Enhancement and Maintenance of the scikit-learn project

1 Goals

The goal of this proposal is to add key features to the open-source machine learning library scikit-learn and aid in maintaining the project.

Scikit-learn powers many applications of machine learning and data science, and it is used in many bio-medical applications [21] including neuroscience [37, 33, 14], pharmaceutics [22, 23, 30], psychiatry [29, 10, 26], cancer research [2, 32, 7], epigenetics [35, 19, 11], radiology [25, 34] and genetics [24, 15, 6]. Thanks to its ease of use, scikit-learn enables domain scientists without extensive machine learning knowledge to efficiently apply machine learning techniques to their core discipline; as a result, the original scikit-learn paper [27] has more than 18,000 citations, and the GitHub repository[1] is used by more than 62,000 repositories. It is also a core upstream dependency of more specialized packages like Starfish[2] for transcriptomics, Scanpy [36] for single-cell analysis, and CellProfiler [17] for biological image analysis which are directly built or funded by CZI. Other downstream packages in biomedical research include QIIME [5, 3, 4], Nilearn [1], MNE [12], Scikit-allel[3], MSMBuilder [13], and MOABB [16].

This proposal aims to fund the work of Andreas Müller (PI) for 4.5 months, and that of Nicolas Hug (co-PI) for 12 months. Müller and Hug are both associate research scientists at Columbia University. Müller has been a core-developer of scikit-learn for over 7 years, and Hug has been a core developer for about six months. We plan to address the following tasks, detailed in the next subsections:

- Maintenance of the library and removal of technological debt
- Improvements to the new fast gradient boosting models
- Faster parameter searches, avoiding redundant computations

1.1 Maintenance of the library

As one of the most widely used machine learning libraries, scikit-learn has a tremendously large code-base, of which some parts are many years old. New contributions are submitted every day, along with bug reports or feature requests. We propose to fund both Müller and Hug to ensure the continued improvement of the library, and to ensure the project evolves to meet the growing needs of the scientific community. Maintaining scikit-learn consists of:

a) Addressing bug reports, prioritizing bug fixes, and ensuring important bugs are fixed in a timely manner.

b) Reviewing new feature contributions and making sure they meet the high quality criteria of the project, in particular regarding code maintainability and user documentation.

[1] https://github.com/scikit-learn/scikit-learn
[2] https://github.com/spacetx/starfish
[3] https://github.com/cggh/scikit-allel

c) Supporting and connecting with the community: this includes maintaining a high-quality documentation, reaching out to users to advertise new features, and decide what directions to take according to community needs. It also involves organizing sprints to engage the community and increase diversity among the contributors.

d) Ensuring backward compatibility is preserved between versions, and that new features are consistent with the needs of the library. This is important for the health of the project and for the numerous downstream projects that rely on scikit-learn.

e) Reducing the current backlog. There are many issues and pull-requests that get stalled for various reasons, most commonly for the lack of time by core developers. Reviewing these pull requests and closing irrelevant ones helps contributors and developers to focus on important matters.

These tasks require experience and a consistent involvement in the project to be carried out efficiently, making the team of Müller and Hug particularly well suited for this task.

1.2 Improvements to Gradient Boosting

Gradient Boosting Decision Trees (GBDTs) are a family of machine learning models used for classification and regression tasks. GBDTs are extremely efficient and often out-perform other models. They are used pervasively in machine learning, including biomedical applications [9, 31].

Recent implementations like LightGBM [20] and XGBoost [8] offer state-of-the-art results, both in terms of prediction accuracy and computation time. A new implementation of GBDTs (by co-PI Hug) was recently released in scikit-learn version 0.21. This new implementation outperforms XGBoost in terms of speed (while matching it's accuracy), and its results (in terms of speed and accuracy) are on a par with LightGBM. Directly including an implementation in scikit-learn ensures compatibility with the core package, and prevents inconsistencies in the API. Moreover, due to scikit-learn's popularity, distributing an implementation with scikit-learn will increase the visibility and adoption of these highly effective algorithms by non-specialists.

While the current implementation is operational, several important features are not implemented yet. Therefore we propose the following enhancements to this new GBDT implementation:

a) Implement native support for missing values. In real-world datasets, feature values might be missing due to measurement errors or incomplete measurements. The current implementation requires practitioners to resort to imputation, which can lead to suboptimal results and is not theoretically grounded [18]. Unlike most machine learning models, GBDTs are able to natively support missing data, without resorting to imputation of missing values. This makes them extremely useful for working with messy data, in particular for non-specialists. Both LightGBM and XGBoost natively support missing values.

b) Implement native support for categorical features. Categorical features are pervasive in biomedical science. For example sex, age, race, or geographic location are potentially relevant for predicting the cancer risk of a patient [28], and could be encoded as categorical variables. The current implementation of GBDTs only supports continuous features, but GBDTs are also able to handle categorical data. For now, users are required to preprocess categorical features using techniques like one-hot-encoding, which is often tedious and can lead to worse results. GBDTs have a native, more efficient way of dealing with categorical variables. Implementing this would free practitioners from having to worry about encoding categorical variables. This feature is implemented in LightGBM, but not in XGBoost.

c) Implement support for sample weights and class weights, which allow the user to specify the confidence they have in their measurements and deal with imbalanced datasets.

d) Document and explain the usage of the new implementation, with an emphasis on missing value support and support for categorical variables. Create detailed examples of typical use-cases.

e) Maintain the code, and fix the potential bugs. The current implementation already has about 5000 lines of Python (and Cython) code. Despite great efforts to provide a thorough test suite, some bugs or numerical instability issues may still be present given the size and complexity of the code base.

f) Improved scalability on multiple cores. The new GBDT implementation is our first use of low-level parallelism (using OpenMP). This enables us to build models much faster, but requires careful engineering and benchmarking to take full advantage of modern multi-core systems.

Please also note that achieving a) and b) would make these models the first ones in scikit-learn to natively support missing data and categorical data. This will demand some careful design steps, and will increase the number of potential bugs and pitfalls.

1.3 Faster parameter searches avoiding redundant computations

Supervised machine learning workflows typically consist of one or multiple preprocessing steps, and a final supervised model. The preprocessing steps, as well as the final model, commonly have hyper-parameters that need to be tuned for the overall procedure to perform well on a specific task: this is called a hyper-parameter tuning. Scikit-learn provides tools for hyper-parameter tuning: namely grid search (exhaustively try parameter combinations along a grid) and random search (parameter combinations are sampled at random).

Performing parameter searches in machine learning workflows often involves redundant computation, and previously computed results can potentially be re-used. As illustrated on Figure 1, when parameters of the final model change, it is unnecessary to recompute the previous (preprocessing) steps. Currently scikit-learn unnecessarily performs redundant computations, wasting computation time.

Leveraging features of the dask library for distributed computing, in particular dasks graph execution engine, the dask-ml package provides equivalent implementations of parameter search logic that are able to re-use previously computed steps. This implementation can be much faster than the implementation available in scikit-learn. However, this package is not as popular as the scikit-learn library, and as a result it is much less likely to be widely used, despite its usefulness. It also relies on dask, adding a dependency and increasing maintenance burden. Having such a utility in scikit-learn would allow non-specialists to train complex pipelines much more efficiently.

We propose to implement parameter search utilities without re-computation within scikit-learn, taking advantage of experience from the dask-ml implementation. This is a complex task that may require intrusive changes to scikit-learn core utilities.

1.4 Expected outcomes, success evaluation and metrics

For open source projects, measuring impact is notoriously hard, as library authors don't have access to telemetry. Download counts are often spread over several distribution channels and highly skewed by automated downloads. The impact of specific additions to open source projects is even harder to measure, in particular since adoption of new features can only be measured with significant

delay. Acknowledging this, we are suggesting proxy metrics that allow at least some quantitative evaluation of the work in the Milestones and Deliverables document. Our high-level goals are outlined below.

1.4.1 Maintenance of the Library

We aim to slow the growth of open issues an pull-requests, ideally even decreasing their number, and shorten the time taken to resolve an issue. We also aim to broaden the community of developers, and in particular the diversity of core developers.

1.4.2 New features for GBDT models

For the GBDT models, our primary goal is merging the proposed features (in particular support for missing values and categorical data) into scikit-learn, and inclusion into a release of the package. We also expect our additions to increase usage of the HistGradientBoosting models, as can be measured by code search on Github.

1.4.3 Faster parameter searches

Given the complexity of the task, an effective implementation of parameter search that reduces redundant computation would require careful design and review by the core team. There are several different approaches to solving this problem, each of which has its own technical constraints. As a result, clearly defining and assessing the pros and cons of each approach would already be an accomplishment and would serve as groundwork for future directions. Our goal here is to reach consensus on an implementation path.

2 Work plan

The funds will be used to:

- Fund Andreas Müller's position for 4.5 months. His long-term experience with the project is essential for evaluating changes in API design and dependencies that are necessary for the planned work.
- Fund Nicolas Hug's position for 12 months. Hug has been involved in maintaining scikit-learn for over a year, and has been a core developer for 6 months. He has contributed key features to the package, in particular the new GBDT implementation mentioned above.
- Fund participations in key Python and scientific conferences, such as the SciPy conference [4].
- Fund participation and organization of sprints, in particular the WiMLDS sprints[5]. See our *Diversity and Inclusion Statement* for details on these events.

We believe that mentoring other contributors is very important for the health of the project: the more contributors are familiar with the code base, the healthier the project. Therefore, a significant part of the effort in implementing the proposed features will go towards reviewing and mentoring other contributors, instead of implementing all of the features ourselves.

All of the proposed features in this proposal are part of the current project roadmap[6].

[4] https://conference.scipy.org
[5] http://wimlds.org/opensourcesprints-2/
[6] https://scikit-learn.org/stable/roadmap.html

3 Existing support

Andreas Müller is a PI on the following grants related to scikit-learn:

- *Building blocks and Search Improvements for Automated Machine Learning Model Selection.* DARPA. $351k. 2018.
- SI2-SSE: *Improving Scikit-learn usability and automation.* NSF. $400k. 2017-2020.
- *Extension and Maintenance of Scikit-learn.* Alfred P. Sloan Foundation. $313k. 2017-2019.

The scikit-learn foundation at INRIA Paris [7] currently funds four full time employees working on scikit-learn and related projects:

- Olivier Grisel, technical lead.
- Jérémie du Boisberranger, research engineer.
- Guillaume Lemaître, research engineer.
- Chiara Marmo, community and operation manager (starting September 2019).

The University of Sydney funds Joel Nothman to work part-time on scikit-learn. Anaconda[8] funds Adrin Jalali as a full time employee to work on scikit-learn and related projects.

[7] https://scikit-learn.fondation-inria.fr/
[8] https://www.anaconda.com/

Figures

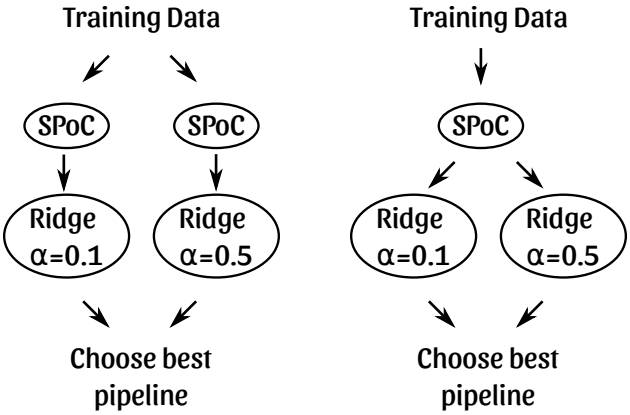

Figure 1: Parameter search for a model predicting electromyography signal from magnetoencephalography activity. Inspired by the MNE package[12].
Left: Current parameter search in scikit-learn. The pre-processing SPoC step is repeated for each of the two pipelines, even though it never changes.
Right: Proposed implementation. The SPoC step is only computed once.

References

[1] Alexandre Abraham et al. "Machine learning for neuroimaging with scikit-learn". In: *Frontiers in Neuroinformatics* 8 (2014), p. 14. ISSN: 1662-5196. DOI: 10.3389/fninf.2014.00014. URL: https://www.frontiersin.org/article/10.3389/fninf.2014.00014.

[2] Spyridon Bakas et al. "Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features". In: *Scientific data* 4 (2017), p. 170117.

[3] Nicholas A. Bokulich et al. "Optimizing taxonomic classification of marker-gene amplicon sequences with QIIME 2's q2-feature-classifier plugin". In: *Microbiome* 6.1 (May 2018), p. 90. ISSN: 2049-2618. DOI: 10.1186/s40168-018-0470-z. URL: https://doi.org/10.1186/s40168-018-0470-z.

[4] Nicholas A. Bokulich et al. "q2-sample-classifier: machine-learning tools for microbiome classification and regression". In: *bioRxiv* (2018). DOI: 10.1101/306167. eprint: https://www.biorxiv.org/content/early/2018/11/28/306167.full.pdf. URL: https://www.biorxiv.org/content/early/2018/11/28/306167.

[5] Evan Bolyen et al. "QIIME 2: Reproducible, interactive, scalable, and extensible microbiome data science". In: *PeerJ Preprints* 6 (Dec. 2018), e27295v2. ISSN: 2167-9843. DOI: 10.7287/peerj.preprints.27295v2. URL: https://doi.org/10.7287/peerj.preprints.27295v2.

[6] Kristopher W Brannan et al. "SONAR discovers RNA-binding proteins from analysis of large-scale protein-protein interactomes". In: *Molecular cell* 64.2 (2016), pp. 282–293.

[7] Kumardeep Chaudhary, Olivier B Poirion, Liangqun Lu, and Lana X Garmire. "Deep learning–based multi-omics integration robustly predicts survival in liver cancer". In: *Clinical Cancer Research* 24.6 (2018), pp. 1248–1259.

[8] Tianqi Chen and Carlos Guestrin. "Xgboost: A scalable tree boosting system". In: *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM. 2016, pp. 785–794.

[9] Xiang Chen, Zhi-Xin Wang, and Xian-Ming Pan. "HIV-1 tropism prediction by the XGboost and HMM methods". In: *Scientific Reports* 9.1 (2019), p. 9997.

[10] Oliver Doehrmann et al. "Predicting Treatment Response in Social Anxiety Disorder From Functional Magnetic Resonance ImagingTreatment Prediction in Social Anxiety Disorder". In: *JAMA Psychiatry* 70.1 (Jan. 2013), pp. 87–97. ISSN: 2168-622X. DOI: 10.1001/2013.jamapsychiatry.5. eprint: https://jamanetwork.com/journals/jamapsychiatry/articlepdf/1356542/yoa120057_87_97.pdf. URL: https://doi.org/10.1001/2013.jamapsychiatry.5.

[11] Meeshanthini V Dogan, Isabella M Grumbach, Jacob J Michaelson, and Robert A Philibert. "Integrated genetic and epigenetic prediction of coronary heart disease in the Framingham Heart Study". In: *PloS one* 13.1 (2018), e0190549.

[12] Alexandre Gramfort et al. "MNE software for processing MEG and EEG data". In: *NeuroImage* 86 (2014), pp. 446–460. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2013.10.027. URL: http://www.sciencedirect.com/science/article/pii/S1053811913010501.

[13] Matthew P Harrigan et al. "MSMBuilder: statistical models for biomolecular dynamics". In: *Biophysical journal* 112.1 (2017), pp. 10–15.

[14] Gerrit Hilgen et al. "Unsupervised spike sorting for large scale, high density multielectrode arrays". In: *bioRxiv* (2016). DOI: 10.1101/048645. eprint: https://www.biorxiv.org/content/early/2016/12/05/048645.full.pdf. URL: https://www.biorxiv.org/content/early/2016/12/05/048645.

[15] Max A Horlbeck et al. "Compact and highly active next-generation libraries for CRISPR-mediated gene repression and activation". In: *Elife* 5 (2016), e19760.

[16] Vinay Jayaram and Alexandre Barachant. "MOABB: trustworthy algorithm benchmarking for BCIs". In: *Journal of neural engineering* 15.6 (2018), p. 066011.

[17] Thouis R. Jones et al. "CellProfiler Analyst: Data Exploration and Analysis Software for Complex Image-Based Screens". In: *BMC Bioinformatics* 9.1 (Nov. 2008), p. 482. ISSN: 1471-2105. DOI: 10.1186/1471-2105-9-482. URL: https://doi.org/10.1186/1471-2105-9-482.

[18] Julie Josse, Nicolas Prost, Erwan Scornet, and Gaël Varoquaux. "On the consistency of supervised learning with missing values". In: *arXiv preprint arXiv:1902.06931* (2019).

[19] Boyko Kakaradov et al. "Early transcriptional and epigenetic regulation of CD8+ T cell differentiation revealed by single-cell RNA sequencing". In: *Nature immunology* 18.4 (2017), p. 422.

[20] Guolin Ke et al. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in Neural Information Processing Systems*. 2017, pp. 3146–3154.

[21] Konrad Paul Kording, A Benjamin, Roozbeh Farhoodi, and Joshua I Glaser. "The roles of machine learning in biomedical science". In: *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2017 Symposium*. National Academies Press. 2018.

[22] Martin Lindh, Anders Karlén, and Ulf Norinder. "Predicting the rate of skin penetration using an aggregated conformal prediction framework". In: *Molecular pharmaceutics* 14.5 (2017), pp. 1571–1576.

[23] Tal Lorberbaum et al. "Systems pharmacology augments drug safety surveillance". In: *Clinical Pharmacology & Therapeutics* 97.2 (2015), pp. 151–158.

[24] Gioele La Manno et al. "Molecular Diversity of Midbrain Development in Mouse, Human, and Stem Cells". In: *Cell* 167.2 (2016), 566–580.e19. ISSN: 0092-8674. DOI: https://doi.org/10.1016/j.cell.2016.09.027. URL: http://www.sciencedirect.com/science/article/pii/S0092867416313095.

[25] Aaron J Masino, Robert W Grundmeier, Jeffrey W Pennington, John A Germiller, and E Bryan Crenshaw. "Temporal bone radiology report classification using open source machine learning and natural langue processing libraries". In: *BMC medical informatics and decision making* 16.1 (2016), p. 65.

[26] Meenal J Patel et al. "Machine learning approaches for integrating clinical and imaging features in late-life depression classification and response prediction". In: *International journal of geriatric psychiatry* 30.10 (2015), pp. 1056–1067.

[27] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[28] Aaron N. Richter and Taghi M. Khoshgoftaar. "Efficient learning from big data for cancer risk modeling: A case study with melanoma". In: *Computers in Biology and Medicine* 110 (2019), pp. 29–39. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2019.04.039. URL: http://www.sciencedirect.com/science/article/pii/S0010482519301477.

[29] André Schmidt et al. “Approaching a network connectivity-driven classification of the psychosis continuum: a selective review and suggestions for future research”. In: *Frontiers in human neuroscience* 8 (2015), p. 1047.

[30] Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. “Generating focused molecule libraries for drug discovery with recurrent neural networks”. In: *ACS central science* 4.1 (2017), pp. 120–131.

[31] Akash A Shah et al. “Development of a machine learning algorithm for prediction of failure of nonoperative management in spinal epidural abscess”. In: *The Spine Journal* (2019).

[32] Alison M. Taylor et al. “Genomic and Functional Approaches to Understanding Cancer Aneuploidy”. In: *Cancer Cell* 33.4 (2018), 676–689.e3. ISSN: 1535-6108. DOI: https://doi.org/10.1016/j.ccell.2018.03.007. URL: http://www.sciencedirect.com/science/article/pii/S1535610818301119.

[33] Antoine M Valera et al. “Stereotyped spatial patterns of functional synaptic connectivity in the cerebellar cortex”. In: *Elife* 5 (2016), e09862.

[34] Ching-Wei Wang et al. “Evaluation and comparison of anatomical landmark detection methods for cephalometric x-ray images: a grand challenge”. In: *IEEE transactions on medical imaging* 34.9 (2015), pp. 1890–1900.

[35] Tina Wang et al. “Epigenetic aging signatures in mice livers are slowed by dwarfism, calorie restriction and rapamycin treatment”. In: *Genome biology* 18.1 (2017), p. 57.

[36] F Alexander Wolf, Philipp Angerer, and Fabian J Theis. “SCANPY: large-scale single-cell gene expression data analysis”. In: *Genome biology* 19.1 (2018), p. 15.

[37] Azar Zandifar et al. “A comparison of accurate automatic hippocampal segmentation methods”. In: *NeuroImage* 155 (2017), pp. 383–393.