AutoClassWeb: a simple web interface for Bayesian clustering of omics data

Pierre Poulain, Jean-Michel Camadro

BMC Research Notes

RESEARCH NOTE

Open Access

AutoClassWeb: a simple web interface for Bayesian clustering of omics data

Pierre Poulain* and Jean-Michel Camadro

Abstract

Objective: Data clustering is a common exploration step in the omics era, notably in genomics and proteomics where many genes or proteins can be quantified from one or more experiments. Bayesian clustering is a powerful unsupervised algorithm that can classify several thousands of genes or proteins. AutoClass C, its original implementation, handles missing data, automatically determines the best number of clusters but is not user-friendly.

Results: We developed an online tool called AutoClassWeb, which provides an easy-to-use and simple web interface for Bayesian clustering with AutoClass. Input data are entered as TSV files and quality controlled. Results are provided in formats that ease further analyses with spreadsheet programs or with programming languages, such as Python or R. AutoClassWeb is implemented in Python and is published under the 3-Clauses BSD license. The source code is available at https://github.com/pierrepo/autoclassweb along with a detailed documentation.

Keywords: Clustering, Genomics, Proteomics, Bayesian, Autoclass, Machine learning

Introduction

In biology, high-throughput technologies (notably in genomics and proteomics) enable identification and quantification of several thousands of genes or proteins in a single experiment. To analyze such a large amount of data, from one or more experiments, clustering algorithms are widely used unsupervised machine-learning methods to group genes or proteins with similar patterns. Bayesian clustering is such an algorithm and one of its implementation in the C programming language (AutoClass C) has been developed in 1996 at the Ames Research Center at NASA [1, 2]. The idea behind Bayesian clustering and the AutoClass algorithm is to find a classification that fits the data with the highest probability. The AutoClass algorithm provides some additional and interesting features: it handles missing data and determines automatically the best number of clusters.

*Correspondence: pierre.poulain@u-paris.fr

Université Paris Cité, CNRS, Institut Jacques Monod, Paris F-75013, France

AutoClass C has been used in a wide variety of applications from clustering cells of the prefrontal cortex in rats and mice [3] to detecting body patterns in the common cuttlefish [4] (see also references [5] and [6] for a detailed list of applications). However, AutoClass C, originally developed by physicists, is not user-friendly: the program is solely accessible through the command line, only 32-bit binaries are available and results files are difficult to parse for subsequent analysis.

More than 10 years ago, Achcar *et al.* published AutoClass@IJM [5], a web interface for AutoClass C. This web service drastically simplified the use of AutoClass C and widen its adoption, especially in biology [3, 7–10]. Unfortunately, this tool is not maintained anymore, and its source code is not publicly available.

To continue to offer this powerful unsupervised Bayesian clustering method to the community, we first developed AutoClassWrapper [6], a Python library that wraps the functionality of AutoClass C. We now go further by providing AutoClassWeb, a new easy-to-use open-source web interface for AutoClass C. AutoClassWeb requires no programming skills, performs a quality control on the

input data, and ships results in formats that support further analysis.

Main text

Implementation

AutoClassWeb utilizes AutoClassWrapper [6], a Python wrapper for the AutoClass C program. This wrapper facilitates the preparation and quality control of data, runs the actual classification, and eventually, prepare results in file formats that allow further analysis.

AutoClassWeb is written in Python [11] and uses the Flask library to build the web interface users interact with. For better reproducibility and sustainability, AutoClassWeb is packaged in a Docker image stored in the BioContainers [12] registry.

The web service itself has been designed to be user-friendly and straightforward. There is no user authentication and by default, results are kept 30 days before being deleted. A comprehensive *help* page provides all the help and guidance the user might need.

Using Docker technology, AutoClassWeb can be quickly deployed on a local machine or on a public web server. To this end and to reduce the installation burden, we provide two companion GitHub repositories with detailed instructions, for local (https://github.com/pierrepo/autoclassweb-app) and server installation (https://github.com/pierrepo/autoclassweb-server).

Data submission

The input data must be formatted as tab-separated values (TSV) files. The first line is a header containing the names of the columns which must be unique. The first column contains the names of the objects studied (*i.e.* protein or gene identifiers).

Missing data is supported and should be coded with an empty value (*i.e.* nothing)..

AutoClass supports three categories of data:

- *real location*: negative and positive values such as position, elevation, microarray log ratio...
- *real scalar*: singly bounded real values, typically bounded below at zero (*i.e.*: length, weight, age).
- *discrete*: qualitative data. For instance, color, phenotype, name...

If the initial input dataset contains several data types (*real scalar*, *real location*, *discrete*), it is recommended to split the initial dataset into several datasets of homogeneous type and submit them in the input form (Figure 1 (A)).

For *real location* and *real scalar* data types, the user can optionally specify an absolute and relative error, respectively. By default, the absolute error for *real location* data is 0.01 and the relative error for *real scalar* data is 0.01 (1%). However, it is worth noting that the AutoClass C algorithm is not very sensitive to the error parameter [5].

Clustering

Upon submission, input data is quality checked and formatted to be usable by AutoClass C. The web interface provides a unique job name, a link to the status page and a quick summary of input data (toggled with the text `Hide/show logs`), as illustrated in Figure 1 (B).

The status page lists running, failed and completed runs with their respective identifier (*Job name*), creation date, status and running time (Figure 1 (C)).

Results

Once a job is completed, a green button allows the user to download results of the clustering. Results are bundled in a zip archive with the following files (where `xxx` stands for the unique identifier of the job):

- `xxx_autoclass_out.cdt` and `xxx_autoclass_out_withproba.cdt` can be viewed with Java TreeView [13], a versatile viewer initially developed for microarray data. The file `xxx_autoclass_out_withproba.cdt` contains the probability for each object (gene or protein) to belong to each class.
- `xxx_autoclass_out_stats.tsv` contains means and standard deviations of numeric columns (*real scalar* and *real location* data types) for each class.
- `xxx_autoclass_out_dendrogram.png` is a dendrogram plot that visualizes the distance between all classes.
- `xxx_autoclass_out.tsv` contains all the data with the class assignment and membership probabilities for all classes. This file is in the TSV format and can be easily parsed with spreadsheet programs such as Microsoft Excel or LibreOffice Calc, or programming languages such as R or Python.

Performance

The AutoClass C algorithm has been designed to run on a single CPU. The running time depends exponentially on the size of the input dataset. Figure 2 illustrates the running time as a function of the input dataset sizes. Dataset size is expressed as the number of rows (usually genes or proteins) times the number of columns (features or properties of interest).

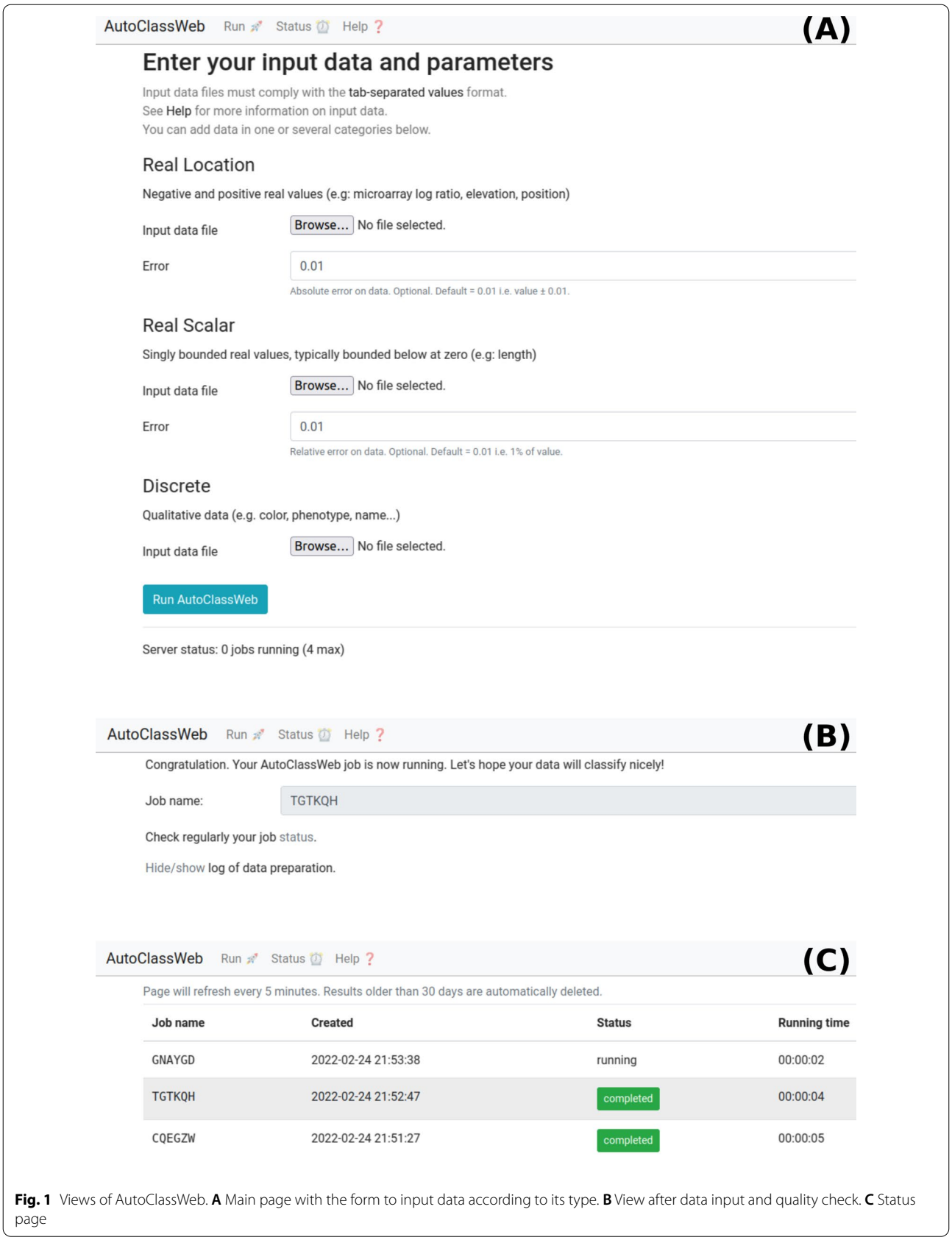

Fig. 1 Views of AutoClassWeb. **A** Main page with the form to input data according to its type. **B** View after data input and quality check. **C** Status page

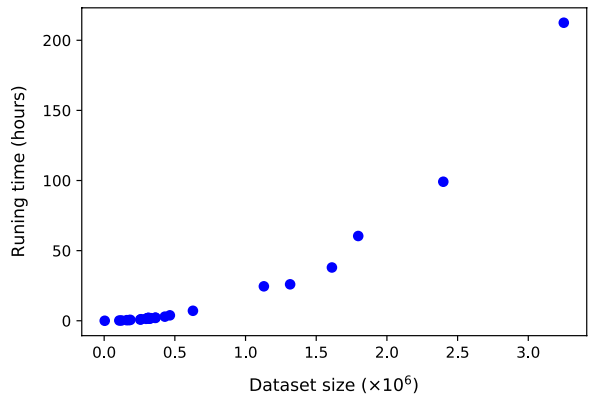

Fig. 2 Running time (in hours) as a function of the input dataset size. Dataset size is expressed as the number of rows (genes or proteins) times the number of columns (features or studied properties)

Conclusion

Data clustering is an essential unsupervised machine-learning approach used in most modern omics data analyses. The AutoClass algorithm, while very powerful, is not widely used, mainly because its original AutoClass C implementation is difficult to use. AutoClassWeb provides an easy-to-use and straightforward web interface for AutoClass C. The project is open-source, packaged in a Docker image available in BioContainers for better reproducibility and sustainability.

Limitations

AutoClassWeb provides a convenient online service to cluster results from high-throughput experiments such as RNA-seq or mass spectrometry based proteomics. However, we would like to point out that the processing time required to cluster data with AutoClass is proportional to the number of genes or proteins to be clustered. A parallel version of AutoClass C that potentially reduces the processing time has been published [14]. Unfortunately, the source code is not available, and the project has been discontinued.

Another limitation of AutoClassWeb requires users to split input data by type (*real location*, *real scalar* or *discrete*) with a special attention to *real location* and *real scalar* which may sometimes be confused.

Abbreviations

CPU: Central processing unit; TSV: Tab-separated values.

Acknowledgements

Authors thank Denis Mestivier for fruitful discussions on AutoClass@IJM.

Author contributions

PP and JMC contributed to the design of the web interface. PP developed the tool and wrote the manuscript. JMC extensively tested the ergonomy. PP and JMC reviewed the manuscript. All authors read and approved the final manuscript.

Funding

This work is funded by the French ministry of research.

Availability of data and materials

The source code of AutoClassWeb is open-source, released under the BSD-3-Clause license, and available on the GitHub development platform https://github.com/pierrepo/autoclassweb. AutoClassWeb source code in its current version 2.2.1 is archived in Software Heritage with the following reference: swh:1:dir:173e846a5137a4b498ca7da9eca980790631bc1a.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 13 April 2022 Accepted: 21 June 2022
Published online: 07 July 2022

References

1. Cheeseman P, Stutz J. Bayesian classification(autoclass): theory and results. In: Fayyad UM, Piatetsky-Shapiro G, Smyth P, Uthurusamy R, editors. Advances in knowledge discovery and data mining. Boston: AAAI/MIT Press; 1996. p. 153–80.
2. Stutz J, Cheeseman P. Autoclass - A Bayesian Approach to Classification. In: Skilling J, Sibisi S, editors., Maximum entropy and bayesian methods. No. 70 in the fundamental theories of physics their clarification, development and application. Dordrecht: Kluwer Academic Publishers; 1996.
3. Elliott MC, Tanaka PM, Schwark RW, Andrade R. Serotonin differentially regulates L5 pyramidal cell classes of the medial prefrontal cortex in rats and mice. ENeuro. 2018;5(1):e0305-17.
4. Crook AC, Baddeley R, Osorio D. Identifying the structure in cuttlefish visual signals. Philos Trans R Soc Lond B Biol Sci. 2002;357(1427):1617–24.
5. Achcar F, Camadro JM, Mestivier D. AutoClass@IJM: a powerful tool for bayesian classification of heterogeneous data in biology. Nucleic Acids Res. 2009;37(Web Server):W63–7.
6. Camadro JM, Poulain P. AutoClassWrapper: a python wrapper for autoclass C classification. J Open Source Softw. 2019;4(39):1390.
7. Wu S, Clevenger JP, Sun L, Visa S, Kamiya Y, Jikumaru Y, et al. The control of tomato fruit elongation orchestrated by sun, ovate and fs8.1 in a wild relative of tomato. Plant Sci. 2015;238:95–104.
8. Léger T, Garcia C, Ounissi M, Lelandais G, Camadro JM. The metacaspase (Mca1p) has a dual role in farnesol-induced apoptosis in *Candida albicans*. Mol Cell Proteomics. 2015;14(1):93–108.
9. Franco M, Vivo JM. Cluster analysis of microarray data. In: Bolón-Canedo V, Alonso-Betanzos A, editors. Microarray bioinformatics, vol. 1986. New York: Springer, New York; 2019. p. 153–83.
10. Duval C, Macabiou C, Garcia C, Lesuisse E, Camadro JM, Auchère F. The adaptive response to iron involves changes in energetic strategies in the pathogen *Candida albicans*. Microbiologyopen. 2020;9:2.
11. van Rossum G. Python tutorial. Amsterdam: Centrum voor Wiskunde en Informatica (CWI); 1995. (**CS-R9526**).
12. da Veiga Leprevost F, Grüning BA, Alves Aflitos S, Röst HL, Uszkoreit J, Barsnes H, et al. BioContainers: an open-source and community-driven framework for software standardization. Bioinformatics. 2017;33(16):2580–2.
13. Saldanha AJ. Java treeview-extensible visualization of microarray data. Bioinformatics. 2004;20(17):3246–8.
14. Pizzuti C, Talia D. P-autoclass: scalable parallel clustering for mining large data sets. IEEE Trans Knowl Data Eng. 2003;15(3):629–41.

Publisher's Note