix the contest problem and infrastructure while implicitly controlling for problem difficulty, data, and computational resources, three of the most challenging sources of endogeneity in observational settings. Note that in our heterogeneity analysis, our skill measures are endogenous, and therefore β_3 should only be interpreted as causal heterogeneity with respect to treatment, not the direct effect of skill.

Due to our uncertainty over the total experiment population before the contest, we assigned teams to treatment or control via Bernoulli randomization; that is, each unit was independently assigned to either treatment or control with equal probability. One danger of a Bernoulli assignment mechanism is the possibility of getting unbalanced numbers of tests and control in a way that reduces the statistical power; however, we ended up with roughly equal size treatment and control (62 vs. 60).

Attrition Analysis and Balance Checks A total of 122 teams took our pre-contest survey and were assigned to a contest track. In Table 3, we present a balance check for all signed-up teams. The results confirm that our randomization worked well, achieving balance across almost all of our covariates.

Track	**Overall**, N = 122	**Restricted**, N = 60	**Unrestricted**, N = 62	**p-value**
`Participated`	56% (68)	62% (37)	50% (31)	0.2
`Team is Pair`	35% (43)	37% (22)	34% (21)	0.7
`Contest Hour`	45.5 (41.5, 47.6)	47.0 (43.6, 47.6)	44.3 (40.1, 47.3)	0.028
`Comfortable with sklearn`	66% (80)	73% (44)	58% (36)	0.076
`Comfortable with Feature Engineering`	48% (59)	52% (31)	45% (28)	0.5
`Comfortable with SQL`	39% (48)	45% (27)	34% (21)	0.2
`Had Prior Data Science Job`	56% (68)	55% (33)	56% (35)	0.9
`High Skill (Total)`	52% (64)	57% (34)	48% (30)	0.4
`High Skill (Specific)`	54% (66)	62% (37)	47% (29)	0.10
`High Skill (General)`	39% (48)	37% (22)	42% (26)	0.6

Table 3: Balance Check for All Teams.

Unfortunately, the experiment was subject to some attrition: only 56% (68 of the 122 teams) submitted Kaggle submissions, with 37 submitting predictions in the Restricted Track and 31 submitting predictions in the Unrestricted Track. Intuitively, we expected that the attrition rate would be higher amongst teams assigned to the restricted track because these teams might feel discouraged after being blocked from using popular, leading to only the more talented participants completing the contest. However, we observed the opposite effect—participants were more likely to submit a score if they were assigned to the restricted track. Because our participants were mostly students, we expect that most of the attrition from the contest was competing priorities faced by the participants, such as problem set deadlines or other extramural opportunities, in a way that is balanced in expectation across treatment conditions.

To explore the possibility of selective attrition more quantitatively, we present balance checks for our post-attrition ("participating") sample in Table 4. We were reassured to find that the post-attrition sample looks even more balanced than the pre-attrition sample. We then formalize the question of differential attrition across treatment conditions as a statistical test in Table 5. First, we show that we cannot reject a null hypothesis that the difference in participation was driven by statistical noise (Model 1, t = 1.3). This null relationship is robust to adjusting for raw skill measures (Model 2), our total aggregate skill measure (Model 3), or our separated specific and general skill measure (Model 4). While we observe that our general and specific skill measures are positive predictors of participation, these measures are not statistically significant. The only statistically significant driver of participation we find is whether the team is a pair, which associates with a 23% higher likelihood of participating.

Finally, given the binary nature of our `Participated` dependent variable, we also explored other non-parametric methods of adjusting for covariates while testing for selective attrition. Specifically, we implement the methods described in Lin (2013) and Guo & Basse (2021), which provide better statistical guarantees when estimating an ATE under model misspecification (such as in our case of a binary DV). We find these methods also unable to re-

Track	**Overall**, N = 122	**Restricted**, N = 60	**Unrestricted**, N = 62	**p-value**
Team is Pair	47% (32)	43% (16)	52% (16)	0.5
Contest Hour	46.8 (43.7, 47.6)	46.9 (44.7, 47.6)	46.7 (41.3, 47.5)	0.4
Comfortable with sklearn	75% (51)	78% (29)	71% (22)	0.5
Comfortable with Feature Engineering	59% (40)	57% (21)	61% (19)	0.7
Comfortable with SQL	46% (31)	43% (16)	48% (15)	0.7
Had Prior Data Science Job	57% (39)	54% (20)	61% (19)	0.5
High Skill (Total)	60% (41)	59% (22)	61% (19)	0.9
High Skill (Specific)	65% (44)	68% (25)	61% (19)	0.6
High Skill (General)	47% (32)	41% (15)	55% (17)	0.2

Table 4: Balance Check for Participating Teams.

	Model 1	Model 2	Model 3	Model 4
Unrestricted	-0.1167 (0.0900)	-0.0816 (0.0877)	-0.1054 (0.0867)	-0.0964 (0.0895)
Team is Pair		0.2648* (0.1030)	0.2497** (0.0924)	0.2447* (0.0947)
Comfortable with sklearn		0.0677 (0.1219)		
Comfortable with Feature Engineering		0.1685 (0.1220)		
Comfortable with SQL		0.0389 (0.1035)		
Had Prior Data Science Job		-0.1362 (0.1002)		
High Skill (Total)			0.1319 (0.0906)	
High Skill (Specific)				0.0731 (0.1007)
High Skill (General)				0.0878 (0.0959)
(Intercept)	0.6167*** (0.0633)	0.4402*** (0.1001)	0.4570*** (0.0814)	0.4411*** (0.0918)
Observations	122	122	122	122
R2	0.01379	0.13509	0.10539	0.10474
Adj. R2	0.00557	0.08997	0.08265	0.07414

Table 5: OLS Results for Attrition Analysis. We also present the results of non-parametric tests of attrition in the body of the paper. For all models, the dependent variable is `Participated` and the number in parentheses are Huber-White Robust Standard Errors.

ject the null hypothesis that our restriction did not drive changes in participation (t-statistics of 0.74 (Lin, 2013), 1.2 (Guo & Basse, 2021), and 1.3 (Guo & Basse, 2021), respectively). We conclude that attrition did not meaningfully affect our ability to interpret differences across treatment conditions as causal effects.

Independence of Experimental Units Despite the above advantages, the prediction contest setting introduces one key empirical challenge that guided our experimental design: given the competitive, simultaneous nature of the competition, one may be concerned about the independence of experimental units. We do not believe this to be relevant for our setting for several reasons. First, we designed the contest to minimize interference between teams. Our contest was run online, and participants did not know each other's contact information. Second, participants were incentivized not to help each other. If others did better in the contest, that would hurt their chances of winning. Lastly, one may be concerned that effort effects passed through the contest leaderboard may drive differences in contest results independently of our treatment. As presented and discussed in our results (Section 4.1), Table 7 shows that effort, as measured in terms of hours spent on the contest, does not differ across treatment conditions; furthermore, it only minimally correlates with the final contest score.

Outcome Distribution To further describe these variables, we present distribution plots of the `Score` and the `Time Spent` on the problem in Figure 3. In the `Score` distribution, we observe that distribution is bimodal, with about $\frac{1}{4}$ of teams at or just above baseline, and $\frac{1}{5}$ of teams close to the best performance. In the `Time Spent` distribution, we observe another bimodal distribution, with modes around 8 and 16 hours spent actively working. Interestingly, the `Score` and `Time Spent` were not very correlated — simple analysis (presented later in Table A5) implies that each additional hour only improves participant `Score`s by a value of 0.006, a statistically insignificant amount (t=1.07)). Overall, we conclude that the teams spent an ecologically meaningful amount of time solving the problem, resulting in meaningful variation in the `Score` measure.

3.4.3 External Validity of Estimates

Our experimental setup was informed by our observations of data scientists and was designed to mimic their typical workflow. Specifically, the problem, outcome measures, and sample population were all designed so as to maximize the generalizability of the results.

Problem Choice We intentionally selected our contest problem because it is similar in its economic structure to many problems faced by modern organizations. For example, binary classification algorithms can be leveraged in maintenance or manufacturing operations (most obviously), sales, document search, and even resource exploration problems such as searching

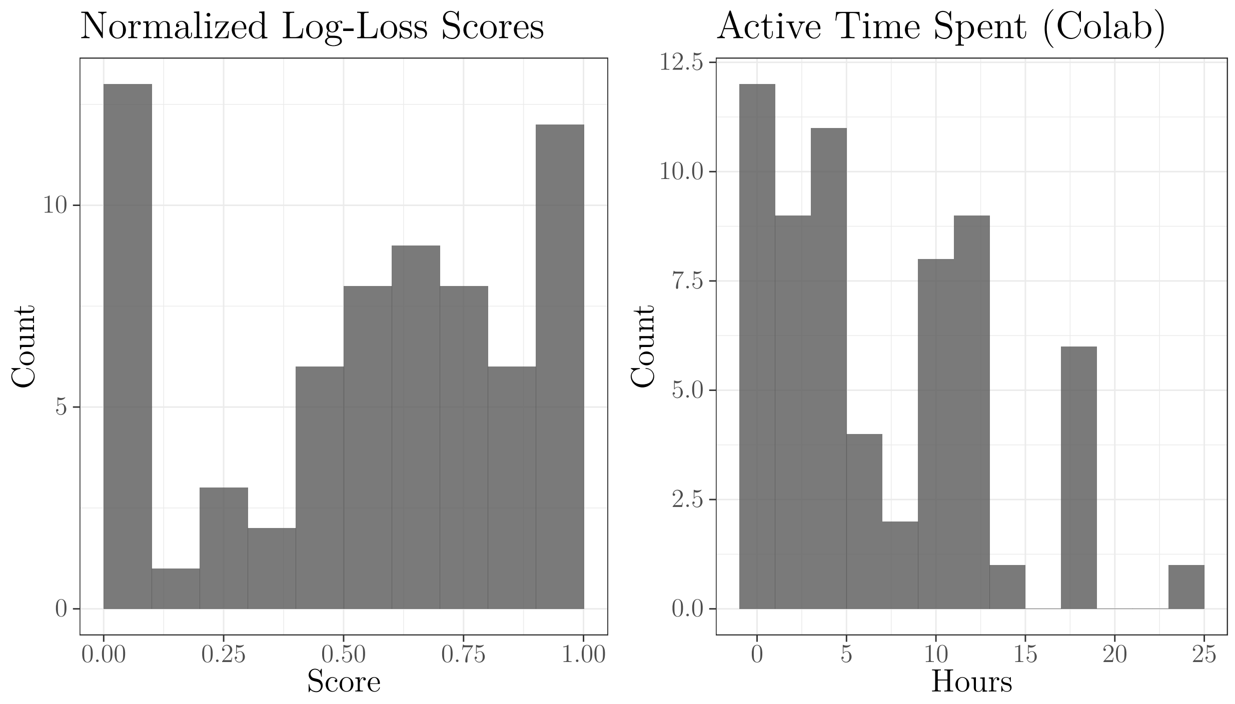


Figure 3: The distribution of (normalized) `Score` and `Time Spent` during the contest, at the team level. `Score` is well supported over all possible scores. Teams spent an average of 8 hours working on the contest, with significant variation in overall time spent.

for gold or oil (Toffel & Aal, 2022; Salesforce, 2022; Snapdocs, 2022; Nagaraj, 2022).

Further, the problem was non-trivial and could be approached through different problem-solving methods (as in the case of most realistic problems). Most obviously, participants could try many different modeling approaches to solve the problem. But the problem also lent itself to substantial engagement with the provided data. Specifically, the data was interpretable by a generally knowledgeable participant, which allows them to leverage substantive knowledge and "common sense" in developing their algorithms. Indeed, significant gains could be made on the problem through even superficial data cleaning. For example, many of the data were categories with too many fields, and one column provided dates marked as 0 CE rather than NA when the date was missing.

Sample Population While the ideal population for our study would be employed data scientists, it is challenging to recruit such highly skilled participants at reasonable costs. (The alternative of paying data scientists by the hour is infeasible given the necessary hours to solve the problem and the relatively high salaries of data scientists in the USA.) Instead, we opted to recruit for our contest through schools, resulting in a larger population of students.

Nevertheless, we believe this is a representative population of many data scientists. To

argue this, we present summary statistics on the backgrounds of our participants[24] in Table 6, and benchmark them to the 2021 Kaggle Data Science Survey (Kaggle, 2022). We find that over 71% of our participants had or were working on graduate degrees; Kaggle finds that 63% of working data scientists have graduate degrees, with over 50% of data scientists falling between the ages of 22 and 34. Our sample also skewed slightly male (57%), reflecting a broader gender bias in STEM fields and data science in general (the Kaggle Survey find that 80% of responding data scientist identify as male). Most importantly, while 83% of our sample was enrolled as a student as of the time of the contest, about 80% of participants had relevant internship or full-time work experiences such as software development or research – and over 47% had significant prior work experiences in data science specifically. Based on the fact that a significant portion of the data science workforce includes recent graduates at our participants' level of education and work background, we infer that our sample is at least approximately representative of data scientists in organizations.
Problem Solving Environment Our contest provides fixed tabular data and problem definition, limits work on the contest problem to fixed time constraints, and incentivizes participation by offering prizes in a contest structure. None of these factors generally appear in realistic business settings, where data scientists develop their own data, continually work on improving models and are paid via salary rather than tournament-style awards. Nevertheless, we argue that these differences do not significantly limit the generalizability of our results.

Our argument has three parts. First, our study focuses only on one aspect of data scientist work – algorithm development. We intentionally fix other aspects of the problem (such as the provided data, team structure, and computing resources) to isolate our variation of interest: the availability of software libraries. Thus, this abstraction is a feature of our design, not a bug. Second, despite the ability to continually improve models without explicit time limits, data scientists in businesses also work under time constraints. There are many demands on data scientist attention – including data work, communicating with their team and management, and working on other models for other parts of the business. Thus, while the *specific* time constraint imposed by the contest is arbitrary, the fact that there is a time constraint is realistic. Further, our contest duration was long enough that many teams were experiencing diminishing returns on time spent – indicative that teams had sufficient time to solve the problem. Finally, a threat to external validity arises only if the contest incentives distort team behavior or effort in a way that significantly differs from data scientists in businesses. However, conditional on participation, we do not believe the contest incentives changed the way that participants solved the problem. We observe that team motivation for

[24]Note that, because some of our teams were pairs, there are more participants than teams.

Variable	**N = 100**
Year Finished Undergrad	2021 (2016, 2022)
Current or Most Recent Degree	
Undergraduate	29%
Masters	49%
PhD	22%
Undergraduate Major	
Data Science	27%
Physical Science or Engineering	56%
Social Science	17%
Current Employment Status	
Student (full-time)	83%
Employed (full-time)	11%
Other	6%
Prior Jobs in Software, Research, or Data Science	
None	21%
Internship	43%
Full Time Work	36%
Prior Jobs in (Only) Data Science	
None	53%
Internship	32%
Full Time Work	15%
Gender	
Male	57%
Female	42%
Prefer Not to Say	1%

Table 6: Background Information on Participants. We see here that our participant population matches the general population of data scientists. This includes the experience level of our participants – the Kaggle 2021 Data Scientist Survey notes that 55% of data scientists have less than 3-years experience in work.

continuing to work on the problem was derived from intrinsic interest in the problem. Many of our participants commented on how they participated despite not actually expecting to win a prize and that their main motivation was to test their data science skills and have fun. We believe this intrinsic intellectual motivation matches the motivation that data scientists in businesses have when they engage in their work – similar to how contests have been used to illuminate aspects of problem solving in other domains (Jeppesen & Lakhani, 2010; Boudreau & Lakhani, 2013).

Percent Accuracy Outcome measure The percent accuracy (for a given visit fraction) outcome provides insight into the economic significance of our observed effects because it is proportional to productivity or revenue outcomes in many applications. Specifically, we designed the measure under the following thought exercise. Imagine that a company employs repair engineers to fix water pumps in rural areas of Tanzania. The main constraint in fixing pumps is not the work to fix a pump, given that it is broken, but rather the limited time to travel to different pumps in different areas. The main benefit of an accurate machine learning algorithm is that it can proactively identify pumps that are likely to fail while reducing the number of wasted visits to pumps that are still functioning correctly. One realistic way that organizations use model predictions is to sequentially visit the pumps in decreasing order of likelihood of needing repair (as predicted by the model). The *Percent Accuracy* of the model, then, would depend on the number of pumps that you were able to visit overall (what we call the *Visit Fraction* r). While this thought exercise may seem stylized, it matches the economic structure of how binary classification models are used in many organizations. Some analogous problems include:

- A plane mechanic decides which parts to replace without knowing (before taking the plane apart) whether the part is in need of repair.
- A sales associate chooses which clients to take meetings with and which ones to pass on.
- A gold mining company decides which regions to seek out possible gold repositories.

For these cases, although the economic magnitude of the decision varies, we observe that the percent accuracy of the model is directly proportional to the productivity or revenue of the operator.

	Model 1	Model 2	Model 3
Dependent Var.:	`Score`	`Percent Accuracy`	`Work Hours`
`Unrestricted`	0.2949***	0.1237***	-0.5745
	(0.0719)	(0.0250)	(1.519)
`(Intercept)`	0.4171***	0.7691***	7.237***
	(0.0479)	(0.0227)	(0.9974)
Observations	68	68	63
R2	0.20372	0.24746	0.00236
Adj. R2	0.19166	0.23606	-0.01400

Table 7: Main Effect Tests (OLS Regression of Score on Unrestricted). For all models, the number in parentheses are Huber-White Robust Standard Errors.

4 Results

4.1 Results: Tools Factor

We beign our presentation of results by demonstrating and quantifying our main effect. Figure 4 presents a density plot of `Score` by `Track` (treatment condition). This simple plot provides clear non-parametric evidence of the main effect, corresponding to a difference of 0.30. Because we randomized treatment, we can interpret this difference as an estimate of the causal effect of machine learning libraries on participant's scores. Figure 4 also provides evidence that our effect is not driven by outliers. We observe that the distribution of scores is roughly similar across both treatments, except that the unrestricted condition bunches at the top of the score distribution. Both treatment conditions had visible variation in scores, including individuals with low scores. To go beyond visual evidence, we apply several different statistical tests. First, to test for differences in distribution, we conduct a KS test and find that the difference between the `Score` distribution of each track is statistically significant (D = 0.56, p = 6e-5), providing completely non-parametric evidence of distributional impact from our treatment. To test for differences in group means, we present regression estimates in Table 7, with the first two models estimating our main effect. We find that the Restricted Track averaged 0.2949 in absolute `Score` less than the Unrestricted Track (Model (1)), and that the effect is statistically significant (t = 4.1, p=1e-4).

To better understand the economic significance of our main effect of treatment on score, we estimate the effect of treatment on percent accuracy in Models (3) and (4). Our treatment leads to an average absolute increase in percent accuracy of about 12 *absolute* percentage points. Given the baseline percent accuracy of 50% and average percent accuracy of 74% (Intercept from Model (3)), 12% improvement corresponds to a *relative* percentage increase

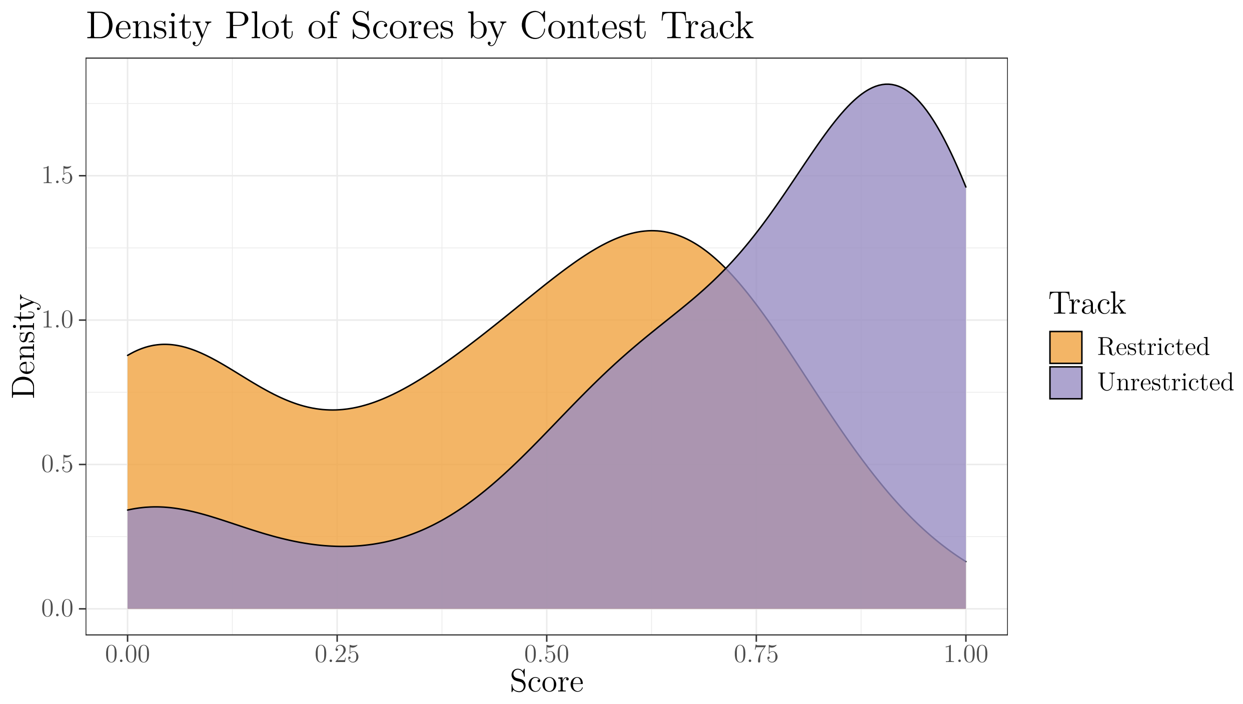


Figure 4: This provides visual evidence of our treatment effect. We find the average difference in `Score` between tracks of 0.30.

of 50% over baseline. This increase in percent accuracy is economically significant — a comparable order of magnitude to the improvement sought in the famous Netflix Prize (Lakhani et al., 2014) and some of the best incentive-based interventions found in the organizational economics literature (see (Gibbons & Roberts, 2012), chapter by Lazear and Oyer). Recall that our definition of percent accuracy seemingly arbitrarily chose to calculate accuracy within the top 30% of predictions. We justify this choice in Figure 5, which shows a box plot of the percent accuracy by treatment condition, varying the size of the sample within which accuracy is calculated (what we call 'visit fraction'). Here, we observe that our treatment condition affects the percent accuracy of teams' models no matter the choice of visit fraction. Further, while the specific treatment effect varies by visit fraction, we find that for a large range ($< 40\%$), the effect is relatively stable, justifying our choice of the top 30%.

As a second approach to benchmarking the economic significance of our main effect, we compare our main effect to the effect of data set size on our outcomes. While we did not randomize the size of the data, we can directly estimate the quality of the participant's models as a function of training data by taking advantage of the fact that we have the code for each team. Specifically, fixing the holdout "test" data set, we randomly sampled down the original "training" data set to successively smaller sizes, retrained the teams' original models, and measured their predictive performance. This allows us to directly compare the

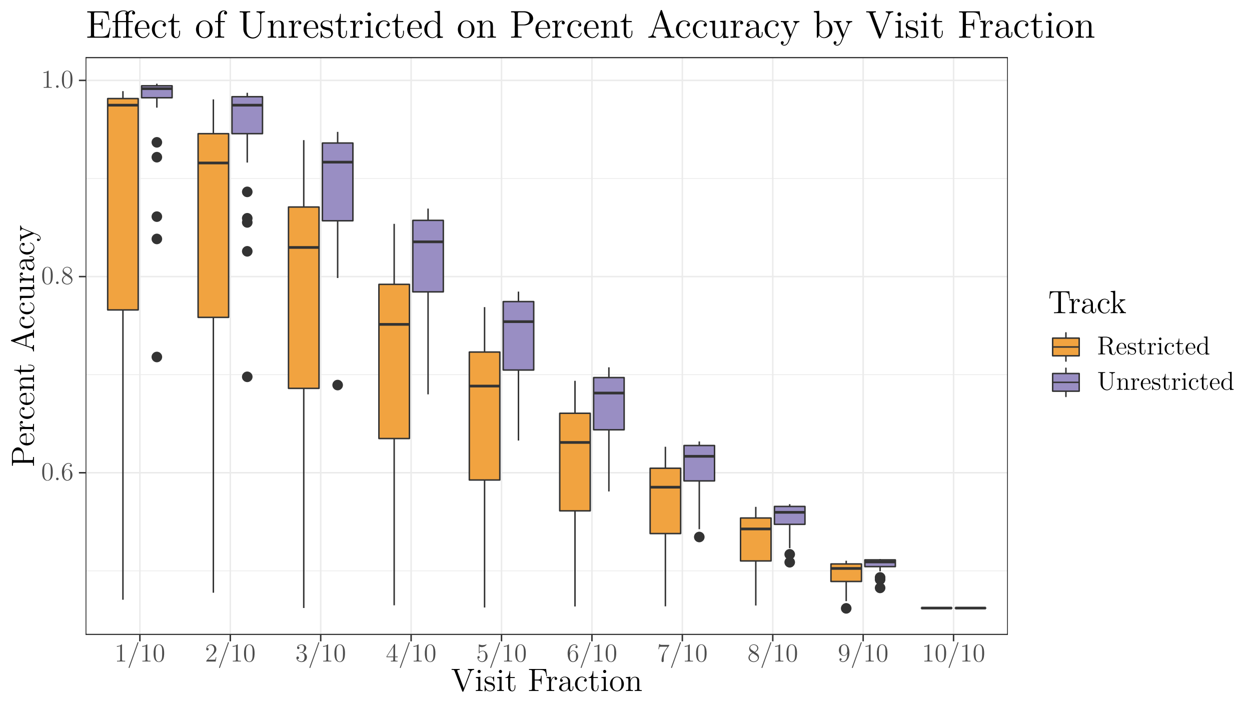


Figure 5: Each column shows the team-level distribution of *Percent Accuracy* by track, when calculated based on the top *Visit Fraction* amount of predictions (as explained in Section 3.3.3 and Section 3.4.3). Concretely, our contest evaluated each team's model based on 12000 predictions, the left-most column takes the top 1200 most-confident predictions from each team's model and evaluates the percentage of those predictions that actually need repair for each team. By contrasting the Unrestricted (Blue) and Restricted (Red) distributions, we see that the treatment effect has an effect no matter the choice of *Visit Fraction*. As such, for our regression, we elect to use *Visit Fraction* = 3/10 to quantify our treatment effect.

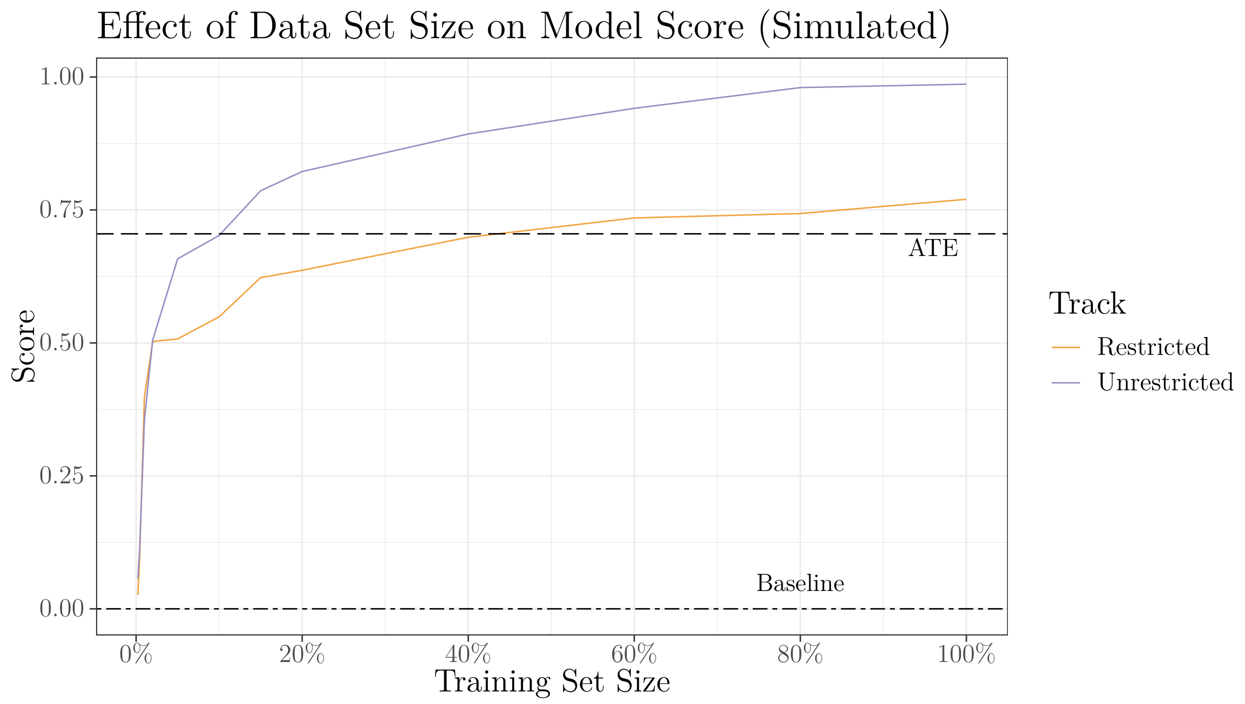


Figure 6: We quantify the effect of data set size (x-axis) on `Score` (y-axis) by rerunning each team's model, trained on specified percentages of the original training set size. Here, we plot the average scores for the top three performing teams in the Restricted track, and two of the top three teams in the Unrestricted track. The dashed-line denotes the value of $1 - \text{ATE}$ based on our ATE estimate from Table 7. We find that the unrestricted teams perform as well with just 15% of the original training set as the restricted teams do with the full 100% of the original training data. Note: The missing model from the Unrestricted track could not be fit with less than 5% of the data due to model parameters that required a minimum number of observations to train.

marginal effect of tools (operationalized as software libraries) and data set size.

We present a plot of normalized log-loss scores top 3x performing teams in each track as a function of training data set size in Figure 6. As expected, the log-loss score worsens as less training data is available. Surprisingly, the *Unrestricted* teams still perform reasonably well with restricted data. In the dashed-lined labeled ATE, we show the model score equal to $1 - \text{ATE}$, which corresponds to how well the teams would have performed if they were restricted from using modeling libraries. By inspecting where this $1 - \text{ATE}$ value intersects the Score of the unrestricted teams under successively smaller training set sizes, we find that not having access to modeling libraries is equivalent to a reduced training set size of 10% of the original training set — a large amount. We take this as evidence in favor of Hypothesis 1.

4.2 Results: Skill Factors

In order to test Hypothesis 2, Table 8 presents tests for heterogeneity captured along our aggregated skill indices. In Model (1), we show that `High Skill (Total)` does not predict `Score`, nor does it interact with our treatment to predict `Score` in a economically or statistically significant way. Model (2) separates total skill into `High Skill (General)` and `High Skill (Specific)`, but similarly finds no evidence of either measure's direct effects on the outcome. However, in Model (3), we present a fully saturated model where we include interaction terms (including the triple interaction effect) – given the binary nature of our predictors, this model estimates a fully non-linear conditional expectation function for `Score`. Here, we find our main heterogeneity result: that the null-effect of skill interacted with treatment (from Model (1)) was the result of two competing effects: a negative interaction with specific skills and a positive interaction with general skills. For those with high general skills, we find that dropping the library restrictions has a much smaller effect on them than on those with low general skills (coef= -0.3028, p=0.018). By contrast, for those with high specific skills, we find that the effect of lifting the restriction is strengthened (coef=0.3309, p=0.009). In Appendix Table A4, we show that these interaction terms are robust to alternative constructions of the skill measure and to use of continuous rather than binary skill indices. Most importantly, our main interaction of interest are on the same order of magnitude as the direct effect, and are therefore also economically significant by the same argument as above.

Using variation in sign-up time, we are able to estimate the direct causal effect of time on final model quality (see Appendix D). Unsurprisingly, we find the effect to be positive. Unfortunately, our design does not allow us to test interaction between skill factors and time available, as we did not exogenously vary the time available to teams in our contest; therefore, the study of the interaction of skill and time must be left to future work.

5 Mechanism

In this section, we propose a mechanism to explain why restrictions on software libraries influence the model quality, which we call "tools-as-skill". Understanding this mechanism is valuable as it demonstrates *why* model libraries lead to improved predictive model development, enabling extrapolation and generalization of our results. First, we define our primary model and assemble the evidence in favor of it. Second, we consider four alternative mechanisms that may explain some of our results and present additional tests to rule them out.

	Model 1	Model 2	Model 3
`Unrestricted`	0.2936** (0.0732)	0.3134** (0.0670)	0.2295** (0.0622)
`High Skill (Total)`	-0.0359 (0.0732)		
`Unrestricted x High Skill (Total)`	-0.0803 (0.1463)		
`High Skill (General)`		-0.0876 (0.0815)	-0.0784 (0.0622)
`Unrestricted x High Skill (General)`			-0.3028* (0.1243)
`High Skill (Specific)`		0.0645 (0.0842)	0.0499 (0.0622)
`Unrestricted x High Skill (Specific)`			0.3309** (0.1243)
`High Skill (General) x High Skill (Specific)`			-0.2656* (0.1243)
`Unrestricted x High Skill (General) x High Skill (Specific)`			0.5071* (0.2487)
`(Intercept)`	0.5639** (0.0366)	0.5642** (0.0421)	0.5770** (0.0311)
Observations	68	68	68
R2	0.20996	0.26470	0.34072
Adj. R2	0.17293	0.21801	0.26380

Table 8: Heterogeneity analyses: OLS Regression of `Score` on `Unrestricted` interacted with Skill Indices. For all models, the dependent variable is `Score` and the number in parentheses are Huber-White Robust Standard Errors.

5.1 The Tools-as-Skills Model

Our model comprises the following three assumptions.

Assumption 1 *Model quality is determined by a standard concave production function with a single input: the product of skill and effort.*

The intuition behind Assumption 1 is that to solve the prediction problem, teams must determine what to do / how to do it (skill) and then exert the effort necessary to do it (effort). The concavity assumption is justified by the observation that almost all prediction problems are capped in how well they can be solved, and experience diminishing marginal returns to skill and effort. We interpret effort in our model as something analogous to "focused time" but choose the term effort to emphasize that we are not taking a stance on the precise functional form through which time enters the production function.

Assumption 2 *A restriction to software libraries additively reduces the effective "stock" of general skill.*

Assumption 2 formalizes the intuition that software libraries substitute for skill (how to reason about data and how to implement analysis approaches) by codifying the relevant knowledge. This is the key assumption in the model that drives our main results.

Assumption 3 *General skill adds directly to the stock of skill, but tool-specific skill moderates the amount of change induced by restrictions to software libraries.*

As discussed in the literature review, general and specific skills have different roles. Our final assumption is justified by the intuition that skill *with* model libraries enables one to benefit more from having them available.

Note that this model does not describe behavior but captures the technological structure of the algorithm production function. The core constraints in the model that limit the ultimate effectiveness of algorithms are the concavity of the production function and the limited amount of effort dedicated to the problem[25].

Formally, we model each team's algorithm quality q as a function of skill (general skill g and tool-specific skill s), library availability l, effort spent e, and noise ϵ:

$$q = f(g, s, l, e) + \epsilon \tag{3}$$

[25]This constraint could be captured by a convex cost function, though we omit this and instead assume that effort is a fixed quantity for simplicity.

where f is some generic function and ϵ is independent of other variables. Combining Assumption 1-3 implies the following restrictions:

$$f(g, s, l, e) = m([\beta_g g + \beta_{sl} sl]e) + \epsilon \tag{4}$$

where the β's are real parameters and m is a generic one-input production function with standard concavity behavior (i.e., $m > 0$, $m' > 0$, $m'' < 0$). Intuitively, we would expect that the β's are positive, meaning that more skill leads to better final model quality; we make this assumption in our following analysis.

The comparative statics of our model rationalize our prior results from Table 8. The derivations follow simply taking derivatives and applying the chain rule, and we present them in Section E.1; here, we illustrate how our model helps us to interpret our empirical results in our discussion here.

First, we find that algorithm quality is increasing in all factors: library availability, skill, and effort ($\partial_l f > 0$, $\partial_g f > 0$ and $\partial_e f > 0$). This is not surprising given Assumptions 1 and 2. However, the more interesting takeaway is the interpretation given by the model – that restrictions to software libraries impact final model quality because less modeling skill makes modeling effort less efficient, leading to lower overall efficiency of time spent.

Second, we find that specific skills and library availability interact positively under mild conditions ($\partial_s \partial_l f > 0$ when $f' + Cf'' > 0$) [26]. This result follows directly from our inclusion of the interaction sl in the production function (Assumption 3). We observe a positive interaction between library availability and specific skill because libraries are irrelevant without the skill needed to use them properly.

Finally, we find that general skills interacts negatively with library availability and with specific skills ($\partial_g \partial_l f < 0$ and $\partial_g \partial_s f < 0$). We observe a negative interaction between library availability and general skills because of Assumption 2 and the concavity assumption from Assumption 1. We observe heterogeneity for library restrictions and general skill because libraries and general skill additively contribute to the same stock of knowledge, and that stock of knowledge influences model quality with diminishing marginal returns. The negative interaction between general and specific skills occurs for the same reason.

We note that while the interaction effects of our model are identified in our experiment, none of the β parameters in the tools-as-skill model are identified. Consequently, we make no attempt to estimate them. The comparative statics exercise here merely confirms that

[26]Taking derivatives reveals this additional mild condition, where $C = \beta_{sl} sle$. What this condition shows is that the interaction is positive in situations where the marginal progress made by participants sufficiently exceeds the loss in progress due to concavity in the production function. Since our contest participants were working in a situation where they were clearly making improvements on the model, we find this assumption to be reasonable in our setting.

our empirical results are consistent with the three assumptions made to derive our model.

The primary purpose of this Tools-as-Skills model is explaining the main heterogeneity results in Table 8, which allows us to generalize and apply our results to other settings where tools and skill may manifest differently. However, our model is *also* useful for investigating empirical questions related to *how* teams go about solving the predictive model development process. In Section E.2.1, we illustrate an application of the model to explaining how access to modeling libraries leads to difference in how teams allocate effort across tasks during the contest. Specifically, we find that access to modeling libraries leads team to spend *more* time on modeling tasks and less time on feature engineering and exploring the data. While this result may be otherwise unintuitive (access to modeling libraries should *free* up time to work on feature engineering), our model clarifies why would expect this effort-substitution pattern – because libraries make time spent on modeling more efficient.

5.2 Ruling Out Alternative Models

Importantly, our results also help us to rule out other potential models of the impact of our software library restriction. In particular there are four other candidate models that may potentially explain our main effect: Discouragement, Unidimensional Skill, Libraries as Effort Saved, and Tools and Skill as Partial Substitutes.

5.2.1 Discouragement

The Discouragement explanation argues that having access to libraries makes the process of working more enjoyable, such that teams without the restriction spend more time on the problem. To rule out a discouragement model – that teams work less because they have a preference to work with libraries – we refer to Model (3) in Table 7, where we show that restrictions to ML libraries did not drive a statistically or ecologically significant difference in the amount of time spent.

5.2.2 Unidimensional Skill

This model assumes that there are no distinctions between skill types. Then, libraries enter the production function multiplicatively, with functional forms like $f = m([\beta_g g + \beta_s s]le)$ or $f = m_l(l)m_e([\beta_g g + \beta_s s]e)$. We can use Table 8 to rule out an explanation involving unidimensional skill that does not distinguish general and tool-specific skill. Specifically, the `Total Skill` coefficient in Model (1) does not interact significantly with `Unrestricted`, and the heterogeneity results of Models 3 and 4 cannot be explained without making such a distinction.

5.2.3 Tools as Effort Saved

In this alternative model, the effect of our libraries would provide a better starting place, by functioning as effort rather than skill. The intuition here is that libraries act like a partner who already started the analysis, and passed it off to a team only partially completed. This would imply a functional form like $f = m(\beta_g ge + \beta_{sl} sl])$.

The effort-saved model, however, would imply the same results as our skill substitution model. How can we distinguish these models? To rule out this competing explanation, we disaggregate our score measure into two components: "initial" score, and then "improvement". We define the initial score as the best submitted Score within the first thirty minutes[27] of any submissions for that team, and the improvement as the difference between the final score and the initial score. We present OLS regressions exploring variation in these new outcomes measures in Table 9. In Models (1) and (2), we present the main effects of our restriction for `First Score` and `Improvement` respectively; in Models (3) and (4), we present the regressions including interactions with skill variables. We find that `First Score` and `Improvement` explain a similar amount of overall score we observe – with comparable intercepts. However, we find that restrictions primarily affect the *improvement* achieved by teams – teams with library restrictions improve less than 0.22 absolute score, over $\frac{2}{3}$ of the total effect (Model 3). By contrast, the restrictions do not have any statistically significant effect on the initial score[28]. Turning to our skill measures, we find that, similarly, the interaction effects are only noteworthy for explaining variation in the improvement. This implies that the specific skills are specifically useful for problem-solving *after* the teams have started to work on the problem. These results rule out the *effort-saved* model, where software libraries simply help teams start from a better place. They show that improvements to models are driven by progress *throughout* the contest, rather than initial starting place.

5.2.4 Tools and Skill as Partial Substitutes

A final alternative model is to argue that while tools and skill do substitute on average, there is some amount of skill that can *never* be substituted for by tools (e.g. software libraries). This alternative model captures the idea that human intuition has 'insight' into data in a way that algorithmic thinking and probabilistic models simply cannot represent. Such a model would justify the idea that data scientists should 'think more carefully about the

[27]We chose thirty minutes to allow for teams to make corrections to formatting that may be associated with initial submission, though the results are robust to changes in this threshold.

[28]We find it reassuring that the separate effects from each model tend to add up to the overall effect that we find in the overall test of experience presented in Table A5. This seems to indicate that our results are reliable, even if they are not statistically significant on their own.

	Model 1	Model 2	Model 3	Model 4
Dependent Var.:	`Score First`	`Improvement`	`Score First`	`Improvement`
`Unrestricted`	0.0784 (0.0827)	0.2195** (0.0809)	0.1587 (0.1431)	0.0826 (0.1361)
`Unrestricted x High Skill (Specific)`			0.0016 (0.1515)	0.3048 (0.1596)
`High Skill (Specific)`			0.1273 (0.0924)	-0.2070* (0.0934)
`Unrestricted x High Skill (General)`			-0.0925 (0.1613)	-0.1546 (0.1641)
`High Skill (General)`			-0.1465 (0.0898)	0.1413 (0.0817)
`(Intercept)`	0.2262*** (0.0468)	0.1916*** (0.0419)	0.1995* (0.0839)	0.2742** (0.0811)
Observations	67	67	67	67
R2	0.01440	0.10915	0.11543	0.17491
Adj. R2	-0.00077	0.09544	0.04292	0.10728

Table 9: First Scores and Incremental Gain. For all models, the number in parentheses are Huber-White Robust Standard Errors.

data' rather than just plugging the data into a model blindly.

In our setting, we would infer from this model that the features that were manually engineered by our teams in the restricted track have predictive power beyond the features that were 'discovered' by the random-forest-based methods that dominated our unrestricted track. Heuristically, we have qualitative evidence that teams that were restricted *did* think more carefully about the data at hand, and engineered variables using that intuition. Specifically, we asked teams that won each of the tracks of our contest to describe their solution approach. The leading example of careful data thinking this comes from our restricted track winners, where the team explained that they combined latitude and longitude data into 'bins' rather than using each of the variables separately, in order to capture the notion of locality intrinsic to that data (the team's process is illustrated in Figure 7). Further excerpts from these winner blog posts are presented in Table A6, and highlight the plug-and-chug approach of Unrestricted Track teams as they sought to test out different type of models rather than think carefully about the data at hand.

We cannot statistically test this alternative model, but we develop a different style of test by combining the *features* from teams in the Restricted track, with the *models* of the teams in the unrestricted track. Our logic is that, if this alternative model were true and

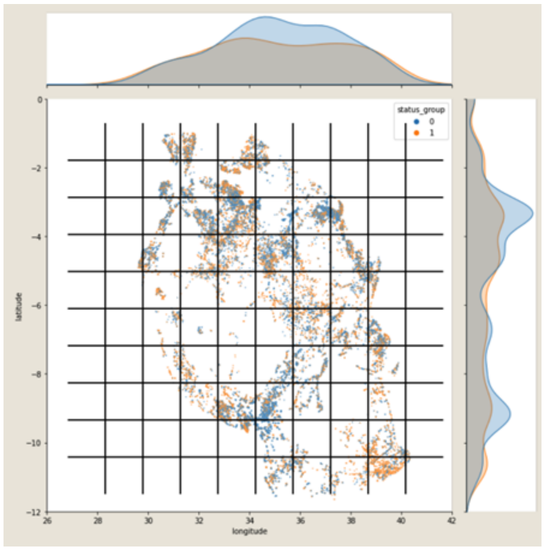


Figure 7: An illustrative graph from a Blog Post of a winning team in the Restricted Track. Here, the team used their intuition to devise a binned measure of latitude and longitude that captures the non-linear notion of locality.

Outcome: `Score`	**Feature Source**			
Model Source	**Original Team**	**Restricted (1st)**	**Restricted (2nd)**	**Restricted (3rd)**
Unrestricted (1st)	1.00	0.92	0.90	0.91
Unrestricted (2nd)	0.99	0.70	0.66	0.85
Unrestricted (3rd)	0.98	0.96	0.90	0.92

Table 10: Summary of Simulation Exercise.

teams in the Restricted track had extra predictive insight embedded in the features that they engineered, then these composite models would do even better than the original Unrestricted Team's performance. A detailed account of this simulated combined model is detailed in Section E.2.2, with our results presented in Table 10. We find that models that combine the features of the Restricted Track with the Models of the Unrestricted track do worse than the Unrestricted Track models with their original feature sets. While we cannot rule out statistically that these combined models are *worse* than the original Unrestricted models, this does provide some evidence against this alternative model of Tools and Skill as partial substitutes.

6 Discussion

Why are some businesses better than others at developing predictive models at scale? In this paper, we answer a key component of this question: What factors drive predictive model development? We develop and apply a methods-tools-skills conceptual framework and demonstrate that tools are as necessary as the methods they codify in explaining predictive model performance. Our results are based on a field experiment demonstrating the impact of library availability on model quality for 68 teams of data scientists. We find that teams that were unrestricted in their access to modeling libraries produced models that were 30% better in log-loss improvement over baseline — corresponding to a 50% relative increase in accuracy improvement over baseline, equivalent to an increase in data set size of 1000%. However, access to libraries does not have the same impact on all teams. For example, we find that teams with high specific modeling-specific skills significantly benefit from model library access, whereas teams with high general data-science skills do not. This is consistent with a mechanism we call 'Tools-as-Skill,' where tools codify and abstract some general data-analytic skills but simultaneously create the need for new tool-specific skills to use them effectively.

6.1 Contributions

Our paper contributes to the literature on IT Productivity and the broader economics of AI. Regarding IT Productivity, we apply a conceptual framework from that literature to a novel technological setting: AI production, particularly predictive model development. Our core contribution is further extending the technology-skills distinction (Tambe, 2014; Tambe & Hitt, 2014; Wu et al., 2018) by distinguishing technology into methods and tools. Additionally, we conceptualize the mechanism by which tools drive predictive model development ('Tools-as-Skill') and present experimental evidence supporting that mechanism. More generally, our contribution is to take ideas from the literature on Skill-biased Technical Change (the distinction between general and specific skills) and use them to guide our empirical study. Beyond this study, this conceptual framework provides a new way to think about other theoretical questions about tools: for example, Teodoridis (2018) notes that it needs to be clarified ex-ante how tools may influence team formation. Our conceptual framework suggests that the impact of tools on team formation depends on the cost of acquiring the specific skills necessary to leverage the tools and the benefit of the general skills that are being replaced by the tool. Finally, we note that the tools that we study here are open-source tools. This study is a step towards quantifying the value of these tools to the broader economy. Given the significant effects found in this paper, this paper contributes to the growing body of evidence arguing that open-source tools are crucial to the modern economy yet under-accounted for (Greenstein & Nagle, 2014; Nagle, 2019; Langenkamp & Yue, 2022).

Regarding the Economics of AI, our core contributions are first conceptualizing AI production as an economic activity worthy of research and second experimentally characterizing the drivers of a critical component of AI production: predictive model development. Theoretically, prior literature suggests the importance of advances in methods, including data, computing, and algorithms (Bessen et al., 2022; DeStefano et al., 2020) to predictive model development; we contribute by expanding this into a framework of factors that includes tools and skills. Empirically, we contribute a novel method for studying this question — a prediction contest, similar to (Cowgill et al., 2020). Furthermore, our method provides a way to measure the effect of tools across teams of varying skill levels, unlike AI Benchmarks, like ImageNet or GLUE, that only capture improvement in methods for the expert teams at the bleeding edge. Our results highlight that tools are a key complementary[29] factor that is co-invented alongside methods, which plays a critical role in the adoption of AI methods in the broader economy.

An important managerial implication of our results is that tools are a key lever managers can adjust to improve predictive model development rather than something that is exogenously 'out there' and available to all firms. Therefore, tools and tool infrastructure at firms is something to actively manage and optimize. We highlight two applications of this implication. First, managers should actively consider the tradeoffs encountered when designing internal tools. Specifically, ML systems commonly have requirements for interpretability, security, or usability that lead to simplifications reducing the available functionality and richness of the available analytic approaches for a specific problem. For example, Amgen can make limited use of black-box algorithms due to regulatory requirements (Amgen, 2022). As another example, some companies offer 'no-code' or 'low-code' solutions that lower the skills needed to access analytic solutions. Our results serve as a way to understand the costs of limiting access to tools (e.g., due to regulatory requirements), or, equivalently, the benefits of adding tools (e.g., in the form of no-code or low-code solutions). This implication is particularly important for firms who find that other levers for improving predictive models (such as data or compute infrastructure) are producing diminishing returns on investment; firms seeking to integrate AI into their operating model should make the investments necessary to couple their internal infrastructure to AI tools (especially open source technologies). As a second application, firms should intentionally develop the skill sets of their data scientists to be optimized around those tools. For organizations with low restrictions on available tools, the most effective model for a center of excellence is to focus on developing tools and infrastructure that codify analytic techniques in a relevant way for the organization. Given

[29]We do not show complementarity by demonstrating the interaction of methods and tools in our empirical analysis; instead, we show the economic and statistical significance of a tools factor juxtaposed with a methods factor. However, we justify the use of the term complementary as follows: the existence of a method is necessary for a tool to implement it. Therefore, the tool has zero value without the method, so methods and tools are complementary.

a sufficient basis of statistics knowledge, the center of excellence can focus on spreading the tool-specific skills needed to leverage that infrastructure rather than investing in further expensive general education. For organizations with more substantial constraints (arising from regulatory or ethical considerations) or for companies that have not had time to invest in robust analytic infrastructure, it's worth investing in fewer extremely high general-skill individuals (such as those with advanced degrees) who can solve important problems independently of available tools.

6.2 Limitations and Future Work

We also acknowledge there are limitations in our approach, which are opportunities for future work. First, although our study is comparable to many published experimental studies (and some observational studies), our sample size is modest. This limits our ability to draw stronger conclusions about mechanisms – especially when analyzing noisier secondary outcomes, such as time allocation. However, we emphasize that our main effects are conclusively demonstrated, with statistical significance due to the large effect size.

Second, we recognize that algorithm production is only one part of a broader process of data science in organizations. For example, our paper does not study the process of *choosing problems* to work on, nor the process of conceiving of, creating, and maintaining relevant data assets. Skills and tools may play a different role in those processes, and we hope there will be more research on these topics in the future.

Finally, we conceive skill as falling into two types (general and specific), which could be enriched. First, it is possible to collect more granular measures; for instance, at the topic level rather than the class level. Second, it would be interesting to explicitly consider a *third* type of skill — domain-specific experience. Whereas our experimental design effectively controls for domain-specific experience (by choosing a relatively obscure problem context), it would be particularly informative to experimentally vary domain-specific experience and study its impact on the complementary between feature engineering and modeling. This is also where we expect our results have the least external validity. We acknowledge that while our experiment was designed with external validity in mind, it is only a single problem in a specific context. While we expect that our tool-as-skill mechanism will generalize to other contexts, we do not expect that our finding regarding the substitutability between feature engineering and modeling will hold in all other contexts.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., … Zheng, X. (2016, March). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems.* arXiv. Retrieved 2022-09-23, from `http://arxiv.org/abs/1603.04467` (Issue: arXiv:1603.04467 arXiv:1603.04467 [cs]) doi: 10.48550/arXiv.1603.04467

Acemoglu, D., & Restrepo, P. (2018). The race between man and machine: Implications of technology for growth, factor shares, and employment. , *108*(6), 1488–1542. Retrieved 2023-03-16, from `https://pubs.aeaweb.org/doi/10.1257/aer.20160696` doi: 10.1257/aer.20160696

Agrawal, A., Gans, J., & Goldfarb, A. (2018). *Prediction Machines: The Simple Economics of Artificial Intelligence.* Harvard Business Press. (Google-Books-ID: wJY4DwAAQBAJ)

Amgen. (2022, September). *Digital Transformation of Pharma and Biotech.* Retrieved 2022-09-23, from `https://www.arcweb.com/blog/digital-transformation-pharma-biotech`

Autor, D. H., Levy, F., & Murnane, R. J. (2003, November). The Skill Content of Recent Technological Change: An Empirical Exploration*. *The Quarterly Journal of Economics*, *118*(4), 1279–1333. Retrieved 2022-09-23, from `https://doi.org/10.1162/003355303322552801` (Number: 4) doi: 10.1162/003355303322552801

Babina, T., Fedyk, A., He, A., & Hodson, J. (2021). Artificial Intelligence, Firm Growth, and Product Innovation. , 89.

Banko, M., & Brill, E. (2001, July). Scaling to Very Very Large Corpora for Natural Language Disambiguation. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics* (pp. 26–33). Toulouse, France: Association for Computational Linguistics. Retrieved 2022-09-23, from `https://aclanthology.org/P01-1005` doi: 10.3115/1073012.1073017

BCG. (2021, April). *Why You Need an Open Source Software Strategy.* Retrieved 2022-09-23, from `https://www.bcg.com/publications/2021/open-source-software-strategy-benefits`

BCG. (2023). *Digital support functions and shared services.* Retrieved 2023-03-16, from `https://www.bcg.com/capabilities/operations/digital-support-functions`

Bessen, J., Impink, S. M., Reichensperger, L., & Seamans, R. (2022, June). The role of data for AI startup growth. *Research Policy*, *51*(5), 104513. Retrieved 2022-09-23, from `https://www.sciencedirect.com/science/article/pii/S0048733322000415` (Number: 5) doi: 10.1016/j.respol.2022.104513

BLS. (2022, September). *Employment to grow 7.7 percent from 2020 to 2030; 1.7 percent excluding COVID-19 recovery : The Economics Daily: U.S. Bureau of Labor Statistics.* Retrieved 2022-09-23, from `https://www.bls.gov/opub/ted/2021/employment-to-grow-7-7-percent-from-2020-to-2030-1-7-percent-excluding-covid-19-recovery.htm`

Bojinov, I., & Gupta, S. (2022, jul 28). Online Experimentation: Benefits, Operational and Methodological Challenges, and Scaling Guide. *Harvard Data Science Review*, *4*(3). (https://hdsr.mitpress.mit.edu/pub/aj31wj81)

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., ... Liang, P. (2022, July). *On the Opportunities and Risks of Foundation Models.* arXiv. Retrieved 2022-09-23, from `http://arxiv.org/abs/2108.07258` (Issue: arXiv:2108.07258 arXiv:2108.07258 [cs]) doi: 10.48550/arXiv.2108.07258

Boudreau, K. J., & Lakhani, K. R. (2013, April). Using the Crowd as an Innovation

Partner. *Harvard Business Review.* Retrieved 2022-09-28, from https://hbr.org/2013/04/using-the-crowd-as-an-innovation-partner

Bresnahan, T., Brynjolfsson, E., & Hitt, L. M. (2002). Information Technology, Workplace Organization, and the Demand for Skilled Labor: Firm-Level Evidence. *The Quarterly Journal of Economics*, *117*(1), 339–376. Retrieved 2019-10-15, from https://econpapers.repec.org/article/oupqjecon/v_3a117_3ay_3a2002_3ai_3a1_3ap_3a339-376..htm (Number: 1)

Bresnahan, T., Greenstein, S., Brownstone, D., & Flamm, K. (1996). Technical progress and co-invention in computing and in the uses of computers. , *1996*, 1–83. Retrieved 2022-11-01, from https://www.jstor.org/stable/2534746 (Publisher: Brookings Institution Press) doi: 10.2307/2534746

Broadbent, M., Weill, P., & Clair, D. S. (1999, June). The Implications of Information Technology Infrastructure for Business Process Redesign. *MIS Quarterly*, *23*(2), 159. Retrieved 2022-09-28, from https://www.jstor.org/stable/249750?origin=crossref doi: 10.2307/249750

Brynjolfsson, E., Rock, D., & Syverson, C. (2021, January). The Productivity J-Curve: How Intangibles Complement General Purpose Technologies. *American Economic Journal: Macroeconomics*, *13*(1), 333–372. Retrieved 2022-09-23, from https://www.aeaweb.org/articles?id=10.1257/mac.20180386 (Number: 1) doi: 10.1257/mac.20180386

Budibund. (2022). *What are internal tools? definitive guide for 2022.* Retrieved 2023-03-16, from https://budibase.com/internal-tools/

Chari, V., & Hopenhayn, H. (1991). Vintage Human Capital, Growth, and the Diffusion of New Technology. *Journal of Political Economy*, *99*(6), 1142–65. Retrieved 2022-09-28, from https://econpapers.repec.org/article/ucpjpolec/v_3a99_3ay_3a1991_3ai_3a6_3ap_3a1142-65.htm

Cho, J., Lee, K., Shin, E., Choy, G., & Do, S. (2016, January). *How much data is needed to train a medical image deep learning system to achieve necessary high accuracy?* arXiv. Retrieved 2022-09-23, from http://arxiv.org/abs/1511.06348 (Issue: arXiv:1511.06348 arXiv:1511.06348 [cs])

Cowgill, B., Dell'Acqua, F., Deng, S., Hsu, D., Verma, N., & Chaintreau, A. (2020, June). *Biased Programmers? Or Biased Data? A Field Experiment in Operationalizing AI Ethics* [SSRN Scholarly Paper]. Rochester, NY. Retrieved 2022-09-23, from https://papers.ssrn.com/abstract=3615404 (Issue: 3615404) doi: 10.2139/ssrn.3615404

CSET. (2022, September). *CSET PARAT.* Retrieved 2022-09-23, from https://parat.cset.tech/

DeepLearningAI. (2021, March). *A Chat with Andrew on MLOps: From Model-centric to Data-centric AI.* Retrieved 2022-09-23, from https://www.youtube.com/watch?v=06-AZXmwHjo

DeStefano, T., Kneller, R., & Timmis, J. (2020). Cloud computing and firm growth. *VOX*. Retrieved from https://voxeu.org/article/cloud-computing-and-firm-growth

Doshi-Velez, F., & Kim, B. (2017, March). *Towards A Rigorous Science of Interpretable Machine Learning.* arXiv. Retrieved 2022-09-23, from http://arxiv.org/abs/1702.08608 (Issue: arXiv:1702.08608 arXiv:1702.08608 [cs, stat]) doi: 10.48550/arXiv.1702.08608

Ferreira, K. J., Lee, B. H. A., & Simchi-Levi, D. (2016). Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Man-*

agement.

Ferreira, K. J., Parthasarathy, S., & Sekar, S. (2022, March). Learning to Rank an Assortment of Products. *Management Science*, *68*(3), 1828–1848. Retrieved 2022-09-23, from `https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2021.4130` (Number: 3 Publisher: INFORMS) doi: 10.1287/mnsc.2021.4130

Frey, L. J., & Fisher, D. H. (1999, January). Modeling decision tree performance with the power law. In *Seventh International Workshop on Artificial Intelligence and Statistics.* PMLR. Retrieved 2022-09-23, from `https://proceedings.mlr.press/r2/frey99a.html` (ISSN: 2640-3498)

Gibbons, R., & Roberts, J. (2012). *The Handbook of Organizational Economics.* Princeton University Press. Retrieved 2022-09-23, from `https://www.degruyter.com/document/doi/10.1515/9781400845354/html` (Publication Title: The Handbook of Organizational Economics) doi: 10.1515/9781400845354

Greenstein, S., Myers, K., & Mehta, S. (2022). Digital Manufacturing at Amgen. , 17.

Greenstein, S., & Nagle, F. (2014, May). Digital dark matter and the economic contribution of Apache. *Research Policy*, *43*(4), 623–631. Retrieved 2022-09-29, from `https://www.sciencedirect.com/science/article/pii/S0048733314000055` doi: 10.1016/j.respol.2014.01.003

Greenstein, S., Yue, D., Herman, K., & Gulick, S. (2022). Hugging face: Serving ai on a platform.

Gregory, R. W., Henfridsson, O., Kaganer, E., & Kyriakou, H. (2021, July). The Role of Artificial Intelligence and Data Network Effects for Creating User Value. *Academy of Management Review*, *46*(3), 534–551. Retrieved 2022-09-23, from `https://journals.aom.org/doi/abs/10.5465/amr.2019.0178` (Number: 3 Publisher: Academy of Management) doi: 10.5465/amr.2019.0178

Guo, K., & Basse, G. (2021, June). The Generalized Oaxaca-Blinder Estimator. *Journal of the American Statistical Association*, *0*(0), 1–13. Retrieved 2022-09-28, from `https://doi.org/10.1080/01621459.2021.1941053` doi: 10.1080/01621459.2021.1941053

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., … Oliphant, T. E. (2020, September). Array programming with NumPy. *Nature*, *585*(7825), 357–362. Retrieved 2022-09-23, from `https://www.nature.com/articles/s41586-020-2649-2` (Number: 7825 Publisher: Nature Publishing Group) doi: 10.1038/s41586-020-2649-2

Hartmann, P., & Henkel, J. (2020, September). The Rise of Corporate Science in AI: Data as a Strategic Resource. *Academy of Management Discoveries*, *6*(3), 359–381. Retrieved 2022-09-23, from `https://journals.aom.org/doi/10.5465/amd.2019.0043` (Number: 3 Publisher: Academy of Management) doi: 10.5465/amd.2019.0043

Hidalgo, C. A., Orghian, D., Canals, J. A., Almeida, F. D., & Martin, N. (2021). *How Humans Judge Machines.* MIT Press. (Google-Books-ID: DhkSEAAAQBAJ)

Hoffman, M., Kahn, L. B., & Li, D. (2018, May). Discretion in Hiring*. *The Quarterly Journal of Economics*, *133*(2), 765–800. Retrieved 2022-09-23, from `https://doi.org/10.1093/qje/qjx042` (Number: 2) doi: 10.1093/qje/qjx042

Huyen, C. (2022, September). *Introduction to Machine Learning Interviews Book · MLIB.* Retrieved 2022-09-23, from `https://huyenchip.com/ml-interviews-book/`

Iansiti, M. (2021). The Value of Data and Its Impact on Competition. *SSRN Electronic*

Journal. Retrieved 2022-09-23, from https://www.ssrn.com/abstract=3890387 doi: 10.2139/ssrn.3890387

Iansiti, M., & Lakhani, K. R. (2020, January). Competing in the Age of AI. *Harvard Business Review.* Retrieved 2022-09-23, from https://hbr.org/2020/01/competing-in-the-age-of-ai (Section: Competitive strategy)

Jeppesen, L. B., & Lakhani, K. R. (2010, October). Marginality and Problem-Solving Effectiveness in Broadcast Search. *Organization Science*, *21*(5), 1016–1033. Retrieved 2022-09-23, from https://pubsonline.informs.org/doi/abs/10.1287/orsc.1090.0491 (Number: 5 Publisher: INFORMS) doi: 10.1287/orsc.1090.0491

Juntu, J., Sijbers, J., De Backer, S., Rajan, J., & Van Dyck, D. (2010). Machine learning study of several classifiers trained with texture analysis features to differentiate benign from malignant soft-tissue tumors in T1-MRI images. *Journal of Magnetic Resonance Imaging*, *31*(3), 680–689. Retrieved 2022-09-23, from https://onlinelibrary.wiley.com/doi/abs/10.1002/jmri.22095 (Number: 3 _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jmri.22095) doi: 10.1002/jmri.22095

Kaggle. (2022, September). *State of Data Science and Machine Learning 2021.* Retrieved 2022-09-23, from https://www.kaggle.com/kaggle-survey-2021

Kohavi, R., & Thomke, S. (2017). The surprising power of online experiments. *Harvard business review*, *95*(5), 74–82.

Koning, R., Hasan, S., & Chatterji, A. (2022, September). Experimentation and Start-up Performance: Evidence from A/B Testing. *Management Science*, *68*(9), 6434–6453. Retrieved 2022-09-29, from https://pubsonline.informs.org/doi/10.1287/mnsc.2021.4209 doi: 10.1287/mnsc.2021.4209

Lagomarsino, E. (2020). Estimating elasticities of substitution with nested CES production functions: Where do we stand? , *88*, 104752.

Lakhani, K., Cohen, W., Ingram, K., Kothalkar, T., Kuzemchenko, M., Malik, S., … Pokrywa, S. (2014). *Netflix: Designing the Netflix Prize (A).* Harvard Business Publishing. Retrieved 2022-09-29, from https://www.hbs.edu/faculty/Pages/item.aspx?num=47650

Langenkamp, M., & Yue, D. N. (2022, July). How Open Source Machine Learning Software Shapes AI. In *Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society* (pp. 385–395). New York, NY, USA: Association for Computing Machinery. Retrieved 2022-09-29, from https://doi.org/10.1145/3514094.3534167 doi: 10.1145/3514094.3534167

Lemus, J., & Marshall, G. (2021, February). Dynamic Tournament Design: Evidence from Prediction Contests. *Journal of Political Economy*, *129*(2), 383–420. Retrieved 2022-09-28, from https://www.journals.uchicago.edu/doi/10.1086/711762 doi: 10.1086/711762

Lin, W. (2013, March). Agnostic notes on regression adjustments to experimental data: Reexamining Freedman's critique. *The Annals of Applied Statistics*, *7*(1), 295–318. Retrieved 2022-09-28, from https://projecteuclid.org/journals/annals-of-applied-statistics/volume-7/issue-1/Agnostic-notes-on-regression-adjustments-to-experimental-data--Reexamining/10.1214/12-AOAS583.full doi: 10.1214/12-AOAS583

LinkedIn. (2022, September). *Building the next version of our infrastructure.* Retrieved 2022-

09-23, from https://engineering.linkedin.com/blog/2019/building-next-infra

Luca, M., Kleinberg, J., & Mullainathan, S. (2016). Algorithms need managers, too. *Harvard business review*, *94*(1), 20.

Mao, J., & Bojinov, I. (2021). Quantifying the value of iterative experimentation. *arXiv preprint arXiv:2111.02334*.

Mattu, J. A., Lauren Kirchner, S., & Larson, J. (2022, September). *How We Analyzed the COMPAS Recidivism Algorithm.* Retrieved 2022-09-23, from https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm

McKinsey. (2021, September). *Global AI Survey 2021 – Desktop.* Retrieved 2022-09-23, from http://ceros.mckinsey.com/global-ai-survey-2020-a-desktop-3-1

McKinsey. (2022). *How to improve your internal operations | McKinsey.* Retrieved 2023-03-16, from https://www.mckinsey.com/capabilities/operations/our-insights/how-good-are-your-internal-operations-really

Metz, C. (2021). *Genius Makers: The Mavericks Who Brought AI to Google, Facebook, and the World.* New York: Dutton.

Nagaraj, A. (2022, January). The Private Impact of Public Data: Landsat Satellite Maps Increased Gold Discoveries and Encouraged Entry. *Management Science*, *68*(1), 564–582. Retrieved 2022-09-23, from https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2020.3878 (Number: 1 Publisher: INFORMS) doi: 10.1287/mnsc.2020.3878

Nagle, F. (2019, March). Open Source Software and Firm Productivity. *Management Science*, *65*(3), 1191–1215. Retrieved 2022-09-29, from https://pubsonline.informs.org/doi/10.1287/mnsc.2017.2977 doi: 10.1287/mnsc.2017.2977

New Vantage Partners. (2022). *Data and ai leadership executive survey 2022.* Retrieved from https://c6abb8db-514c-4f5b-b5a1-fc710f1e464e.filesusr.com/ugd/e5361a_2f859f3457f24cff9b2f8a2bf54f82b7.pdf

Oladele, S. (2021). *A comprehensive guide on how to monitor your models in production.*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., … Cournapeau, D. (2011). Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 6.

PricewaterhouseCoopers. (2022, September). *PwC 2022 AI Business Survey.* Retrieved 2022-09-23, from https://www.pwc.com/us/en/tech-effect/ai-analytics/ai-business-survey.html

Ransbotham, S., Khodabandeh, S., Kiron, D., Candelon, F., Chu, M., & LaFountain, B. (2020). *Expanding ai's impact with organizational learning.* Retrieved from https://web-assets.bcg.com/f1/79/cf4f7dce459686cfee20edf3117c/mit-bcg-expanding-ai-impact-with-organizational-learning-oct-2020.pdf

Rock, D. (2021, September). Engineering Value: The Returns to Technological Talent and Investments in Artificial Intelligence. *SSRN Electronic Journal.* Retrieved 2021-04-27, from https://papers.ssrn.com/abstract=3427412 doi: 10.2139/ssrn.3427412

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986, October). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536. Retrieved 2022-09-23, from https://www.nature.com/articles/323533a0 (Number: 6088 Publisher: Nature Publishing Group) doi: 10.1038/323533a0

Salesforce. (2022, September). *Binary Classification Use Case.* Retrieved 2022-09-23, from https://help.salesforce.com/s/articleView?id=sf.bi_edd_about_use_case _binary.htm&type=5

Senoner, J., Netland, T., & Feuerriegel, S. (2022, August). Using Explainable Artificial Intelligence to Improve Process Quality: Evidence from Semiconductor Manufacturing. *Management Science*, *68*(8), 5704–5723. Retrieved 2022-09-23, from https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2021.4190 (Number: 8 Publisher: INFORMS) doi: 10.1287/mnsc.2021.4190

Sevilla, J., & Villalobos, P. (2022, September). *Parameter counts in Machine Learning - AI Alignment Forum.* Retrieved 2022-09-23, from https://www.alignmentforum.org/posts/GzoWcYibWYwJva8aL/parameter-counts-in-machine-learning

Smaldone, F., Ippolito, A., Lagger, J., & Pellicano, M. (2022, June). Employability skills: Profiling data scientists in the digital labour market. *European Management Journal.* Retrieved 2022-09-23, from https://www.sciencedirect.com/science/article/pii/S0263237322000810 doi: 10.1016/j.emj.2022.05.005

Snapdocs. (2022, September). *Snapdocs Case Study.* Retrieved 2022-09-23, from https://aws.amazon.com/solutions/case-studies/snapdocs-case-study/

Spotify. (2021). *Invisible made visible: The value of designing tools for internal teams.* Retrieved 2023-03-16, from https://spotify.design/article/invisible-made-visible-the-value-of-designing-tools-for-internal-teams

Tambe, P. (2014). Big data investment, skills, and firm value. , *60*(6), 1452–1469. Retrieved 2021-03-03, from http://pubsonline.informs.org/doi/abs/10.1287/mnsc.2014.1899 doi: 10.1287/mnsc.2014.1899

Tambe, P., & Hitt, L. M. (2014). Job hopping, information technology spillovers, and productivity growth. , *60*(2), 338–355. Retrieved 2023-03-16, from https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2013.1764 (Publisher: INFORMS) doi: 10.1287/mnsc.2013.1764

Teodoridis, F. (2018). Understanding team knowledge production: The interrelated roles of technology and expertise. , *64*(8), 3625–3648. Retrieved 2021-06-15, from http://pubsonline.informs.org/doi/abs/10.1287/mnsc.2017.2789 (Publisher: INFORMS) doi: 10.1287/mnsc.2017.2789

Thomke, S. H. (1998). Managing experimentation in the design of new products. *Management science*, *44*(6), 743–762.

Thomke, S. H. (2020). *Experimentation works: The surprising power of business experiments.* Harvard Business Press.

Toffel, M., & Aal, Y. A. (2022, September). *Intenseye: Powering Workplace Health and Safety with AI - Case - Faculty & Research - Harvard Business School.* Retrieved 2022-09-23, from https://www.hbs.edu/faculty/Pages/item.aspx?num=60944

Varian, H. (2018, January). Artificial Intelligence, Economics, and Industrial Organization. In *The Economics of Artificial Intelligence: An Agenda* (pp. 399–419). University of Chicago Press. Retrieved 2022-09-23, from https://www.nber.org/books-and-chapters/economics-artificial-intelligence-agenda/artificial-intelligence-economics-and-industrial-organization

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., … van Mulbregt, P. (2020, March). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. Retrieved 2022-09-23, from https://www.nature.com/articles/s41592-019-0686-2 (Number: 3 Publisher: Nature Publishing Group) doi: 10.1038/s41592-019-0686-2

Wu, L., Jin, F., & Hitt, L. M. (2018). Are all spillovers created equal? a network perspective on information technology labor movements. , *64*(7), 3168–3186. Retrieved 2023-03-16, from https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2017.2778 (Publisher: INFORMS) doi: 10.1287/mnsc.2017.2778

Wu, L., Lou, B., & Hitt, L. (2019, October). Data Analytics Supports Decentralized Innovation. *Management Science*, *65*(10), 4863–4877. Retrieved 2019-10-21, from http://pubsonline.informs.org/doi/10.1287/mnsc.2019.3344 (Number: 10) doi: 10.1287/mnsc.2019.3344

Yu, D., Seltzer, M. L., Li, J., Huang, J.-T., & Seide, F. (2013). Feature learning in deep neural networks - studies on speech recognition tasks. Retrieved 2022-09-29, from https://arxiv.org/abs/1301.3605 doi: 10.48550/ARXIV.1301.3605

Zhang, D., Mishra, S., Brynjolfsson, E., Etchemendy, J., Ganguli, D., Grosz, B., … Perrault, R. (2021, March). *The AI Index 2021 Annual Report.* arXiv. Retrieved 2022-09-23, from http://arxiv.org/abs/2103.06312 (Issue: arXiv:2103.06312 arXiv:2103.06312 [cs])

A Contest Overview

A.1 Recruitment & Pre-Contest Logistics

Qualified undergraduate and graduate students were recruited for the contest from leading universities in the United States, including Harvard, MIT, CMU, Columbia, JHU, and the University of Washington. We marketed the contest by sending emails to professors, program directors, and on-campus organizations active in the fields of statistics, computer science, data science, and related disciplines. We explained that the contest would require familiarity with the Python programming language, and knowledge of data science, machine learning, and statistical modeling. Our contacts at these universities forwarded information about the contest (including our website[30]) to relevant class mailing lists and posted a flyer we created (Figure A1) around their campuses. To entice students to participate, these materials emphasized that the contest would allow students to (i) grow their data science skills, (ii) win cash prizes and recognition from the data science community, and (iii) contribute to the science of data science. The monetary prizes advertised for the competition are given in Table A1. These prizes were awarded within each track (meaning the top scorer in the restricted and unrestricted tracks were each awarded $1,000).

After these marketing materials were distributed, over 400 students from 50 universities registered to participate in the datathon. One month before the competition, we sent the email shown in Figure A2 to all who registered. This email included a calendar hold and provided logistical details about the competition. On February 1, two weeks before the datathon, we sent an additional email shown in Figure A3 with additional details about the competition and a video encouraging everyone who registered to participate.

Placement	**Prize**
1st Place	$1,000
2nd Place	$500
3rd Place	$250
Top 10	$50
Above Baseline	$10 Amazon gift card

Table A1: Prizes awarded in each track of the competition.

On the day of the competition, the kickoff email in Figure A4 was sent to everyone who registered. This email instructed registrants to complete a pre-contest survey on Qualtrics (detailed in Appendix B), which collected basic information about the registrants and allowed us to monitor how many registrants were actually competing. There were 191 responses to

[30]https://www.datathonatlish.com/home

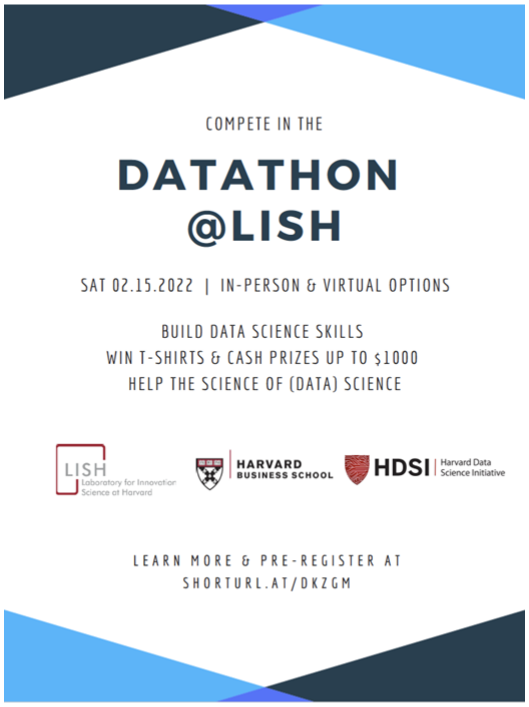
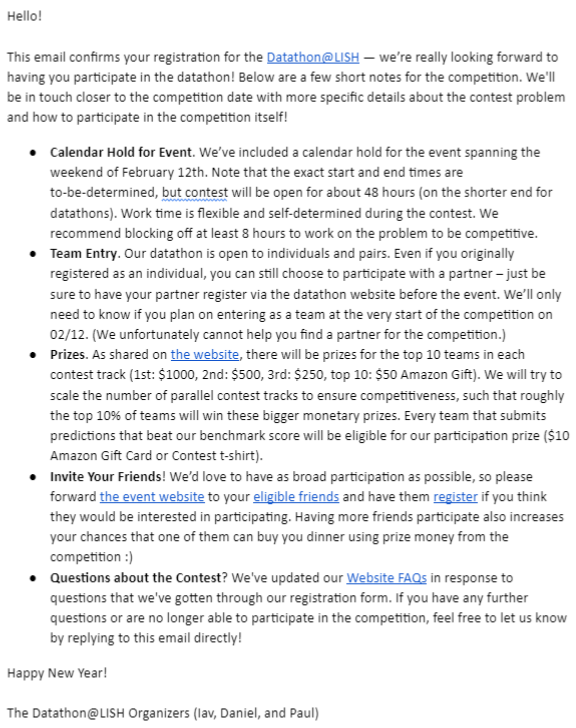

COMPETE IN THE
DATATHON @LISH
SAT 02.15.2022 | IN-PERSON & VIRTUAL OPTIONS
BUILD DATA SCIENCE SKILLS
WIN T-SHIRTS & CASH PRIZES UP TO $1000
HELP THE SCIENCE OF (DATA) SCIENCE
LISH Laboratory for Innovation Science at Harvard
HARVARD BUSINESS SCHOOL
HDSI Harvard Data Science Initiative
LEARN MORE & PRE-REGISTER AT
SHORTURL.AT/DKZGM

Figure A1: The marketing flyer used to recruit contest participants.



Hello!

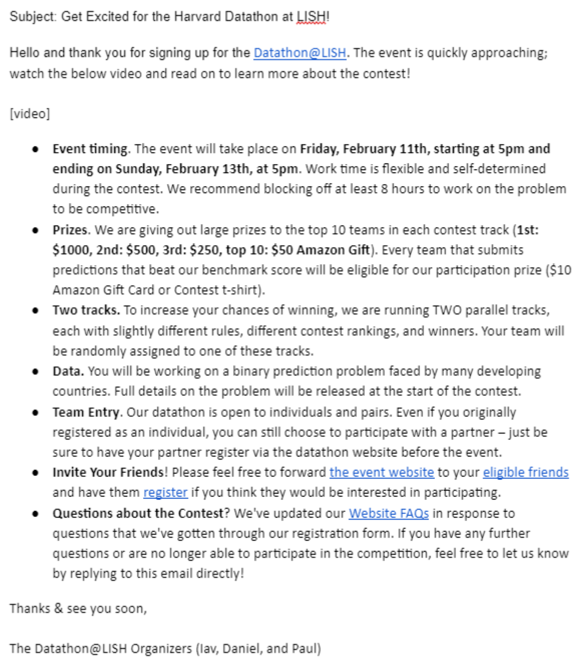
This email confirms your registration for the Datathon@LISH — we're really looking forward to having you participate in the datathon! Below are a few short notes for the competition. We'll be in touch closer to the competition date with more specific details about the contest problem and how to participate in the competition itself!

- **Calendar Hold for Event**. We've included a calendar hold for the event spanning the weekend of February 12th. Note that the exact start and end times are to-be-determined, but contest will be open for about 48 hours (on the shorter end for datathons). Work time is flexible and self-determined during the contest. We recommend blocking off at least 8 hours to work on the problem to be competitive.
- **Team Entry**. Our datathon is open to individuals and pairs. Even if you originally registered as an individual, you can still choose to participate with a partner – just be sure to have your partner register via the datathon website before the event. We'll only need to know if you plan on entering as a team at the very start of the competition on 02/12. (We unfortunately cannot help you find a partner for the competition.)
- **Prizes**. As shared on the website, there will be prizes for the top 10 teams in each contest track (1st: $1000, 2nd: $500, 3rd: $250, top 10: $50 Amazon Gift). We will try to scale the number of parallel contest tracks to ensure competitiveness, such that roughly the top 10% of teams will win these bigger monetary prizes. Every team that submits predictions that beat our benchmark score will be eligible for our participation prize ($10 Amazon Gift Card or Contest t-shirt).
- **Invite Your Friends**! We'd love to have as broad participation as possible, so please forward the event website to your eligible friends and have them register if you think they would be interested in participating. Having more friends participate also increases your chances that one of them can buy you dinner using prize money from the competition :)
- **Questions about the Contest**? We've updated our Website FAQs in response to questions that we've gotten through our registration form. If you have any further questions or are no longer able to participate in the competition, feel free to let us know by replying to this email directly!

Happy New Year!

The Datathon@LISH Organizers (Iav, Daniel, and Paul)

Figure A2: The initial registration email sent one month before the competition.


Subject: Get Excited for the Harvard Datathon at LISH!

Hello and thank you for signing up for the Datathon@LISH. The event is quickly approaching; watch the below video and read on to learn more about the contest!

[video]

- **Event timing**. The event will take place on **Friday, February 11th, starting at 5pm and ending on Sunday, February 13th, at 5pm**. Work time is flexible and self-determined during the contest. We recommend blocking off at least 8 hours to work on the problem to be competitive.
- **Prizes**. We are giving out large prizes to the top 10 teams in each contest track (**1st: $1000, 2nd: $500, 3rd: $250, top 10: $50 Amazon Gift**). Every team that submits predictions that beat our benchmark score will be eligible for our participation prize ($10 Amazon Gift Card or Contest t-shirt).
- **Two tracks.** To increase your chances of winning, we are running TWO parallel tracks, each with slightly different rules, different contest rankings, and winners. Your team will be randomly assigned to one of these tracks.
- **Data.** You will be working on a binary prediction problem faced by many developing countries. Full details on the problem will be released at the start of the contest.
- **Team Entry**. Our datathon is open to individuals and pairs. Even if you originally registered as an individual, you can still choose to participate with a partner – just be sure to have your partner register via the datathon website before the event.
- **Invite Your Friends**! Please feel free to forward the event website to your eligible friends and have them register if you think they would be interested in participating.
- **Questions about the Contest**? We've updated our Website FAQs in response to questions that we've gotten through our registration form. If you have any further questions or are no longer able to participate in the competition, feel free to let us know by replying to this email directly!

Thanks & see you soon,

The Datathon@LISH Organizers (Iav, Daniel, and Paul)

Figure A3: The second email sent two weeks before the competition.

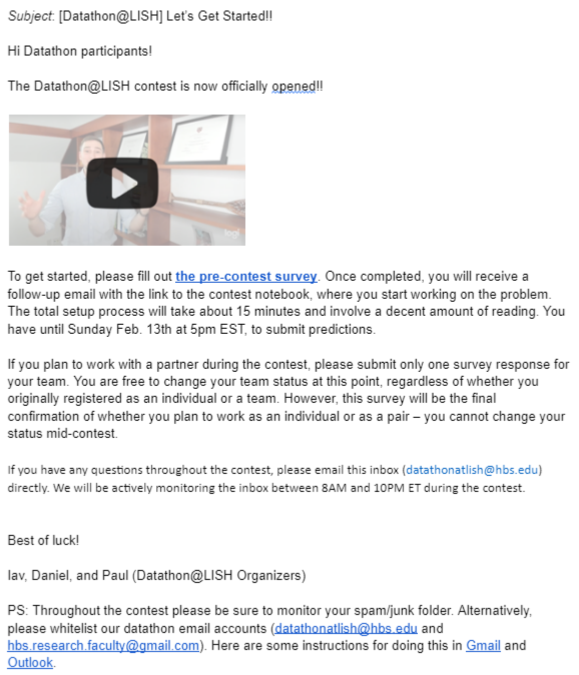
this survey. After registrants completed the survey, they were automatically sent the final email in Figure A5, which described the details of the prediction problem and included a link to the code notebook for the competition. Half of the registrants who completed the pre-contest survey were sent the email with a link to the code notebook for the restricted track, and the other half were sent a link for the unrestricted track. This is how we implemented our randomization procedure.

A.2 Colab Notebooks

All resources and information the participants needed to participate in the competition were contained in the Colab notebook they were sent after completing the pre-contest survey. Half of the participants were sent a notebook labelled ”Track Lagrange”[31] (our internal coding for the unrestricted track), while the other half were sent a notebook labelled ”Track Euler”[32] (our internal coding for the restricted track). The two versions of the notebooks were almost completely identical, with the exception of the section within the rules that described package usage. The notebooks were divided into the following sections:

- **Competition Rules and Submission Process for the Datathon@LISH** - This section defined the rules for the competition and provided instructions for participants

[31]https://colab.research.google.com/drive/1vvTstNGRvBMiPXEmcWODhT-KLrNUCb1a?usp=sharing
[32]https://colab.research.google.com/drive/1jnD0yP3JxXG9lOkRnvj-WfJVi-21XWZO?usp=sharing



Subject: [Datathon@LISH] Let's Get Started!!

Hi Datathon participants!

The Datathon@LISH contest is now officially opened!!

To get started, please fill out **the pre-contest survey**. Once completed, you will receive a follow-up email with the link to the contest notebook, where you start working on the problem. The total setup process will take about 15 minutes and involve a decent amount of reading. You have until Sunday Feb. 13th at 5pm EST, to submit predictions.

If you plan to work with a partner during the contest, please submit only one survey response for your team. You are free to change your team status at this point, regardless of whether you originally registered as an individual or a team. However, this survey will be the final confirmation of whether you plan to work as an individual or as a pair – you cannot change your status mid-contest.

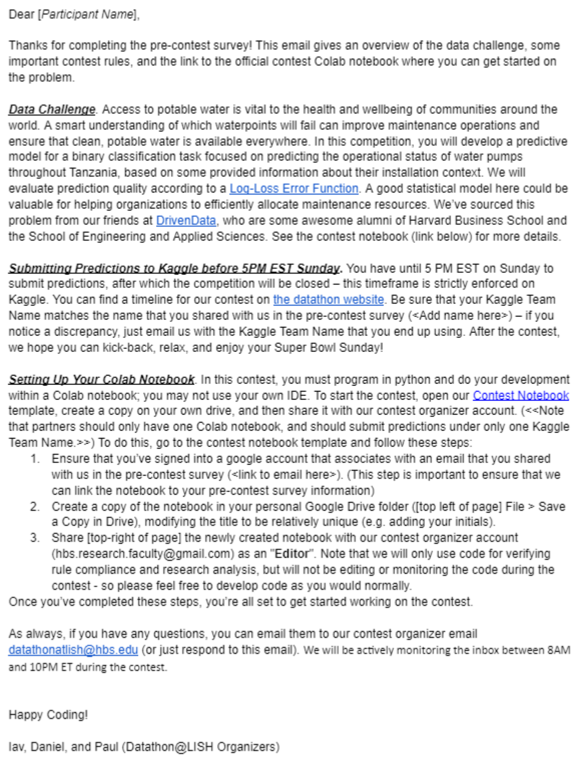
If you have any questions throughout the contest, please email this inbox (datathonatlish@hbs.edu) directly. We will be actively monitoring the inbox between 8AM and 10PM ET during the contest.

Best of luck!

Iav, Daniel, and Paul (Datathon@LISH Organizers)

PS: Throughout the contest please be sure to monitor your spam/junk folder. Alternatively, please whitelist our datathon email accounts (datathonatlish@hbs.edu and hbs.research.faculty@gmail.com). Here are some instructions for doing this in Gmail and Outlook.

Figure A4: The kickoff email sent on the day of the competition.



Dear [*Participant Name*],

Thanks for completing the pre-contest survey! This email gives an overview of the data challenge, some important contest rules, and the link to the official contest Colab notebook where you can get started on the problem.

Data Challenge. Access to potable water is vital to the health and wellbeing of communities around the world. A smart understanding of which waterpoints will fail can improve maintenance operations and ensure that clean, potable water is available everywhere. In this competition, you will develop a predictive model for a binary classification task focused on predicting the operational status of water pumps throughout Tanzania, based on some provided information about their installation context. We will evaluate prediction quality according to a Log-Loss Error Function. A good statistical model here could be valuable for helping organizations to efficiently allocate maintenance resources. We've sourced this problem from our friends at DrivenData, who are some awesome alumni of Harvard Business School and the School of Engineering and Applied Sciences. See the contest notebook (link below) for more details.

Submitting Predictions to Kaggle before 5PM EST Sunday. You have until 5 PM EST on Sunday to submit predictions, after which the competition will be closed – this timeframe is strictly enforced on Kaggle. You can find a timeline for our contest on the datathon website. Be sure that your Kaggle Team Name matches the name that you shared with us in the pre-contest survey (<Add name here>) – if you notice a discrepancy, just email us with the Kaggle Team Name that you end up using. After the contest, we hope you can kick-back, relax, and enjoy your Super Bowl Sunday!

Setting Up Your Colab Notebook. In this contest, you must program in python and do your development within a Colab notebook; you may not use your own IDE. To start the contest, open our Contest Notebook template, create a copy on your own drive, and then share it with our contest organizer account. (<<Note that partners should only have one Colab notebook, and should submit predictions under only one Kaggle Team Name.>>) To do this, go to the contest notebook template and follow these steps:

1. Ensure that you've signed into a google account that associates with an email that you shared with us in the pre-contest survey (<link to email here>). (This step is important to ensure that we can link the notebook to your pre-contest survey information)
2. Create a copy of the notebook in your personal Google Drive folder ([top left of page] File > Save a Copy in Drive), modifying the title to be relatively unique (e.g. adding your initials).
3. Share [top-right of page] the newly created notebook with our contest organizer account (hbs.research.faculty@gmail.com) as an "**Editor**". Note that we will only use code for verifying rule compliance and research analysis, but will not be editing or monitoring the code during the contest - so please feel free to develop code as you would normally.

Once you've completed these steps, you're all set to get started working on the contest.

As always, if you have any questions, you can email them to our contest organizer email datathonatlish@hbs.edu (or just respond to this email). We will be actively monitoring the inbox between 8AM and 10PM ET during the contest.

Happy Coding!

Iav, Daniel, and Paul (Datathon@LISH Organizers)

Figure A5: The email sent after registrants completed the pre-contest survey.

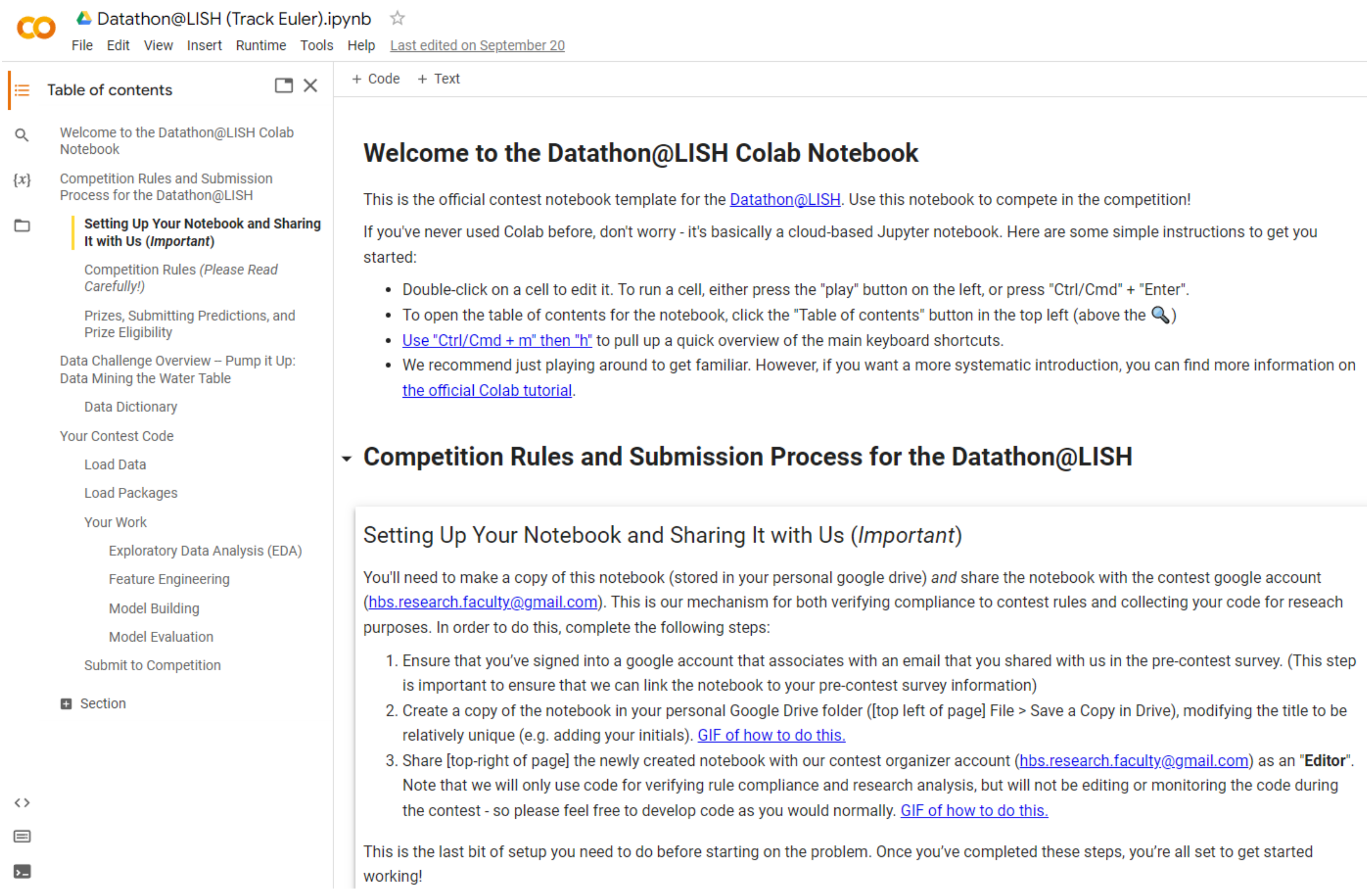
to submit their models to Kaggle. This section was identical across the two versions of the notebooks except for an additional rule in the ”Track Euler” notebook, which stated that participants could not import any machine learning modeling functions from open-source software libraries like `sklearn` or `pytorch`.

- **Data Challenge Overview - Pump it Up: Data Mining the Water Table** - This section provided some background on the prediction problem and included a data dictionary that defined each of the features in the data set.

- **Your Contest Code** - This section provided space for the participants to work on the problem. It was broken into separate sections for exploratory data analysis (EDA), feature engineering, model building, and model evaluation. We intentionally divided the notebook into these sections so we could track how much time participants spent on each task.

The Colab notebooks were set to auto-save every 60 seconds. This allowed us to track how much time the participants spent on each part of the problem throughout the competition. See Figure A6 for an image of the notebook from the restricted track. See the Supplement for the complete notebooks from the two tracks.

Once a participant was satisfied with their model, the next step was to submit the model's predictions on the test set to our competition on Kaggle (described in the next section). The link to the appropriate Kaggle competition was provided within each notebook.


Datathon@LISH (Track Euler).ipynb
File Edit View Insert Runtime Tools Help Last edited on September 20
Table of contents
+ Code + Text

Welcome to the Datathon@LISH Colab Notebook

This is the official contest notebook template for the Datathon@LISH. Use this notebook to compete in the competition!

If you've never used Colab before, don't worry - it's basically a cloud-based Jupyter notebook. Here are some simple instructions to get you started:

- Double-click on a cell to edit it. To run a cell, either press the "play" button on the left, or press "Ctrl/Cmd" + "Enter".
- To open the table of contents for the notebook, click the "Table of contents" button in the top left (above the 🔍)
- Use "Ctrl/Cmd + m" then "h" to pull up a quick overview of the main keyboard shortcuts.
- We recommend just playing around to get familiar. However, if you want a more systematic introduction, you can find more information on the official Colab tutorial.

Competition Rules and Submission Process for the Datathon@LISH

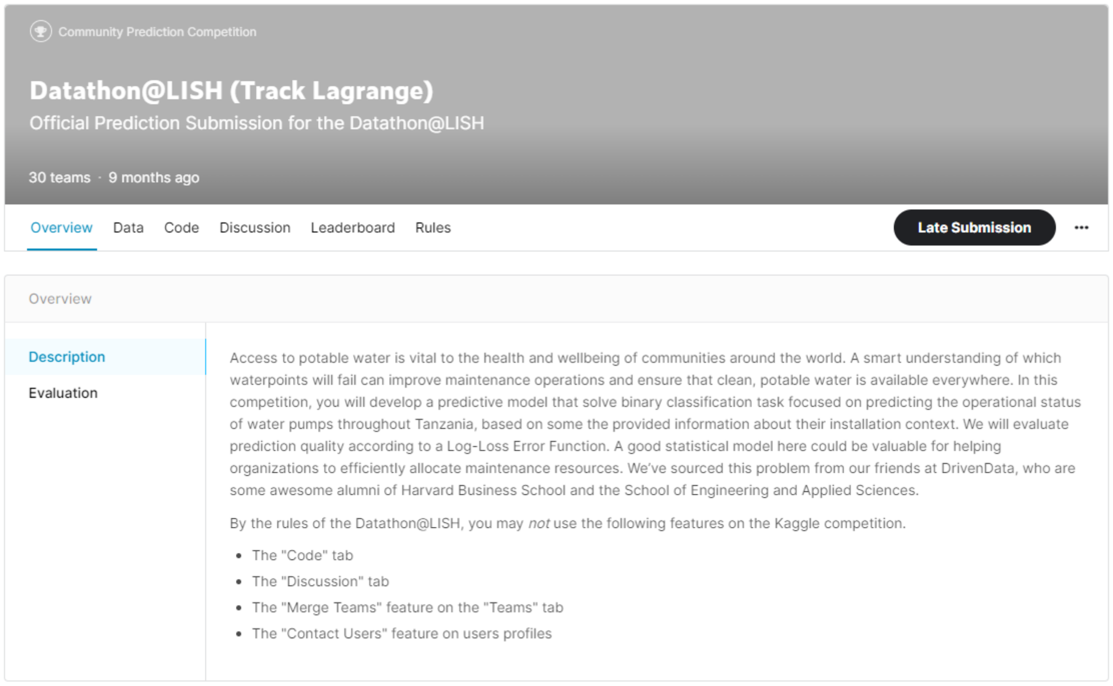
Setting Up Your Notebook and Sharing It with Us (*Important*)

You'll need to make a copy of this notebook (stored in your personal google drive) *and* share the notebook with the contest google account (hbs.research.faculty@gmail.com). This is our mechanism for both verifying compliance to contest rules and collecting your code for reseach purposes. In order to do this, complete the following steps:

1. Ensure that you've signed into a google account that associates with an email that you shared with us in the pre-contest survey. (This step is important to ensure that we can link the notebook to your pre-contest survey information)
2. Create a copy of the notebook in your personal Google Drive folder ([top left of page] File > Save a Copy in Drive), modifying the title to be relatively unique (e.g. adding your initials). GIF of how to do this.
3. Share [top-right of page] the newly created notebook with our contest organizer account (hbs.research.faculty@gmail.com) as an "**Editor**". Note that we will only use code for verifying rule compliance and research analysis, but will not be editing or monitoring the code during the contest - so please feel free to develop code as you would normally. GIF of how to do this.

This is the last bit of setup you need to do before starting on the problem. Once you've completed these steps, you're all set to get started working!

Figure A6: A screenshot of the contest Colab notebook.


Community Prediction Competition
Datathon@LISH (Track Lagrange)
Official Prediction Submission for the Datathon@LISH
30 teams · 9 months ago
Overview
Data
Code
Discussion
Leaderboard
Rules
Late Submission
Overview
Description
Evaluation

Access to potable water is vital to the health and wellbeing of communities around the world. A smart understanding of which waterpoints will fail can improve maintenance operations and ensure that clean, potable water is available everywhere. In this competition, you will develop a predictive model that solve binary classification task focused on predicting the operational status of water pumps throughout Tanzania, based on some the provided information about their installation context. We will evaluate prediction quality according to a Log-Loss Error Function. A good statistical model here could be valuable for helping organizations to efficiently allocate maintenance resources. We've sourced this problem from our friends at DrivenData, who are some awesome alumni of Harvard Business School and the School of Engineering and Applied Sciences.

By the rules of the Datathon@LISH, you may *not* use the following features on the Kaggle competition.

- The "Code" tab
- The "Discussion" tab
- The "Merge Teams" feature on the "Teams" tab
- The "Contact Users" feature on users profiles

Figure A7: The home page of the Kaggle competition for the Lagrange track.

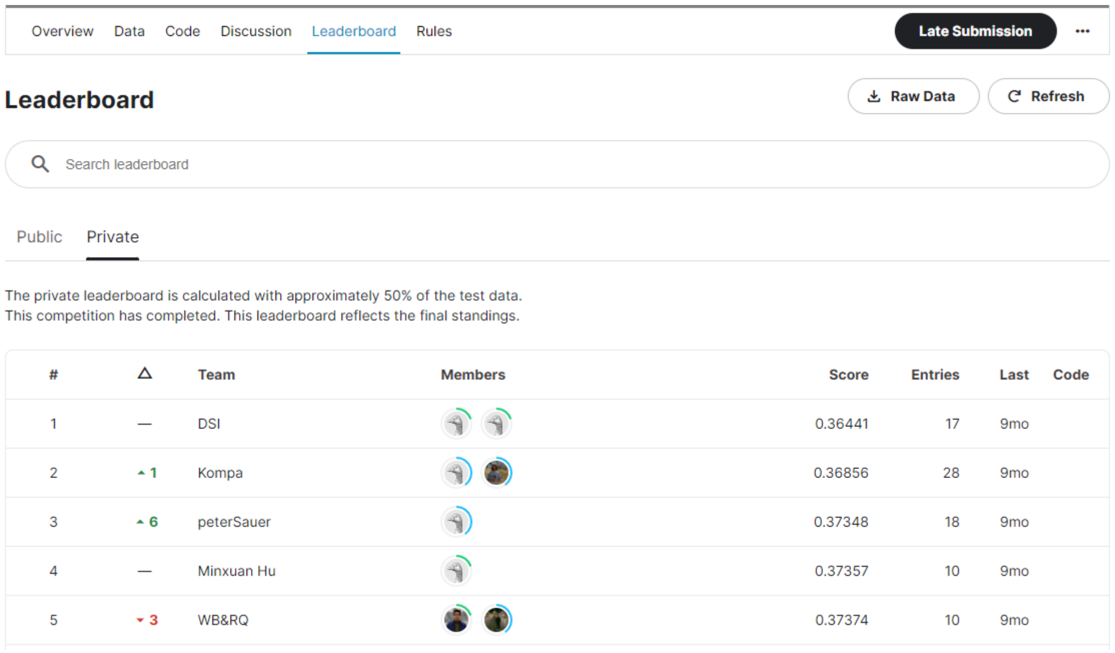
A.3 Kaggle Competition

We created a separate competition on Kaggle for each of the two tracks (see Figure A7). To submit to Kaggle, participants would generate a file with their model's predictions on the test set, then upload this file to Kaggle to be scored. 50% of the test set was used to automatically calculate the model's log loss on the ”public” leader board, which all participants could see throughout the competition. The remaining 50% of the test set was reserved for the ”private” leader board (see Figure A8). This is the leader board that was used to determine the winners of the competition, and the participants' performance on this leader board was not revealed until the contest was over.

B Survey Details

The pre-contest survey was administered on Qualtrics and allowed us to collect basic information from each participant for identification purposes (their Google account name, Kaggle username, affiliated university, etc.) We also asked information about their educational background and prior experience, including:

- Their undergraduate concentration, and coursework in specific areas (Data Science, Machine Learning, Statistics & Probability, etc.);



The private leaderboard is calculated with approximately 50% of the test data.
This competition has completed. This leaderboard reflects the final standings.

#	△	Team	Members	Score	Entries	Last	Code
1	—	DSI		0.36441	17	9mo	
2	▲ 1	Kompa		0.36856	28	9mo	
3	▲ 6	peterSauer		0.37348	18	9mo	
4	—	Minxuan Hu		0.37357	10	9mo	
5	▼ 3	WB&RQ		0.37374	10	9mo	

Figure A8: The private leader board of the Kaggle competition for the Lagrange track.

- Their familiarity with different programming languages;
- Their familiarity with different statistical modeling techniques and machine learning algorithms;
- Their familiarity with different software libraries in Python.

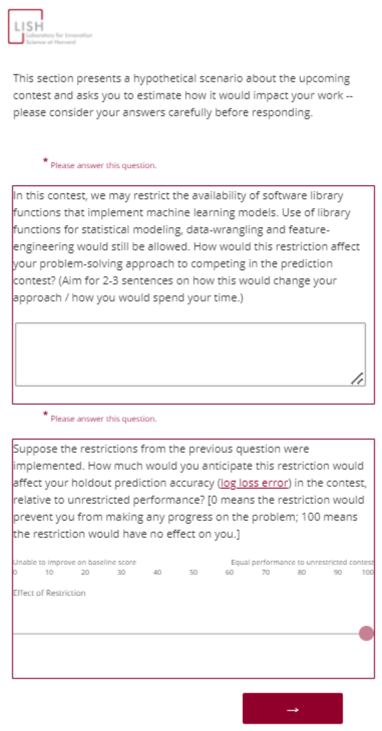
The pre-contest survey also asked respondents to answer the hypothetical question shown in Figure A9. This question asks participants to estimate the extent to which their success in the competition would be impacted by a restriction in their access to relevant software libraries.

The short post-contest survey asked participants to re-evaluate the degree to which library restrictions impacted (or would have impacted) their success now that they contest was over. It also asked them to describe their general problem-solving process, and reflect on what other approaches they might have taken if they had been given more time.

The full pre- and post-contest surveys can be found in the Supplement.

C Skill Analysis

Construction of Indices. Of the 39 variables that we constructed from our Qualtric Survey (detailed in Appendix B), we coded them according to our conceptual framework in Section



LISH Laboratory for Innovation Science at Harvard

This section presents a hypothetical scenario about the upcoming contest and asks you to estimate how it would impact your work -- please consider your answers carefully before responding.

* Please answer this question.

In this contest, we may restrict the availability of software library functions that implement machine learning models. Use of library functions for statistical modeling, data-wrangling and feature-engineering would still be allowed. How would this restriction affect your problem-solving approach to competing in the prediction contest? (Aim for 2-3 sentences on how this would change your approach / how you would spend your time.)

* Please answer this question.

Suppose the restrictions from the previous question were implemented. How much would you anticipate this restriction would affect your holdout prediction accuracy (log loss error) in the contest, relative to unrestricted performance? [0 means the restriction would prevent you from making any progress on the problem; 100 means the restriction would have no effect on you.]

Unable to improve on baseline score
Equal performance to unrestricted contest
0
10
20
30
40
50
60
70
80
90
100
Effect of Restriction
→

Figure A9: A hypothetical question asking respondents to estimate the impact of libraries on their work.

2.2.2, and then removed a variables according to the rules outlined in Section 3.3.4. We list the results of this exercise in Table A2. Ultimately, 16/39 (41%) of our originally measured variables were included our skill indices. 3x variables were removed because they were not balance between treatment and control conditions. 7x were removed because they did not vary sufficiently (they were either 1 or 0 for over 75% of respondents). 8x were specific skills that were removed because they were skills for tools that were not relevant to our problem. Lastly, 6x were removed because our specific/general distinction couldn't be applied clearly to those variables – they refer to prior experience with the predictive model development process and could be taken as measures of both specific or general skills.

Robustness of Skill Heterogeneity Results. Our results in Table 8 show the economic and statistical significance of the interaction of our skill indices with our treatment. Here, we present slightly modified tests to show the robustness of this result to alternative specification.

1. First, to better illustrate the main way we interpret results, we show the same specification, but using (selected) individual measures rather than our aggregate measure, in Table **??**. Specifically, we chose 1) whether the team had worked as a software developer, 2) whether they majored in a STEM major, and 3) whether they were com-

Skill Variable	**Skill Type**	**Reason Removed**
`prioremploy_research_any`	General	Difference between Tracks
`priorcourses_programming_gte2`	General	Low Variation
`priorcourses_stats_gte2`	General	Low Variation
`priorlanguages_python_comfortable`	General	Low Variation
`priorlibraries_numpy_comfortable`	General	Low Variation
`is_MajorStem`	General	
`priorcourses_algorithms_gte2`	General	
`priorcourses_datascience_gte2`	General	
`priorcourses_os_gte1`	General	
`prioremploy_softwaredev_any`	General	
`priorlanguages_sql_comfortable`	General	
`priorstatmodels_glms_comfortable`	Specific	Difference between Tracks
`priorlanguages_julia_comfortable`	Specific	Irrelevant to Problem
`priorlanguages_matlab_comfortable`	Specific	Irrelevant to Problem
`priorlanguages_stata_comfortable`	Specific	Irrelevant to Problem
`priorlibraries_tensorflow_comfortable`	Specific	Irrelevant to Problem
`priorlibraries_pytorch_comfortable`	Specific	Irrelevant to Problem
`priorlibraries_xgboost_comfortable`	Specific	Irrelevant to Problem
`priormlmodels_ensemble_comfortable`	Specific	Irrelevant to Problem
`priormlmodels_neuralnet_comfortable`	Specific	Irrelevant to Problem
`priormlmodels_svms_comfortable`	Specific	Irrelevant to Problem
`priorstatmodels_linreg_comfortable`	Specific	Low Variation
`priorstatmodels_logreg_comfortable`	Specific	Low Variation
`priorlibraries_pandas_comfortable`	Specific	Low Variation
`is_MajorDatasci`	Specific	
`priorcourses_ml_gte2`	Specific	
`prioremploy_datascience_any`	Specific	
`priorlanguages_r_comfortable`	Specific	
`priorlibraries_matplotlib_comfortable`	Specific	
`priorlibraries_scipy_comfortable`	Specific	
`priorlibraries_sklearn_comfortable`	Specific	
`priormlmodels_randomforest_comfortable`	Specific	
`priorstatmodels_regul_comfortable`	Specific	
`priormlstages_featureengi_comfortable`	Unclear Theory	Unclear
`priormlstages_modelbuild_comfortable`	Unclear Theory	Unclear
`priormlstages_modeleval_comfortable`	Unclear Theory	Unclear
`priorcontests_official_gte1`	Unclear Theory	Unclear
`priorcontests_unofficial_gte1`	Unclear Theory	Unclear
`priormlstages_edas_comfortable`	Unclear Theory	Unclear

Table A2: A list of all variables collected in our Qualtrics survey, and whether they were included in construction of skill indices that we describe in section 3.3. If variable was removed from the index, the reason is given. If no reason is listed, then the variable was included in our index. 16/39 variables were included in the final indices.

fortable with SQL as our examples of general skills. We chose 1) whether they were comfortable with sklearn, 2) whether they were comfortable with regularization, and 3) whether they had taken prior courses on data science as our examples of specific skills. Here we can see that our results directionally extend to the individual variable level, albeit with varying magnitudes and levels of statistical significance. The defining feature of our result – that the size of the interaction exceeds that of the direct effect of skills – also extends to the individual variable level. The point here is that our notion of general and specific skills provides our conceptual framework for *organizing* these individual-measure specifications into a coherent and consistent framework.

2. Second, one may be concerned with the specific variable we removed from our skill indices. We argue that each of our filters outlined in 3.3.4 (low variation in variable, differences between tracks, irrelevance to contest problem, or unclear theory) strengthen the interpretability of our result. The only purely statistical reason to remove variable was the first reason: low variation in variable. In Table A4, Model (2), we run an alternative specification where we include the Low Variation variables into the indices. We find that our results are robust in both economic and statistical significance to inclusion of these variables.

3. Lastly, one may be concerned that our results rely on a binary index. We run the same specification, but leveraging a standardized, *continuous* measure of our skill measures, presenting the results in Table A4, Model (3). Here, we show that our results are consistent in terms of direction, economic, and statistical significance with this alternative specification. Further, this alternative specification provides a sense of *magnitude* of our interaction effect – it can be interpreted as saying that a 1-standard-deviation increase in general skills over average *negatively* interacts with `Unrestricted`, with normalized Score an average of -0.21 lower than those with average general skills. By contract, a 1-standard-deviation in specific skills *positively* interacts with `Unrestricted`, with a normalized Score an average of 0.22 higher than those with average specific skills. We prefer our binary indicator specification because the precise meaning of '1-standard-deviation' requires an assumption that all the skills we measure are of equal importance, which is unjustified a priori. Our binary indicator better represents our assumption that we can capture a direction change in skill, but not an accurate absolute measure.

	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6
Dependent Var.:	score	score	score	score	score	score
unrestricted	0.2688** (0.0730)	0.3243** (0.0638)	0.2908** (0.0709)	0.2152** (0.0802)	0.2549** (0.0762)	0.2990** (0.0756)
had_prior_softwaredev_job	-0.0553 (0.0730)					
had_prior_softwaredev_job x unrestricted	-0.2760^ (0.1460)					
is_MajorStem		-0.1586* (0.0638)				
is_MajorStem x unrestricted		-0.2331^ (0.1277)				
sql_comfortable			-0.0997 (0.0709)			
sql_comfortable x unrestricted			-0.2203 (0.1418)			
sklearn_comfortable				0.0374 (0.0802)		
regularization_comfortable					0.1075 (0.0762)	
prior_datasci_courses						0.0313 (0.0756)
(Intercept)	0.5682** (0.0365)	0.5809** (0.0319)	0.5632** (0.0355)	0.5615** (0.0401)	0.5411** (0.0381)	0.5594** (0.0378)
Observations	68	68	68	68	68	68
R2	0.25090	0.28333	0.25119	0.25550	0.23182	0.20661
Adj. R2	0.21578	0.24974	0.21609	0.22060	0.19581	0.16942

Table A3: Individual Skill Measures - Interaction Effects.

	Model 1	Model 2	Model 3
Test:	Original	Add Low Variation Measures	Continuous Skill Variable
Unrestricted	0.2812** (0.0593)	0.2359** (0.0609)	0.2749** (0.0599)
High Skill (General)	-0.0980 (0.0617)	-0.0694 (0.0611)	-0.0767* (0.0341)
Unrestricted x High Skill (General)	-0.2655* (0.1233)	-0.2516* (0.1225)	-0.2133** (0.0682)
High Skill (Specific)	0.0772 (0.0603)	0.0488 (0.0606)	0.0474^ (0.0282)
Unrestricted x High Skill (Specific)	0.2787* (0.1205)	0.3345** (0.1218)	0.2190** (0.0563)
High Skill (General) x High Skill (Specific)	-0.2656* (0.1243)	-0.2657* (0.1210)	-0.0557 (0.0339)
Unrestricted x High Skill (General) x High Skill (Specific)	0.5071* (0.2487)	0.5443* (0.2452)	0.1296^ (0.0678)
(Intercept)	0.5908** (0.0296)	0.5979** (0.0307)	0.5872** (0.0300)
Observations	68	68	68
R2	0.34072	0.29993	0.42513
Adj. R2	0.26380	0.21825	0.35806

Table A4: Robustness of Heterogeneity Measure.

D Time-Spent Analysis

As time spent on the contest is not random, we cannot directly measure its effect on our outcomes of interests. Instead, leveraging the observation that our contest had a variable start-time but a fixed end-time, we use an instrumental variable (IV) approach. Specifically, we develop an instrument based on the intuition that teams with a later start-time will exogenously spend less time on the contest. As the contest spanned a 48-hour period over three days (starting at day 1 at 5PM and ending on day 3 at 5PM), participants were allowed to complete the pre-contest survey any time after 5PM on day 1, after which they were given details of the contest problem. However, some teams did not complete the survey and start work until as late as the morning of day 3. Regardless of team start-time, the contest ended at 5PM on for all teams. We, therefore, define a variable `Contest Time` that measures the amount of time left in the contest when the team completes the survey and starts working on the contest. This variable satisfies the necessary IV assumptions for estimating the causal effect of `Work Hours` on score.

- First Stage (Relevance). Less overall time to work on the contest problem reduces the amount of time that a team spends working on the problem.
- Exclusion Restriction. Less time to work on the problem affects the team's final score only through the way that it limits the amount of time that teams can spend on the problem. It does not affect the nature of the problem or the resources teams have available to solve the problem.
- Exogeneity (Random assignment of the instrument). Teams did not know anything about the details of the contest problem or the treatment restriction until they filled out the pre-contest survey. So, any variation in start-time must be independent of the details of the problem itself, or participant's expectation of winning conditional on the problem details. Therefore, we argue that this variable is driven by participant busyness on day 1 and day 2 (recall our participants are largely students in the middle of an academic semester) – in a way that is exogenous to participant skills or motivation.
- Monotonicity. All else equal, having less overall time to work on the contest problem can only reduce a team's score. If a team were (counterfactually) granted more time, they are unlikely to do worse.

Therefore, we can leverage contest start-time as an instrument for time-spent on the contest. We implement this idea through a 2SLS model, where we control for the effect of our variation in the tools.

	(1) OLS	(2) First Stage	(3) 2SLS
Work Hours	0.006		0.051
	(0.006)		(0.028)
Contest Hours		0.189	
		(0.077)	
Unrestricted	0.289	-0.160	0.315
	(0.074)	(1.478)	(0.102)
(Intercept)	0.377	-1.110	0.056
	(0.081)	(3.631)	(0.221)
Num.Obs.	63	63	63
R2	0.214	0.095	-0.466
R2 Adj.	0.174	0.049	-0.540
F	5.346	2.059	

Table A5: Effect Time Spent on Score (2SLS).

We explore the effect of time on model quality. Surprisingly, in Table A5, Model (1), we find that time spent on the contest is barely correlated with final score (a coefficient of 0.006 Score improvement / hour spent). However, this result cannot be interpreted as the causal effect of time on model quality because the time-spent is endogenous to a variety of other factors such as skill. We present our instrumental variables model in Model (2) and (3). We find that, leveraging our IV specification, our estimate of the effect of time-spent on score increases an order of magnitude, to 0.05 Score improvement / hour spent (p=0.075). This provides evidence that there is some omitted variable that increases the score but reduces the endogenous amount of time spent working on the problem – for example, unobserved aspects of team experience. However, we can only speculate what this might be.

Given that the median amount of time spent on the contest was 4.9 hours and our standard deviation as 6.0 hours, a back-of-the-envelope calculator would expect that a one standard deviation increase in time spent corresponds to an improvement of contest score by 0.30. This implies that, on average, this score improvement is a comparable order of magnitude to that of our treatment restriction. This is perhaps surprising – we expected the effect of time to be larger than the effect of libraries. The magnitude estimate implies that equipping data scientists with the right tools makes them much more effective, perhaps even as effective as doubling the time available for them to work.

E Tools-as-Skills Mechanism

E.1 Comparative Statics

We present here the derivation of the comparative statics highlighted in Section 5.1. They are straightforward and simply involve repeated application of the chain rule and product rules for derivatives. First, we assume, per our assumptions above, that all $\beta > 0$, that $m > 0, m' > 0, m'' \leq 0$, and that all our variables are stocks, meaning that they are positive-valued. Then

Direct Effects:

$$\begin{aligned}
\partial_l f &= \beta_{sl} se \times m'([\beta_g g + \beta_{sl} sl]e) > 0 \\
\partial_g f &= \beta_g e \times m'([\beta_g g + \beta_{sl} sl]e) > 0 \\
\partial_s f &= \beta_{sl} le \times m'([\beta_g g + \beta_{sl} sl]e) > 0
\end{aligned}$$

Interaction Effects:

$$\begin{aligned}
\partial_s \partial_l f &= \beta_{sl} e \times m'([\beta_g g + \beta_{sl} sl]e) + \beta_{sl}^2 sle^2 \times m''([\beta_g g + \beta_{sl} sl]e) \\
&= \beta_{sl} e(f' + \beta_{sl} slef'') \\
&\implies \partial_s \partial_l f > 0 \iff |f'| > \beta_{sl} sle|f''| \\
\partial_g \partial_l f &= \beta_{sl} \beta_g se^2 \times m''([\beta_g g + \beta_{sl} sl]e) < 0 \\
\partial_g \partial_s f &= \beta_{sl} \beta_g le^2 \times m''([\beta_g g + \beta_{sl} sl]e) < 0
\end{aligned}$$

E.2 Extensions to the Mechanism

E.2.1 Effort Allocation across Tasks

We now develop a simple extension of our model to explain *how* teams allocate their effort in each treatment condition. The intuition captured in our model is as follows: because productivity is driven by the product of skill and effort, and because libraries are like skills, library availability directly affects the productivity of effort spent. Therefore, in any model where there are multiple tasks and participants must allocate their effort across these tasks, the allocation of effort will be affected by the available libraries for the tasks. As we restricted software libraries for modeling but not feature engineering, we expect that the availability of libraries for modeling purposes should shift participants' effort allocation towards modeling.

Formally, we assume model quality q is the product of two production functions, f_d and f_m, which correspond to the production functions for exploratory data analysis and modeling, respectively. We make the functional form of these production functions identical to Equation 5.1 (including the same sign assumption on the derivatives), except we remove the dependency of f_e on l, to match the structure of our actual treatment (recall that our library restrictions were only imposed on library modeling functions, not feature engineering library functions). We assume that effort is fixed and allocated by the team between the two tasks, where we parameterize effort modeling as the variable e. Finally, we assume that effort is allocated so as to maximize model quality.

$$\begin{aligned} q &= f_d([\beta_{gd} + \beta_{sd}s][1-e]) + f_m([\beta_{gm}g + \beta_{slm}sl]e) \\ e^* &= \operatorname{argmax}_e q(e) \end{aligned} \tag{5}$$

Even given relatively general assumptions, the model yields a sharp prediction about how library availability affects how effort is spent: it predicts that as l increases, e^* also increases. That is, as library availability increases so does the effort. We present a proof of this in the appendix, but the intuition is straightforward. In this model, the allocation of effort is an explicit tradeoff between the quality of the two tasks. Therefore, the optimal effort allocation balances the benefit of adding one infinitesimal unit of effort spent on feature engineering to the cost of removing that unit of effort from modeling. When l increases, the marginal benefit of effort on modeling increases – and so the effort allocation shifts towards modeling. An alternative viewpoint might expect that removing libraries would require teams to spend *more* effort on the task, but this view does not take into account that per-task effort is not fixed and participants can choose how to allocate effort across multiple tasks.

We present three sources of evidence consistent with this prediction: content from contest winner blog posts, results from our survey, and results from teams' contest notebook data. Although none of our evidence constitutes a formal statistical test, the data is convergent on the idea that library restrictions shift effort allocation across tasks.

First, Table A6 presents excerpts from short blog posts written by our contest winners detailing the problem-solving approaches they took during the contest. The blog posts differ across tracks both in their description of feature engineering and model building. For feature engineering, the restricted teams focused on developing a more intuitive understanding of the data. As one particularly salient example of this, each of the top three winners in the restricted track noted the relevance of spatial clusters in predicting pump status, in a way that directly influenced their modeling approach. One group operationalized this understanding by choosing to use hierarchical clustering as a way to summarize relevant covariates and structure their logistic regression. Another group focused on constructing grid measures that captured the inherent nonlinearity present in latitude and longitude data (see Figure 7). Notably, these restricted teams also stated that, while intuitively-motivated features like spatial clusters were useful, more automatically generated features based on arbitrary interactions were not beneficial to model performance. By contrast, even though the unrestricted top three winners did a large amount of feature engineering, they engaged more abstractly with the data. These team primarily focused on data issues like redundancy, missing values, grouping of categories, and systematic generation of features and feature interactions. For model building, the restricted track top three winners scarcely commented on their modeling approach, with each group settling on variations of logistic regression. They instead allocated attention to searching for hyperparameters to guide feature selection (via regularization). By contrast, each of the three unrestricted track winners spent a substantial amount of effort and computational resources exploring the usage of different modeling packages. After some experimentation, each group landed on some form of ensemble model, with different forms of random forests serving as the final base model in all cases.

Second, we present evidence from the post-contest survey, where we asked teams to reflect on their experience with the competition and describe their problem-solving approaches. We manually coded team responses into three categories: Exploratory Data Analysis, Feature Transformation, and Model Selection & Tuning, with each team's response possibly falling into multiple categories. To verify the qualitative answers with the respondents' submitted notebooks to ensure that their narrative was consistent with their submitted code. Our results in Table A7 show that around two-thirds of respondents emphasized only one of these three approaches. The remaining third identified two or more approaches as vital to their solution. Most notably, 86% of the participants in the unrestricted track emphasized the importance of Model Selection & Tuning in developing their solution, with responses describing many different algorithmic approaches (random forest, gradient boosting, neural networks, etc.) and tuning grids of different model hyperparameters. By contrast, only 33% of the restricted team emphasized their modeling approach, describing traditional model selection with generalized linear models and hyperparameter tuning for regularization. Additionally, 72% of participants in the unrestricted track stressed the importance of Feature Transformation, compared to only 43% of participants in the restricted. We saw a similar difference for Exploratory Data Analysis. These results show stark differences in how participants in the two tracks perceived the importance of different tasks.

Finally, we develop evidence for differences in time allocation across tasks based on our data on time spent by notebook subsection. In theory, our data can be used to directly test for differences in effort driven by our library restrictions. However, due to our small sample

Track	**Unrestricted**	**Restricted**
Data Engineering	We found many redundant variables that could be dropped, and we reduced the remaining categorical variables to their top K categories, choosing K to be the elbow in each categorical variable's sorted frequency plot. ...We also found that a few variables would benefit from some median-value imputation, such as construction year and GPS height. The possibility of ordinal encoding and one hot encoding was ruled out for the categorical variables by experimentation due to the absence of a clear relationship between classes and due to the presence of a significantly high number of classes in a feature.	For high cardinality fields, we chose a top-k threshold based on elbow graph of term frequency and lumped the tail terms into ”other”. We also discovered that ”lga” and ”ward” are two more informative location variables... We learned that mal-functioning wells usually have ”dry” for quantity and ”other” for water point type, and they tend to cluster geospatially. We hypothesize that there existed discrete clusters within the data that could be used to improve our predictive performance... We performed “Hierarchical K-Means Clustering” to form a tree of clusters and performed regression on each leaf cluster.
Modeling Approach	Different tree-based and Boosting models such as Random Forest, XGBoost, LightGBM, CATBoost were tested and the best model (CATBoost) was selected based on the validation accuracy. We did a grid search over [random forests] and [XGBoost's Ensembles with Hyperparameter Tuning], beating our previous scores. In parallel, we attempted a massive Bayesian hyperparameter search for an [Multi Layer Perceptron], but after hundreds of iterations, we never saw a validation log loss less than 0.45 and abandoned the MLP altogether.	*[Every single one used logistic regression with grid search for regularization. No one commented on this much.]*
How to Allocate Time	A lot of the time was actually spent on figuring out how to put everything together, researching potential pitfalls to implementing random forests, and time spent on ideas not working out whether due to results (like regularizing the model) or inability to implement on time (like oversampling and undersampling).	Given more time, we would explore adding interactive terms and stacking an ensemble layer to current models.

Table A6: Qualitative Evidence from Winner's Blog Post.

	Exploratory Data Analysis	**Feature Transformation**	**Model Selection & Tuning**
Unrestricted	6 (15%)	28 (72%)	13 (33%)
Restricted	2 (7%)	12 (43%)	24 (86%)

Table A7: Coded Analysis of Survey Responses. Note: Percentages are normalized within competition track. The results sum to greater than 100% because some participants mentioned more than one of the approaches in their response.

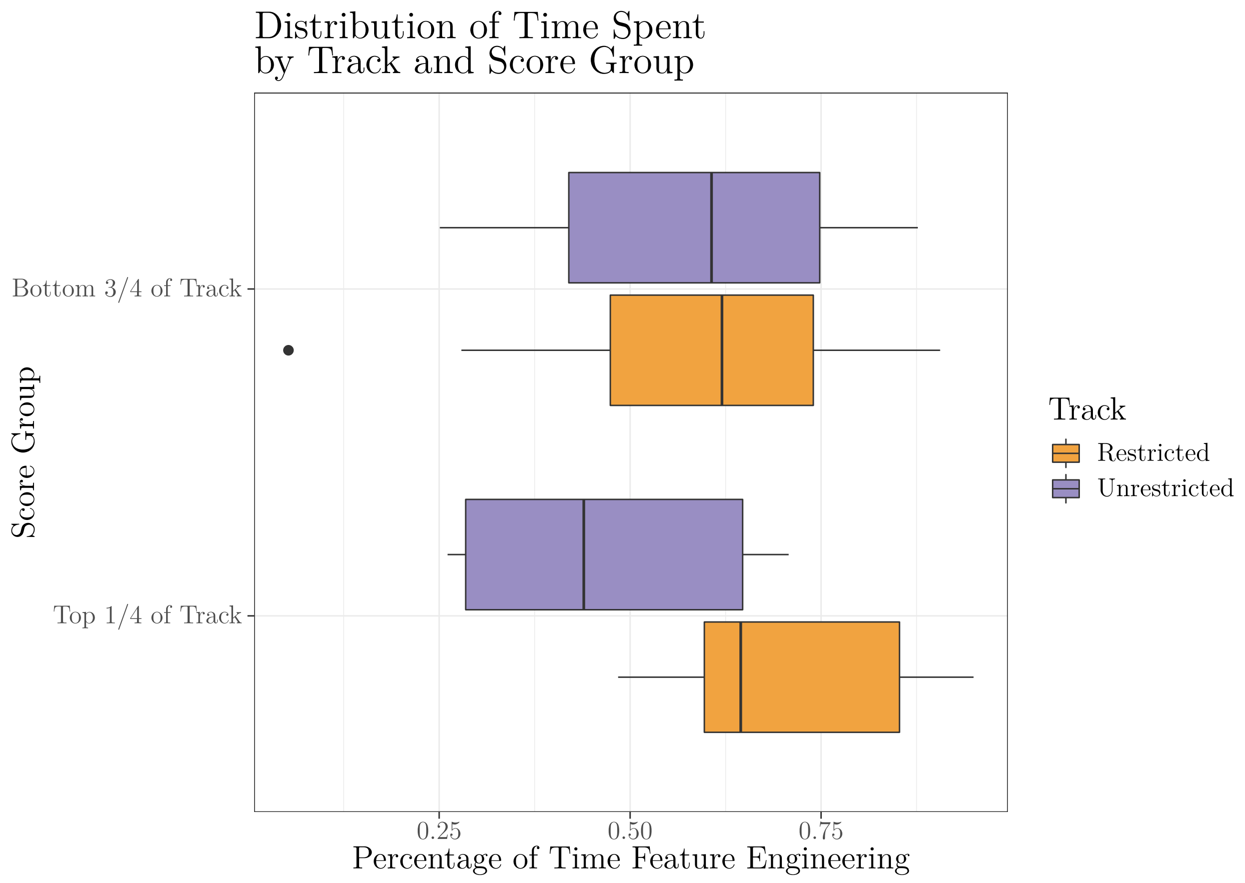


Figure A10: This boxplot compares the distribution of `Time Spent` between the Restricted and Unrestricted tracks for the bottom 3/4 and top 1/4 of each track.

and large variability, we are underpowered to detect any meaningful difference. Instead, we present suggestive but striking visual evidence of an effect in Figure A10. In this figure, we plot the distribution of percent time spent on feature engineering for each team. We further split our population into two subsamples based on contest score: one sample comprising the bottom three quartiles' score in each track and the other comprising the top quartile by score. In the sample of the bottom three quartiles, we do not observe any differences in the distribution of time allocation; however, in the top quartile, there is a large difference consistent with our theoretical prediction and other qualitative findings. This suggests that our task allocation model applies primarily to teams that are more skilled and know how to maximize the value of their time; conversely, it suggests that less skilled participants may not know how to best allocate their effort given the library restrictions. This result is qualitatively robust to different cutoffs used to define the subsamples. However, we want to emphasize that it is not statistically significant and is by no means conclusive.

These three streams of evidence suggest that library availability changed participant effort allocation, causing them to spend more effort and, in turn, more focus modeling.

E.2.2 Tasks as Substitutes

The above section justifies the empirical usefulness of including two types of tasks in our model as a way to explain effort allocation between feature engineering and modeling. However, such a choice raises a natural question: are these two types of tasks inherently complementary or substitutable? Our model does not make a prediction on this front and can be generalized to accommodate either type of relationship between the two task types via the functional form:

$$q = f(f_d + f_m) \tag{6}$$

where have suppressed the arguments of f_e and f_m for notational clarity. Here, we assume that f is positive and monotonically increasing in the input. Given the monotonicity of f, the model supports the theoretical predictions developed in prior sections. However, the model exhibits complementarity or substitution between feature engineering and modeling if f is convex or concave, respectively[33].

Understanding if the two types of tasks are inherently complementary or substitutable has significant practical implications. On the one hand, theoretical discussions describe advanced ML techniques like deep learning models as automated "feature learning" (Yu et al., 2013). This suggests that advanced ML models remove the need to understand the non-linearities in the data generating model, or similarly, that understanding the non-linearities present in the data generating model would eliminate the need for advanced ML models. On the other hand, substantive knowledge of the problem is thought to allow data scientists to improve their predictive model, justifying the practices of including subject matter experts on teams of data scientists. Whether these tasks are complementary or substitutable would thus have implications for the design of teams.

We argue that, in our context, feature engineering and modeling are substitutable with each other – that quality modeling mitigates the benefit of feature engineering work. An ideal statistical test of this relationship would exogenously vary the amount of effort on feature engineering or modeling and then study the interaction effect between those quantities and the final model score. We, unfortunately, cannot conduct this type of test given our data. However, we can run a simulation combining the engineered data from the winners of the

[33]We assume q is a nested model, which could be, for example, parameterized as a nested CES model like the ones used in energy economics (Lagomarsino, 2020)

restricted track and the (untrained) model structure of the unrestricted track winners. The logic of this exercise is that the restricted track focuses more on feature engineering and, therefore, should have data features that more directly capture nonlinearities present in the data, whereas unrestricted track teams should have better models. Consequently, we expect this synthetic predictive model to outperform the original models of the unrestricted track winners.

To do this, we trained synthetic models in two ways. In the first approach, we extracted the complete set of features engineered by the top three winners of the restricted track ("the variables") and the final model specifications of the top winners ("the models") of the unrestricted track. The models were re-trained using the variables directly, including optimizing the hyperparameter over the same grid the top winners used to tune their final model. In the second approach, we broadened the set of model specifications considered, rerunning the complete grid of model specifications attempted in the actual competition by the unrestricted track winners. For example, if a participant tested several different algorithms and eventually chose a random forest as their final model specification, we re-trained all the algorithms they tried using the additional data and selected the best specification.

The results of this simulation exercise are presented in Table 10. We surprisingly find that, under every situation considered, the synthetic models did not perform better than the best-performing models in the unrestricted track. In some cases, the synthetic model does much worse than the original model. This is surprising because, under most models with separate tasks, even moderate amounts of substitution between feature engineering and modeling would result in better synthetic models. Instead, the results of our (simple) simulation exercise suggest that feature engineering and model substitution are strongly substitutable. This would be consistent with the idea that more advanced machine learning algorithms used by the unrestricted teams were already capturing the non-linearities in the data set that the restricted participants captured with extensive feature engineering.