GigaBlog

Reproducible Classification. Q&A on ShinyLearner & the CODECHECK certificate, pt. 2

Scott Edmunds

Published April 8, 2020

Citation
Edmunds, S. (n.d.). In *GigaBlog*. GigaBlog. https://doi.org/10.59350/zxcdd-ns360

Keywords
Technology, Code Ocean, Containers, Docker, Open Science
Feature Image

ShinyLearnerThis week we showcased a new way of peer reviewing software, testing code in an independent manner and providing a CODECHECK *"certificate of reproducible computation"* when the results in the paper can be reproduced. We've written a post on the CODECHECK process featuring a Q&A with CODECHECK founder Stephen Eglan, and here we'll provide a follow up with a Q&A with the author of the software that was subject to the CODECHECK certification process. Another Stephen, author Stephen Piccolo talks here about his new paper on ShinyLearner, a benchmarking tool for machine-learning classification algorithms.

ShinyLearner authorThe paper stands out by making this process very systematic and reproducible, and we ask Stephen about why classification algorithms are so important, why he uses reproducible and open research in his work, and how the CODECHECK review process was from an authors perspective. Stephen is an Assistant Professor in the College of Life Science at Brigham Young University, working in an open and reproducible way to integrate knowledge and techniques across biology, computer science, medicine, and statistics.**
**

Why are there so many classification algorithms and why do they need to be benchmarked?

Computer scientists and statisticians have been creating and refining classification algorithms for decades. Research papers have been published on hundreds of these algorithms, and implementations are available in the public domain for many of them. In many research areas (biological or otherwise), classification algorithms can help to identify patterns that distinguish two or more groups and then be used to predict the group to which new observations belong. However, it is difficult for scientists to know which classification algorithm is best for a particular research application. In addition, most algorithms support hyperparameters, which allow the scientist to modify the algorithm's behavior, but it is inefficient and bias-prone to tune these algorithms in an *ad hoc* manner. Benchmark analyses address this problem by making the algorithm/parameter comparisons systematic in nature. We can apply tens or even hundreds of algorithm variations to benchmark datasets and identify which tend to perform best.

Why did you build ShinyLearner and what technical approaches did you take to design it?

We created ShinyLearner to make it easier to perform benchmark comparisons. Several open-source software libraries are available for performing classification analyses. But most require the scientist to write computer code to perform analysis. Generally, these libraries perform parameter tuning, but they provide little insight into this process. We sought to provide a tool that requires no coding to perform the analysis and that generates simply formatted output files so users can more easily gain insight into benchmark results. It encapsulates 4 open-source, machine-learning libraries into a "software container" and provides a consistent interface for working with any of them. The use of software containers makes the installation process easier because it already includes all the software you need to execute the analysis. All you need is to install the Docker software (or a related tool for working with containers) and download the container image. The user then executes the software at the command line (terminal). To ease this process, we created a web application that helps the user construct the

commands they need to execute. One more tidbit: within the software container, we use a combination of 4 programming languages to piece everything together: Python, R, Java, and bash scripting. Most projects would limit themselves to 1-2 programming language, but we needed to write code in these various languages to interface with the different machine-learning libraries but also because in some cases, it seemed more efficient to use one language over another to implement the desired functionality. By using software containers, the end user doesn't need to worry about this complexity or install (specific versions of) runtimes for all of these languages.

This was an open source tool that used many techniques for computational reproducibility, so what drove you to follow open science practices?

Nowadays, most scientists will not take seriously a journal article describing a piece of research software unless it is open source. They want to be able to see the code and verify its design and functionality directly. This is a great thing! I am still baffled when I see a research article that describes software or an algorithm but doesn't make the code available.

In this paper, we also followed open-science practices by sharing our analysis scripts (in addition to the actual ShinyLearner software). We wanted others to be able to see the exact code we used to generate the figures in our paper. We used CodeOcean, a cloud-computing service that provides a free tier for open science projects [see previous blog on the platform]. We did this for the sake of transparency and because we knew that when we submitted the paper to a journal, at least one reviewer would want us to tweak our analysis. Because we had already packaged it up nicely and made it available for others, it was also easy for us to remember each step of the analysis and tweak what we needed to tweak. Finally, we used open-science practices because that's what we like to see in others.

How was the process of undergoing a CODECHECK review as an author?

For me, the CODECHECK process was easy. The reviewer set everything up. I just needed to verify that his review was reasonable and had been configured properly [see the certificate here].

From an author's perspective does the CODECHECK certificate seem a good incentive and reward for making your work easy to use? Do you think it may drive authors to improve the way they release and write up software papers?

CODECHECK certificateIt's nice to have that stamp of approval and peace of mind knowing that at least one outside person was able to reproduce our work. I wasn't aware of this certificate before submitting to this journal. So it wasn't a big motivation for me at the time. But going forward, I would be motivated by the certificate because it would tell me the journal is serious about following good scientific practices. Few other journals make the extra effort to really verify your code when it is open source. I would love to see this become standard practice. It's more work for everyone in the short term, but in the long run it's very beneficial. It won't always prevent research misconduct or ensure that a scientific analysis is impactful, but it's a simple way to encourage open science.

GigaBlog

If authors knew that they would need to pass a CODECHECK from the beginning, it would motivate them to follow practices early in the process that would support open, reproducible science. This would benefit themselves but also the broader community. I touch on this in more detail in an earlier *Gigascience* paper (Tools and techniques for computational reproducibility).

Read more in our previous post on CODECHECK. Stephen Eglen presented CODECHECK at The 14th Munin Conference on Scholarly Publishing 2019 and you can watch a video recording here.

References

Piccolo SR. et al., ShinyLearner: A containerized benchmarking tool for machine-learning classification of tabular data, *GigaScience*, Volume 9, Issue 4, April 2020, doi:10.1093/gigascience/giaa026

Eglen SJ. CODECHECK Certificate 2020-001. Zenodo. 2020 http://doi.org/10.5281/zenodo.3674056

Piccolo SR, Frampton MB. Tools and techniques for computational reproducibility. *Gigascience*. 2016;5(1):30. Published 2016 Jul 11. doi:10.1186/s13742-016-0135-4

The post Reproducible Classification. Q&A on ShinyLearner & the CODECHECK certificate, pt. 2 appeared first on GigaBlog.