Logging Data Scientists: Collecting Evidence for Data Science Automation

If we really want to automate data science we need to know how data scientists behave. In other words, we have to apply data science to data scientists. However, it seems very difficult to track all the activities a data scientist (or a data science team) is doing. Indeed, apart from a few surveys (about the tools and times they devote to every stage of the whole process), there is a lack of evidence about what data scientists really do and the decisions and actions they take, especially at a high granularity level. The introduction of data mining tools in the past two decades, such as SPSS Clementine (then IBM Modeler), Weka KnowledgeFlow, SAS Enterprise Miner, RapidMiner and many others that followed, made it possible, for the first time, to incorporate most of a data mining process into the same tool. However, logging the actions of the users had to be done locally, with the difficulty of obtaining a relative good number of expert experiences. Collaborative or competitive platforms such as Kaggle or Github can also be a source of data, but it is difficult to extract information about sequential workflow or the particular actions that have to be taken for all the stages of a data science project. This is aggravated by the recent "back to programming" trend, where the products of data scientists in these platforms are programs (usually in R or python), and not a sequence of actions over a structured set of possibilities. In fact, some tools that try to automate the process are based on the "knowledge, experience and best practices" of data scientists, such as DataRobot, but not based on the evidence of real logs at a high granularity level.

Things are different for cloud data science tools, and many old platforms are migrating or are native there, such as BigML[1], DataRobot[2], Azure ML[3], ClowdFlows[4] and others [Jain, 2016]. In these platforms we can log the activity of data scientists and use that activity to recommend actions according to the interactions of many data scientists on the same platforms for similar situations.

Tracking user behaviour through software interaction is a common topic in areas such as web usability (by the use of technologies involving mouse or eye tracking) [Granka et al., 2004, Mueller and Lockerd, 2001] and business intelligence (using behavioural analytics) [Vera-Baquero et al., 2013, zur Muehlen and Shapiro, 2010]. Activity logging is also common in ambient intelligence from fixed sensors or from mobile devices [Martín et al., 2013]. Nevertheless, the goal of tracking and exploiting non-trivial operational processes is represented by the area of *process mining* [van der Aalst, 2016]. Process mining seeks to process "event logs" (information about business processes stored by information systems) so as to discover, monitor and improve processes (i.e., check the conformance of processes, detect bottlenecks or predict execution problems) by means of process analytics. However, to our knowledge, process mining has not been applied to the data science process itself.

The first thing we want to analyse in this short paper is what to log and how to represent the events in data science tools such that we are able to track the full data science process. It is important that we distinguish the data or knowledge flow, as represented by many graphical data science tools (see Figure 1, bottom) from the log of actions and events (the process) that led to that flow. It is especially important that we can track mistakes, trials and other attempted actions that are not finally represented by the flow, and this can only be done from the log. Figure 1 (top) shows the procedure of extracting the events (by using specific http/network inspector tools) from Clowdflows

[1] `https://bigml.com/`

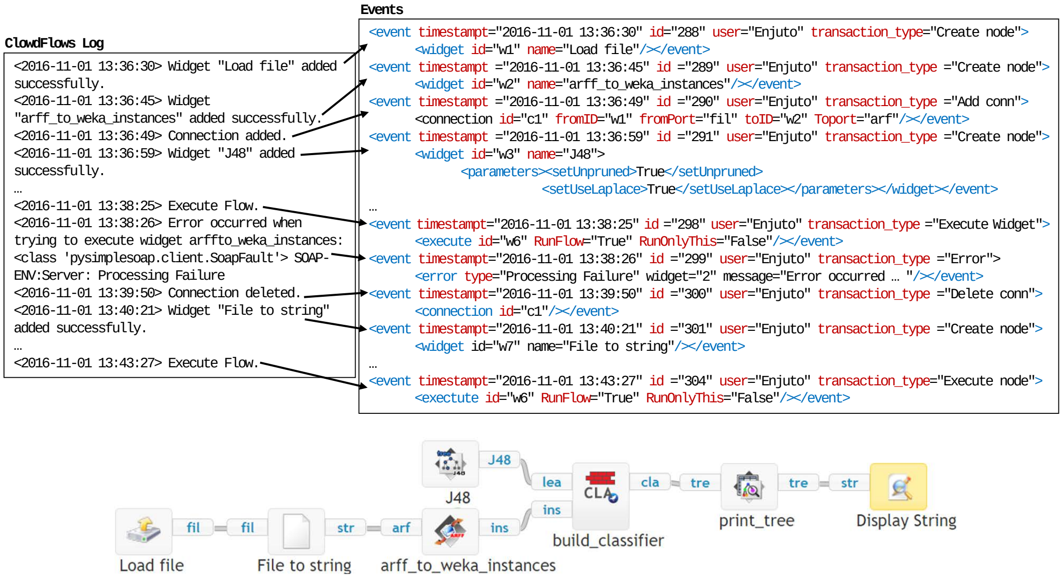
[2] `https://www.datarobot.com/`

[3] `https://azure.microsoft.com/en-us/services/machine-learning/`

[4] `http://www.clowdflows.org/`

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

(an online visual data science tool, [Kranjc et al., 2012]) including fixed attributes (timestamp, user, resource, transaction type, etc.) and event-specific ones (node identifiers, parameters, ports, inputs, outputs, etc.).



ClowdFlows Log

```
<2016-11-01 13:36:30> Widget "Load file" added
successfully.
<2016-11-01 13:36:45> Widget
"arff_to_weka_instances" added successfully.
<2016-11-01 13:36:49> Connection added.
<2016-11-01 13:36:59> Widget "J48" added
successfully.
…
<2016-11-01 13:38:25> Execute Flow.
<2016-11-01 13:38:26> Error occurred when
trying to execute widget arffto_weka_instances:
<class 'pysimplesoap.client.SoapFault'> SOAP-
ENV:Server: Processing Failure
<2016-11-01 13:39:50> Connection deleted.
<2016-11-01 13:40:21> Widget "File to string"
added successfully.
…
<2016-11-01 13:43:27> Execute Flow.
```

Events

```
<event timestampt="2016-11-01 13:36:30" id="288" user="Enjuto" transaction_type="Create node">
      <widget id="w1" name="Load file"/></event>
<event timestampt ="2016-11-01 13:36:45" id ="289" user="Enjuto" transaction_type ="Create node">
      <widget id="w2" name="arff_to_weka_instances"/></event>
<event timestampt ="2016-11-01 13:36:49" id ="290" user="Enjuto" transaction_type ="Add conn">
      <connection id="c1" fromID="w1" fromPort="fil" toID="w2" Toport="arf"/></event>
<event timestampt ="2016-11-01 13:36:59" id ="291" user="Enjuto" transaction_type ="Create node">
      <widget id="w3" name="J48">
            <parameters><setUnpruned>True</setUnpruned>
                       <setUseLaplace>True</setUseLaplace></parameters></widget></event>
…
<event timestampt="2016-11-01 13:38:25" id ="298" user="Enjuto" transaction_type ="Execute Widget">
      <execute id="w6" RunFlow="True" RunOnlyThis="False"/></event>
<event timestampt="2016-11-01 13:38:26" id ="299" user="Enjuto" transaction_type ="Error">
      <error type="Processing Failure" widget="2" message="Error occurred … "/></event>
<event timestampt="2016-11-01 13:39:50" id ="300" user="Enjuto" transaction_type ="Delete conn">
      <connection id="c1"/></event>
<event timestampt="2016-11-01 13:40:21" id ="301" user="Enjuto" transaction_type ="Create node">
      <widget id="w7" name="File to string"/></event>
…
<event timestampt="2016-11-01 13:43:27" id ="304" user="Enjuto" transaction_type="Execute node">
      <exectute id="w6" RunFlow="True" RunOnlyThis="False"/></event>
```

Figure 1: Top: Example of a log using Clowdflows and its formatting into events. Complete information about events is extracted by means of http/network inspector tools. Bottom: the corresponding data-to-knowledge flow finally completed by the data scientist.

The second issue to consider is how to analyse the data from the formatted events and the flow. As the tracked events are at a very low-level (e.g., connect a node X with Y, execute node Y), we have to use abstraction in order to match series of them with more high-level events (e.g., perform feature selection). There are several approaches to activity recognition in the literature, but we are considering two different approaches: a logical approach using event calculus, as done by [Monserrat et al., 2016], where processes can be matched with given instructions (e.g., an exercise given to a student or a data scientist), and a reinforcement learning approach, where the set of possible actions at each point is limited by the use of contextual information, where repetitive tasks can be spotted.

The extent and possibilities of this analysis and the use of other AI tools depend on the particular application, what parts of the pipeline are to be analysed and the understandability of the insight. As we can gather data science processing information from the use of Clowdflows (and perhaps in the future from some other tools, such as BigML), we are encouraging expert data scientists, practitioners and students to be logged, so that we can use all this information for a better understanding of the data science process, common mistakes and recipes for success. In terms of automation, the introduction of assistants in the same tools can be a first step, but we envisage some other possibilities along the lines of some of the ongoing initiatives for the automation of data science: Chalearn's AutoML [Guyon et al., 2016], the Automatic Statistician [Lloyd et al., 2014] and the SYNTH project [De Raedt, 2016].

Overall, the generation of this logged information as open source data can be very useful for the data science community in general, but especially useful for the application of AI to data science and ultimately for the (semi-)automation of data science in a more holistic way.

References

L. De Raedt. SYNTH Project: Synthesising inductive data models, 2016. URL http://synth.cs.kuleuven.be/.

L. A. Granka, T. Joachims, and G. Gay. Eye-tracking analysis of user behavior in www search. In *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 478–479. ACM, 2004.

I. Guyon, I. Chaabane, H. J. Escalante, S. Escalera, D. Jajetic, J. R. Lloyd, N. Macía, B. Ray, L. Romaszko, M. Sebag, A. Statnikov, S. Treguer, and E. Viegas. A brief review of the ChaLearn AutoML challenge. In *Proc. of AutoML 2016 at ICML*, 2016.

A. Jain. 19 Data Science Tools for people who aren't so good at Programming, May 2016. URL `https://www.analyticsvidhya.com/blog/2016/05/19-data-science-tools-for-people-dont-understand-coding/`.

J. Kranjc, V. Podpečan, and N. Lavrač. Clowdflows: a cloud based scientific workflow platform. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 816–819. Springer, 2012.

J. R. Lloyd, D. Duvenaud, R. Grosse, J. B. Tenenbaum, and Z. Ghahramani. Automatic construction and natural-language description of nonparametric regression models. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1242–1250. AAAI Press, 2014.

H. Martín, A. M. Bernardos, J. Iglesias, and J. R. Casar. Activity logging using lightweight classification techniques in mobile devices. *Personal and ubiquitous computing*, 17(4):675–695, 2013.

C. Monserrat, J. Hernandez-Orallo, J.-F. Dolz, M.-J. Ruperez, and P. Flach. Knowledge acquisition by abduction for skills monitoring: Application to surgical skills. In *Inductive Logic Programming*. Springer, 2016.

F. Mueller and A. Lockerd. Cheese: tracking mouse movement activity on websites, a tool for user modeling. In *CHI'01 extended abstracts on Human factors in computing systems*, pages 279–280. ACM, 2001.

W. van der Aalst. *Process Mining: Data Science in Action*. Springer, 2016.

A. Vera-Baquero, R. Colomo-Palacios, and O. Molloy. Business process analytics using a big data approach. *IT Professional*, 15(6):29–35, 2013.

M. zur Muehlen and R. Shapiro. Business process analytics. In *Handbook on Business Process Management 2*, pages 137–157. Springer, 2010.