Tooling for Developing Data-Driven Applications: Overview and Outlook

Thomas Weber
LMU Munich
Munich, Germany
thomas.weber@ifi.lmu.de

Heinrich Hußmann
LMU Munich
Munich, Germany
hussmann@ifi.lmu.de

ABSTRACT

Machine Learning systems are, by now, an essential part of the software landscape. From the development perspective this means a paradigmatic shift, which should be reflected in the way we write software. For now, the majority of developers relies on traditional tools for data-driven development, though. To determine how research into tools is catching up, we conducted a systematic literature review, searching for tools dedicated to data-driven development. Of the 1511 search results, we analyzed 76 relevant publications in detail. The diverse sample indicated a strong interest in this topic from different domains, with different approaches and methods. While there are a number of common trends, e.g. the use of visualization, in these tools, only a limited, although increasing, number of these tools has so far been evaluated comprehensively. We therefore summarize trends, strengths and weaknesses in the status quo for data-driven development tools and conclude with a number of potential future directions this field.

CCS CONCEPTS

• General and reference → Surveys and overviews; **• Software and its engineering → Software notations and tools**; **• Computing methodologies →** *Machine learning*; *Artificial intelligence.*

KEYWORDS

literature review, software development, tools, machine learning, data-driven development

ACM Reference Format:
Thomas Weber and Heinrich Hußmann. 2022. Tooling for Developing Data-Driven Applications: Overview and Outlook. In *Mensch und Computer 2022 (MuC '22), September 4–7, 2022, Darmstadt, Germany*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3543758.3543779

1 INTRODUCTION

The ubiquity of systems that leverage large amounts of data, and often Machine Learning, have turned their development from an academic exercise to an established development paradigm. Dedicated data-driven development is becoming quite common and will, in all likelihood, eventually be just another part of industrial software development.

Yet, the most common way for defining how a software behaves has not dramatically shifted in decades. Particularly the popular programming languages rely on explicitly defined behavioral patterns, typically described by a sequence of instructions. And even with more declarative tools, like functional or logical programming, the knowledge of the desired behavior must be transferred from the developers mind to the machine via some form of, typically written, explicit definition of behavior.

For Machine Learning systems, on the other hand, the description of the behavior is typically implicit and embedded in the data and the developers task is to provide the necessary framework to extract it. So, while at first glance, a piece of code that extensively uses Machine Learning may appear very similar to its traditional counterpart. Yet, while in the latter the developer has defined the algorithm in detail, the former code would usually describe the management and directing of data such that the behavior can be inferred from it, making it much closer to the idea of *programming by example* [57]. This marks an important change in the nature of the programs and the activities of their developers, leading to some referring to data-driven software as "Software 2.0" [41].

While this may mark a paradigmatic change in the approach to software development, from the tooling perspective many of the traditional tools, like simple text editors, command line interfaces and integrated development environments (IDEs) remain the popular choice for developers. These offer a wide array of support mechanisms, e.g. for dependency management or debugging. Management of a large volume of data and its exploration, however, are often done not as easily and traditional debugging with breakpoints can only get a developer so far when the programs behavior is embedded in the data and not in the lines of code.

Using existing tools certainly works, and those who pioneered Machine Learning and those who now use it had and have to build on top of eixisting technology and use the tools at hand. However, if the data-driven development paradigm is to be carried into everyday development, now may be a good time to consider and evaluate whether tools that drive software development since its early beginnings are still adequate for this change in paradigm, new practices and a broader audience that may not have a traditional programming background but wishes to participate in the development process.

The fact that the simple text editor and IDEs are very common for data-driven development does, also, not mean that there are no alternatives. Both industry and academia have created a myriad of tools of the years to make software development, including for data-driven software, easier and more accessible. This is particularly relevant when development becomes a team effort, for not everyone

MuC '22, September 4–7, 2022, Darmstadt, Germany

ACM ISBN 978-1-4503-9690-5/22/09…$15.00
https://doi.org/10.1145/3543758.3543779

may be comfortable using typical programming tools. For all their benefits, their adoption is lagging though.

Their limited use in practice may have many reasons that are hard to pinpoint, particularly when there is only little effort to evaluate existing tools. Moving forward with new types of tool support in the future, it is crucial to understand what has already been tried, what tools exist, and what issues have already been addressed, though.

To provide an overview of the tooling landscape and particularly systematic evaluations of tools for the development of data-driven applications, this paper reports on a review of the academic literature (Sect. 2). Through systematic labeling (Sect. 2.2) , we were able to determine some popular avenues that have already been explored and evaluated, representing the state-of-the-art, and areas that still remain open (Sect. 3). Based on these, we discuss possible future directions and what upcoming tools may look like and what needs to be done to ensure that they are successful (Sect. 4).

2 METHOD

In order to gauge the state of the literature and to get an overview of existing development tools for data-driven software and their evaluations, we conducted a literature research, following the PRISMA guidelines [67] which we outline in the following section. The goal of this survey is to determine which aspects of data-driven software development must currently rely on previously existing tools are which are already well supported with novel and dedicated tools. We are particularly interested in indicators that tell us how well tools work, that were specifically created to support data-driven development.

2.1 Search and Filtering

To determine the efficacy of existing tools, we need to rely on form of analysis or evaluation. We therefore queried large publication databases, specifically the ACM Digital Library, the IEEExplore and the proceedings of the AAAI conference, for publications that matched this research focus. The first two represent the major publication databases for computer science literature, while we included the AAAI proceedings specifically, because the AAAI conference is a major venue for related literature that is typically not listed in either of the other databases.

Arguably, there are also tools coming exclusively from industry that try to address the needs of developers. However, either there is little to no evaluations publicly available for some of these or these tools have been evaluated as part of a publication and should therefore show up when querying the publication databases. We accept the possibility of missing out on tools that have been evaluated and the results being available *somewhere* but not as part of a scientific publication, since comprehensively finding such evaluations is unfeasible and their number should be relatively small. This would include especially in-house tools, which may be used at some industry companies to great success, but as long as they are not public, they offer only limited value for developers at large and thus cannot contribute to this overview.

Given that scientific research in the fields of Machine Learning, Artificial Intelligence and data-driven application is very active, terminology can sometimes be in flux. We therefore decided to start our search with broad queries and refine the results later based on manual selection.

We therefore queried each of the publication databases using the following search criteria:

- The term "tool" had to be present in the abstract to limit the results to publications that had tooling as a focus.
- "data-driven", "machine learning" or "artificial intelligence" as the general research field also had to occur in the abstract.
- "developer" or "development" as our intended target group could be anywhere in the text. We decided for this broader query here because an initial scan of promising literature revealed that the target group was often left implicit in the abstract.
- Considering the major pace at which the field is changing, we furthermore decided to only consider publications of the last ten years. Tools that are more than a decade old and have not received any attention in the meantime clearly have limited relevance for current and future research and development.

Joining these criteria was trivial for the IEEExplore and the ACM DL, both of which provide a powerful advanced search. For the AAAI database, no such option was available though, so we resorted to downloading the metadata of 9 717 available publications from the last ten years and filtered them ourselves. Naturally, since we could not download these nearly 10,000 full texts, we filtered the third criterion, i.e. the term "developer" or "development" only in the available metadata (abstract, keywords, etc.) for the AAAI publications. Additionally, we omitted the query pertaining to Artificial Intelligence, as this is the topic of the conference series to begin with and should trivially be the subject of all the publications published there.

The results of our query are listed in Table 1. The table also provides the number of results, which remained after screening, which we did in an multi-stage process (cf. [67]).

After extracting the list of meta-data for each publication from their respective databases, we first filtered them by title. Since our search focused on tool support for development experts, the primary inclusion criterion here was whether the publication would introduce or evaluate a new or existing tools in this context respectively. At this stage we also excluded all those texts where the title clearly indicated that the tools were not intended for software development but for example for end users of various domains, e.g. software tools for medical diagnoses.

Given the ambiguity and broad use of the search terms, particularly "tool" and "development", this already substantially reduced the number of relevant publications to about a tenth of the initial search results.

In the second stage we then read the abstracts of the remaining papers, applying the same filter criteria, i.e. include tools that were specifically concerned with the creation of data-driven software.

We excluded publications that only applied data-driven principles or ML to domain problems, e.g. those that apply Machine Learning to medicine, manufacturing, etc. but also those that apply these techniques to, for example, conventional software repositories to automatically detect defects via ML. While the latter certainly fall in the intersection of software engineering and data-driven software, the goal of this survey is to determine how to specifically enhance development experience *of* data-driven, not *using* data-driven software.

Technical publications that reported on the development or improvement of algorithms but did not integrate them or provide them in a dedicated tool, we also removed from the list of relevant texts at this point. This left us with a manageable amount of 76 publications in total. A full list can be found in the addendum to this text.

2.2 Labeling

Using the full text of those 76, we labeled each publication in two ways.

First, we had a number of pre-defined criteria that we were interested in:

- As a rough classification we looked at what kind of tool support each paper describes? Is it about an existing tools, did the authors improve a tool from prior work or did they creates a completely new solution. This provides us with a general overview of the tool landscape in this area.
- Since, as mentioned, adoption of existing tools from academia is limited, we also recorded whether publications went beyond a purely technical description and reported on on real-world application of their tools or any other forms of practical evaluation in the form of case studies, field studies, etc. to gauge the effectiveness of these tools.
- To determine areas in need of additional research, another focus was what aspect of the development process these tools focus on? Since each tool can support steps specific to data-driven development but also integration into conventional software engineering, we use the terminology of Hesenius et al. and the EDDA process [31]. The EDDA process is an extension of traditional software engineering processes to capture the additional requirements of data-driven development. Specifically, it adds additional steps for assessing the suitability of Machine Learning, *data exploration* and subsequent *model requirements*, *model development*, and *integration*. By categorizing the tools according to the steps in this process, we can determine whether the whole process is similarly well supported or whether some steps receive more attention than others.

Beyond this we performed open labeling of the publications, which we subsequently clustered into categories. The following chapter will list these labels and further results.

3 RESULTS

Our search and filtering resulted in 76 publications pertaining to tools for data-driven development. The following section will describe these publications in further detail and highlight shared topics and point out differences.

3.1 Metadata

First we looked at the metadata of the publications, starting with the publication date. While we specifically selected only publications from the last ten years, it is noteworthy that even during this period of time, we can see a trend of increasing research interest.

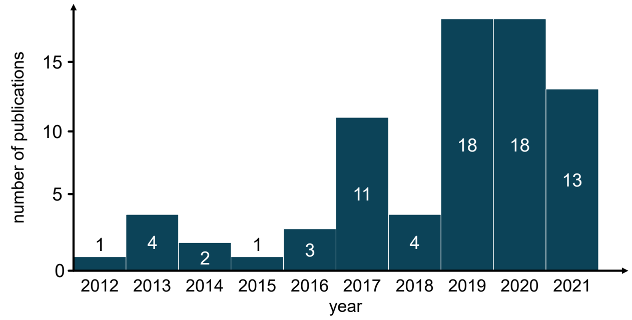


Figure 1: Distribution of the publication date of the papers in our selection.

Fig. 1 shows this steady increase. This also matches the results of similar surveys of various topics related to Machine Learning, which generally find that this development paradigm is becoming more popular in practice and research [95].

This is also reflected in the breadth of domains that publish about data-driven development tools. Naturally, Machine Learning and data-management venues are represented by our sample of papers (16 times), but also other areas of computing, like general software engineering (12 times), education (5 times), and particularly human-computer-interaction (16 times). Considering that tools a form of human-machine interaction, this comes as no surprise, though.

Notable, however, is the fact that our search criteria only yielded a single publication from the AAAI conference series. While, certainly, the complete database of AAAI publications is smaller than ACM DL and IEEExplore, we found the primary reason for this to be that these publications tend to be either highly focused on the underlying technology or report on case studies where Machine Learning was applied to solve a problem in practice.

3.2 Research Topics

Opposed to that, of the 76 publications we extracted, only one reported on algorithmic improvements in the context of automating and thus simplifying development [18]. **Automating** development in general was a popular topic in the full set of search results, but often with the goal of eliminating the developer from the software creation process – and thus of limited interest to our question of how tools can support developers. However, partial automation also was a topic of interest for 14 of the 76 publications, again with an increase in the recent years, e.g. for entity matching [89], prediction [73] and optimization [64]. Particularly the use of meta-learning [87], or "AutoML" systems, i.e. ML systems that automatically learn their ideal configuration, seem to be of interest for tool developers [56, 78, 89, 94].

Just as automation is of interest from a technical perspective, they also end up in dedicated tools for developers with the goal of simplifying model development and optimization (e.g. [56, 89]. ATMSeer [90], for examples, extends AutoML systems with visualizations to give developers feedback about the systems performance and progression.

Researchers, however, recognize that complete automation may not always be feasible or desirable [94]. Particularly the general

Table 1: Total number of publications throughout the search and filtering process.

	AAAI Proceedings	ACM Digital Library	IEEExplore	**Total**
Records retrieved from databases	80	839	592	**1511**
Screened by title	9	95	67	**171**
Screened by abstract	1	48	27	**76**

opacity and black-box nature is viewed critically. If even the developers no longer have means for understanding their complex software, it will become hard to debug [46], evaluate [32, 49], and control [79]. Works like "ATMSeer" by Wang et al. [90] or "CertifAI" by Sharma et al. [79] thus attempt to make the advances from the research on **Explainable AI (XAI)** accessible as tools for developers through interaction or examples and visualization respectively (cf. also [23, 46]). In this field, there are also a number of structured evaluations, e.g. a study by Kaur et al. [42] of how XAI tools are used by developers and whether they achieve their goal of informing them.

For these explanations, but also for development tools in general, another major topic appears to be how to visualize the complexity of ML. Many researchers recognize that in the traditional, code-based format, it can be hard to understand what is going on. For this reason, 12 publications specifically investigate the use of **visualizations**, e.g. for data preparation [84], development [44, 92] and evaluation [49]. Typically, they use various graphical presentations to offer real-time, and sometimes interactive, visualizations of various metrics of the ML models as shown in Fig. 2.

In addition, many of the tools mentioned or investigated in 24 of the publications have some visual component a use graphical programming. Especially the graphical programming aspect is often directly inspired by the domain of model-driven development (MDD), like the work by Zhang et al. [96], which aspires to bring MDD and data-driven development together.

This naturally also includes a number of analyses and evaluations of existing graphical tools for data-driven development, like RapidMiner [7, 38, 72] or Orange [80], which Bjaoui et al. [7] consider for novices and Shastri [80] for non-programmers respectively.

Some of these tools have been around for quite a while – in the case of RapidMiner for two decades now [72] – and have been used for various domains (e.g. [27, 47, 54, 59]). It therefore comes as not surprise that the evaluations highlight a number of benefits of these graphical tools, particularly with respect to visual organization and managing complexity. Yet, no single graphical tool has managed to become as popular as code-based tools like Jupyter Notebooks [39].

Kery et al. [44] and Zhao et al. [98] therefore report on attempts to bridge the divide between code and visualization via widgets that allow switching between these representations. The results of their evaluations in the form of a user study indicate that data scientists view this flexibility and the ability to hide and show code on demand very favorable, which may be an important factor for adoption.

3.3 Supporting the Development Process

As previously mentioned, we also specifically looked into which steps of the development process different tools aim to support.

Table 2: The steps of the EDDA process [31] and the publications that introduce or evaluate tools that support this step in some way. Publications not listed did not introduce tools to specifically support individual steps of the development but either provided general support or helped with activities related to the development but not captured by the EDDA process.

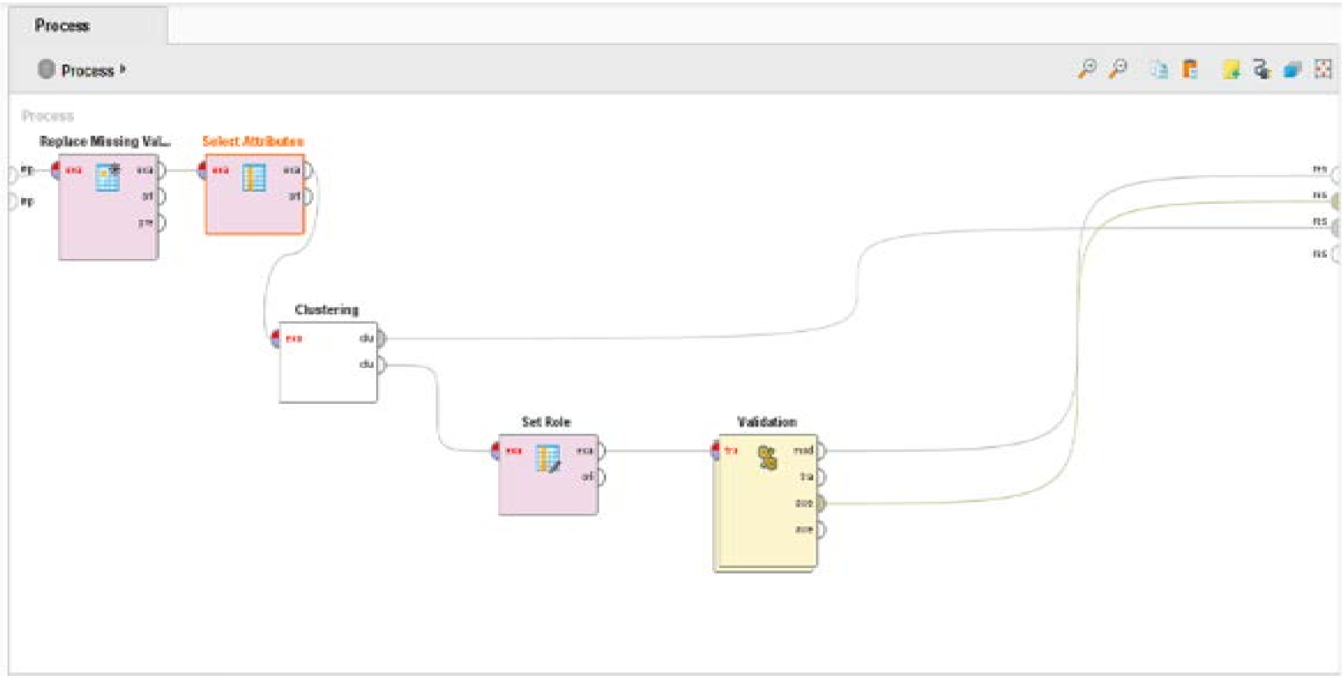
Activity	Publications
Requirements Engineering	[99]
Specification and Design	—
Evaluation of ML suitability	[17, 99]
Data Exploration	[2, 3, 12, 26, 70, 83, 84]
Model Requirements	—
Model development	[3, 12, 18, 30, 32, 48, 56, 58, 65, 68, 74, 85, 88, 90, 92, 98]
Model integration	[50, 96]
System Test & Evaluation	[11, 23, 32, 33, 46, 49, 79, 90, 97]
Operation & Maintenance	—
End-to-end	[10, 30, 49]

Table 2 provides an overview of the different steps of the EDDA process [31] and the publications from our selection that support each step.

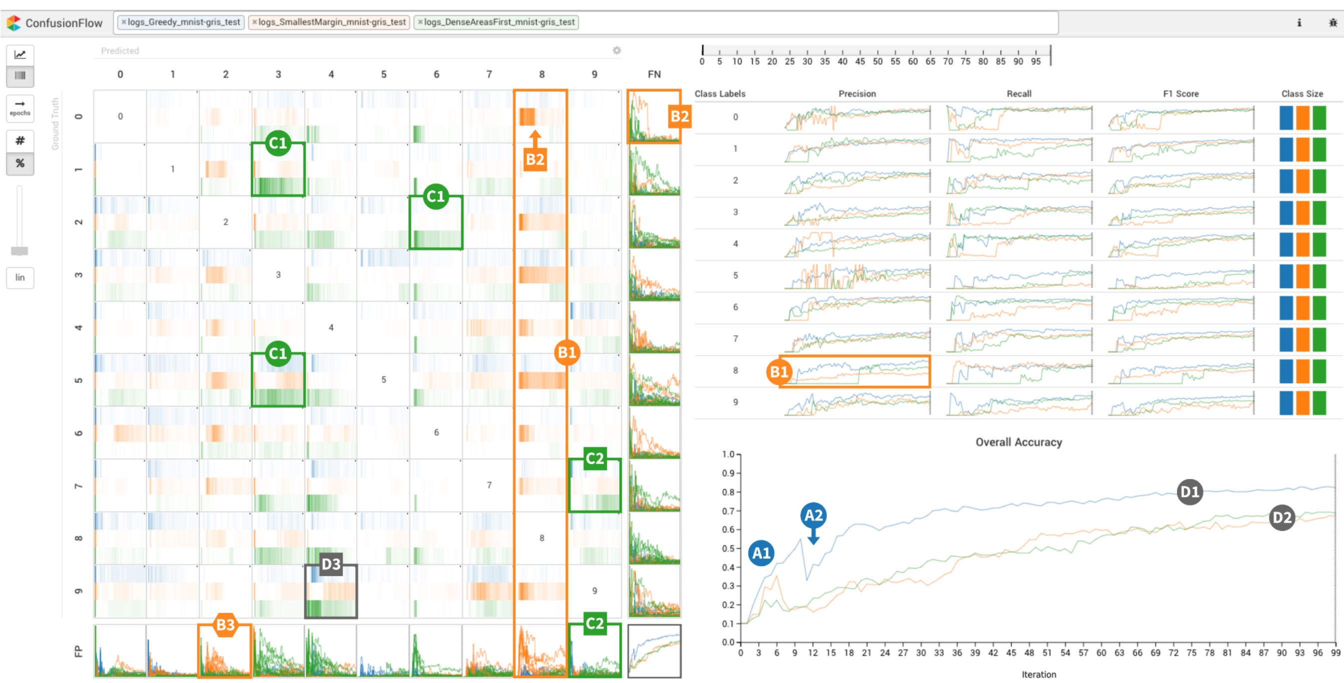
Very clearly there are some steps that seem to be of more interest to researchers than others. Especially data exploration, model development and testing and evaluation seem to get a lot of attention and tooling. Some of this may be an effect of our search and screening strategy, since we only queried the databases for tools for data-driven development. Some steps of the EDDA process are in part covered by conventional software engineering, so existing and established tools may suffice.

For data-centric steps in the process there are quite a number of novel tools, however, e.g. for labeling the data where Alaghbari et al. [2] explore the use of gamification or TagRuler by Choi et al. [12], which combines Natural Language Processing and programming by example to automate this usual tedious process.

The concrete implementation of ML models is very heavily supported from various perspectives, be it the selection of the appropriate model [3, 58, 68] and architecture [98] or its implementation, [22, 56, 85] training and optimization [32, 64, 74, 90]. Optimizing ML models is of course also a question of optimizing hardware-specific code for running the complex models efficiently on GPUs or other specialized hardware. Liou et al's. [48] GEVO-ML is an example for a tool that addresses this particular challenge of ML via automation, detecting and applying certain optimizations.



(a) RapidMiner (image from [7]) allows the development of a data processing pipeline by composition of functional blocks



(b) ConfusionFlow (image taken from [32]) supports developers with visualizations during performance analysis of classifiers

Figure 2: Two examples for tools for the development of data-driven software that heavily rely on a graphical presentation.

Optimization of course is also very much related to the testing and evaluation step, where many of the XAI and visualization systems play a role, either in order to give the developers a better understanding of the metrics and machinations [30, 32, 79, 92] or explicitly for debugging [32, 46].

Some of the authors attempt to maintain a holistic view though [3, 10, 60], addressing issues that are relevant throughout the development process like traceability [63] and asset management [36].

Much less in the focus appears to be the question of ML requirements, which according to the EDDA process [31] should include questions about acceptance criteria, i.e. when a ML model is adequate. For the integration of ML systems, we found two publications in our search: Liu et al. [50] which explores how different ML systems and libraries can interoperate via model transformation and Zhang et al. [96] who approach ML tools from the perspective of MDD and highlight the existing integration challenges in the tool infrastructure.

Overall it appears that certain aspects of data-driven development are already very actively explored – whether to a degree of saturation we cannot yet say – while others still provide ample opportunity for improvements. Still, the breadth of tools for many niche and specialized aspects also highlights a certain fragmentation. Instead of integrating solutions into tools with a large user base, researchers currently appear to prefer building their solutions from scratch and as stand-alone solutions. This may be in part due to the nature of research, focusing on small, well defined aspects. Another factor, though, may also be the matter of interoperability and integration, as previously mentioned [96].

There are notable exceptions like Malviya et al. [52], who advocate for a plugin-based architecture, and others [5, 38, 44, 53, 73] who build plugins or extensions to existing tools. However, based on the publications we investigated, the tooling landscape remains in flux, with Jupyter Notebooks – currently – a solid contender as the tool of choice [39]. Alternatively, it is also possible to leverage the existing tooling landscape and adapt it to support the development of data-driven applications too [16, 96], which not only is an efficient reuse of resources but could also lead to a greater integration of software tools and thus counteract a potential fragmentation.

3.4 Evaluation Methodology

A slow adoption of tools can of course also be the result of insufficient quality and poor usability. We therefore specifically noted the evaluation methods, if any, in the publications. Positively, almost half (31) of the publications we analyzed did perform some for of evaluation.

However, when comparing the method of evaluation, we found that there are three mostly disjoint groups: publications from the more technically inclined venues tend to favor benchmarks with analytical metrics to compare their implementations with prior work (7), while particularly researchers from the field of human-computer interaction rely more on user studies, interviews and surveys to capture to how their tools are perceived (13). A method that both groups occasionally rely on are case studies, i.e. applying their tool in a realistic setting (10). Here we could, however, not reliably determine from the text whether these case studies were selected after the tools were completed or whether the tools were built with the application in mind.

While all these evaluation methods are very much valid to determine the quality of the tools which people build, they each of course only address a limited number of quality criteria. Unfortunately, our analysis shows that only a handful of publications use more than one of those methods, so even though researchers may claim their tool to be high quality, we typically get an incomplete picture, focusing often either on analytical quality or the human perception but rarely both.

3.5 Related Literature Surveys

Beyond direct evaluation on individual tools, we found 14 publications in our sample that reported not on single tools but on comparative surveys across multiple existing tools, eight of which are from the last two years. This, to us, indicates an increase in reflection on the status quo and a desire to understand the current situation before moving forward [25, 66].

Being from different domains, these publications focused on different aspects of data-driven tools, be it where the tools are used, e.g. in education [39], or how they support specific steps in the development from data preparation [26] to model selection [68] or training and optimization [74]. In contrast to our survey they did not specifically focus on the benefits for development experts but contrasted the expert perspective to that of data science novices [4, 15, 74]. This focus is often motivated by the complexity and opacity of current machine learning models and the process for developing and tuning them [74], which makes existing tools hard to use for less experienced users. At the same time, it is acknowledged that even experts often require a considerable amount of skills and training for effectively using current tools [65] which suggests that there is still much to do to achieve easy to use tools for data science, regardless of the target group.

Nonetheless, these types of comparisons and overviews, particularly when applied to practical, real-word use cases, provide an additional perspective for evaluation the existing tool landscape for data-driven development.

4 DISCUSSION

From the body of publications, we were able to determine a number of commonalities and shared directions but also highlight some of the outliers and shortcomings in existing scientific literature. The following section will now summarize some of these and draw conclusion regarding tooling for data-driven development in the future.

4.1 Common Goals

The generally high – and increasing – interest in the topic of data-driven development and tools and methods for its support already suggest that is by now well understood that Machine Learning etc. have moved from academic exercise to be the driver of real-world change and thus needs to be treated as such. This view is shared by many of the reviews and shows in a decent number of evaluations, which also suggest a increase in reflection.

Such introspection will help with transitioning from data *science* to an *engineering* discipline but considering how diverse the field still is with many disjoint methods and process but a limited consensus on best practices, prerequisites and goals [28], there may be a need for a coordinated effort to consolidate and steer the field similar to the movement that brought us software engineering [61].

Another fact that has been acknowledged is that it can be quite challenging for developers to move from conventional programming to developing data-driven software. While the former builds on a defined sequence of instructions, the latter has many of its functionality and intricacies implicit in the data where it is learned by the software but remains hidden to the developer.

Certainly, the field of XAI is working on this challenge, but, at the current point in time, the preferred target group seems to be end-users [13, 14, 20]. While this yields interesting results, it is still unclear whether end users need or want explainability [21, 91]. Developers, on the contrary, very clearly benefit from opening up the black box that is ML for the purpose of debugging [32, 46], identifying biases [33] and just general fairness [79], transparency [90] and an ethical use of the technology [86]. Focusing some of the XAI efforts on developers first may, therefore, yield more immediate, actionable successes. In addition, given that someone has to develop the end-user explainability, making sure that developers precisely understand what is going on, may also prevent a situation where flawed mental models are propagated and should in turn improve the situation for all (cf. [55]).

4.2 Common Openings

As highlighted in Tab. 2, not all aspects of data-driven development, in this case represented by steps in the EDDA-process [31], have received an equal amount of attention in the form of publications. As mentioned before some of these areas may already be covered by prior research from traditional software engineering though. In addition, a high number of publications and approaches in one area must not mean that this step has been solved but can also indicate a high need for diverse solutions or many incremental improvements. It is therefore hard to precisely determine, where exactly more needs to be done.

It it, however, noteworthy that the large gaps appear to lie at the interface between data-driven and more traditional software development: there is little to nothing for specifying the requirements of the data-driven aspect in the context of the larger software and similarly little for integrating the data-driven part into the software and pushing it into the world. This may be due to a diffusion of responsibility between traditional and data-driven development, both expecting the other side to handle this. Another possible explanation may be that these are the points of friction where we have not yet found adequate solutions. Both these potential reasons, however, strongly suggests that these areas will require some attention and should not be neglected going forward.

4.3 Quo Vadis

Research into the development of data-driven explanations certainly is only getting started, though. Nonetheless, the question arises which immediate issues need to be addressed now to steer the field in a good direction. For tooling in particular it is important to make Machine Learning and related technologies accessible to a broad audience to make sure many can participate in a technology that affects everyone. Likewise, we need to determine the major impediments for its common adoption first before we rush into niche solutions for specialized problems.

Based on the literature we described in this paper, we see a number of properties that tools in the nearby future should try to achieve:

Vertical Integration. One issue that is not just bothersome but can also introduce an arbitrary number of problems is when a developer has to use a specialized tool for each step in the development. It is no wonder that for traditional software engineering, many professionals use Integrated Development Environments (IDEs), i.e. tools that attempt to integrate support mechanisms for a slew of activities in a single application.

As our literature review shows quite clearly, there are many helpful tools for many important steps in the development process but most of them are stand-alone applications. In order to use them, developers will have to export, import and transform their data, which will be the source of errors.

Arguably, this may very well be an artifact of the fact that academic projects often focus on one aspect only. Still, a growing number of projects, including from academia, provide their support mechanisms integrated into the Jupyter Notebook environment, as one of the most popular tool. Such a close integration into an established platform not only does reduces the likelihood of errors due to for example data-transfer, but also simplified usage and increases usability, which saves time and should positively affect adoption. We therefore encourage this practice for future development efforts.

Horizontal Integration. Of course, consolidating tools leads to another open topic: how to integrate data, models, etc. from various tools, libraries and systems. This will require some form of universal interface between applications, which is highly non-trivial, but there are early efforts for interaction between software systems, both in our sample of the literature, e.g. the AI-ESTATE standard [37, 83], or beyond, like the ONNX standard and platform[1] which allows for a shared representation and thus exchange of ML models. Unfortunately, in our literature sample, we could find only very few examples where these, admittedly relatively new, standards are being used.

Currently, the more common practice (e.g. [3]) is to leverage the RESTful architecture, as well established in traditional software engineering, and provide ML functionality as a service with an API. Adherence to this de facto standard method also allows fairly simple interoperability with traditional software systems; a topic that was barely mentioned in our literature sample.

Automation. A major trend in the publications from our sample seems to be automation though, with one in five publications introducing a tool that automates some aspect of the development. As mentioned before, complete automation is viewed critically in some of the analyses though, for it will exasperate the opaque nature of ML, which is undesirable for developers and unacceptable in many domains like for example healthcare [34, 69, 76].

This balance between automation and control is of course not new, but in this case of software and data-driven development is

[1] https://onnx.ai

somewhat reflexive in that the developers of the automation would potentially automate their own work.

To ensure this balance, we should strive to make the transition across the spectrum of automation as easy as possible. This means that for any step that can be automated, developers should have the choice of letting the computer solve the problem for them, manually solve it themselves or rely on any form of collaborative interaction in between with little overhead for switching between these options.

Graphical and Code-based Tool. Another very common theme is the use of visualizations, which were used in a number of publications. They way they are often presented suggests that for many data-driven development has a strong graphical component, be it as a means of tackling its complexity or as a property inherent to this development paradigm.

Naturally, many have leveraged this in some form or another, yielding a number of graphical programming tools for data-driven development. Still, Jupyter notebooks, which are at their core code-based, remain among the most popular tools [39].

Various evaluations in our sample suggests, though, that the graphical aspects provides a number of benefits, particularly when dealing with complexity at the scale of many data-driven applications. Code, on the other hand, is very concise and expressive, allowing a high degree of control and flexibility. Both therefore are valid presentations.

Work like that of Kery et al. [44] already show that these two options must not stand distinct from each other but can be combined. Solutions like that, which allow a seamless transition from established to novel interfaces may convince more people to try interfaces beyond what they already know and use – particularly when integrated into a platform that developers already use, like Jupyter notebooks. The division of Jupyter Notebooks into individual blocks that individually can be switched between graphical and code-based view lends itself especially well for this, but this notion of decomposition has been used to a further degree already. Silva et als. [81] DBSnap++ uses graphical programming, where queries of data are built from individual blocks, or consider RapidMiner [7], which builds the whole data processing pipeline by composing blocks.

In fact, the pipeline-nature of data-driven software lends itself very much to this idea inspired by prior work in MDD, since the data-flow from a high-level perspective is straight forward. To still facilitate the aforementioned control and flexiblity, tools could allow a fluid switch to code [44] for individual steps or even a hierarchical structure where developers can drill down through multiple levels of abstraction per block until they reach the level of detail they require.

However, one should not forget that this way of abstraction, composition and general approach to complexity has been state-of-the-art for many years in MDD but did manage to convince the majority of software engineers to use it in everyday practice.

Holistic Evaluation. Why exactly MDD is not as widely adopted as it might be for all its benefits is beyond the scope of this paper, but some recent trends in that field [6] and some aspects of our data may hold some clues how to prevent this for data-driven development.

To be adopted a tool should fulfill certain criteria, among which are that it should provide sufficient benefits at a low cost. For data-driven tools the benefits can include an increase in model and software quality, while the cost can come in work overhead and general usability.

Part of this is often the topic of an increasing number of evaluations in this field of research – a development that is to be encouraged. Still, looking at how the tools in our sample are evaluated, we saw that the majority is only evaluated for one of these two aspects: either some form of usability evaluation or performance benchmarking. However, even if a tool is superior to other in the benchmarks, few people will adopt it if it is only usable by highly experienced experts. Likewise, if a tool is a delight to use but provides only limited practical value, it will likely also be rejected.

This is not to say that tools from technology experts are unusable or those from HCI are technically unsound – with the current evaluations we cannot tell – but to convince potential early adopters, an holistic evaluation from multiple perspective may be more convincing.

This should also encourage and facilitate collaboration across domains, which generally is desirable and should lead to better, more rounded tools. However, such collaboration must be supported on an organizational level also, for example by encouraging more human-centered work at technical venues and vice versa. Considering how tooling and support for developers is distributed across the different venues in our data set, there still seems be a bit of work left to do.

4.4 Limitations

At the same time, one would assume that an area like software development, where the target groups is capable of building their own solutions, would be self-regulating in that developers will likely build the tools for their own most pressing issues. So, our selection of scientific literature cannot fully reflect the full breath of tooling for data-driven development, since it will not include all the individual solutions that did not make it into a scientific publication. Likewise we also cannot judge the tooling landscape that large corporations might employ internally, as long as they remain behind closed doors.

These tools, however, often are created in an ad-hoc fashion and without publicly accessible evaluations it is hard to determine their actual value and benefit. Particularly for data-driven software, where many factors like unknown biases can be the deciding factor between impressive results and hidden flaws, a healthy degree of skepticism for un-tested and un-evaluated tools is appropriate. Consequently publication, scientific or otherwise, an open-source mentality and, in general, continuous evaluation are all what we should strive for to ensure that data-driven software increases in quality and its development becomes easier, more accessible and more reliable.

That is, of course, not to say that many of the current tools we looked at are necessarily flawed, but with only about half of them providing an evaluation, there is still some room for making their benefits more convincing and improving transparency.

Naturally, our results for these tools depend on the search and filter criteria we applied and can only reflect a snapshot of the status quo. Given how fast the field of ML is changing, many more tools are probably already being developed to address many of the open issues and at the same time the technology may change such that existing tools become obsolete. Given that software developers do still use tools that have been around for decades, the adoption is not a quick though. So, the question really is not which individual tool will be used by how many people but which overarching theme will be prevalent and can convince developers to change habits. These themes, some of which have been described in this paper, are more stable and much more than just trends, but therefore should also be grounded in the results of systematic evaluations of what works and what does not.

5 CONCLUSION

Motivated by the growing importance of software powered by data and Machine Learning, we conducted a survey of the current scientific literature to find current means for supporting the software developers behind this software.

Filtering the search results down to 76 relevant publications, we determined common topics and shortcomings of the literature at this point. A considerable number of these publications introduced novel tools that try to solve, automate or simplify various aspects of the development process, from data acquisition, model creation and tuning to evaluation and interpretation of the resulting software.

In spite of the abundance of tools a number of step in the development of data-driven software have not yet received as much attention and support as others, particularly those steps at the intersection with traditional software engineering, e.g. determining the requirements of these systems.

For other steps there are many existing tools. While there is a growing culture of evaluation in this domain for these tools, the majority of them is only scrutinized from one perspective, either as a case study, by benchmarking or by eliciting user feedback, but rarely are these methods combined.

To make sure that all these tools actually meet a wider demand and are not just a one-off solution for a single problem, we need more cross-domain collaboration for both requirements and evaluation specifically and for supporting developers and the development teams generally.

The community of tool developers for data-driven development does appear to be very active and ML software is in high demand, so we can expect many more tools, hopefully with some comparative studies in between. Based on our findings we also provide a series of suggestions for future tool development, which may give this community even more to work on.

The long term goal of simplifying the development of data-driven applications is not only one that concerns this small group of developers, though. Certainly, better tools has an immediate impact on their work experience but it will also affect many of the involved team members, not all of which will be experts, and it will also affect the quality of the software they write. This software in turn can have a wide ranging impact on society and thus should be as good as we can get it. It is therefore in everyone's interest that, for example, decisions about ethical implications like privacy, biases, etc. are made deliberately by an informed development team so that they can then be implemented correctly. In addition, when software development gets easier and we get a better understanding of these systems, they also become more accessible to regular people up to the point of end-user programming. Improving the development process and increasing the understanding of data-driven applications is therefore a matter that in the long term affects just about everyone, which makes it endeavor where broad collaboration across many domains and areas of expertise should be the goal.

REFERENCES

References marked with • are part of the set of papers selected for this review.

[1] • Parvej Aalam and Tamanna Siddiqui. 2016. Comparative study of data mining tools used for clustering. In *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)*. 3971–3975.

[2] • Sarah Alaghbari, Annett Mitschick, Gregor Blichmann, Martin Voigt, and Raimund Dachselt. 2020. Achiever or explorer? gamifying the creation process of training data for machine learning. In *Mensch und Computer 2020 - Tagungsband, Magdebug, Germany, September 6-9, 2020*, Florian Alt, Stefan Schneegass, and Eva Hornecker (Eds.). ACM, 173–181. https://doi.org/10.1145/3404983.3405519

[3] • Maqbool Ali, Rahman Ali, Wajahat Ali Khan, Soyeon Caren Han, Jae Hun Bang, Tae Ho Hur, Dohyeong Kim, Sungyoung Lee, and Byeong Ho Kang. 2018. A Data-Driven Knowledge Acquisition System: An End-to-End Knowledge Engineering Process for Generating Production Rules. *IEEE Access* 6 (2018), 15587–15607. https://doi.org/10.1109/ACCESS.2018.2817022

[4] • Saleema Amershi, James Fogarty, Ashish Kapoor, and Desney S. Tan. 2011. Effective End-User Interaction with Machine Learning. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2011, San Francisco, California, USA, August 7-11, 2011*, Wolfram Burgard and Dan Roth (Eds.). AAAI Press. http://www.aaai.org/ocs/index.php/AAAI/AAAI11/paper/view/3736

[5] • R. Anil, R. Danymol, Harsha Gawande, and R. Gandhiraj. 2014. Machine learning plug-ins for GNU Radio Companion. In *2014 International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*. 1–5. https://doi.org/10.1109/ICGCCEE.2014.6922250

[6] Mojtaba BAgherzadeh, Francis Bordeleau, Jean-Michel Bruel, Jurgen Dingel, Sébastien Gérad, Nicolas Hili, and Sebastian Voss. 2017. Summary of Workshop on Model-Driven Engineering Tools (MDETools'17). In *Proceedings of MODELS 2017 Satellite Event: Workshops (ModComp, ME, EXE, COMMitMDE, MRT, MULTI, GEMOC, MoDeVVa, MDETools, FlexMDE, MDEbug), Posters, Doctoral Symposium, Educator Symposium, ACM Student Research Competition, and Tools and Demonstrations co-located with ACM/IEEE 20th International Conference on Model Driven Engineering Languages and Systems (MODELS 2017), Austin, TX, USA, September, 17, 2017 (CEUR Workshop Proceedings)*. CEUR-WS.org. http://ceur-ws.org/Vol-2019

[7] • Mohamed Bjaoui, Houneida Sakly, Mourad Said, Naoufel Kraiem, and Mohamed Salim Bouhlel. 2020. Depth Insight for Data Scientist with RapidMiner « an Innovative Tool for AI and Big Data towards Medical Applications». In *Proceedings of the 2nd International Conference on Digital Tools & Uses Congress* (Virtual Event, Tunisia) *(DTUC '20)*. Association for Computing Machinery, New York, NY, USA, Article 21, 6 pages. https://doi.org/10.1145/3423603.3424059

[8] • Luís C. Borges, Viriato M. Marques, and Jorge Bernardino. 2013. Comparison of data mining techniques and tools for data classification. In *International C* Conference on Computer Science & Software Engineering, C3S2E13, Porto, Portugal - July 10 - 12, 2013*, Bipin C. Desai, Ana Maria Almeida, Jorge Bernardino, and Sudhir P. Mudur (Eds.). ACM, 113–116. https://doi.org/10.1145/2494444.2494451

[9] • Reece A. Boyd and Salvador E. Barbosa. 2017. Reinforcement Learning for All: An Implementation Using Unreal Engine Blueprint. In *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*. 787–792. https://doi.org/10.1109/CSCI.2017.136

[10] • Eli Brumbaugh, Atul Kale, Alfredo Luque, Bahador Nooraei, John Park, Krishna Puttaswamy, Kyle Schiller, Evgeny Shapiro, Conglei Shi, Aaron Siegel, Nikhil Simha, Mani Bhushan, Marie Sbrocca, Shi-Jing Yao, Patrick Yoon, Varant Zanoyan, Xiao-Han T. Zeng, Qiang Zhu, Andrew Cheong, Michelle Gu-Qian Du, Jeff Feng, Nick Handel, Andrew Hoh, Jack Hone, and Brad Hunter. 2019. Bighead: A Framework-Agnostic, End-to-End Machine Learning Platform. In *2019 IEEE International Conference on Data Science and Advanced Analytics, DSAA 2019, Washington, DC, USA, October 5-8, 2019*, Lisa Singh, Richard D. De Veaux, George Karypis, Francesco Bonchi, and Jennifer Hill (Eds.). IEEE, 551–560. https://doi.org/10.1109/DSAA.2019.00070

[11] • Alberto Castellini, Francesco Masillo, Riccardo Sartea, and Alessandro Farinelli. 2019. eXplainable Modeling (XM): Data Analysis for Intelligent Agents. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, Edith Elkind, Manuela Veloso, Noa Agmon, and Matthew E. Taylor (Eds.). International

Foundation for Autonomous Agents and Multiagent Systems, 2342–2344. http://dl.acm.org/citation.cfm?id=3332106

[12] • Dongjin Choi, Sara Evensen, Çagatay Demiralp, and Estevam Hruschka. 2021. TagRuler: Interactive Tool for Span-Level Data Programming by Demonstration. In *Companion of The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia (Eds.). ACM / IW3C2, 673–677. https://doi.org/10.1145/3442442.3458602

[13] Michael Chromik. 2021. Making SHAP Rap: Bridging Local and Global Insights Through Interaction and Narratives. In *Human-Computer Interaction – INTERACT 2021*, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen (Eds.). Springer International Publishing, Cham, 641–651.

[14] Michael Chromik, Malin Eiband, Felicitas Buchner, Adrian Krüger, and Andreas Butz. 2021. I Think I Get Your Point, AI! The Illusion of Explanatory Depth in Explainable AI. In *26th International Conference on Intelligent User Interfaces* (College Station, TX, USA) *(IUI '21)*. Association for Computing Machinery, New York, NY, USA, 307–317. https://doi.org/10.1145/3397481.3450644

[15] • Arthur R. T. de Lacerda and Carla S. R. Aguiar. 2019. FLOSS FAQ chatbot project reuse: how to allow nonexperts to develop a chatbot. In *Proceedings of the 15th International Symposium on Open Collaboration, OpenSym 2019, Skövde, Sweden, August 20-22, 2019*, Björn Lundell, Jonas Gamalielsson, Lorraine Morgan, and Gregorio Robles (Eds.). ACM, 3:1–3:8. https://doi.org/10.1145/3306446.3340823

[16] • Rob DeLine. 2015. Modern software is all about data. development environments should be, too. (keynote). In *Companion Proceedings of the 2015 ACM SIGPLAN International Conference on Systems, Programming, Languages and Applications: Software for Humanity, SPLASH 2015, Pittsburgh, PA, USA, October 25-30, 2015*, Jonathan Aldrich and Patrick Eugster (Eds.). ACM, 4. https://doi.org/10.1145/2814189.2833213

[17] • Amanda Gonçalves Dias, Evangelos E. Milios, and Maria Cristina Ferreira de Oliveira. 2019. TRIVIR: A Visualization System to Support Document Retrieval with High Recall. In *Proceedings of the ACM Symposium on Document Engineering 2019, Berlin, Germany, September 23-26, 2019*, Sonja Schimmler and Uwe M. Borghoff (Eds.). ACM, 10:1–10:10. https://doi.org/10.1145/3342558.3345401

[18] • Gonzalo I. Diaz, Achille Fokoue-Nkoutche, Giacomo Nannicini, and Horst Samulowitz. 2017. An effective algorithm for hyperparameter optimization of neural networks. *IBM J. Res. Dev.* 61, 4-5 (2017), 9:1–9:11. https://doi.org/10.1147/JRD.2017.2709578

[19] • Drazen Draskovic. 2019. Development of intelligent systems and application of gamification in artificial intelligent learning. In *2019 27th Telecommunications Forum (TELFOR)*. 1–1. https://doi.org/10.1109/TELFOR48224.2019.8971360

[20] Malin Eiband, Daniel Buschek, and Heinrich Hussmann. 2021. How to Support Users in Understanding Intelligent Systems? Structuring the Discussion. In *IUI '21: 26th International Conference on Intelligent User Interfaces, College Station, TX, USA, April 13-17, 2021*, Tracy Hammond, Katrien Verbert, Dennis Parra, Bart P. Knijnenburg, John O'Donovan, and Paul Teale (Eds.). ACM, 120–132. https://doi.org/10.1145/3397481.3450694

[21] Malin Eiband, Daniel Buschek, Alexander Kremer, and Heinrich Hussmann. 2019. The Impact of Placebic Explanations on Trust in Intelligent Systems. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Regan L. Mandryk, Stephen A. Brewster, Mark Hancock, Geraldine Fitzpatrick, Anna L. Cox, Vassilis Kostakos, and Mark Perry (Eds.). ACM. https://doi.org/10.1145/3290607.3312787

[22] • Alexander Eremeev, Sergey Ivliev, and Alexander Kozhukhov. 2020. Tool Environment for Creating Training Prototypes of Intelligent Decision Support Systems. In *2020 V International Conference on Information Technologies in Engineering Education (Inforino)*. 1–4. https://doi.org/10.1109/Inforino48376.2020.9111781

[23] • Oscar Gomez, Steffen Holter, Jun Yuan, and Enrico Bertini. 2020. ViCE: visual counterfactual explanations for machine learning models. In *IUI '20: 25th International Conference on Intelligent User Interfaces, Cagliari, Italy, March 17-20, 2020*, Fabio Paternò, Nuria Oliver, Cristina Conati, Lucio Davide Spano, and Nava Tintarev (Eds.). ACM, 531–535. https://doi.org/10.1145/3377325.3377536

[24] • Taisa Guidini Gonçalves and Ana Regina Cavalcanti da Rocha. 2019. Development process for intelligent user interfaces: an initial approach. In *Proceedings of the XVIII Brazilian Symposium on Software Quality, SBQS 2019, Fortaleza, Brazil, October 28 - November 1, 2019*, Adriano Bessa Albuquerque and Ana Luiza Bessa de Paula Barros (Eds.). ACM, 210–215. https://doi.org/10.1145/3364641.3364665

[25] • Danielle Gonzalez, Thomas Zimmermann, and Nachiappan Nagappan. 2020. The State of the ML-universe: 10 Years of Artificial Intelligence & Machine Learning Software Development on GitHub. In *MSR '20: 17th International Conference on Mining Software Repositories, Seoul, Republic of Korea, 29-30 June, 2020*, Sunghun Kim, Georgios Gousios, Sarah Nadi, and Joseph Hejderup (Eds.). ACM, 431–442. https://doi.org/10.1145/3379597.3387473

[26] • Mazhar Hameed and Felix Naumann. 2020. Data Preparation: A Survey of Commercial Tools. *SIGMOD Rec.* 49, 3 (2020), 18–29. https://doi.org/10.1145/3444831.3444835

[27] Jianchao Han, Juan C. Rodriguez, and Mohsen Beheshti. 2008. Diabetes Data Analysis and Prediction Model Discovery Using RapidMiner. In *International Conference on Bio-Science and Bio-Technology, BSBT 2008, part of the Second International Conference on Future Generation Communication and Networking, FGCN 2008, Volume 3, Hainan Island, China, December 13-15, 2008*. IEEE Computer Society, 96–99. https://doi.org/10.1109/FGCN.2008.226

[28] Guy S. Handelman, Hong Kuan Kok, Ronil V. Chandra, Amir H. Razavi, Shiwei Huang, Mark Brooks, Michael J. Lee, and Hamed Asadi. 2019. Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods. *American Journal of Roentgenology* 212, 1 (2019), 38–43. https://doi.org/10.2214/AJR.18.20224 arXiv:https://doi.org/10.2214/AJR.18.20224 PMID: 30332290.

[29] • Andrew Head, Fred Hohman, Titus Barik, Steven Mark Drucker, and Robert DeLine. 2019. Managing Messes in Computational Notebooks. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.). ACM, 270. https://doi.org/10.1145/3290605.3300500

[30] • Tim Hegeman, Matthijs Jansen, Alexandru Iosup, and Animesh Trivedi. 2021. GradeML: Towards Holistic Performance Analysis for Machine Learning Workflows. In *ICPE '21: ACM/SPEC International Conference on Performance Engineering, Virtual Event, France, April 19-21, 2021, Companion Volume*, Johann Bourcier, Zhen Ming (Jack) Jiang, Cor-Paul Bezemer, Vittorio Cortellessa, Daniele Di Pompeo, and Ana Lucia Varbanescu (Eds.). ACM, 57–63. https://doi.org/10.1145/3447545.3451185

[31] Marc Hesenius, Nils Schwenzfeier, Ole Meyer, Wilhelm Koop, and Volker Gruhn. 2019. Towards a software engineering process for developing data-driven applications. In *Proceedings of the 7th International Workshop on Realizing Artificial Intelligence Synergies in Software Engineering, RAISE@ICSE 2019, Montreal, QC, Canada, May 28, 2019*, Tim Menzies and Burak Turhan (Eds.). IEEE / ACM, 35–41. https://doi.org/10.1109/RAISE.2019.00014

[32] • Andreas Hinterreiter, Peter Ruch, Holger Stitz, Martin Ennemoser, Jurgen Bernard, Hendrik Strobelt, and Marc Streit. 2020. ConfusionFlow: A model-agnostic visualization for temporal analysis of classifier confusion. *IEEE Transactions on Visualization and Computer Graphics* (2020), 1–1. https://doi.org/10.1109/TVCG.2020.3012063

[33] • Fred Hohman, Andrew Head, Rich Caruana, Robert DeLine, and Steven Mark Drucker. 2019. Gamut: A Design Probe to Understand How Data Scientists Understand Machine Learning Models. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.). ACM, 579. https://doi.org/10.1145/3290605.3300809

[34] Andreas Holzinger, Chris Biemann, Constantinos S. Pattichis, and Douglas B. Kell. 2017. What do we need to build explainable AI systems for the medical domain? *CoRR* abs/1712.09923 (2017). arXiv:1712.09923 http://arxiv.org/abs/1712.09923

[35] • Qingyuan Hu. 2021. A Survey of Adversarial Example Toolboxes. In *2021 2nd International Conference on Computing and Data Science (CDS)*. 603–608. https://doi.org/10.1109/CDS52072.2021.00109

[36] • Samuel Idowu, Daniel Strüber, and Thorsten Berger. 2021. Asset Management in Machine Learning: A Survey. In *43rd IEEE/ACM International Conference on Software Engineering: Software Engineering in Practice, ICSE (SEIP) 2021, Madrid, Spain, May 25-28, 2021*. IEEE, 51–60. https://doi.org/10.1109/ICSE-SEIP52600.2021.00014

[37] • IEEE. 2014. IEEE Guide for the Use of Artificial Intelligence Exchange and Service Tie to All Test Environments (AI-ESTATE). , 102 pages. https://doi.org/10.1109/IEEESTD.2014.6922153

[38] • Dietmar Jannach, Michael Jugovac, and Lukas Lerche. 2016. Supporting the Design of Machine Learning Workflows with a Recommendation System. *ACM Trans. Interact. Intell. Syst.* 6, 1 (2016), 8:1–8:35. https://doi.org/10.1145/2852082

[39] • Jeremiah W. Johnson and Karen H. Jin. 2020. Jupyter Notebooks in Education. *J. Comput. Sci. Coll.* 35, 8 (April 2020), 268–269.

[40] • Mohammed Abbas Kadhim, M. Afshar Alam, and Harleen Kaur. 2013. Design and Implementation of Intelligent Agent and Diagnosis Domain Tool for Rule-Based Expert System. In *2013 International Conference on Machine Intelligence and Research Advancement*. 619–622. https://doi.org/10.1109/ICMIRA.2013.129

[41] Andrej Karpathy. 2017. Software 2.0. https://karpathy.medium.com/software-2-0-a64152b37c35

[42] • Harmanpreet Kaur, Harsha Nori, Samuel Jenkins, Rich Caruana, Hanna M. Wallach, and Jennifer Wortman Vaughan. 2020. Interpreting Interpretability: Understanding Data Scientists' Use of Interpretability Tools for Machine Learning. In *CHI '20: CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, April 25-30, 2020*, Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik (Eds.). ACM, 1–14. https://doi.org/10.1145/3313831.3376219

[43] • Mustafa Kaytan and İbrahim Berkan Aydilek. 2017. A review on machine learning tools. In *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*. 1–4. https://doi.org/10.1109/IDAP.2017.8090257

[44] • Mary Beth Kery, Donghao Ren, Kanit Wongsuphasawat, Fred Hohman, and Kayur Patel. 2020. The Future of Notebook Programming Is Fluid. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, CHI 2020, Honolulu, HI, USA, April 25-30, 2020*, Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik (Eds.). ACM, 1–8. https://doi.org/10.1145/3334480.3383085

[45] • Chris Kim. 2019. A modular framework for collaborative multimodal annotation and visualization. In *Proceedings of the 24th International Conference on Intelligent User Interfaces: Companion, Marina del Ray, CA, USA, March 16-20, 2019*. ACM, 165–166. https://doi.org/10.1145/3308557.3308730

[46] • Sanjay Krishnan and Eugene Wu. 2017. PALM: Machine Learning Explanations For Iterative Debugging. In *Proceedings of the 2nd Workshop on Human-In-the-Loop Data Analytics, HILDA@SIGMOD 2017, Chicago, IL, USA, May 14, 2017*, Carsten Binnig, Joseph M. Hellerstein, and Aditya G. Parameswaran (Eds.). ACM, 4:1–4:6. https://doi.org/10.1145/3077257.3077271

[47] Jesse Lane and Hak J. Kim. 2015. Big data: web-crawling and analysing financial news using RapidMiner. *Int. J. Bus. Inf. Syst.* 19, 1 (2015), 41–57. https://doi.org/10.1504/IJBIS.2015.069064

[48] • Jhe-Yu Liou, Xiaodong Wang, Stephanie Forrest, and Carole-Jean Wu. 2020. GEVO-ML: a proposal for optimizing ML code with evolutionary computation. In *GECCO '20: Genetic and Evolutionary Computation Conference, Companion Volume, Cancún, Mexico, July 8-12, 2020*, Carlos Artemio Coello Coello (Ed.). ACM, 1849–1856. https://doi.org/10.1145/3377929.3398139

[49] • Shixia Liu, Jiannan Xiao, Junlin Liu, Xiting Wang, Jing Wu, and Jun Zhu. 2018. Visual Diagnosis of Tree Boosting Methods. *IEEE Trans. Vis. Comput. Graph.* 24, 1 (2018), 163–173. https://doi.org/10.1109/TVCG.2017.2744378

[50] • Yu Liu, Cheng Chen, Ru Zhang, Tingting Qin, Xiang Ji, Haoxiang Lin, and Mao Yang. 2020. Enhancing the interoperability between deep learning frameworks by model conversion. In *ESEC/FSE '20: 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, Virtual Event, USA, November 8-13, 2020*, Prem Devanbu, Myra B. Cohen, and Thomas Zimmermann (Eds.). ACM, 1320–1330. https://doi.org/10.1145/3368089.3417051

[51] • Nirav Malsattar, Tomo Kihara, and Elisa Giaccardi. 2019. Designing and Prototyping from the Perspective of AI in the Wild. In *Proceedings of the 2019 on Designing Interactive Systems Conference, DIS 2019, San Diego, CA, USA, June 23-28, 2019*, Steve Harrison, Shaowen Bardzell, Carman Neustaedter, and Deborah G. Tatar (Eds.). ACM, 1083–1088. https://doi.org/10.1145/3322276.3322351

[52] • Vikas K. Malviya, Sawan Rai, and Atul Gupta. 2018. Development of a plugin based extensible feature extraction framework. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing, SAC 2018, Pau, France, April 09-13, 2018*, Hisham M. Haddad, Roger L. Wainwright, and Richard Chbeir (Eds.). ACM, 1840–1847. https://doi.org/10.1145/3167132.3167328

[53] • Sofia Meacham, Vaclav Pech, and Detlef D. Nauck. 2020. Classification Algorithms Framework (CAF) to Enable Intelligent Systems Using JetBrains MPS Domain-Specific Languages Environment. *IEEE Access* 8 (2020), 14832–14840. https://doi.org/10.1109/ACCESS.2020.2966630

[54] Marcin Michalak, Katarzyna Dusza, Dominik Korda, Krzysztof Kozlowski, Bartlomiej Szwej, Michal Kozielski, Marek Sikora, and Lukasz Wróbel. 2016. Application of RapidMiner and R Environments to Dangerous Seismic Events Prediction. In *Proceedings of the 2016 Federated Conference on Computer Science and Information Systems, FedCSIS 2016, Gdańsk, Poland, September 11-14, 2016 (Annals of Computer Science and Information Systems, Vol. 8)*, Maria Ganzha, Leszek A. Maciaszek, and Marcin Paprzycki (Eds.). IEEE, 249–252. https://doi.org/10.15439/2016F188

[55] Tim Miller, Piers Howe, and Liz Sonenberg. 2017. Explainable AI: Beware of Inmates Running the Asylum Or: How I Learnt to Stop Worrying and Love the Social and Behavioural Sciences. *CoRR* abs/1712.00547 (2017). arXiv:1712.00547 http://arxiv.org/abs/1712.00547

[56] • Armin Moin. 2021. Data Analytics and Machine Learning Methods, Techniques and Tool for Model-Driven Engineering of Smart IoT Services. In *43rd IEEE/ACM International Conference on Software Engineering: Companion Proceedings, ICSE Companion 2021, Madrid, Spain, May 25-28, 2021*. IEEE, 287–292. https://doi.org/10.1109/ICSE-Companion52605.2021.00130

[57] Brad A. Myers. 1986. Visual programming, programming by example, and program visualization: a taxonomy. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI 1886, Boston, Massachusetts, USA, April 13-17, 1986*, Marilyn M. Mantei and Peter Orbeton (Eds.). ACM, 59–66. https://doi.org/10.1145/22627.22349

[58] • Gopala Krishna Murthy N., V. S. N. Bhushana Rao, Naga Raju Orsu, Ramya Chiluvuri, and Suresh B. Mudunuri. 2013. Web-Weka1.0: Online Tool for Comparison of Classification Algorithms of Weka. In *19th International Conference on Management of Data, COMAD 2013, Ahmedabad, India, December 19-21, 2013*, Bipin V. Mehta, Nikos Mamoulis, Arnab Bhattacharya, and Maya Ramanath (Eds.). Computer Society of India, 95–96. http://comad.in/comad2013/pdfs/demo-murthy.pdf

[59] Gabor Nagy, Gergo Barta, and Tamás Henk. 2015. Portfolio optimization using local linear regression ensembles in RapidMiner. *CoRR* abs/1506.08690 (2015). arXiv:1506.08690 http://arxiv.org/abs/1506.08690

[60] • Varun Natu and Rahul Ghosh. 2019. EasyDist: An End-to-End Distributed Deep Learning Tool for Cloud. In *Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, COMAD/CODS 2019, Kolkata, India, January 3-5, 2019*, Raghu Krishnapuram and Parag Singla (Eds.). ACM, 265–268. https://doi.org/10.1145/3297001.3297037

[61] Peter Naur and Brian Randell (Eds.). 1969. Software Engineering, Report on a conference sponsored by the NATO science comittee. (1969).

[62] • N. Rich Nguyen. 2020. Toward an Open-source Toolkit for Machine Learning Education. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education, SIGCSE 2020, Portland, OR, USA, March 11-14, 2020*, Jian Zhang, Mark Sherriff, Sarah Heckman, Pamela A. Cutter, and Alvaro E. Monge (Eds.). ACM, 1400. https://doi.org/10.1145/3328778.3372531

[63] • Aquilas Tchanjou Njomou, Alexandra Johanne Bifona Africa, Bram Adams, and Marios Fokaefs. 2021. MSR4ML: Reconstructing Artifact Traceability in Machine Learning Repositories. In *28th IEEE International Conference on Software Analysis, Evolution and Reengineering, SANER 2021, Honolulu, HI, USA, March 9-12, 2021*. IEEE, 536–540. https://doi.org/10.1109/SANER50967.2021.00061

[64] • Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, and Jason H. Moore. 2016. Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference, Denver, CO, USA, July 20 - 24, 2016*, Tobias Friedrich, Frank Neumann, and Andrew M. Sutton (Eds.). ACM, 485–492. https://doi.org/10.1145/2908812.2908918

[65] • Erol Ozan. 2021. A Novel Browser-based No-code Machine Learning Application Development Tool. In *2021 IEEE World AI IoT Congress (AIIoT)*. 0282–0284. https://doi.org/10.1109/AIIoT52608.2021.9454239

[66] • Szilárd Pafka. 2017. Machine Learning Software in Practice: Quo Vadis?. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*. ACM, 25. https://doi.org/10.1145/3097983.3106683

[67] Matthew J. Page, Joanne E. McKenzie, Patrick M. Bossuyt, Isabelle Boutron, Tammy C. Hoffmann, Cynthia D. Mulrow, Larissa Shamseer, Jennifer M. Tetzlaff, Elie A. Akl, Sue E. Brennan, Roger Chou, Julie Glanville, Jeremy M. Grimshaw, Asbjørn Hróbjartsson, Manoj M. Lalu, Tianjing Li, Elizabeth W. Loder, Evan Mayo-Wilson, Steve McDonald, Luke A. McGuinness, Lesley A. Stewart, James Thomas, Andrea C. Tricco, Vivian A. Welch, Penny Whiting, and David Moher. 2021. The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. *Journal of Chronic Diseases* 134 (June 2021), 178–189. https://doi.org/10.1016/j.jclinepi.2021.03.001

[68] • Ki Sun Park, Kyoung-Soon Hwang, Keon Myung Lee, Chan Sik Han, and Jin Han. 2019. Machine learning modeling assistance for non-expert developers. In *Proceedings of the Conference on Research in Adaptive and Convergent Systems, RACS 2019, Chongqing, China, September 24-27, 2019*, Chih-Cheng Hung, Qianbin Chen, Xianzhong Xie, Christian Esposito, Jun Huang, Juw Won Park, and Qinghua Zhang (Eds.). ACM, 117–122. https://doi.org/10.1145/3338840.3355680

[69] Urja Pawar, Donna O'Shea, Susan Rea, and Ruairi O'Reilly. 2020. Explainable AI in Healthcare. In *2020 International Conference on Cyber Situational Awareness, Data Analytics and Assessment, CyberSA 2020, Dublin, Ireland, June 15-19, 2020*. IEEE, 1–2. https://doi.org/10.1109/CyberSA49311.2020.9139655

[70] • Haoyue Ping, Julia Stoyanovich, and Bill Howe. 2017. DataSynthesizer: Privacy-Preserving Synthetic Datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management, Chicago, IL, USA, June 27-29, 2017*. ACM, 42:1–42:5. https://doi.org/10.1145/3085504.3091117

[71] • Edward Raff. 2017. JSAT: Java Statistical Analysis Tool, a Library for Machine Learning. *J. Mach. Learn. Res.* 18 (2017), 23:1–23:5. http://jmlr.org/papers/v18/16-131.html

[72] Oliver Ritthoff, Ralf Klinkenberg, Simon Fischer, Ingo Mierswa, and Sven Felske. 2001. Yale: Yet Another Learning Environment.

[73] • Manuel Martin Salvador, Marcin Budka, and Bogdan Gabrys. 2019. Automatic Composition and Optimization of Multicomponent Predictive Systems With an Extended Auto-WEKA. *IEEE Trans Autom. Sci. Eng.* 16, 2 (2019), 946–959. https://doi.org/10.1109/TASE.2018.2876430

[74] • Eldon Schoop, Forrest Huang, and Björn Hartmann. 2020. SCRAM: Simple Checks for Realtime Analysis of Model Training for Non-Expert ML Programmers. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, CHI 2020, Honolulu, HI, USA, April 25-30, 2020*, Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik (Eds.). ACM, 1–10. https://doi.org/10.1145/3334480.3382879

[75] • Roger Selzler, Adrian D. C. Chan, and James R. Green. 2021. TSEA: An Open Source Python-Based Annotation Tool for Time Series Data. In *IEEE International Symposium on Medical Measurements and Applications, MeMeA 2021, Lausanne,*

Switzerland, June 23-25, 2021. IEEE, 1–6. https://doi.org/10.1109/MeMeA52024.2021.9478712

[76] Arash Shaban-Nejad, Martin Michalowski, John S. Brownstein, and David L. Buckeridge. 2021. Guest Editorial Explainable AI: Towards Fairness, Accountability, Transparency and Trust in Healthcare. *IEEE J. Biomed. Health Informatics* 25, 7 (2021), 2374–2375. https://doi.org/10.1109/JBHI.2021.3088832

[77] • James Shanahan and Liang Dai. 2017. Large Scale Distributed Data Science from scratch using Apache Spark 2.0. In *Proceedings of the 26th International Conference on World Wide Web Companion, Perth, Australia, April 3-7, 2017*, Rick Barrett, Rick Cummings, Eugene Agichtein, and Evgeniy Gabrilovich (Eds.). ACM, 955–957. https://doi.org/10.1145/3041021.3051108

[78] • Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, and Tim Kraska. 2019. Democratizing Data Science through Interactive Curation of ML Pipelines. In *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019*, Peter A. Boncz, Stefan Manegold, Anastasia Ailamaki, Amol Deshpande, and Tim Kraska (Eds.). ACM, 1171–1188. https://doi.org/10.1145/3299869.3319863

[79] • Shubham Sharma, Jette Henderson, and Joydeep Ghosh. 2020. CERTIFAI: A Common Framework to Provide Explanations and Analyse the Fairness and Robustness of Black-box Models. In *AIES '20: AAAI/ACM Conference on AI, Ethics, and Society, New York, NY, USA, February 7-8, 2020*, Annette N. Markham, Julia Powles, Toby Walsh, and Anne L. Washington (Eds.). ACM, 166–172. https://doi.org/10.1145/3375627.3375812

[80] • Dvijesh J. Shastri. 2020. Machine Learning for Non-Programmers. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, CHI 2020, Honolulu, HI, USA, April 25-30, 2020*, Regina Bernhaupt, Florian 'Floyd' Mueller, David Verweij, Josh Andres, Joanna McGrenere, Andy Cockburn, Ignacio Avellino, Alix Goguey, Pernille Bjøn, Shengdong Zhao, Briane Paul Samson, and Rafal Kocielnik (Eds.). ACM, 1–3. https://doi.org/10.1145/3334480.3375051

[81] • Yasin N. Silva, Anthony Nieuwenhuyse, Thomas G. Schenk, and Alaura Symons. 2018. DBSnap++: creating data-driven programs by snapping blocks. In *Proceedings of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education, ITiCSE 2018, Larnaca, Cyprus, July 02-04, 2018*, Irene Polycarpou, Janet C. Read, Panayiotis Andreou, and Michal Armoni (Eds.). ACM, 170–175. https://doi.org/10.1145/3197091.3197114

[82] • Utkarsh Singh and Chris Crawford. 2021. AvaFlow: Learning to Design Intelligent Virtual Agents through Flow-Based Programming. In *SIGCSE '21: The 52nd ACM Technical Symposium on Computer Science Education, Virtual Event, USA, March 13-20, 2021*, Mark Sherriff, Laurence D. Merkle, Pamela A. Cutter, Alvaro E. Monge, and Judithe Sheard (Eds.). ACM, 1307. https://doi.org/10.1145/3408877.3439620

[83] • Liessman Sturlaugson, Nathan Fortier, Patrick Donnelly, and John W. Sheppard. 2013. Implementing AI-ESTATE with prognostic extensions in Java. In *2013 IEEE AUTOTESTCON*. 1–8. https://doi.org/10.1109/AUTEST.2013.6645086

[84] • Yunjia Sun, Edward Lank, and Michael A. Terry. 2017. Label-and-Learn: Visualizing the Likelihood of Machine Learning Classifier's Success During Data Labeling. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces, IUI 2017, Limassol, Cyprus, March 13-16, 2017*, George A. Papadopoulos, Tsvi Kuflik, Fang Chen, Carlos Duarte, and Wai-Tat Fu (Eds.). ACM, 523–534. https://doi.org/10.1145/3025171.3025208

[85] E. G. Sysoletin, Konstantin A. Aksyonov, X. D. Nisskhen, and Olga P. Aksyonova. 2017. Development of Components of Multi-agent CASE-System for Describing the Logic of Behavior of Mobile Robots. In *2017 European Modelling Symposium (EMS), Manchester, United Kingdom, survey = "•", November 20-21, 2017*. IEEE, 3–8. https://doi.org/10.1109/EMS.2017.12

[86] • Ville Vakkuri, Kai-Kristian Kemell, and Pekka Abrahamsson. 2021. Technical Briefing: Hands-On Session on the Development of Trustworthy AI Software. In *43rd IEEE/ACM International Conference on Software Engineering: Companion Proceedings, ICSE Companion 2021, Madrid, Spain, May 25-28, 2021*. IEEE, 332–333. https://doi.org/10.1109/ICSE-Companion52605.2021.00142

[87] Joaquin Vanschoren. 2019. Meta-Learning. In *Automated Machine Learning - Methods, Systems, Challenges*, Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren (Eds.). Springer, 35–61. https://doi.org/10.1007/978-3-030-05318-5_2

[88] • Junfeng Wang and Andrew Hogue. 2020. CVNodes: A Visual Programming Paradigm for Developing Computer Vision Algorithms. In *17th Conference on Computer and Robot Vision, CRV 2020, Ottawa, ON, Canada, May 13-15, 2020*. IEEE, 174–181. https://doi.org/10.1109/CRV50864.2020.00031

[89] • Pei Wang, Weiling Zheng, Jiannan Wang, and Jian Pei. 2021. Automating Entity Matching Model Development. In *37th IEEE International Conference on Data Engineering, ICDE 2021, Chania, Greece, April 19-22, 2021*. IEEE, 1296–1307. https://doi.org/10.1109/ICDE51399.2021.00116

[90] • Qianwen Wang, Yao Ming, Zhihua Jin, Qiaomu Shen, Dongyu Liu, Micah J. Smith, Kalyan Veeramachaneni, and Huamin Qu. 2019. ATMSeer: Increasing Transparency and Controllability in Automated Machine Learning. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Stephen A. Brewster, Geraldine Fitzpatrick, Anna L. Cox, and Vassilis Kostakos (Eds.). ACM, 681. https://doi.org/10.1145/3290605.3300911

[91] Thomas Weber, Heinrich Hußmann, and Malin Eiband. 2021. Quantifying the Demand for Explainability. In *Human-Computer Interaction - INTERACT 2021 - 18th IFIP TC 13 International Conference, Bari, Italy, August 30 - September 3, 2021, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 12933)*, Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen (Eds.). Springer, 652–661. https://doi.org/10.1007/978-3-030-85616-8_38

[92] • James Wexler, Mahima Pushkarna, Tolga Bolukbasi, Martin Wattenberg, Fernanda B. Viégas, and Jimbo Wilson. 2020. The What-If Tool: Interactive Probing of Machine Learning Models. *IEEE Trans. Vis. Comput. Graph.* 26, 1 (2020), 56–65. https://doi.org/10.1109/TVCG.2019.2934619

[93] • Jiachi Xie, Chelsea M. Myers, and Jichen Zhu. 2019. Interactive Visualizer to Facilitate Game Designers in Understanding Machine Learning. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Regan L. Mandryk, Stephen A. Brewster, Mark Hancock, Geraldine Fitzpatrick, Anna L. Cox, Vassilis Kostakos, and Mark Perry (Eds.). ACM. https://doi.org/10.1145/3290607.3312851

[94] • Doris Xin, Eva Yiwei Wu, Doris Jung Lin Lee, Niloufar Salehi, and Aditya G. Parameswaran. 2021. Whither AutoML? Understanding the Role of Automation in Machine Learning Workflows. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*, Yoshifumi Kitamura, Aaron Quigley, Katherine Isbister, Takeo Igarashi, Pernille Bjørn, and Steven Mark Drucker (Eds.). ACM, 83:1–83:16. https://doi.org/10.1145/3411764.3445306

[95] Jun Yuan, Changjian Chen, Weikai Yang, Mengchen Liu, Jiazhi Xia, and Shixia Liu. 2021. A survey of visual analytics techniques for machine learning. *Comput. Vis. Media* 7, 1 (2021), 3–36. https://doi.org/10.1007/s41095-020-0191-7

[96] Qishen Zhang, Tamás Kecskés, Janos L. Mathe, and Janos Sztipanovits. 2019. Towards bridging the gap between model- and data- driven tool suites for cyber-physical systems. In *Proceedings of the 5th International Workshop on Software Engineering for Smart Cyber-Physical Systems, SEsCPS@ICSE 2019, Montreal, QC, Canada, May 28, 2019*, Tomás Bures, Bradley R. Schmerl, John S. Fitzgerald, and Danny Weyns (Eds.). IEEE / ACM, 7–13. https://doi.org/10.1109/SEsCPS.2019.00009

[97] • Xufan Zhang, Ziyue Yin, Yang Feng, Qingkai Shi, Jia Liu, and Zhenyu Chen. 2019. NeuralVis: Visualizing and Interpreting Deep Learning Models. In *34th IEEE/ACM International Conference on Automated Software Engineering, ASE 2019, San Diego, CA, USA, November 11-15, 2019*. IEEE, 1106–1109. https://doi.org/10.1109/ASE.2019.00113

[98] • Chunqi Zhao, I-Chao Shen, Tsukasa Fukusato, Jun Kato, and Takeo Igarashi. 2022. ODEN: Live Programming for Neural Network Architecture Editing. In *IUI 2022: 27th International Conference on Intelligent User Interfaces, Helsinki, Finland, March 22 - 25, 2022*, Giulio Jacucci, Samuel Kaski, Cristina Conati, Simone Stumpf, Tuukka Ruotsalo, and Krzysztof Gajos (Eds.). ACM, 392–404. https://doi.org/10.1145/3490099.3511120

[99] • Zhibin Zhou, Qing Gong, Zheting Qi, and Lingyun Sun. 2019. ML-Process Canvas: A Design Tool to Support the UX Design of Machine Learning-Empowered Products. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*, Regan L. Mandryk, Stephen A. Brewster, Mark Hancock, Geraldine Fitzpatrick, Anna L. Cox, Vassilis Kostakos, and Mark Perry (Eds.). ACM. https://doi.org/10.1145/3290607.3312859

[100] • Zhi-Jie Zhou, Guan-Yu Hu, Chang-Hua Hu, Cheng-Lin Wen, and Leilei Chang. 2021. A Survey of Belief Rule-Base Expert System. *IEEE Trans. Syst. Man Cybern. Syst.* 51, 8 (2021), 4944–4958. https://doi.org/10.1109/TSMC.2019.2944893