The Journal of Open Source Software

PerMetrics: A Framework of Performance Metrics for Machine Learning Models

Nguyen Van Thieu [1¶]

1 Faculty of Computer Science, Phenikaa University, Yen Nghia, Ha Dong, Hanoi, 12116, Vietnam. ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Performance metrics are pivotal in machine learning field, especially for tasks like regression, classification, and clustering (Saura, 2021). They offer quantitative measures to assess the accuracy and efficacy of models, aiding researchers and practitioners in evaluating, contrasting, and enhancing algorithms and models. In regression tasks, where continuous predictions are made, metrics like MSE, RMSE, and Coefficient of Determination (Nguyen et al., 2018; Nguyen, Nguyen, & Nguyen, 2019) can reveal how well models capture data patterns. In classification tasks, metrics such as accuracy, precision, recall, F1-score, and AUC-ROC (Luque et al., 2019) assess a model's ability to classify instances correctly, detect false results, and gauge overall predictive performance. Clustering tasks aim to discover inherent patterns and structures within unlabeled data by grouping similar instances together. Metrics like Silhouette coefficient, Davies-Bouldin index, and Calinski-Harabasz index (Nainggolan et al., 2019) measure clustering quality, helping evaluate how well algorithms capture data distribution and assign instances to clusters. In general, performance metrics serve multiple purposes. They enable researchers to compare different models and algorithms (Ahmed et al., 2021), identify strengths and weaknesses (Nguyen, Nguyen, Nguyen, & Nguyen, 2019), and make informed decisions about model selection and parameter tuning (Nguyen, Hoang, et al., 2020). Moreover, it also plays a crucial role in the iterative process of model development and improvement. By quantifying the model's performance, metrics guide the optimization process (Nguyen, Nguyen, et al., 2020), allowing researchers to fine-tune algorithms (Nguyen et al., 2021), explore feature engineering techniques, and address issues such as overfitting, underfitting, and bias (Van Thieu et al., 2023). This paper introduces a Python framework named PerMetrics (PERformance METRICS), designed to offer comprehensive performance metrics for machine learning models. The library, referred to as permetrics, is open-source and written in Python. It provides a wide number of metrics to enable users to evaluate their models effectively. PerMetrics is hosted on GitHub and is under continuous development and maintenance by the dedicated team. The framework is accompanied by comprehensive documentation, examples, and test cases, facilitating easy comprehension and integration into users' workflows.

Statement of need

Permetrics is a Python project developed in the field of performance assessment and machine learning. To the best of our knowledge, it is the first open-source framework that contributes a significant number of metrics, totaling 103 methods, for three fundamental problems: regression, classification, and clustering. This library relies exclusively on only two well-known third-party Python scientific computing packages: NumPy (Harris et al., 2020) and SciPy (Virtanen et al., 2020). The modules of permetrics are extensively documented, and the automatically generated API provides a complete and up-to-date description of both the object-oriented and functional implementations underlying the framework.

To gain a better understanding of the necessity of permetrics library, this section will compare it to several notable libraries currently are available. Most notably, Scikit-Learn (Pedregosa et al., 2011), which also encompasses an assortment of metrics for regression, classification, and clustering problems. Nevertheless, a few classification metrics present in Scikit-Learn lack support for multiple outputs, such as the Matthews correlation coefficient (MCC) and Hinge loss. Furthermore, critical metrics such as root mean square error, mean absolute percentage error, NSE, and KGE are absent. PerMetrics addresses these deficiencies. Additionally, Scikit-Learn is deficient in various vital clustering metrics, including but not limited to Ball Hall index, Banfeld Raftery index, sum of squared error, Duda Hart index, and Hartigan index (Desgraupes, 2013).

Another popular package is Metrics (Hamner, 2015). It provides a variety of metrics for different programming languages such as Python, MATLAB, R, and Haskell. However, the development team has ceased activity since 2015. They offers a limited number of metrics because they focused on creating a single set of metrics for multiple languages. Additionally, the metrics are not packaged as a complete library but rather exist as repository code on GitHub.

TorchMetrics (Nicki Skafte Detlefsen et al., 2022) is a widely recognized framework for performance metrics developed for PyTorch users. The library includes over 100 metrics, covering various domains such as regression, classification, audio, detection, and text. However, TorchMetrics does not provide metrics specifically for clustering tasks. Although it offers a substantial number of metrics, it falls short compared to permetrics. Moreover, it relies heavily on other major libraries such as NumPy, Torch, Typing-extensions, Packaging, and Lightning-utilities. Additionally, using this library may not be easy for beginners in Python programming, as it requires a deep understanding of the Torch library to utilize TorchMetrics effectively.

Other popular libraries such as TensorFlow (Abadi et al., 2016), Keras (Chollet & others, 2015), CatBoost (Prokhorenkova et al., 2018), and MxNet (Chen et al., 2015) also contain modules dedicated to metrics. However, the issue with these libraries is that their metric modules are specific to each respective one. It is challenging to combine metric modules from different libraries with each other. If it is possible to combine them, it often requires installing numerous related libraries. Furthermore, the metric modules within each library are tailored to users who are familiar with that specific one, requiring users to learn multiple libraries, syntax structures, and necessary commands associated with each framework to use them in combination. These are significant obstacles when using metrics from such libraries.

All the aforementioned challenges are addressed by our permetrics library. It not only offers a simple and concise syntax and usage but also does not require any knowledge of other major libraries such as TensorFlow, Keras, or PyTorch. Additionally, it can be seamlessly integrated with any computational or machine learning library. In the future, we plan to expand permetrics to include other domains such as text metrics, audio metrics, detection metrics, and image metrics.

Test paper

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., & others. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv Preprint arXiv:1603.04467*.

Ahmed, A. N., Van Lam, T., Hung, N. D., Van Thieu, N., Kisi, O., & El-Shafie, A. (2021). A comprehensive comparison of recent developed meta-heuristic algorithms for streamflow time series forecasting problem. *Applied Soft Computing*, *105*, 107282. https://doi.org/10.1016/j.asoc.2021.107282

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., & Zhang, Z. (2015). Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv Preprint arXiv:1512.01274*.

Chollet, F., & others. (2015). *Keras*. https://keras.io

Desgraupes, B. (2013). Clustering indices. *University of Paris Ouest-Lab Modal'X*, *1*(1), 34.

Hamner, B. (2015). *Metrics*. https://github.com/benhamner/Metrics

Harris, C. R., Millman, K. J., Van Der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., & others. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362.

Luque, A., Carrasco, A., Martín, A., & De Las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, *91*, 216–231. https://doi.org/10.1016/j.patcog.2019.02.023

Nainggolan, R., Perangin-angin, R., Simarmata, E., & Tarigan, A. F. (2019). Improved the Performance of the K-Means Cluster Using the Sum of Squared Error (SSE) optimized by using the Elbow Method. *Journal of Physics: Conference Series*, *1361*(1), 012015. https://doi.org/10.1088/1742-6596/1361/1/012015

Nguyen, T., Hoang, B., Nguyen, G., & Nguyen, B. M. (2020). A new workload prediction model using extreme learning machine and enhanced tug of war optimization. *Procedia Computer Science*, *170*, 362–369. https://doi.org/10.1016/j.procs.2020.03.063

Nguyen, T., Nguyen, B. M., & Nguyen, G. (2019). Building resource auto-scaler with functional-link neural network and adaptive bacterial foraging optimization. *International Conference on Theory and Applications of Models of Computation*, 501–517. https://doi.org/10.1007/978-3-030-14812-6_31

Nguyen, T., Nguyen, G., & Nguyen, B. M. (2020). EO-CNN: An enhanced CNN model trained by equilibrium optimization for traffic transportation prediction. *Procedia Computer Science*, *176*, 800–809. https://doi.org/10.1016/j.procs.2020.09.075

Nguyen, T., Nguyen, T., Nguyen, B. M., & Nguyen, G. (2019). Efficient time-series forecasting using neural network and opposition-based coral reefs optimization. *International Journal of Computational Intelligence Systems*, *12*, 1144–1161. https://doi.org/10.2991/ijcis.d.190930.003

Nguyen, T., Nguyen, T., Vu, Q.-H., Huynh, T. T. B., & Nguyen, B. M. (2021). Multi-objective sparrow search optimization for task scheduling in fog-cloud-blockchain systems. *2021 IEEE International Conference on Services Computing (SCC)*, 450–455.

Nguyen, T., Tran, N., Nguyen, B. M., & Nguyen, G. (2018). A resource usage prediction system using functional-link and genetic algorithm neural network for multivariate cloud metrics. *2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA)*, 49–56. https://doi.org/10.1109/SOCA.2018.00014

Nicki Skafte Detlefsen, Jiri Borovec, Justus Schock, Ananya Harsh, Teddy Koker, Luca Di Liello, Daniel Stancl, Changsheng Quan, Maxim Grechkin, & William Falcon. (2022). *TorchMetrics - Measuring Reproducibility in PyTorch*. https://doi.org/10.21105/joss.04101

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, *31*. https://catboost.ai

Saura, J. R. (2021). Using Data Sciences in Digital Marketing: Framework, methods, and performance metrics. *Journal of Innovation & Knowledge*, *6*(2), 92–102. https://doi.org/10.1016/j.jik.2020.08.001

Van Thieu, N., Deb Barma, S., Van Lam, T., Kisi, O., & Mahesha, A. (2023). Groundwater level modeling using Augmented Artificial Ecosystem Optimization. *Journal of Hydrology*, *617*, 129034. https://doi.org/10.1016/j.jhydrol.2022.129034

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., & others. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, *17*(3), 261–272.