Title

Brain Predictability toolbox: a Python library for neuroimaging-based machine learning.

Permalink

https://escholarship.org/uc/item/5dj042r3

Journal

Bioinformatics, 37(11)

ISSN

1367-4803

Authors

Hahn, Sage
Yuan, De Kang
Thompson, Wesley K
et al.

Publication Date

2021-07-12

DOI

10.1093/bioinformatics/btaa974

Peer reviewed

OXFORD

Bioimage informatics

Brain Predictability toolbox: a Python library for neuroimaging-based machine learning

Sage Hahn [1,*], De Kang Yuan[1], Wesley K. Thompson[2], Max Owens[1], Nicholas Allgaier[1] and Hugh Garavan[1]

[1]Department of Psychiatry and Complex Systems, University of Vermont, Burlington, VT 05401, USA and [2]Division of Biostatistics, Department of Family Medicine and Public Health, University of California, San Diego, La Jolla, CA 92093, USA

*To whom correspondence should be addressed.

Associate Editor: Lu Zhiyong

Abstract

Summary: Brain Predictability toolbox (BPt) represents a unified framework of machine learning (ML) tools designed to work with both tabulated data (e.g. brain derived, psychiatric, behavioral and physiological variables) and neuroimaging specific data (e.g. brain volumes and surfaces). This package is suitable for investigating a wide range of different neuroimaging-based ML questions, in particular, those queried from large human datasets.

Availability and implementation: BPt has been developed as an open-source Python 3.6+ package hosted at https://github.com/sahahn/BPt under MIT License, with documentation provided at https://bpt.readthedocs.io/en/latest/, and continues to be actively developed. The project can be downloaded through the github link provided. A web GUI interface based on the same code is currently under development and can be set up through docker with instructions at https://github.com/sahahn/BPt_app.

Contact: sahahn@uvm.edu

1 Introduction

Large datasets in all domains are becoming increasingly prevalent as data from smaller existing studies are pooled and larger studies are funded. This increase in available data offers an unprecedented opportunity for researchers interested in applying machine learning (ML) based methodologies, especially those working in domains such as neuroimaging where data collection is quite expensive. This article considers neuroimaging-based ML (analyses of brain data) as an example domain in which the toolbox can be applied.

While there are a number of existing libraries for performing general ML based workflows within Python and other languages, the Brain Predictability toolbox (BPt) offers a high level user interface with specific consideration made toward neuroimaging based ML. BPt is designed to supplement the experience currently offered by similar popular libraries such as scikit-learn (Pedregosa et al., 2011) and nilearn (Abraham et al., 2014), rather than replace. BPt leverages existing ML libraries along with new functionality in order to provide a resource suitable for guiding users through the full research ML workflow; from loading data to interpreting results.

2 Description

2.1 Usability

BPt offers both a python based api and a web interface application, each with overlapping utility and distinct strengths and weaknesses.

In this way, BPt seeks to balance 'user friendliness' and expressiveness, with the goal of creating a framework suitable for both beginners, and one with enough flexibility to be used by advanced ML practitioners. That said, this library is not explicitly designed as a tutorial for new users. Some baseline knowledge of machine learning is required as well as some background Python knowledge, though the web interface version of the project (BPt_app) seeks to eliminate the latter prerequisite. A comprehensive documentation is provided along with several detailed examples for the Python api. Examples, found at https://github.com/sahahn/BPt/tree/master/Examples, are provided as jupyter notebooks and explore a range of problem types on real world data.

2.2 Best practices

The underlying structure of the library guides users to follow best practices in regard to cross validation, namely; perform a global train-test split, using the training set for model pipeline exploration and ultimately evaluating on the testing set. Performance from each step is easily reported over multiple user-defined metrics. The general structure of both the library and web application further guides users through a recommended workflow.

2.3 Data loading

BPt allows a user to easily load, manipulate and interactively view input neuroimaging datasets. Loading functions are equipped to

help perform outlier detection, handling of missing data, loading of specific variables and detection of duplicate variables among a number of other utilities. Data visualization tools are implemented in order to facilitate active data exploration.

2.4 ML pipelines
Diverse and complex ML pipelines can easily be created with a number of predefined choices across a range of state-of-the-art ML techniques. BPt strives to include as broad and as recent a selection of different ML algorithms as possible, as well as to directly integrate these choices with custom and preset hyperparameter distributions. Users can further express the choice between one or more algorithms or pipeline steps as hyperparams, allowing for the easy inclusion of model selection as properly nested within cross-validation.

2.5 Problem type support
All common ML problem types are supported (regression, binary and categorical), with low level implementation issues abstracted away, and new wrapper functions written to provide extended problem type support.

2.6 Covariates and feature importance
Properly handling covariates within neuroimaging-based machine learning is rarely straightforward. BPt supports a range of techniques for estimating the influence of covariates, including: feature importance, leave-out group CV (e.g. leave-out site for multi-site neuroimaging data), experiments on one group (e.g. sex-specific classifier), post-stratifying raw predictions (e.g. by race) and others. Feature importance in particular is supported by extracting base measures (e.g. beta weights from linear models), in addition to calculating SHapley Additive exPlanations (Lundberg and Lee, 2017) and permutation-derived feature importances (Altmann et al., 2010).

2.7 Reproducibility
By conducting loading, preprocessing and modeling within the same script, analyses can be easily reproduced and shared. Automatic logs are generated within the python workflow and similarly within the web app projects can be easily created and saved. These tools allow previous analyses to be easily retrievable.

2.8 Convenience
Most researchers working on neuroimaging-based ML applications, or other applied academic ML, have little background in software engineering, which means that writing code for loading data and building ML models can often take longer than expected or introduce unexpected bugs. Instead, by leveraging BPt, researchers can quickly move from ideas to experimentation and, importantly, results.

2.9 Backend libraries
BPt makes use of a few other libraries within the scientific Python community, which without their contribution this project would not be possible, most notably: Numpy (Oliphant, 2006), pandas (McKinney, 2010) and scikit-learn (Pedregosa et al., 2011). Plotting functionality makes use of the matplotlib library (Hunter, 2007). Extra classifiers and pipeline objects beyond those included with scikit-learn are used from python libraries: lightgbm (Ke et al., 2017), xgboost (Chen and Guestrin, 2016), imbalanced-learn (Lemaître et al., 2017) and DESlib (Cruz et al., 2018). Hyperparameter optimizers are implemented through FaceBook's Nevergrad library, and additional feature importance support is added with the Shap library (Lundberg and Lee, 2017).

Data Availability

All relevant data and code can be found at https://github.com/sahahn/BPt and https://github.com/sahahn/BPt_app.

Conflict of Interest: none declared.

References

Abraham,A. et al. (2014) Machine learning for neuroimaging with scikit-learn. *Front. Neuroinf.*, **8**, 14.

Altmann,A. et al. (2010) Permutation importance: a corrected feature importance measure. *Bioinformatics*, **26**, 1340–1347.

Chen,T. and Guestrin,C. (2016) Xgboost: a scalable tree boosting system.Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM. pp. 785–794.

Cruz,R.M., et al. (2020). DESlib: A Dynamic ensemble selection library in Python. *Journal of Machine Learning Research*, **21**, 1–5.

Hunter,J.D. (2007) Matplotlib: a 2D graphics environment. *Comput. Sci. Eng.*, **9**, 90.

Ke,G. et al. (2017) Lightgbm: a highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pp. 3146–3154.

Lemaître,G. et al. (2017) Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning. *J. Mach. Learn. Res.*, **18**, 559–563.

Lundberg,S.M. and Lee,S.I. (2017) A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 4765–4774.

McKinney,W. (2010) Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, **Vol. 445**, pp. 51–56.

Oliphant,T.E. (2006) *A Guide to NumPy*, **Vol. 1**. Trelgol Publishing, USA, p. 85.

Pedregosa,F. et al. (2011) Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.*, **12**, 2825–2830