Data and text mining

ClassificaIO: machine learning for classification graphical user interface

Raeuf Roushangar[1,2], George I. Mias[1,2,*]

[1]Department of Biochemistry and Molecular Biology,

[2]Institute for Quantitative Health Science and Engineering,

Michigan State University, East Lansing MI 48824

*To whom correspondence should be addressed.

Abstract

Summary: ClassificaIO is an open-source Python graphical user interface (GUI) for machine learning classification for the scikit-learn module. ClassificaIO aims to provide an easy-to-use interactive way to train, validate, and test data on a range of classification algorithms. The GUI enables fast comparisons within and across classifiers, and facilitates uploading and exporting of trained models, and both validated, and tested data results.

Availability: ClassificaIO is implemented as a Python application and is available for download and installation through the Python Package Index (PyPI) (http://pypi.python.org/pypi/ClassificaIO) and it can be deployed using the "import" function once installed. The application is distributed under an MIT license and source code is available for download (for Mac OS X, Unix and Microsoft Windows) through PyPI and GitHub (http://github.com/gmiaslab/ClassificaIO), and at https://doi.org/10.5281/zenodo.1133266.

Contact: gmias@msu.edu

Supplementary information: Supplementary data are available online.

1 Introduction

Recent advances in high-throughput technologies, especially in genomics, have led to an explosion of large-scale structured data (e.g. RNA-sequencing and microarray data) (Mias and Snyder, 2013). Machine learning methods (classification, regression, clustering, etc.) are routinely used in mining these data to extract biological insights in a range of areas within genetics and genomics (Libbrecht and Noble, 2015).

Many advanced machine learning algorithms have been developed in the recent years, scikit-learn (Pedregosa, et al., 2011) is one of the most popular machine learning libraries in Python with a plethora of thoroughly tested and well-maintained machine learning algorithms. However, these algorithms are primarily aimed at users with computing background, which may pose limitations for many biomedical scientists wanting to apply machine learning to their research. Several open source machine learning tools, such as KNIME and Weka (Berthold, et al., 2008; Frank, et al., 2004) written in Java and Orange (Demsar, et al., 2013) written in Python, have been developed as graphical user interfaces (GUIs). However, and to the best of our knowledge, no standalone GUIs exist for the scikit-learn library.

Here, we present ClassificaIO, an open-source Python GUI for supervised machine learning classification for the scikit-learn library. ClassificaIO aims to provide easy-to-use access to a range of state-of-the-art classification algorithms to researchers with minimal machine learning background, allowing them to use machine learning and apply it to their research. What distinguishes ClassificaIO from other similar applications is:

- Interactive point-and-click GUI to 25 supervised classification algorithms in scikit-learn
- Fast comparisons within and across classifiers to train, validate, and test data
- Upload and export of ClassificaIO trained models (for future of a trained model without the need to retrain), and export of both validated, and tested data results

2 Implementation and System Requirements

ClassificaIO has been developed using the standard Python interface Tkinter module to the Tk GUI toolkit (Ousterhout, 1994). The GUI can be easily deployed using the "import" function. It uses external packages: Pillow, Pandas (McKinney, 2010), NumPy (Oliphant, 2006), and scikit-learn. ClassificaIO requires Python version 3.5 or higher, and we recommend using the Spyder integrated development environment (IDE) in Anaconda Navigator (https://www.anaconda.com/download/) on Mac OS High Sierra (10.13) and Microsoft Windows 10 or higher. A user manual is also distributed with ClassificaIO and can be accessed directly through the 'Help' button at the upper left of the main Window that points to: (https://github.com/gmiaslab/manuals/blob/master/ClassificaIO/ClassificaIO_UserManual.pdf).

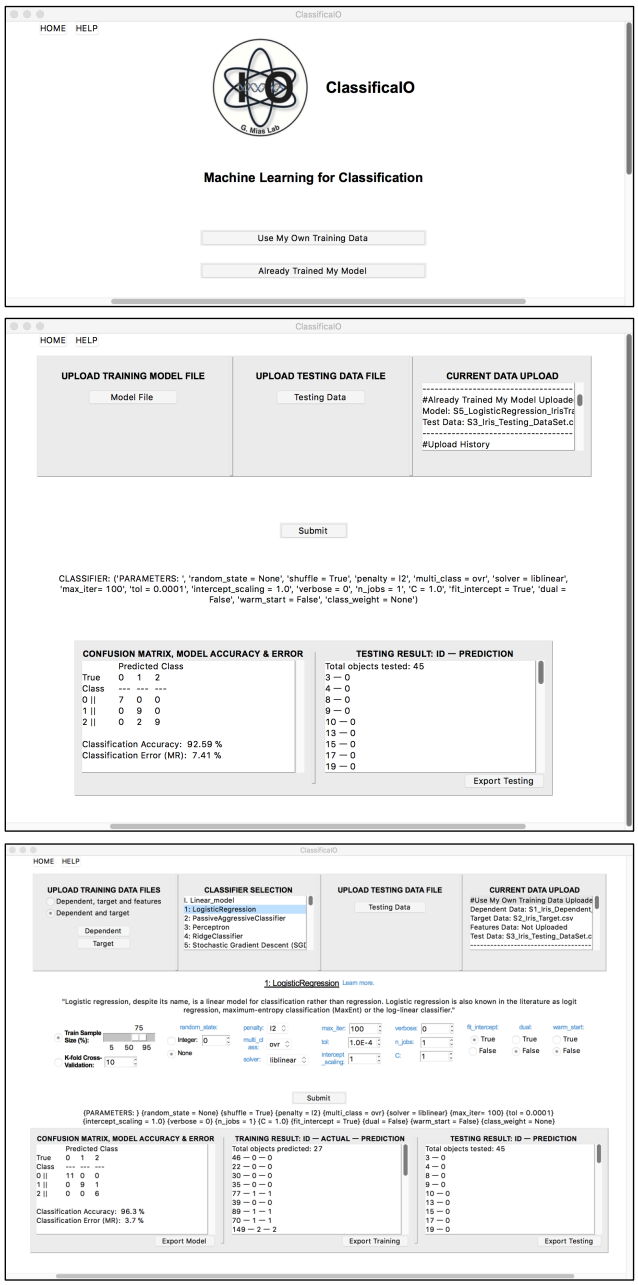

Figure. 1. ClassificaIO graphical user interface (Mac OS shown). As described in the summary implementation, **(a)** *Main* window appears on the screen after typing 'ClassificaIO.gui()' in a Spyder IDE or other python interpreter directly. **(b)** an example *Use My Own Training Data* window with uploaded training and testing data files, selected logistic regression classifier, populated classifier parameters, and output results for classification. **(c)** a corresponding *Already Trained My Model* window with uploaded ClassificaIO trained model and testing data file, and output result for classification.

3 Description

ClassificaIO consists of three windows: (a) *Main, (b) Use My Own Training Data*, and (c) *Already Trained My Model*. Each window is actually implemented within the code as a class with several functions/methods that are dynamically connected to provide the graphical user interface, **Figure 1**.

ClassificaIO's *Main* window, **Figure 1(a)** has two buttons: (i) the *Use My Own Training Data* button, which when clicked allows the user to train and test classifiers using their own training and testing data, (ii) the *Already Trained My Model* button, which when clicked allows the user to use their own already ClassificaIO trained model and testing data.

Data Input. For the *Use My Own Training Data* window, **Figure 1b**, by clicking the corresponding buttons in the 'UPLOAD TRAINING DATA FILES' and 'UPLOAD TESTING DATA FILE' panels, a file selector directs the user to upload all required comma-separated values (CSV) data files ('Dependent and Target' or 'Dependent, Target and Features' and 'Testing Data'). For more details on data files format, see Supplementary data and user manual . A history of all uploaded data files (file name and directory) is automatically saved in the 'CURRENT DATA UPLOAD' panel.

Classifier Selection. After the data is uploaded, the user can select between 25 different well-known classification algorithms (including logistic regression, perceptron, support vector machines, k-nearest neighbors, decision tree, random forest, neural network multi-layer perceptron, and more). The algorithms are integrated from the scikit-learn library, and allow the user to train and test models using their own uploaded data. Each classifier can be easily selected by clicking the corresponding classifier name in the 'CLASSIFER SELECTION' panel. Once classifier selection is completed, the classifier parameters will populate, and provide the user with an interactive point-and-click interface to set, modify, and test the influence of each parameter on their data. The user can switch classifiers and parameters through point-and-click, which enables fast comparisons within and across classifier models.

Model Training. Both train-validate split and cross-validation methods (which are necessary to prevent/minimize overfitting) will populate with each classifier that can be used for training. Training, validating and testing are all performed after pressing the submit button, **Figure 1(b)**.

Results Output. After training is completed, the confusion matrix, classifier accuracy and error are displayed in the 'CONFUSION MATRIX, MODEL ACCURACY & ERROR' panel, bottom of **Figure 1b**. Model validation data results are displayed in the 'TRAINING RESULT: ID – ACTUAL – PREDICTION' panel, and testing data results are displayed in the 'TESTING RESULT: ID – PREDICTION' panel.

Model Export. By clicking on the 'Export Model' button (**Figure 1b** bottom left), the user can export trained models to preserve for future use without having to retrain. A previously exported ClassificaIO model can then be used for testing of new data in the *Already Trained My Model* window, **Figure 1c**, by clicking the 'Model file' button in the 'UPLOAD TRAINING MODEL FILE'.

Results Export. Full results (trained models, and both validated and tested data) for both windows, **Figure 1b-c**, can be exported as CSV files for further analysis for publication, sharing, or later use (for more details on the export data file formats, see Supplementary data).

4 Summary and Future Direction

We have presented ClassificaIO as a GUI that implements scikit-learn supervised machine learning classification algorithms. The scikit package is one of the most popular in Python, and many of its machine leaning algorithms are currently used for analyzing large and complex data sets in genomics. Our interface provides an easy-to-use entry point to machine learning analysis, without the requirement of advanced computational and machine learning knowledge. ClassificaIO is provided as an open source software, and its back-end classes and functions allow for rapid development. We anticipate further development, aided by the scikit-learn library developer community to integrate additional classifi-

cation algorithms, and extend ClassificaIO to include other machine leaning methods such as regression, clustering, and anomaly detection, to name but a few.

Funding

RR has been supported by The Paul and Daisy Soros Fellowship for New Americans. GIM and research reported in this publication have been supported by a Jean P. Schultz Endowed Biomedical Research Fund award, and by NIH/NHGRI Grant HG0006785.

Conflict of Interest: none declared.

References

Berthold, M.R., *et al.* KNIME: The Konstanz Information Miner. *Stud Class Data Anal* 2008:319-326.

Demsar, J., *et al.* Orange: Data Mining Toolbox in Python. *J Mach Learn Res* 2013;14:2349-2353.

Frank, E., *et al.* Data mining in bioinformatics using Weka. *Bioinformatics* 2004;20(15):2479-2481.

Libbrecht, M.W. and Noble, W.S. Machine learning applications in genetics and genomics. *Nat Rev Genet* 2015;16(6):321-332.

McKinney, W. Data structures for statistical computing in python. In, *Proceedings of the 9th Python in Science Conference*. 2010. p. 51-56.

Mias, G.I. and Snyder, M. Personal genomes, quantitative dynamic omics and personalized medicine. *Quant Biol* 2013;1(1):71-90.

Oliphant, T.E. A guide to NumPy. Trelgol Publishing USA; 2006.

Ousterhout, J.K. Tcl and the Tk toolkit. Reading, Mass.: Addison-Wesley; 1994.

Pedregosa, F., *et al.* Scikit-learn: Machine Learning in Python. *J Mach Learn Res* 2011;12:2825-2830.