The Journal of Open Source Software

GAMA: Genetic Automated Machine learning Assistant

Pieter Gijsbers[1] **and Joaquin Vanschoren**[1]

1 Eindhoven University of Technology

DOI: 10.21105/joss.01132

Software

- Review
- Repository
- Archive

Submitted: 19 November 2018
Published: 30 January 2019

Summary

Successful machine learning applications hinge on a plethora of design decisions, which require extensive experience and relentless empirical evaluation. To train a successful model, one has to decide which algorithms to use, how to preprocess the data, and how to tune any hyperparameters that influence the final model. Automating this process of algorithm selection and hyperparameter optimization in the context of machine learning is often called AutoML (Automated Machine Learning).

The usefulness of AutoML is twofold. It makes creating good machine learning models more accessible to non-experts, as AutoML systems often support simple to use interfaces. For experts, it takes away a time consuming process of model selection and hyperparameter tuning. They can instead focus on related tasks such as interpreting the model or curating the data.

GAMA is an AutoML package for end-users and AutoML researchers. It uses genetic programming to efficiently generate optimized machine learning pipelines given specific input data and resource constraints. A machine learning pipeline contains data preprocessing as well as a machine learning algorithm, with fine-tuned hyperparameter settings. By default, GAMA uses scikit-learn (Pedregosa et al., 2011) implementations for preprocessing (e.g. Normalizer, PCA, ICA) and learners (e.g. Gaussian Naive Bayes, Support Vector Machine, Random Forest).

GAMA can also combine multiple tuned machine learning pipelines together into an ensemble, which on average should help model performance. At the moment, GAMA is restricted to classification and regression problems on tabular data.

In addition to its general use AutoML functionality, GAMA aims to serve AutoML researchers as well. During the optimization process, GAMA keeps an extensive log of progress made. Using this log, insight can be obtained on the behaviour of the population of pipelines. It can answer questions such as which mutation operator is most effective, how fitness changes over time, and how much time each algorithm takes. For example, Figure 1 shows fitness over time (in blue) and pipeline length over time (in red), and is (simplified) output that can be generated from the analysis log.

Related Work

There are already many AutoML systems, both open-source and closed-source. Amongst these are auto-sklearn (Feurer et al., 2015), TPOT (Olson et al., 2016), ML-Plan (Mohr, Wever, & Hüllermeier, 2018) and Auto-WEKA (Thornton, Hutter, Hoos, & Leyton-Brown, 2013). They differ in optimization strategy, programming language, target audience or underlying machine learning package.

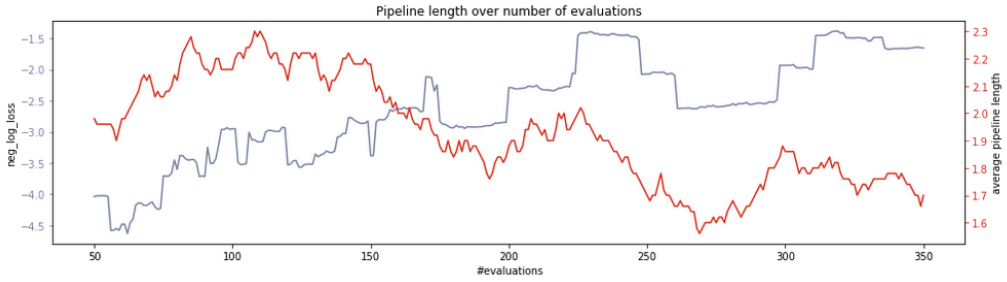

Figure 1: A simplified visualization of the optimization process based on log data. The blue line indicates the moving average of performance over time (left y-axis, higher is better). The red line represents the moving average of number of steps in the pipeline over time (right y axis, lower is better).

Most closely related to GAMA is TPOT (Olson et al., 2016). TPOT is also a Python package performing AutoML based on genetic programming using scikit-learn pipelines. The major difference between the two is the evolutionary algorithm used. TPOT uses a synchronous evolutionary algorithm, whereas GAMA uses an asynchronous algorithm. Other differences include having a CLI (TPOT), exporting independent Python code (TPOT), direct ARFF support (GAMA) and visualization of the optimization process (GAMA). Further comparison is contained in GAMA's documentation.

Acknowledgements

This software was developed with support from the Data Driven Discovery of Models (D3M) program run by DARPA and the Air Force Research Laboratory.

References

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., & Hutter, F. (2015). Efficient and robust automated machine learning. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems 28* (pp. 2962–2970). Curran Associates, Inc. Retrieved from http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf

Mohr, F., Wever, M., & Hüllermeier, E. (2018). ML-plan: Automated machine learning via hierarchical planning. *Machine Learning*, *107*(8-10), 1495–1515. doi:10.1007/s10994-018-5735-z

Olson, R. S., Urbanowicz, R. J., Andrews, P. C., Lavender, N. A., Kidd, L. C., & Moore, J. H. (2016). Applications of evolutionary computation: 19th european conference, evoapplications 2016, porto, portugal, march 30 – april 1, 2016, proceedings, part i. In G. Squillero & P. Burelli (Eds.), (pp. 123–137). Springer International Publishing. doi:10.1007/978-3-319-31204-0_9

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Thornton, C., Hutter, F., Hoos, H. H., & Leyton-Brown, K. (2013). Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In *Proc. of kdd-2013* (pp. 847–855).