A Machine Learning Application that Matters

Geoff Holmes[1], Dale Fletcher[2], Peter Reutemann[3], and Martijn van Oostrum[4]

[1] University of Waikato, Hamilton, NZ, geoff@waikato.ac.nz
[2] University of Waikato, Hamilton, NZ, dale@waikato.ac.nz
[3] University of Waikato, Hamilton, NZ, fracpete@waikato.ac.nz
[4] BLGG AgroXpertus, Wageningen, NL,
martijn.vanoostrum@blgg.agroxpertus.com

Abstract. TODO

1 Introduction

- Machine learning that matters?? See Wagstaffs paper (citations) http://scholar.google.com/scholar?oi=bibs&hl=en&cites=2774690908240883628
- Martijn - business side

2 ADAMS

ADAMS, the Advanced Data mining and Machine learning System, is a modular, scientifc workflow engine written in Java. Currently available modules include support for Weka, MOA, R, image processing (ImageJ, JAI, ImageMagick, Gnuplot), PDF management and display, spreadsheet manipulation (CSV, Gnumeric, Excel, ODF), scripting (Groovy, Jython), GIS support (OpenStreetMap), Twitter, time-series analysis, network support (SSH, SCP, SFTP/FTP, Email), XML/HTML/JSON processing facilities and webservice capabilities.

In contrast to current versions of other workflow engines, like Kepler, KNIME or RapidMiner, where the user places operators on a canvas and connects them manually, ADAMS organizes the operators (or actors) of a workflow automatically in a tree structure without explicit connections. Instead, so called control actors determine how data flows between actors. Examples of control actors are, e.g., Sequence, Branch, Tee, Trigger, IfThenElse and Switch. Apart from flow control, there are two further aspects to an actor: functional (primitive actor or manages nested actors) and procedural in terms of input/output of data (standalone, source, transformer, sink).

Using a tree layout has advantages and disadvantages. In terms of advantages, the layout is very compact, it scales to thousands of actors, avoids having to manually rearrange the workflow in order to include additional actors (disconnect/reconnect), is context aware when adding actors (data types of input and output limit what actors can be inserted) and has customizable rules for suggesting actors depending on context for common sequences of actors. Disadvantages are, the tree layout is less intuitive compared to a canvas-based approach and it

only supports 1-to-n connections. The 1-to-n limitation is mitigated using callable actors (multiple actors can channel data into a single actor using its name), containers for storing multiple outputs, variables and internal storage (re-using data in multiple locations). Variables can be either used in expressions, e.g., ones for evaluating mathematical formulas, or attached to parameters of operators. The latter allows for influencing the flow execution, e.g., for the turning on/off of sub-flows or dynamically changing the setup of a learning algorithm. The scope of variables and internal storage can be limited using the LocalScope control actor.

Some further feature highlights: Though ADAMS is a data-driven workflow, transporting the data in so called tokens, by design rather than an event-based one, i.e., actors get executed if there is data available for them to process, it is also possible to trigger sub-flows using cronjobs. This allows, for instance, for recurring clean-up operations. Interactive actors are very useful for developing workflow applications. These actors either prompt the user to enter or select a value, controlling sub-flow execution, or require the user to inspect data, e.g., visual inspection of images, influencing the data flow. By supporting scripting (Groovy/Jython), it is possible to quickly prototype new actors without the need of compiling Java code and restarting the workflow application. Once a workflow has been developed, it is not necessary to use the graphical user interface for executing it, the command-line can be used as well (e.g., for use in a headless server environment). Java-code generation from existing flows is possible as well.

TODO screenshots, simple example

3 AgroXpertus

ADAMS - meta-flows, S2000

4 Discussion

- Cost-benefit analysis (Martijn)
- Number of predictions (Martijn)
- Percentages going to wet chemistry (Martijn)
- Six Impact Challenges:
  2. $100M saved through improved decision making provided by an ML system → S2000
- Lack of Follow-through
  ADAMS → easily integrate ML system into business processes (opposed to plain Weka)

5 Conclusion

ADAMS matters!

References

1. Kiri L. Wagstaff (2012): Machine Learning that Matters. Proceedings of the Twenty-Ninth International Conference on Machine Learning (ICML), p. 529-536, 2012.
2. Peter Reutemann and Joaquin Vanschoren (2012): Scientific Workflow Management with ADAMS. Proceedings of the Machine Learning and Knowledge Discovery in Databases (ECML-PKDD), Part II, LNCS 7524, 2012, pp 833-837, Bristol, UK, 2012.