IBM AutoAI: Human-in-the-Loop Automated Machine Learning Supports Data Scientists to Build Better Models

Dakuo Wang
IBM Research
dakuo.wang@ibm.com

Josh Andres
IBM Research

Justin Weisz
IBM Research

Erick Oduor
IBM Research

Udayan Khurana
IBM Research

Horst Samulowitz
IBM Research

Arunima Chaudhary
IBM Research

Abel Valente
IBM Research

Dustin Torres
IBM Research

Casey Dugan
IBM Research

ABSTRACT

Data science (DS) often refers to the data-driven decision making process, which consists a set of highly interdisciplinary labor-intensive tasks. Lately, researchers have built various automated machine learning (AutoML) systems to automated various DS tasks and have yield some promising results. In this paper, we introduce IBM AutoAI, a human-in-the-loop automated machine learning system that can support data scientists with various DS tasks. Further, we conduct a between-subject user study to evaluate the system's usability with a focus on the modeling building task. Professional data scientists (N = 30, Years of Practice = 3.5) were randomly assigned into one of the two conditions: in the Notebook condition, participants were asked to build a model in Jupyter Notebook, which replicated their existing work practice; in the AutoAI condition, the participants were firstly introduced to AutoAI, and then were asked to build a model using AutoAI. Each participant was given the same dataset and the same amount of time (30 mins). Our preliminary result showed that participants in the AutoAI condition built models three times faster, and the model had a higher accuracy score, when compared to the Notebook condition. We discussed the implications for human-in-the-loop automated data science system design and future research directions.

CCS CONCEPTS

• **Human-centered computing** → *User studies*; *Empirical studies in HCI*; • **Computing methodologies** → *Artificial intelligence.*

KEYWORDS

Data science, automated data science, automated machine learning, AutoML, model building, human-in-the-loop, AI

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
DaSH@KDD, August 24, 2020, Virtual Conference
© 2020 Association for Computing Machinery.

ACM Reference Format:
Dakuo Wang, Josh Andres, Justin Weisz, Erick Oduor, Udayan Khurana, Horst Samulowitz, Arunima Chaudhary, Abel Valente, Dustin Torres, and Casey Dugan. 2020. IBM AutoAI: Human-in-the-Loop Automated Machine Learning Supports Data Scientists to Build Better Models. In *Proceedings of (DaSH@KDD).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION AND BACKGROUND

Data Science (DS) refers to the practice of applying statistical and machine learning approaches to analyze data and generate insights for decision making or knowledge discovering [6, 30, 32]. It involves a wide range of tasks from understanding the problem to coding and training a machine learning model [32, 42]. Thus, a DS project often requires an interdisciplinary DS team (e.g., domain experts and data scientists) [28]. Previous literature suggest that as much as 80% of DS team's time [39] spent on low-level activities such as tweaking data [18] or trying and selecting various candidate algorithms [45]; thus, they do not have enough time for the actual knowledge discovery activities. To cope with this challenge, researchers have started exploring the use of machine learning algorithm (i.e. bayesian optimization) to automated the low-level activities, such as automatically training and selecting a best algorithm from all the candidates [25, 27, 29, 45], this group of work is called Automated Data Science (or **AutoDS** in short) [1].

Many research communities, universities, and companies have recently made significant investments in AutoDS research, under the promise that the world has so much data and there is an unprecedented demand for data scientists [17]. For example, Google released AutoML in 2018 [13]. Startups like H2O [16] and Data Robot [5] both have their branded products. There are also Auto-sklearn [10] and TPOT [8, 33] from the open source communities.

However, little is known on how data scientists would actually use an AutoDS system. To fill this gap, we designed a prototype AutoDS system, IBM AutoAI[2], and conducted this between-subject user study We recruited 30 professional data scientists and assigned

[1] Some researchers also refer to Automated Artificial Intelligence (AutoAI). In this paper, we use **AutoDS** to refer the collection of all these technologies, AutoML, AutoDS, AutoDS.
[2] When this study was conducted in 2019, IBM AutoAI was a prototype system [41]. Now, it is available at https://www.ibm.com/cloud/watson-studio/autoai.

them into one of the two conditions: half of them were asked to build models using AutoAI, and the other half were asked to buld models using Python in a Jupyter Notebook environment [20]. We acknowledge there are many tasks in a data scientist's workflow (e.g., data preparation, model building, model deployment, etc. [32, 42]), we focus only on the **model building** task. Then, we collected various measurements to quantify the participant's performance and the final outcome, such as how long each participant spent in building the model v.s. evaluating the model, whether they made human-error, and the final model's accuracy.

We found that the participants with AutoAI's support generated more models (8 models v.s. 3.13 models), faster (5 minutes v.s. 15 minutes), and with less human errors (0 out of 15 v.s 7 out of 15 made mistakes). To the best of our knowledge, this is the first work that reports a systematic investigation of users' behaviors in using an AutoDS system in a controlled experiment setting. Our result can shed lights on future designs of AutoDS systems.

2 RELATED WORK

2.1 Human-Centered Machine Learning

Many researchers have studied how data scientists work with data and models. For example, it was suggested that 80 percent of the time of a data science project is spent in data preparation [15, 32, 44, 45]. As a result, data scientists often do not have enough time to complete a comprehensive data analysis [40]. Passi and Jackson suggested that when data scientists preprocess their data, it is often a rule-based, not rule-bound, process [34]. Pine and Liboiron further investigated how data scientists made those rules [37].

A popular topic in this domain is the user experience of interactive machine learning. **Interactive machine learning** research aims to design better user experiences for human users to provide more information to the machine learning tool [1]. These are often labellers of a data sample or domain experts who hve better domain knowledge that the machine learning model does not easily reflect.

Based on these empirical findings, many tools were built to support data scientists' work practices [24, 38**?**]. For example, Jupyter Notebook [20] and its variations such as Google Colab [14] and Jupyter-Lab [21] are widely adopted by the data science community. These systems provide an easy code-and-test environment with a graphical user interface so that data scientists can quickly iterate their model crafting and testing process [26, 32]. Another group of tools includes the Data Voyager [19] and TensorBoard [4] systems that provide visual analytic support to data scientists to explore their data, but they often stop in the data preparation stage and thus do not provide automated support for model building tasks.

Today's AutoDS systems focus mostly on the model building and data analysis tasks, with only a few exceptions that cover the data preparation tasks. In the context of AutoDS in this paper, we focus only on the model building task, and study how data scientists can collaborate with AutoDS to build models.

2.2 Human-in-the-Loop AutoDS

AutoDS refers to a group of technologies that can automate the manual processes of data pre-processing, feature engineering, model selection, etc. [45]. Several technologies have been developed with different specializations. For example, Google has developed a set of AutoDS products under the umbrella of Cloud AutoML, such that even non-technical users can build models for visual, text, and tabular data [13]. H2O is java-based software for data modelling that provides a python module, which data scientists can import into their code file to use the automation capability [16].

Despite the extensive work in building AutoDS systems and algorithms, only a few recent efforts have focused on the interaction between humans and AutoDS. Gil and collaborators proposed a guideline for designing AutoDS systems [12]. However, they envisioned new design features for AutoDS based on their understanding of how data scientists manually build models, and their understandings were from surveying the previous literature and their personal experience. In our study, we conducted an experiment to systematically examine how data scientists actually used AutoDS in a lab setting.

Wang et al. presented a multi-granularity visualization system, ATMSeer, that enables users to browse AutoDS processes at the algorithm, hyperpartition, and hyperparameter levels [45]. Their results indicated that users with a higher level of expertise in machine learning were more willing to interact with ATMSeer [43]. But their work only focused on the visualization aspect and revealed information on existing AutoDS pipelines, and they did not measure the data scientists' behaviors. Thus, the feedback from these users were limited to the AutoDS visualization user interface, not to the automated model building functionality.

Another recent work studied data scientists' perceptions and attitudes towards AutoDS systems through an interview with 30 data scientists [42]. The interviewees, who have never interacted with AutoDS before, believed that AutoDS could potentially change their work practice. They also believed automation in data science work was the future, but they certainly hoped that such automation can support their jobs instead of sabotaging them. We follow this line of research. We aim to provide an account for the actual user behaviors when a data scientist uses an AutoDS system to build a model.

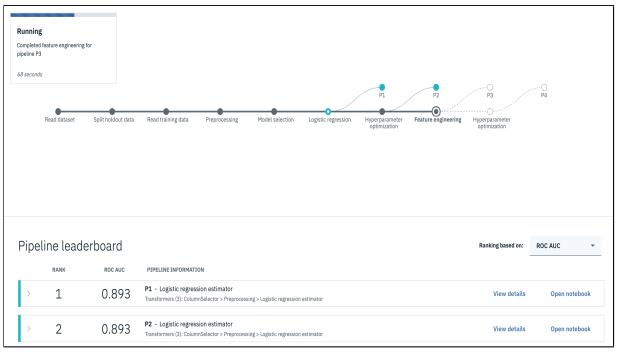
3 METHOD

3.1 Experimental Procedure

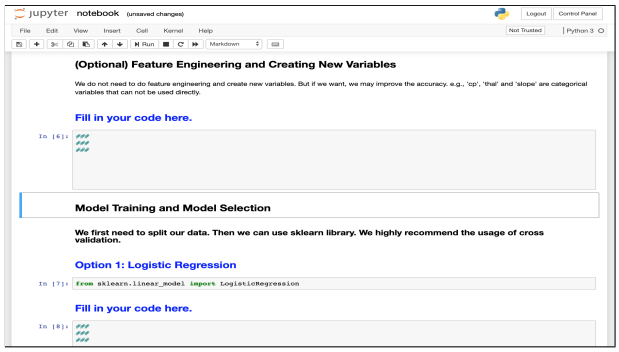
We designed and conducted a between-subject lab experiment to understand how the introduction of AutoAI can change the work practices of data scientists, following a lab experiment study set up [11]. The prototype AutoAI system (shown in Figure 1 (a)) is a web-based system that allows user to upload any tabular datasets and it can then automatically detect the problem typs (e.g., classification or regression), clean the data, train and select models, and present a list of model results in a leaderboard and tree-chart visualization. The user can download a model result as a model pickle file or as a Jupyter Notebook for further evaluation or deployment.

Participants were asked to try their best to build a machine learning model for the given UCI Heart Disease dataset [3], either by writing Python code in a Jupyter Notebook (**Notebook Condition**) [36] or by using the AutoAI prototype system (**AutoDS Condition**), shown in Figure 1.

To simplify the study setting and also to be focused only on the model building task, we selected the UCI Heart Disease dataset as it contains only 303 patients' basic medical record information



(a) Screenshot of results from our experimental AutoDS system. This figure shows how four pipelines are constructed (top visualization), as well as a leaderboard that ranks models according to a selected metric (bottom list). Both models shown use logistic regression, but model pipeline 2 (P2) includes a hyperparmaeter optimization step, differentiating it from model P1. Models P3 and P4 are in process. The "Open notebook" button automatically generates and presents the Python code for a selected pipeline.



(b) Screenshot from the Jupyter Notebook used in the study. The Notebook contained skeleton code to load relevant data science libraries (numpy, pandas, skikit-learn), as well as the data set. Instructional sections in the Notebook were provided to remind participants to perform the following tasks: data preprocessing (optional), feature engineering (optional), model training (with 5 different algorithms recommended), and model comparison (optional).

Figure 1: (a) Experimental AutoDS user interface. (b) Jupyter Notebook user interface.

(13 features with both continuous and categorical values) that can be used to predict whether the patients have heart disease (target feature with 0 represents no disease). It was widely used benchmark dataset in research papers as well as in machine learning competitions (e.g., Kaggle [22]). No further data cleaning or preparation was needed. In both conditions, the dataset (and required libraries in Notebook) was pre-loaded so participants did not need to find the data.

Participants were given up to 30 minutes to build their best model in one of the two conditions. During this time, participants were free to try out more than one model. By the end of the experiment, participants were asked to submit the best model's name and ROC AUC score[3]. They were allowed to submit early if they were satisfied with the model. All experiment sessions were conducted remotely using a video conferencing system, and participants used their own laptops during the experiment. We asked the participants to share their screens, and with their consent, we recorded each session for further analysis.

3.2 AutoDS Condition

Participants in the **AutoDS Condition** used an AutoAI prototype system, shown in Figure 1. AutoAI can automate the following portions of the data science process: reading data; detecting the type of prediction needed (e.g. binary classification, multi-class classification, or regression); splitting the data into separate training, testing, and validation subsets; training all candidate models and selecting the top ones; engineering new features; and optimizing hyperparameters.

As a default setting, AutoAI can generate a list of the 4 model pipelines. A model pipeline corresponds to all of the steps taken in generating the model, from reading data to optimizing hyperparameters. Not all pipelines had the hyperparameter optimization step or feature engineering step, leading to the variety of generated models and their accuracy scores. Participants could modify AutoAI setting and get up to 8 model pipelines.

Participants could inspect the generated model results with a visualization and a leaderboard showing information such as the confusion matrix, a feature importance chart, and tables with various evaluation metrics (e.g. ROC AUC, precision, F1, etc.) and descriptions of feature transformations. In addition, the AutoDS systems can automatically generate Python codes in the Jupyter Notebook format for user to download. Thus, participants were also able to view, execute, or edit the generated code and further improve the model generated by AutoAI.

3.3 Notebook Condition

Participants in the **Notebook Condition** were provided with an online Jupyter Notebook environment to replicate their current work environment. To simplify the task and to provide a fair comparison, we designed the task to help participants focus only on the model building part of coding. Thus, we provided notebooks with pre-scripted code sections and a skeleton of instructions. The notebooks were preloaded with data, and necessary libraries. And we provided instructions in the markdown cells in different sections. For example, we listed "(Optional) Feature Engineering section", "(Required) Modeling Training and Selection section", and blank code cells for users to fill in their code. In the model training and selection section, we suggest five commonly used algorithms for this particular task: Logistic Classifier, KNN, SVM, Decision Tree, and Random Forest [35]. Note that from our pilot runs, a data scientist only needed to write no more than 10 lines of code to get the expected result: split train and test data subsets, define a model variable, and run a cross-validation function to train the model

[3] one of the widely-used model performance metrics [35]. For simplicity, we refer to this metric as "accuracy" in the rest of paper.

and report the accuracy score. Out of the 5 algorithms, Random Forest gives the best performance with accuracy score at "0.925" on a 3-fold cross-validation set up.

3.4 Measures

We are interested in how data scientists users may perform differently in the two conditions. Thus, we manually analyzed the video recording data to extract measures such as how many models each participant generated, what the final algorithm they selected, and what its accuracy was. We also captured time-related measures such as how long they spent on the model building task.

We performed three pilot study sessions, all our pilot participants were able to accomplish the task in less than 30 minute. We considered a participant completed the task if they got at least one model. However, some of their solutions contained human-errors. For example, in the Notebook condition, some participants forgot to split the data, and thus reported a very high but incorrect accuracy score on the training dataset. We created one additional marker to track whether a participant made mistake or not.

3.5 Participants

We recruited 30 professional data scientists in IBM and randomly assigned them to one of the two conditions (AutoDS v.s. Notebook). These participants come from different locations from the U.S., Africa, and Australia. Seven participants out of 30 were female (23%), which was similar to the 16.8% ratio reported in the Kaggle 2018 survey of data science [23].

Our recruitment criteria was that the participants were professional data scientists and they practiced data science or machine learning works in their day-to-day work. Participants reported practicing data science for an average of 3.5 years (SD = 2.7). Six participants (20%) rated themselves as beginners, 17 (57%) as intermediates, and 7 (23%) as experts in data science. Participants had a moderate amount of experience (M (SD) = 3.2 (.97) out of 5) with SK-learn, the Python library we used for our study.

4 RESULTS

All 15 participants (100%) in AutoDS condition and eight participants (53%) in the Notebook condition finished the model building task without any human error. We refereed to these participants as the "successful" participants. In contrast, seven participants (47%) from Notebook condition made one or more mistakes in the process. Common human-errors included the participant forgot to split the dataset into training and testing subsets, or they reported a different accuracy metric other than the required ROC AUC score

The experiment time was limited up to 30 minutes for both the conditions, but participants could finish the task early if they were confident in the model they generated. Out of all 30 participants, the fastest participant to reach a state of completion was P23 (Notebook), who finished the task early in 17.9 minutes with an accuracy score of 0.923. This was an extraordinarily good performance of human data scientists in the Notebook condition, because on average participants spent 15 minutes (SD = 6.8 minutes) in generating 1 model. For comparison, AutoDS generated 4 model pipelines in about 5 minutes. **This is a 300% productivity increase.**

As for the model chosen as the final model, 18 participants chose Random Forest (also the best one we achieved prior to the study), 9 chose Logistic Regression, 2 used SVM and 1 chose ExtraTree. The AutoDS condition outcome was dominated by Random Forest (13 out of 15), while 2 participants chose Logistic regression. In the Notebook condition, participants' choices were more diverse.

4.1 How Data Scientists Worked with AutoAI

In this subsection, we report how participants built models using AutoAI.

Participants needed to explore the data and understand the problem, they did so in the AutoDS configuration step, e.g., viewing the data distributions and setting prediction type to binary classification (M (SD) = 2.3 (.96) min).

Feature engineering, model training, and model selection steps were fully automated by AutoDS; we tracked time as participants waiting for AutoDS to complete runs (M (SD) = 2.1 (.59) min).[4]

> "It's fast and it gives you visualisations to compare the models, this saves me a lot of time" (P8, M, AutoAI)

Because it was fast for data scientists in AutoDS condition to generate models (in total less than 5 minutes), participants had more time to further examine or improve the model pipeline results produced by AutoDS. They spent most of their time in examining the leaderboard and visualization as shown in Figure 1 (M (SD) = 9.7 (3.4) min). Then, they also went into particular model's detailed information page for further information (M (SD) = 3.9 (2.2) min), and some of them downloaded the generated notebook and changed the code in it to further improve the model (7 out 15 participants, spent an average of 2.3 (SD = .70) minutes).

The path of selecting the best model in AutoDS condition seemed straightforward: a participant ran AutoDS and selected the model with the "highest performance AUC ROC score". But some participants did not select the top performance model as their final choice. Of the 8 possible models that could be produced with AutoDS, 8 participants chose the highest-scoring one (model pipeline 7 "P7") with a Random Forest algorithm, and 7 chose a different model with a lower ROC AUC score than P7. It appeared that even though P7 with the highest ROC AUC score was available to the participant, in some cases they had already invested time and effort in inspecting and tweaking parameters within an earlier generated pipeline, in the end selecting a pipeline with a lower ROC AUC score than P7. The mean ROC AUC score for the final selected model was 0.919 (SD = .01).

4.2 How Data Scientists Worked with Notebooks

We collected the same measurements from the Notebook condition as a baseline to compare with AutoDS condition. But because the work practices are fundamentally different, for some measurements, we could only collect proxy indicators. we found that participants spent 15 minutes (SD = 6.8 minutes) in **building the first model**. Note that this measure also included the time participants spent on **pre-processing** and **splitting data** (9.4 minutes, SD = 5.5 minutes),

[4] In our study, the dataset was simple, so it took AutoDS only a few minutes to get the first model generated; with a more complicated dataset (e.g., hundreds of MB data file), this training step could take hours.

so the actual model building task did not take that long. A couple participants (N = 2) were marked "un-successfull" as they did not perform the necessary data splitting step.

In investigating the dataset, seven participants (47%) performed some form of exploratory data analysis, either by generating a description of the tables, or by producing graphs and charts. From the post-study interview, almost all participants mentioned that if they had had more time, they would have loved to conduct more data exploration or maybe to learn more domain knowledge about the heart disease field, and consulted with domain experts.

> "... I need to talk with domain experts or doctors to understand the data and feature ... whether my current model makes sense ..." (P29, M, Notebook)

Only four participants (27%) performed some form of feature engineering, by transforming, scaling, or combining existing features to create new features. Some argued that they did not have time, while a few others stated they did not need to do so for this simple dataset.

About the quantity of the outcome, Notebook participants tried on average 3.13 (SD = 1.69) models. Three participants focused all of their attention on just one model, and four participants tried all five suggested models. Despite finishing the task minutes early, participants who finished only one model argued they did not have enough time to try more models, or so they believed.

In summary, the results from the Notebook condition echoed what previous literature reported as a data science workflow. This result suggested our Notebook condition study design replicated data scientists' current day-to-day work practice. And by tracking various behavioral and perceptional measures, we had a solid baseline metrics to compare with the measures we collected from the AutoDS condition.

4.3 Manually-Crafted Models v.s. AutoDS-Generated Models

We will use this sub-section to report the differences between the model building practices in the two conditions.

4.3.1 AutoDS can build more models, use less time, and have fewer human-errors. AutoDS easily generated 8 models with various algorithm and hyperparameter combinations, whereas participants in Notebook condition generated 3.13 (SD = 1.69) models. To provide a fair comparison, participants in AutoDS examined in detail 3.8 (SD = 1.2) models out of the 8 models, which served as a candidate set for participants to select the final model. That count was still higher than those in Notebook condition, though the t-test shows the difference is not signification $t(28) = 1.25$, $p = .223$.

About the accuracy of the final selected model, 7 participants in the Notebook condition reported either an accuracy score other than the ROC AUC metric, or reported the score from the same training data set; as a result, their scores were not considered as valid model scores. We conducted an analysis of variance (ANOVA) to compare the ROC AUC scores between participants with AutoDS and only the successful Notebook participants. When reporting ANOVA results, we included effect size as partial η^2, which is the proportion of variance accounted for by each factor in the model, controlling for all other factors. To interpret effect sizes, Miles & Shevlin [31] gives an guidance that partial $\eta^2 \geq .14$ is a large effect, $\geq .06$ is a medium-sized effect, and $\geq .01$ is a small effect. Participants in AutoDS conndition generated and selected the final model with higher ROC AUC scores (M (SD) = .919 (.01)) than participants with Notebooks (M (SD) = .899 (.02)), $F[1, 21] = 7.61$, $p = .01$, partial $\eta^2 = .27$.

As aforementioned, participants spent much more time in building models in Notebook condition than in AutoDS condition. Even if we compared only the time participants spent on the first model in Notebook (time to first model M (SD) = 15.0 (6.8) min) v.s. the time they spent in generating 8 models in the AutoDS condition (config + run time M (SD) = 4.3 (1.0) min), that difference is still significant $t(14.7) = 6.0$, $p < .001$. One factor that contributes to this time difference could be that participants in the Notebook condition spent a significantly greater amount of time searching for SK-learn library documentation on the web (M (SD) = 6.4 (2.6) min) than participants in the AutoDS condition (M (SD) = 0.93 (1.4)) (B3), $t(21.3) = -7.2$, $p < .001$.

4.3.2 Data scientists trust more in manually-crafted models than AutoDS-generated models. Other than the productivity of model building and the performance score of the model result, we also collected how much trust each participant has in their final selected model at the end of the study. Overall ratings of confidence landed in the middle of the scale (M (SD) = 2.9 (.97) out of 1 to 5, 5 is strongly trusted), suggesting that participants were not satisfied with the produced model and they could have produced even better models if they had had more time.

Participants in AutoDS condition were significantly less confident (M (SD) = 2.4 (.98)) in their models than the participants had in the Notebook condition (M (SD) = 3.3 (.72)), $t(25.7) = -2.9$, $p < .01$. An important consideration here was that more confidence did not equate to better models. This may suggest that while the participants in the AutoDS condition had more model results generated by AutoDS very faster, their counterparts in the Notebook condition were slowly invested in writing code line by line. This investment of time and effort into a model could influence users' perspectives when it came to how confident they were.

5 DISCUSSION

Motivated by literature, we began the investigation on how data scientists use an AutoDS system in model building task. Through our analyses, we found that participants working with AutoDS can create more models, faster, and with fewer human-errors, comparing to working alone with a Jupyter Notebook coding environment. With AutoDS's help, data scientists can afford to spend more time in understanding the results and inspecting the codes generated by AutoDS, whereas in the baseline condition participants had to spent lots of their efforts and time in searching for library documentations and writing code from scratch. However, despite such productivity gain from AutoDS, participants still trust their manually-crafted models more than the AutoDS generated model. These results enlighten us about the future of data scientists work, and suggest design improvement for the existing technology.

5.1 Toward The Future of Human-in-the-Loop AutoDS Research and System Design

We believe AutoDS technologies will become available to more and more data scientists, affording them a chance to craft models together with AI. We agree with the majority of our participants that AutoDS will not replace data scientists' job. Instead, it will play an important assistant role to the human data scientists in the data science workflow. Data scientists' role will shift, as suggested by our findings: they may spend less time in the tedious modeling training and hyperparameter tuning tasks, but spend more time in understanding the data, communicating with the domain experts, and selecting the model to best fit the domain and context. In addition, they will need to spend a big proportion of their time to rationalize the AutoDS decisions and results, and translate those decisions to present to other human stakeholders. The future in which data scientists can work together with AutoDS is promising, but there are works needed to be done to achieve that vision.

Our result showed that some participants worked over a wider range of AutoDS results, some others focused on improving the code of one particular model. It might be possible that the more successful participants were the ones who tried a wider range of approaches rather than focusing all of their attention on just one model. This finding echoed wisdom in the artistic domain, where the most creative works emerge when favoring quantity over quality [2]. And this was not isolated that the data science work appears linked with a practice in the art design domain, as [32] argued that "Data science workers work is a new type of crafting", while in the AutoDS context, users are crafting the AutoDS generated results as a material.

Our results suggested that AutoDS prototype's code generation feature, where it can automatically translate a resulting model into Python code, is very welcomed by the participants. They could inspect the codes to see the decisions made by AutoDS. They can rationalize some of those decisions, and they could even change the code and see if that changes the result as expected. Furthermore, we observed there are a couple participants who kept working on the code generated by AutoDS to achieve a better model, that use case again supports our claim that data scientists and AutoDS together could deliver better results. Other than the possible use of the entire code, data scientists could also easily migrate or replicate a segment of the generated code, maybe with a transformation function, some hyperparameters, or other decisions made by AutoDS into a different task and context. Thus, we highly recommend AutoDS systems to provide such code-generation feature.

The fact that people loved the code-generation feature could also attribute to the needs of a transparent and trusted AutoDS. Participants wanted to understand not only what the decisions are (code-generation satisfied that), but also why AutoDS made such decisions. However, only the generated code for the result model was not enough. We need to design AutoDS systems to provide better explanation for why certain decisions were made. For example, when showing users the top performed algorithm and the optimized set of hyperparameter values, we could also reveal the information about all the other candidates considered, and why AutoDS eliminated those options. Another example is that for the new features engineered by AutoDS, users should have a better view of how those features are generated and chosen in the middle of the feature engineering process, because users may have some domain knowledge to prevent the nonsense transformations (e.g., absolute value of gender). We are glad to see that recently there are some researchers moving toward this direction [7]. We should also caution readers that we are not arguing to show a user all the information AutoDS has. Full transparency causes information overload and is no better than no transparency. This links back to a classic social computing theory: "social translucent" [9], where researchers argue to show the information at the right moment and at the right level of details, and we should learn from our past.

This collaborative step that emerged over practice, between data scientist and the AutoDS, appears to us as a *demythification* of the AutoDS replacing the job of the data scientist, and instead, offers a perspective on how the relationship between this parties towards a collaborative partnership is maturing. We argue a **Human-AI Collaboration** future, where human data scientists can work on the understanding of domain knowledge and context tasks, while AutoDS systems work on the computation tasks. Then human data scientists can work on result interpretation tasks, while AutoDS can work on deployment tasks. They hand over information to each other and share responsibility. Together, AutoDS plays a partner role in the future of human-in-the-loop data science practice.

5.2 Limitations

We acknowledge our works limitations as follows: this is a lab experiment study, thus it has the common limitations as any other lab experiments [11]. For instance, the task and data set is simple and specific to model building task, thus participants may have different behaviors and perceptions toward AutoDS in other tasks or dataset. But we argue that we used the same task and data set for the AutoDS and Notebook conditions, thus the comparison result between these two stands. Another limitation is that all the participants are professional data scientists coming from a technology company, there may be some selection bias in their background. Another limitation is that the results collected from this lab experiment could be different from the ones in a real-world deployment. We hope to further explore these in future research.

6 CONCLUSION

Our work reveals collaborative details pertaining the indispensable role that humans and AutoDS play to create better models together, through a between-subject experiment with 30 professional data scientists building models with Notebook or with AutoDS. Grounded by the results, we propose design implications and also extend the discussion of Human-AI Collaboration to contexts beyond data science.

ACKNOWLEDGMENTS

We thank all the participants who contributed their time participate our experiment. We also want to thank Alex Swain, Dillon Eversman, and Voranouth Supadulya for their inspiring UX design sketches, and Gregory Bramble, Theodoros Salonidis, Peter Kirchner, and Alex Gray for their input on the algorithm implementation.

REFERENCES

[1] Saleema Amershi, Maya Cakmak, William Bradley Knox, and Todd Kulesza. 2014. Power to the people: The role of humans in interactive machine learning. *AI Magazine* 35, 4 (2014), 105–120.

[2] David Bayles and Ted Orland. 2001. *Art & fear: Observations on the perils (and rewards) of artmaking*. Image Continuum Press.

[3] Catherine L Blake and Christopher J Merz. 1998. UCI repository of machine learning databases, 1998.

[4] Tommy Dang, Fang Jin, et al. 2018. Predict saturated thickness using tensorboard visualization. In *Proceedings of the Workshop on Visualisation in Environmental Sciences*. Eurographics Association, 35–39.

[5] DataRobot. [n.d.]. Automated Machine Learning for Predictive Modeling. Retrieved 3-April-2019 from https://www.datarobot.com/

[6] David Donoho. 2017. 50 years of data science. *Journal of Computational and Graphical Statistics* 26, 4 (2017), 745–766.

[7] Jaimie et al. Drozdal. 2020. Exploring Information Needs for Establishing Trust in Automated Data Science Systems. In *IUI'20*. ACM, in press.

[8] EpistasisLab. [n.d.]. *tpot*. Retrieved 3-April-2019 from https://github.com/EpistasisLab/tpot

[9] Thomas Erickson and Wendy A Kellogg. 2000. Social translucence: an approach to designing systems that support social processes. *ACM transactions on computer-human interaction (TOCHI)* 7, 1 (2000), 59–83.

[10] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. 2015. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*. 2962–2970.

[11] Darren Gergle and Desney S Tan. 2014. Experimental research in HCI. In *Ways of Knowing in HCI*. Springer, 191–227.

[12] Yolanda Gil, James Honaker, Shikhar Gupta, Yibo Ma, Vito D'Orazio, Daniel Garijo, Shruti Gadewar, Qifan Yang, and Neda Jahanshad. 2019. Towards human-guided machine learning. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*. ACM, 614–624.

[13] Google. [n.d.]. *Cloud AutoML*. Retrieved 3-April-2019 from https://cloud.google.com/automl/

[14] Google. [n.d.]. *Colaboratory*. Retrieved 3-April-2019 from https://colab.research.google.com

[15] Philip J Guo, Sean Kandel, Joseph M Hellerstein, and Jeffrey Heer. 2011. Proactive wrangling: mixed-initiative end-user programming of data transformation scripts. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 65–74.

[16] H2O. [n.d.]. *H2O*. Retrieved 3-April-2019 from https://h2o.ai

[17] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. 2018. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 784–800.

[18] Jeffrey Heer and Ben Shneiderman. 2012. Interactive dynamics for visual analysis. *Queue* 10, 2 (2012), 30.

[19] Jeffrey Heer, Fernanda B Viégas, and Martin Wattenberg. 2007. Voyagers and voyeurs: supporting asynchronous collaborative information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1029–1038.

[20] Project Jupyter. [n.d.]. *Jupyter Notebook*. Retrieved 3-April-2019 from https://jupyter.org

[21] Project Jupyter. [n.d.]. *JupyterLab*. https://www.github.com/jupyterlab/jupyterlab

[22] Kaggle. 2017. *The State of Data Science & Machine Learning*. https://www.kaggle.com/surveys/2017

[23] Kaggle. 2018. *Kaggle Data Science Survey 2018*. Retrieved 17-September-2019 from https://www.kaggle.com/sudhirnl7/data-science-survey-2018/

[24] Sean Kandel, Andreas Paepcke, Joseph Hellerstein, and Jeffrey Heer. 2011. Wrangler: Interactive visual specification of data transformation scripts. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 3363–3372.

[25] James Max Kanter and Kalyan Veeramachaneni. 2015. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE, 1–10.

[26] Mary Beth Kery, Marissa Radensky, Mahima Arya, Bonnie E John, and Brad A Myers. 2018. The story in the notebook: Exploratory data science using a literate programming tool. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 174.

[27] Udayan Khurana, Deepak Turaga, Horst Samulowitz, and Srinivasan Parthasrathy. 2016. Cognito: Automated feature engineering for supervised learning. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 1304–1307.

[28] Peter Krensky, Pieter den Harner, Erick Brethenoux, Jim Hare, Svetlana Sicular, and Shubhangi Vashisth. 2020. Magic Quadrant for data science and machine-learning platforms. *Gartner, Inc* (2020).

[29] Hoang Thanh Lam, Johann-Michael Thiebaut, Mathieu Sinn, Bei Chen, Tiep Mai, and Oznur Alkan. 2017. One button machine for automating feature engineering in relational databases. *arXiv preprint arXiv:1706.00327* (2017).

[30] Yaoli Mao, Dakuo Wang, Michael Muller, Kush Varshney, Ioana Baldini, Casey Dugan, and Aleksandra Mojsilovic. 2020. How Data Scientists Work Together With Domain Experts in Scientific Collaborations. In *Proceedings of the 2020 ACM conference on GROUP*. ACM.

[31] Jeremy Miles and Mark Shevlin. 2001. *Applying regression and correlation: A guide for students and researchers*. Sage.

[32] Michael Muller, Ingrid Lange, Dakuo Wang, David Piorkowski, Jason Tsay, Q. Vera Liao, Casey Dugan, and Thomas Erickson. 2019. How Data Science Workers Work with Data: Discovery, Capture, Curation, Design, Creation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, UK) *(CHI '19)*. ACM, New York, NY, USA, Forthcoming.

[33] Randal S Olson and Jason H Moore. 2016. TPOT: A tree-based pipeline optimization tool for automating machine learning. In *Workshop on Automatic Machine Learning*. 66–74.

[34] Samir Passi and Steven Jackson. 2017. Data vision: Learning to see through algorithmic abstraction. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*. ACM, 2436–2447.

[35] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[36] Fernando Perez and Brian E Granger. 2015. Project Jupyter: Computational narratives as the engine of collaborative data science. *Retrieved September* 11, 207 (2015), 108.

[37] Kathleen H Pine and Max Liboiron. 2015. The politics of measurement and action. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 3147–3156.

[38] Tye Rattenbury, Joseph M Hellerstein, Jeffrey Heer, Sean Kandel, and Connor Carreras. 2017. *Principles of data wrangling: Practical techniques for data preparation*. " O'Reilly Media, Inc.".

[39] Armand Ruiz. 2017. The 80/20 data science dilemma. *InfoWorld* (2017).

[40] Charles Sutton, Timothy Hobson, James Geddes, and Rich Caruana. 2018. Data diff: Interpretable, executable summaries of changes in distributions for data wrangling. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2279–2288.

[41] Dakuo Wang, Parikshit Ram, Daniel Karl I Weidele, Sijia Liu, Michael Muller, Justin D Weisz, Abel Valente, Arunima Chaudhary, Dustin Torres, Horst Samulowitz, et al. 2020. AutoAI: Automating the End-to-End AI Lifecycle with Humans-in-the-Loop. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*. 77–78.

[42] Dakuo Wang, Justin D. Weisz, Michael Muller, Parikshit Ram, Werner Geyer, Casey Dugan, Yla Tausczik, Horst Samulowitz, and Alexander Gray. 2019. Human-AI Collaboration in Data Science: Exploring Data Scientists' Perceptions of Automated AI. *To appear in Computer Supported Cooperative Work (CSCW)* (2019).

[43] 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*. ACM, 681.

[44] Amy X Zhang, Michael Muller, and Dakuo Wang. 2020. How do data science workers collaborate? roles, workflows, and tools. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (2020), 1–23.

[45] Marc-André Zöller and Marco F Huber. 2019. Survey on Automated Machine Learning. *arXiv preprint arXiv:1904.12054* (2019).