Journal of Machine Learning Research 25 (2024) 1-10 Submitted 11/23; Revised 3/24; Published 9/24

aeon: a Python Toolkit for Learning from Time Series

Matthew Middlehurst[1] M.B.MIDDLEHURST@SOTON.AC.UK
Ali Ismail-Fawaz[2] ALI-EL-HADI.ISMAIL-FAWAZ@UHA.FR
Antoine Guillaume[3] ANTOINE.GUILLAUME@NOVAHE.FR
Christopher Holder[1,4] C.HOLDER@UEA.AC.UK
David Guijo-Rubio[4,5] DGUIJO@UCO.ES
Guzal Bulatova GUZALBULATOVA@GMAIL.COM
Leonidas Tsaprounis LEONIDAS.TSAP@GMAIL.COM
Lukasz Mentel LUKASZ.MENTEL@PM.ME
Martin Walter MARTIN.FRIEDRICH.WALTER@GMAIL.COM
Patrick Schäfer[6] PATRICK.SCHAEFER@HU-BERLIN.DE
Anthony Bagnall[1,4] A.J.BAGNALL@SOTON.AC.UK
The authors are the developers of aeon at the time of submission, with the following affiliations:
[1]ECS, University of Southampton, United Kingdom [2]IRIMAS, Université de Haute-Alsace, France [3]Novahé and Constellation, France [4]CMP, University of East Anglia, United Kingdom [5]DIAN, University of Córdoba, Spain [6]Humboldt-Universität zu Berlin, Germany

Editor: Pradeep Ravikumar

Abstract

`aeon` is a unified Python 3 library for all machine learning tasks involving time series. The package contains modules for time series forecasting, classification, extrinsic regression and clustering, as well as a variety of utilities, transformations and distance measures designed for time series data. `aeon` also has a number of experimental modules for tasks such as anomaly detection, similarity search and segmentation. `aeon` follows the `scikit-learn` API as much as possible to help new users and enable easy integration of `aeon` estimators with useful tools such as model selection and pipelines. It provides a broad library of time series algorithms, including efficient implementations of the very latest advances in research. Using a system of optional dependencies, `aeon` integrates a wide variety of packages into a single interface while keeping the core framework with minimal dependencies. The package is distributed under the 3-Clause BSD license and is available at `https://github.com/aeon-toolkit/aeon`.

Keywords: Python, open source, time series, machine learning, data mining, forecasting, classification, extrinsic regression, clustering

1. Introduction

Time series appear in all areas of scientific research and play a central role in all business analytics. Time Series Machine Learning (TSML) research involves developing and using algorithms that exploit the unique characteristics of ordered data: interesting features may be based on the interaction between observations. TSML encompasses standard machine learning tasks such as classification, clustering, extrinsic regression and anomaly detection in addition to time series specific problems such as forecasting and segmentation.

TSML is an active research field, and aeon plays a central role in facilitating reproducible research (Middlehurst et al., 2024; Guijo-Rubio et al., 2024; Holder et al., 2023).

aeon is a Python toolkit for TSML tasks, preprocessing and benchmarking. The package is designed to be easy to use, especially for those familiar with scikit-learn (Pedregosa et al., 2011). Where possible the scikit-learn API and style is followed, and where there are shared learning tasks aeon objects aim to be compatible with the available utilities such as model selection and pipelining. aeon is forked from version v0.16.0 of the sktime (Löning et al., 2021) package. Since the community split, the packages have diverged significantly in terms of available modules and included estimators. In the following, we provide an overview of the packages available in aeon, as well as some experimental and upcoming modules. This document describes version v0.5.0 of aeon. However, the toolkit is being constantly enhanced. We gave two tutorials on aeon in 2024. Details on the classification and regression modules are available in our hands-on SIGKDD tutorial (Bagnall et al., 2024). Slides and code for this tutorial are all open source and available at this URL: https://aeon-tutorials.github.io/KDD-2024/. An overview of the whole toolkit was presented at an ECML-PKDD tutorial (see https://aeon-tutorials.github.io/ECML-2024/). aeon supports all versions from Python 3.8 onwards in version v0.5.0. Extensive online documentation is available at https://aeon-toolkit.org.

2. Code Design and Implementation

In aeon, the package design is modular, with algorithms grouped by learning tasks. The primary TSML modules such as clustering and classification are encapsulated as much as possible and do not import from each other. Supporting modules such as distances and transformations exist to hold functions and classes useable in multiple learning tasks, as well as standalone if required.

aeon uses object-oriented design for most cases to fit into the scikit-learn estimator interface. Some modules are more functional in design, such as distance measures and performance metrics. aeon TSML classes follow an inheritance structure with each module having its own base class. An example of this structure for some of the main aeon modules is shown in Figure 1. The base class for a module contains mandatory methods such as fit to be inherited, as well default functionality such as converting and verifying input data.

Estimators have tags that identify the types of data the estimators can take and their functionality. So, for example, a classifier with the following capability tags can handle both multivariate and unequal length time series input.

```
_tags = {
    "capability:multivariate": True,
    "capability:unequal_length": True
}
```

In aeon we aim to keep the package core dependencies to a minimum. The primary dependency is scikit-learn, which we base our estimator interface off. Shared with scikit-learn, the package contains extensive usage of the numpy (Harris et al., 2020) and scipy (Virtanen et al., 2020) libraries. scikit-learn makes use of cython (Behnel et al., 2010) to optimise its implementation, aeon depends on numba (Lam et al., 2015) and aims to use numba compiled just-in-time (JIT) functions where possible.

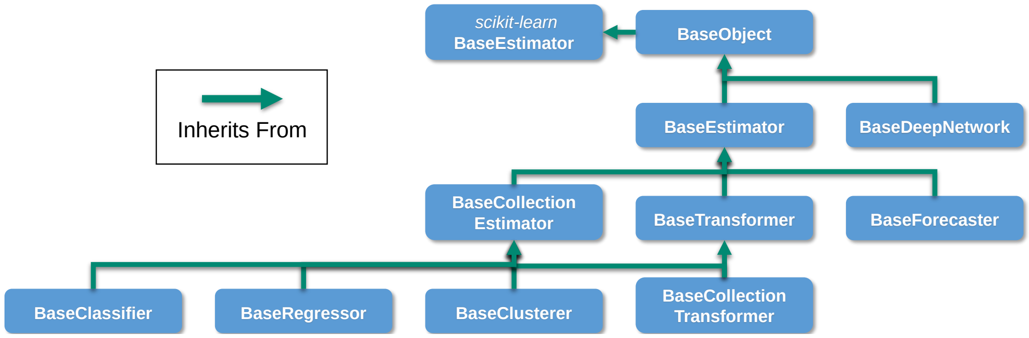

Figure 1: A flowchart of base class inheritance for the main `aeon` modules. Classification algorithms will inherit from `BaseClassifier`, for example.

As well as the core dependencies, `aeon` also includes a range of optional/soft dependencies to packages such as `statsmodels` (Seabold and Perktold, 2010), `tensorflow` (Abadi et al., 2015), and `tsfresh` (Christ et al., 2018). These are commonly used to create wrappers for algorithms present in these packages or used as a framework for estimators such as deep learners. Creating a singular framework for these TSML algorithms allows for easier benchmarking and reproducibility.

3. Time Series Modules

`aeon` splits different TSML tasks into modules. In the following we go over some of the stable core modules of `aeon`, as well as some experimental modules. Figure 2 displays a flowchart for different TSML modules, with situations where one would want to use each.

3.1 Forecasting

At its simplest, forecasting involves predicting the next values in sequence of a given time series. In `aeon` there are mechanisms for reducing forecasting to regression through windowing and a range of tools to help produce better forecasts.

The forecasting module in `aeon` is, at the time of writing, undergoing significant revision. Please look at the latest version or talk to us on slack for the latest information. The forecasting interface is built using a similar *fit* and *predict* structure to `scikit-learn`. The basic usage is as follows.

```
from aeon.datasets import load_airline
from aeon.forecasting.trend import TrendForecaster
y = load_airline()
forecaster = TrendForecaster()
forecaster.fit(y)  # fit the forecaster
pred = forecaster.predict()  # predict the next value
```

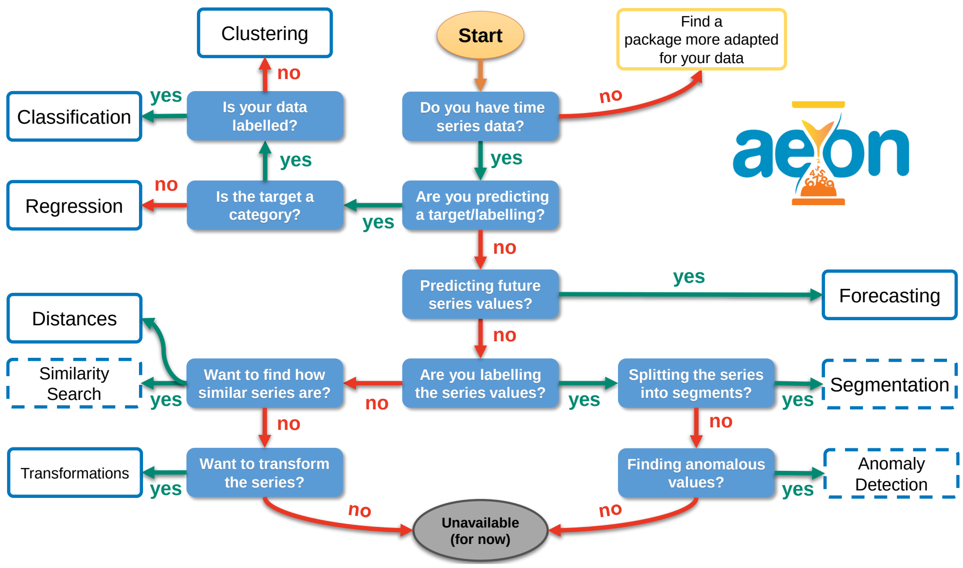

Figure 2: A flowchart for selecting the correct time series learning module in `aeon`. Learning tasks with dashed borders are experimental.

3.2 Classification, Clustering and Regression

Time series classification, clustering and extrinsic regression tasks all have the same *fit* and *predict* interface and are compatible with the `scikit-learn` interface for the same tasks. All of these estimators inherit from a `BaseCollectionEstimator`, where tags are defined and conversions are performed. Typically, collections of series are stored in a 3D `numpy` array, while unequal length series are stored in a list of 2D `numpy` arrays. The shape for both of these data types is (n_cases, n_channels, n_timepoints). Using classification as an example, the most basic usage is as follows:

```python
from aeon.datasets import load_classification
from aeon.classification.convolution_based import RocketClassifier
train_X, train_y, _ = load_classification("GunPoint", split="Train")
test_X, _, _ = load_classification("GunPoint", split="Test")
clf = RocketClassifier()
clf.fit(train_X,train_y)  # fit the classifier
preds = clf.predict(test_X)  # make a prediction for each test case
```

3.2.1 Classification

Classification algorithms are sometimes grouped on the basis of the transformation used (Bagnall et al., 2017; Middlehurst et al., 2024). `aeon` contains an extensive range of classifiers, using such a taxonomy to create sub-packages of approaches. The classification module includes a wide breadth of algorithmic approaches types: convolution based (Dempster et al., 2020; Tan et al., 2022); deep learning (Fawaz et al., 2019, 2020; Ismail-Fawaz et al., 2022); dictionary based (Schäfer, 2015; Middlehurst et al., 2020b; Schäfer and Leser, 2023; Bennett and Abdallah, 2023); distance based (Lines and Bagnall, 2015; Lucas et al., 2019); feature based (Lubba et al., 2019; Middlehurst and Bagnall, 2022); hybrids (Lines et al., 2018; Middlehurst et al., 2021); interval based (Deng et al., 2013; Middlehurst et al., 2020a; Cabello et al., 2023); and shapelet based (Bostrom and Bagnall, 2017; Nguyen and Ifrim, 2021; Guillaume et al., 2022). Our documentation and the research field survey in Middlehurst et al. (2024) has more information on each category and algorithm.

3.2.2 Clustering

Clustering works in conjunction with the distances module to provide k-means based and k-medoids based clustering. k-means works with over ten elastic distance functions implemented (Holder et al., 2023) and can be used with barycentre averaging (Petitjean et al., 2011; Ismail-Fawaz et al., 2023b). We are also working on including a range of feature based and deep learning clustering algorithms (Lafabregue et al., 2022).

3.2.3 Regression

Extrinsic regression is a recently defined task (Tan et al., 2021). `aeon` contains a range of algorithms adapted from classification (Guijo-Rubio et al., 2024; Middlehurst and Bagnall, 2023), including popular deep learning algorithms (Foumani et al., 2023). We hope that keeping open implementations in `aeon` will help the field in growing, and we intend to keep the library up to date with the latest state-of-the-art.

3.3 Transformations

Transformers are objects that transform data from one representation to another. `aeon` contains time series specific transformers which can be used in pipelines in conjunction with other estimators. `aeon` transformers follow the `scikit-learn` design with a *fit* and *transform* method.

General transformers aim to accept all input types whether that is a single series or a collection, and will attempt to restructure the data or broadcast to multiple transformer objects if necessary to fit the input data to the data structure used by the transformer. Collection transformers are structured to efficiently process collections of series, and can only take that kind of input. Transformations are used extensively in all other modules.

Transformers come in multiple types. They can be series-to-series transformations which both take and output a time series, such as the Fourier transform or channel selection for multivariate series (Dhariyal et al., 2023). Alternatively, transformers can be series-to-features which take a series input but output a feature vector such as basic summary statistics or TSFresh (Christ et al., 2018).

3.4 Experimental Modules

An experimental module is an area of the code base that in progress of being developed at the time of writing and can still rapidly change. Current experimental modules include: Segmentation (Hallac et al., 2019; Ermshaus et al., 2023); Anomaly detection (Nakamura et al., 2020; Talagala et al., 2021); Similarity search; and benchmarking (Ismail-Fawaz et al., 2023a).

4. Conclusions

There are several other Python toolkits that implement TSML algorithms. These include `tslearn` (Tavenard et al., 2020) and `pyts` (Faouzi and Janati, 2020) for classification and clustering, `adtk` for anomaly detection, numerous time series distance packages such as `dtaidistance` and matrix profile tools such as `stumpy` (Law, 2019). We believe `aeon` is the most comprehensive toolkit for time series machine learning. We hope that our attempt at unifying diverse research fields and communities such as forecasting, classification and anomaly detection will benefit the development of the Python time series ecosystem. `aeon` joined numFOCUS as an affiliate project in 2024 and we hope to work towards full membership in the near future.

Acknowledgments

`aeon` is supported by the EPSRC under a special call grant EP/W030756/2. Thank you to everyone who has contributed to `aeon` whether that be through code, documentation or other means. There are too many to credit individually, but everyone has a hand in making the project great.

References

M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31(3):606–660, 2017.

A. Bagnall, M. Middlehurst, G. Forestier, A. Ismail-Fawaz, A. Guillaume, D. Guijo-Rubio, C. W. Tan, A. Dempster, and G. I. Webb. A hands-on introduction to time series classification and regression. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '24, 2024. URL `https://doi.org/10.1145/3637528.3671443`.

S. Behnel, R. Bradshaw, C. Citro, L. Dalcin, D. S. Seljebotn, and K. Smith. Cython: The best of both worlds. *Computing in Science & Engineering*, 13(2):31–39, 2010.

L. A. Bennett and Z. S. Abdallah. Red comets: An ensemble classifier for symbolically represented multivariate time series. In *International Workshop on Advanced Analytics and Learning on Temporal Data*. Springer, 2023.

A. Bostrom and A. Bagnall. Binary shapelet transform for multiclass time series classification. In *Transactions on Large-Scale Data-and Knowledge-Centered Systems XXXII*, pages 24–46. Springer, 2017.

N. Cabello, E. Naghizade, J. Qi, and L. Kulik. Fast, accurate and explainable time series classification through randomization. *Data Mining and Knowledge Discovery*, pages 1–64, 2023.

M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing*, 307: 72–77, 2018.

A. Dempster, F. Petitjean, and G. I. Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

H. Deng, G. Runger, E. Tuv, and M. Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.

B. Dhariyal, T. Le Nguyen, and G. Ifrim. Scalable classifier-agnostic channel selection for multivariate time series classification. *Data Mining and Knowledge Discovery*, 37(2): 1010–1054, 2023.

A. Ermshaus, P. Schäfer, and U. Leser. Clasp: parameter-free time series segmentation. *Data Mining and Knowledge Discovery*, 37(3):1262–1300, 2023.

J. Faouzi and H. Janati. pyts: A python package for time series classification. *The Journal of Machine Learning Research*, 21(1):1720–1725, 2020.

H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, 2019.

H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.

N. M. Foumani, L. Miller, C. W. Tan, G. I. Webb, G. Forestier, and M. Salehi. Deep learning for time series classification and extrinsic regression: A current survey. *arXiv preprint arXiv:2302.02515*, 2023.

D. Guijo-Rubio, M. Middlehurst, G. Arcencio, D. F. Silva, and A. Bagnall. Unsupervised feature based algorithms for time series extrinsic regression. *Data Mining and Knowledge Discovery*, 2024.

A. Guillaume, C. Vrain, and W. Elloumi. Random dilated shapelet transform: A new approach for time series shapelets. In *International Conference on Pattern Recognition and Artificial Intelligence*, pages 653–664. Springer, 2022.

D. Hallac, P. Nystrup, and S. Boyd. Greedy gaussian segmentation of multivariate time series. *Advances in Data Analysis and Classification*, 13(3):727–751, 2019.

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

C. Holder, M. Middlehurst, and A. Bagnall. A review and evaluation of elastic distance functions for time series clustering. *Knowledge and Information Systems*, pages 1–45, 2023.

A. Ismail-Fawaz, M. Devanne, J. Weber, and G. Forestier. Deep learning for time series classification using new hand-crafted convolution filters. In *2022 IEEE International Conference on Big Data (Big Data)*, pages 972–981. IEEE, 2022.

A. Ismail-Fawaz, A. Dempster, C. W. Tan, M. Herrmann, L. Miller, D. F. Schmidt, S. Berretti, J. Weber, M. Devanne, G. Forestier, et al. An approach to multiple comparison benchmark evaluations that is stable under manipulation of the comparate set. *arXiv preprint arXiv:2305.11921*, 2023a.

A. Ismail-Fawaz, H. I. Fawaz, F. Petitjean, M. Devanne, J. Weber, S. Berretti, G. I. Webb, and G. Forestier. Shapedba: Generating effective time series prototypes using shapedtw barycenter averaging. 2023b.

B. Lafabregue, J. Weber, P. Gançarski, and G. Forestier. End-to-end deep representation learning for time series clustering: a comparative study. *Data Mining and Knowledge Discovery*, 36(1):29–81, 2022.

S. K. Lam, A. Pitrou, and S. Seibert. Numba: A llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, pages 1–6, 2015.

S. M. Law. Stumpy: A powerful and scalable python library for time series data mining. *Journal of Open Source Software*, 4(39):1504, 2019.

J. Lines and A. Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015.

J. Lines, S. Taylor, and A. Bagnall. Time series classification with hive-cote: The hierarchical vote collective of transformation-based ensembles. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(5):52, 2018.

C. H. Lubba, S. S. Sethi, P. Knaute, S. R. Schultz, B. D. Fulcher, and N. S. Jones. catch22: Canonical time-series characteristics. *Data Mining and Knowledge Discovery*, 33(6):1821–1852, 2019.

B. Lucas, A. Shifaz, C. Pelletier, L. O'Neill, N. Zaidi, B. Goethals, F. Petitjean, and G. I. Webb. Proximity forest: an effective and scalable distance-based classifier for time series. *Data Mining and Knowledge Discovery*, 33(3):607–635, 2019.

M. Löning, A. Bagnall, S. Ganesh, V. Kazakov, J. Lines, and F. Király. sktime: A unified interface for machine learning with time series. *arXiv preprint arXiv:1909.07872*, 2021.

M. Middlehurst and A. Bagnall. The freshprince: A simple transformation based pipeline time series classifier. In *International Conference on Pattern Recognition and Artificial Intelligence*, pages 150–161. Springer, 2022.

M. Middlehurst and A. Bagnall. Extracting features from random subseries: A hybrid pipeline for time series classification and extrinsic regression. In *International Workshop on Advanced Analytics and Learning on Temporal Data*, pages 113–126. Springer, 2023.

M. Middlehurst, J. Large, and A. Bagnall. The canonical interval forest (cif) classifier for time series classification. In *2020 IEEE international conference on big data (big data)*, pages 188–195. IEEE, 2020a.

M. Middlehurst, J. Large, G. Cawley, and A. Bagnall. The temporal dictionary ensemble (tde) classifier for time series classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 660–676. Springer, 2020b.

M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom, and A. Bagnall. Hive-cote 2.0: a new meta ensemble for time series classification. *Machine Learning*, 110(11):3211–3243, 2021.

M. Middlehurst, P. Schäfer, and A. Bagnall. Bake off redux: a review and experimental evaluation of recent time series classification algorithms. *Data Mining and Knowledge Discovery*, 2024.

T. Nakamura, M. Imamura, R. Mercer, and E. Keogh. Merlin: Parameter-free discovery of arbitrary length anomalies in massive time series archives. In *2020 IEEE international conference on data mining (ICDM)*, pages 1190–1195. IEEE, 2020.

T. L. Nguyen and G. Ifrim. Mrsqm: Fast time series classification with symbolic representations. In *International Workshop on Advanced Analytics and Learning on Temporal Data*. Springer, 2021.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

F. Petitjean, A. Ketterlin, and P. Gançarski. A global averaging method for dynamic time warping, with applications to clustering. *Pattern recognition*, 44(3):678–693, 2011.

P. Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29(6):1505–1530, 2015.

P. Schäfer and U. Leser. Weasel 2.0: a random dilated dictionary transform for fast, accurate and memory constrained time series classification. *Machine Learning*, 112(12):4763–4788, 2023.

S. Seabold and J. Perktold. Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th Python in Science Conference*, volume 57, pages 10–25080. Austin, TX, 2010.

P. D. Talagala, R. J. Hyndman, and K. Smith-Miles. Anomaly detection in high-dimensional data. *Journal of Computational and Graphical Statistics*, 30(2):360–374, 2021.

C. W. Tan, C. Bergmeir, F. Petitjean, and G. I. Webb. Time series extrinsic regression: Predicting numeric values from time series data. *Data Mining and Knowledge Discovery*, 35:1032–1060, 2021.

C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb. Multirocket: Multiple pooling operators and transformations for fast and effective time series classification. *Data Mining and Knowledge Discovery*, pages 1–24, 2022.

R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar, et al. Tslearn, a machine learning toolkit for time series data. *The Journal of Machine Learning Research*, 21(1):4686–4691, 2020.

P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.