Tests for machine learning

Tiffany A. Timbers

Table of contents

1 Context

Checklists have been shown to decrease errors in safety critical systems (Gawande 2010), and the use of a reproducibility checklist at the Machine Learning NeurIPS 2019 conference led to an increase in the percentage of authors submitting the code for their work [from 50% to 75%; Pineau et al. (2021)]. Thus, in an effort to make applied machine learning software more trustworthy by increasing its robustness, we aim to create a general and robust checklist for creating software tests for applied machine learning code. Such a checklist may be used by data scientists and machine learning engineers to guide the manual writing of tests. It may also act as a source for engineering large-language model (LLM) prompts that act as reliable starting points for engineering reproducible test data and software tests themselves for each item on the checklist.

Such a checklist should include tests for data presence, quality and ingestion at the beginning of the analysis, the model fitting and evaluation, as well as tests for the artifacts (presence and quality) which are created by the analysis. To write a comprehensive checklist for creating software tests for applied machine learning code we are researching industrial best practices (e.g., as documented in published guides, such as Microsoft Industry Solutions Engineering Team (2024), and blog posts of experienced machine learning software engineers, for example Jordan (2020)), as well as the published academic literature (Braiek and Khomh 2020; Openja et al. 2023; Silva and De França 2023). Openja et al. (2023) in particular is of significant interest as they comprehensibly studied the testing strategies of 10 open source machine learning projects; and identified 15 major categories of testing strategies used by machine learning engineers and 13 different types of tests. Furthermore, they quantified the frequency of these to identify which are most common across the projects studied.

The identification and categorization testing strategies and types of tests used by machine learning engineers in the wild (Openja et al. 2023) is very informative as to which strategies and test types should be used for future machine learning projects. Additionally, it would be wise to link each item on the comprehensive machine learning testing checklist we are developing to each test strategy and test type identified in Openja et al. (2023). One outstanding question from this work however is how representative are those 10 machine learning projects studied by Openja et al. (2023)?

It would be very informative and worthwhile to examine additional machine learning projects to assess whether their findings do extend to other projects. In particular projects that are more complete, such as published machine learning analyses, as well as projects where machine learning is being applied as the type of analysis to answer predictive questions in a given domain. There are two potential sources of data for this, the first being a subset of the papers identified in Wattanakriengkrai et al. (2022). In that study, they linked 377 GitHub repositories to academic papers, as well as coded each study as to what they were focused on (e.g., deep learning, computer vision, NLP, machine learning, sensors, etc). This data set is open and available here. These studies appear to primarily be methodology papers. A second source for studies is the xDD/GeoDeepDive API. This API aims to allow for the extraction of "dark data" from scientific works, which were previously only available from manually reading of the literature. This API could be used to identify papers where machine learning is being applied as the type of analysis to answer predictive questions in a given domain. It is very possible that both types of work (methodological machine learning studies and studies where machine learning is applied to answer a predictive question in a given domain) use and need different types of tests and testing strategies. This is currently unknown.

2 Annotated checklist of tests for machine learning

Below is a tentative checklist of items that should go on the comprehensive checklist for applied machine learning projects. It is by no means "comprehensive"

at this point and currently only includes ideas from Microsoft Industry Solutions Engineering Team (2024) and Jordan (2020).

2.0.1 Saving and loading data

- ☐ Loading data file function works as expected (Microsoft Industry Solutions Engineering Team 2024).
- ☐ Saving data/figures function works as expected

2.0.2 Data Validation

- ☐ Files contain data (Microsoft Industry Solutions Engineering Team 2024).
- ☐ Data/images in the expected format (Microsoft Industry Solutions Engineering Team 2024).
- ☐ Data does not contain null values or outliers (Microsoft Industry Solutions Engineering Team 2024).

2.0.3 Cleaning and transforming data

- ☐ Cleaning and transforming functions works as expected (Microsoft Industry Solutions Engineering Team 2024).

2.0.4 Modeling (pre-train tests)

- ☐ Does the model accept the correct inputs and produce the correctly shaped outputs (Microsoft Industry Solutions Engineering Team 2024)?
- ☐ Do the weights of the model update when running fit (Microsoft Industry Solutions Engineering Team 2024)?
- ☐ Does model output aligns with expectations (for example, in classification, are the labels what are expected based on input) (Jordan 2020)?
- ☐ Do the output ranges align with our expectations (eg. the output of a classification model should be a distribution with class probabilities that sum to 1) (Jordan 2020)?
- ☐ Does a single gradient step on a batch of data yield a decrease in your loss (Jordan 2020)?
- ☐ Is there leakage between your training, validation and test datasets (Jordan 2020)?

2.0.5 Other potential tests for modeling

We can also write tests that assess whether the machine learning model logic is correct, these tests are referred to as post-train tests, or behavioural tests Ribeiro et al. (2020). As well as tests to assess model performance, these tests are referred to as evaluation tests (Yan 2020). These evaluation tests may be particularly useful for determining data distribution shifts when models are in production.

3 Machine learning testing examples

There exist several examples/demos of how to test machine learning code. The data set and code used in these examples, may be an excellent starting place for the first test case to assess if the checklist can be used by LLM's to generate reproducible test data and high quality test cases.

- `testing-ml` by Eugene Yan and the accompanying article uses the Titanic data set
- `mercury-robust` by BBVA uses the Titanic, Tips and default credit card data sets
- `breast_cancer_predictor_py` by Timbers *et al.* uses the Madison Wisconsin Breast Cancer data set

4 Cookiecutter project templates

Having a function specifications for the functions needed for an applied machine learning project would be useful for pairing with the prompts for LLM test data and test suite generation. Below is a potential Cookiecutter project templates that might be of interest for modification and/or extension with the checklist items and prompts:

- `cookiecutter-data-science` by @drivendata (very popular and well structured project with scripts and pipeline, no function specifications however)

5 How to assess test quality of automated tests

There are several ways test quality can be assessed:

1. Human expert quality control (QC) - a human expert reviews the test cases and evaluates the test suite quality and case coverage. This assessment can be of great utility and high quality, but it is manual and thus can be quite slow. This can also be limited by the availability of the human expert.

2. Coverage (e.g., branch or line coverage) - the lines, or branches, of code executed by the test suite is calculated as a percentage of the code base. This assessment can be completely automated, and is therefore very efficient, it however does not assess the quality of the test cases.
3. Mutation testing - the code under test is intentionally changed/mutated to induce bugs, and the test suite is evaluated for its ability to detect the mutations. This can be a nice balance between human expert QC and coverage, as although human expertise is needed to design and create the mutated version of the code base, once created, assessing it can be automated. See Cheng et al. (2018) for an example where this was used to assess the presence of bugs in Weka machine learning code.

References

Braiek, Houssem Ben, and Foutse Khomh. 2020. "On Testing Machine Learning Programs." *Journal of Systems and Software* 164: 110542. https://doi.org/https://doi.org/10.1016/j.jss.2020.110542.

Cheng, Dawei, Chun Cao, Chang Xu, and Xiaoxing Ma. 2018. "Manifesting Bugs in Machine Learning Code: An Explorative Study with Mutation Testing." In *2018 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, 313–24. https://doi.org/10.1109/QRS.2018.00044.

Gawande, Atul. 2010. *Checklist Manifesto, the (HB)*. Penguin Books India.

Jordan, Jeremy. 2020. "Effective Testing for Machine Learning Systems." https://www.jeremyjordan.me/testing-ml/.

Microsoft Industry Solutions Engineering Team. 2024. *Engineering Fundamentals Playbook: Testing Data Science and MLOps Code Chapter*. Microsoft.

Openja, Moses, Foutse Khomh, Armstrong Foundjem, Zhen Ming, Mouna Abidi, Ahmed E Hassan, et al. 2023. "Studying the Practices of Testing Machine Learning Software in the Wild." *arXiv Preprint arXiv:2312.12604*.

Pineau, Joelle, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché-Buc, Emily Fox, and Hugo Larochelle. 2021. "Improving Reproducibility in Machine Learning Research (a Report from the Neurips 2019 Reproducibility Program)." *Journal of Machine Learning Research* 22 (164): 1–20.

Ribeiro, Marco Tulio, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. "Beyond Accuracy: Behavioral Testing of NLP Models with CheckList." *arXiv Preprint arXiv:2005.04118*.

Silva, Sara, and Breno Bernard Nicolau De França. 2023. "A Case Study on Data Science Processes in an Academia-Industry Collaboration." In *Proceedings of the XXII Brazilian Symposium on Software Quality*, 1–10.

Wattanakriengkrai, Supatsara, Bodin Chinthanet, Hideaki Hata, Raula Gaikovina Kula, Christoph Treude, Jin Guo, and Kenichi Matsumoto. 2022. "GitHub Repositories with Links to Academic Papers: Public Access, Traceability, and Evolution." *Journal of Systems and Software* 183: 111117.

Yan, Eugene. 2020. "How to Test Machine Learning Code and Systems." https://eugeneyan.com/writing/testing-ml/#model-evaluation-to-ensure-satisfactory-performance.