Test checklist for machine learning applications

Simon Goring Yingzi Jin John Shiu Tony Shum
Orix Au Yeung Rohan Alexander Tiffany A. Timbers

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 aimed to create a general and robust checklist for software tests for applied machine learning code. This checklist includes 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. The test checklist items were derived from software tests for applied machine learning code which either industry, or the scholarly literature have deemed as important for correct and robust applied machine learning software, as well as from examining things that commonly go wrong in machine learning code (threat model). 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 evaluating the quality of existing applied machine learning code, as well as for engineering reproducible test data and software tests themselves for each item on the checklist.

Table of contents

1 Introduction

- Problem with software robustness with applied machine learning code. Include and cite examples where non-robust code has led to negative societal and economic impacts (e.g., misinformation (Belanger 2024), social bias (Nunwick 2023), substantial financial losses (Regidi 2019), and safety hazards (Shepardson 2023))
- Checklists as a solution to safety critical systems (Gawande 2010)
- Checklists impact in machine learning reproducibility (Pineau et al. 2021)
- Goal: a checklist for tests for applied machine learning projects
- Scope: test checklist is aimed at applications of machine learning, not new machine learning methodologies. In future, a checklist for that could be developed, but that is out of scope for this paper.
- Potential impact: guide for practitioners, code reviewers, educators, as well as LLM prompts starting points for automated code test suite evaluation and automated test data and software test generation.

2 Background

- What are formal software tests and why are they used?
- On what kind of systems has most software testing literature been well studied and focused?
- What kinds of software tests exist
- What do we know about software testing in machine learning code, summarize: Braiek and Khomh (2020), Openja et al. (2023) and Silva and De França (2023) answering what kinds of software tests have been observed to be used in the wild.
- Recommendations for testing from industry, summarize: Microsoft Industry Solutions Engineering Team (2024) and Jordan (2020)
- Propose an edited version of the testing triangle that is more understandable

3 The Test Checklist for Machine Learning Applications

- Discuss possible threats to a robust and trustworthy applied machine learning project, possible issues include:
  - General errors in code (e.g., bug in code that leads to data labels being shifted by one, or misnaming a file being written to disk)

  - Mismatch of machine learning model choices with respect to the data used for training and evaluation (e.g., linear regression for binomial data)
  - Data quality issues (e.g., missing data, duplicate data, data anomalies, imbalances in categorical data, etc)
  - Data leakage between training and test set, leading to overfitting (e.g., test data being used to create pre-processing object)
  - Issues with hyperparameter tuning (e.g., too small of a search space when choosing hyperparameters)
  - General model performance (e.g., for example, in classification, are the labels what are expected based on input)
  - Model stability issues (e.g., a different train-validation split leads to a large change in the model)
  - Model behaviour/learning issues (e.g., model learns shortcut to predictions that can learn to erroneous prediction in certain cases)
  - Bias/fairness issues (e.g., model makes different predictions for particular subgroups of observations)
  - Reproducibility issues (e.g., model training and prediction outputs are different on different computers or operating systems)
  - Data drift (e.g., model performance decreases over time in production and the new data appears to being coming from a different distribution than the test data)
  - Communication of results/predictions issues and/or user interface issues (this itself is very very broad… and may have to be split out)

- Not all issues presented above can be adequately addressed through writing software tests, and so we will reduce the list for the purpose of our software test checklist for machine learning to the following issues:

1. Errors in code (e.g., bug in code that leads to data labels being shifted by one, or misnaming a file being written to disk)
2. Data quality issues (e.g., missing data, duplicate data, data anomalies, imbalances in categorical data, etc)
3. Data leakage between training and test set, leading to overfitting (e.g., test data being used to create pre-processing object)
4. General model performance (e.g., for example, in classification, are the labels what are expected based on input)
5. Model stability issues (e.g., a different train-validation split leads to a large change in the model)
6. Model behaviour/learning issues (e.g., model learns shortcut to predictions that can learn to erroneous prediction in certain cases)
7. Bias/fairness issues (e.g., model makes different predictions for particular subgroups of observations)
8. Reproducibility issues (e.g., model training and prediction outputs are different on different computers or operating systems)

9. Data drift (e.g., model performance decreases over time in production and the new data appears to being coming from a different distribution than the test data)

3.0.1 Applied machine learning checklist items

1. Errors in code
    - ☐ Loading data/model file function works as expected (Microsoft Industry Solutions Engineering Team 2024)
    - ☐ Saving data/model/figures function works as expected
    - ☐ Data cleaning and transforming functions works as expected (Microsoft Industry Solutions Engineering Team 2024).
2. Data quality issues
    - ☐ Code checks that files contain data and that it is in the expected format (Microsoft Industry Solutions Engineering Team 2024).
    - ☐ Code checks that data does not contain null values, duplicates, wrong types or outliers (Microsoft Industry Solutions Engineering Team 2024).
3. Data leakage between training and test set, leading to overfitting
    - ☐ Data is split into training and test set
    - ☐ Code checks that there are no duplicate records in the training and test set
    - ☐ Pre-processor is only created from the test set
4. General model performance
    - ☐ The model accept the correct inputs and produce the correctly shaped outputs (Microsoft Industry Solutions Engineering Team 2024)
    - ☐ The weights of the model update when running fit (Microsoft Industry Solutions Engineering Team 2024)
    - ☐ The model output aligns with expectations (Jordan 2020)
    - ☐ 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)
5. Model stability issues
    - ☐ Code checks the model weights stability during training (e.g., different training/validation/cross-validation splits shouldn't significantly change the weights)
6. Model behaviour/learning issues
    - ☐ Code checks for invariance for predictions Ribeiro et al. (2020)
    - ☐ Code checks for directionality of predictions Ribeiro et al. (2020)
7. Bias/fairness issues

   ☐ ???

8. Data drift

   ☐ Code checks/compares the distribution of features from the training data and the prediction data to determine whether they are different

- Use breast cancer toy example/demo to walk through 3-5 items from the checklist (maybe one from each kind of test from the testing triangle?)
- Report evaluation of n (not sure what number n should be…) applied machine learning projects in the wild with respect to the checklist (creates a benchmark)

4 Discussion

- Recall goal: a checklist for tests for applied machine learning projects
- Highlight # of checklist items, and number across each test category
- Summarize findings from evaluation of n applied machine learning projects with regards to the checklist
- Discuss potential immediate applications of checklist
  - professional data analysts/machine learning engineers working on applied machine learning projects (guide in the actual writing of tests)
  - Senior tech leads evaluating their team's applied machine learning project
  - Data science educators grading student projects
  - Data science students working on their own projects (guide in the actual writing of tests)
- Discuss further off applications of the checklist:
  - a source for engineering large-language model (LLM) prompts that act as reliable starting points for evaluating the quality of existing machine learning code
  - a source for engineering large-language model (LLM) prompts that generate code for reproducible test data and software tests themselves for each item on the checklist
- Come back to societal and economic impacts this tool may have (revisit misinformation (Belanger 2024), social bias (Nunwick 2023), substantial financial losses (Regidi 2019), and safety hazards (Shepardson 2023))

References

Belanger, Ashley. 2024. "Air Canada Must Honor Refund Policy Invented by Airline's Chatbot." Ars Technica. https://arstechnica.com/tech-policy/2024/02/air-canada-must-honor-refund-policy-invented-by-airlines-chatbot/.

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.

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.

Nunwick, Alice. 2023. "ITutorGroup Settles AI Hiring Lawsuit Alleging Age Discrimination." Verdict. https://www.verdict.co.uk/itutorgroup-settles-ai-hiring-lawsuit-alleging-age-discrimination/.

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.

Regidi, Asheeta. 2019. "SEBI's Circular: The Black Box Conundrum and Misrepresentation in AI-Based Mutual Funds." Firstpost. https://www.firstpost.com/business/sebis-circular-the-black-box-conundrum-and-misrepresentation-in-ai-based-mutual-funds-6625161.html.

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*.

Shepardson, David. 2023. "GM's Cruise Recalling 950 Driverless Cars After Pedestrian Dragged in Crash." Reuters. https://www.reuters.com/business/autos-transportation/gms-cruise-recall-950-driverless-cars-after-accident-involving-pedestrian-2023-11-08/.

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.