The Journal of Open Source Software

PyCVI: A Python package for internal Cluster Validity Indices, compatible with time-series data

Natacha Galmiche [1]

1 University of Bergen, Norway

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

PyCVI is a Python package specialized in internal Clustering Validity Indices (CVIs) compatible with both time-series and non time-series data.

Clustering is a task that aims at finding groups within a given dataset. CVIs are used to select the best clustering among a pre-computed set of clusterings. In other words, CVIs select the division of the dataset into groups that best ensures that similar datapoints belong to the same group and non-related datapoints are in different groups.

PyCVI implements 12 state-of-the-art *internal* CVIs to improve clustering pipelines as well as the Variation of Information (VI) (Meilă, 2003), a distance measure between clusterings. VI can have many purposes, among which being used as an *external* CVI and to evaluate internal CVIs or clustering methods when true labels are known. The *internal* qualifier here refers to the general case in practice where no *external* information is available about the dataset such as the true association of the datapoints with groups, as opposed to *classification* tasks.

Statement of need

There exist many mature libraries in python for machine learning and in particular clustering: scikit-learn (Pedregosa et al., 2011), TensorFlow (Abadi et al., 2015), PyTorch (Paszke et al., 2019), scikit-learn-extra (*Scikit-learn Extra*, n.d.), and even several specifically for time series data: aeon (*Aeon*, n.d.), sktime (Löning et al., 2019), tslearn (Tavenard et al., 2020).

However, although being fundamental to clustering tasks and being an active research topic, very few internal CVIs are implemented in standard python libraries (only 3 in scikit-learn, more were available in R but few were maintained and kept in CRAN (Charrad et al., 2014)). Thus for a given CVI, there is currently no corresponding maintained and public implementation. This is despite the well-known limitations of all existing CVIs (Arbelaitz et al., 2013), (Gagolewski et al., 2021), (Gurrutxaga et al., 2011), (Theodoridis & Koutroumbas, 2009) and the need to use the right one(s) according to the specific dataset at hand, similarly to matching the right clustering method with the given problem. A crucial step towards developing better CVIs would be an easy access to an implementation of existing CVIs in order to facilitate larger comparative studies.

In addition, all CVIs rely on the definition of a distance between datapoints and most of them on the notion of cluster center.

For static data, the distance between datapoints is usually the euclidean distance and the cluster center is defined as the usual average. Libraries such as scipy, numpy, scikit-learn, etc. offer a large selection of distance measures that are compatible with their main functions.

For time-series data however, the common distance used is Dynamic Time Warping (DTW) (Berndt & Clifford, 1994) and the barycenter of a group of time series is then not defined as the usual mean, but as the DTW Barycentric Average (DBA) (Petitjean et al., 2011). Unfortunately, DTW and DBA are not compatible with the libraries mentioned above. This, among other reasons, made additional machine learning libraries specialized in time series data such as aeon, sktime and tslearn necessary.

PyCVI fills that gap by implementing 12 state-of-the-art internal CVIs: Hartigan (Strauss & Hartigan, 1975), Calinski-Harabasz (Calinski & Harabasz, 1974), GapStatistic (Tibshirani et al., 2001), Silhouette (Rousseeuw, 1987), ScoreFunction (Saitta et al., 2007), Maulik-Bandyopadhyay (Maulik & Bandyopadhyay, 2002), SD (Halkidi et al., 2000), SDbw (Halkidi & Vazirgiannis, 2001), Dunn (Dunn, 1974), Xie-Beni (Xie & Beni, 1991), XB* (Kim & Ramakrishna, 2005) and Davies-Bouldin (Davies & Bouldin, 1979). Furthermore, in PyCVI their definition is extended in order to make them compatible with DTW and DBA in addition to static data. Finally, PyCVI is entirely compatible with scikit-learn, scikit-learn-extra, aeon and sktime, in order to be easily integrated into any clustering pipeline in python. To ensure a fast a reliable computation of DTW and DBA, PyCVI relies on the aeon library.

Example

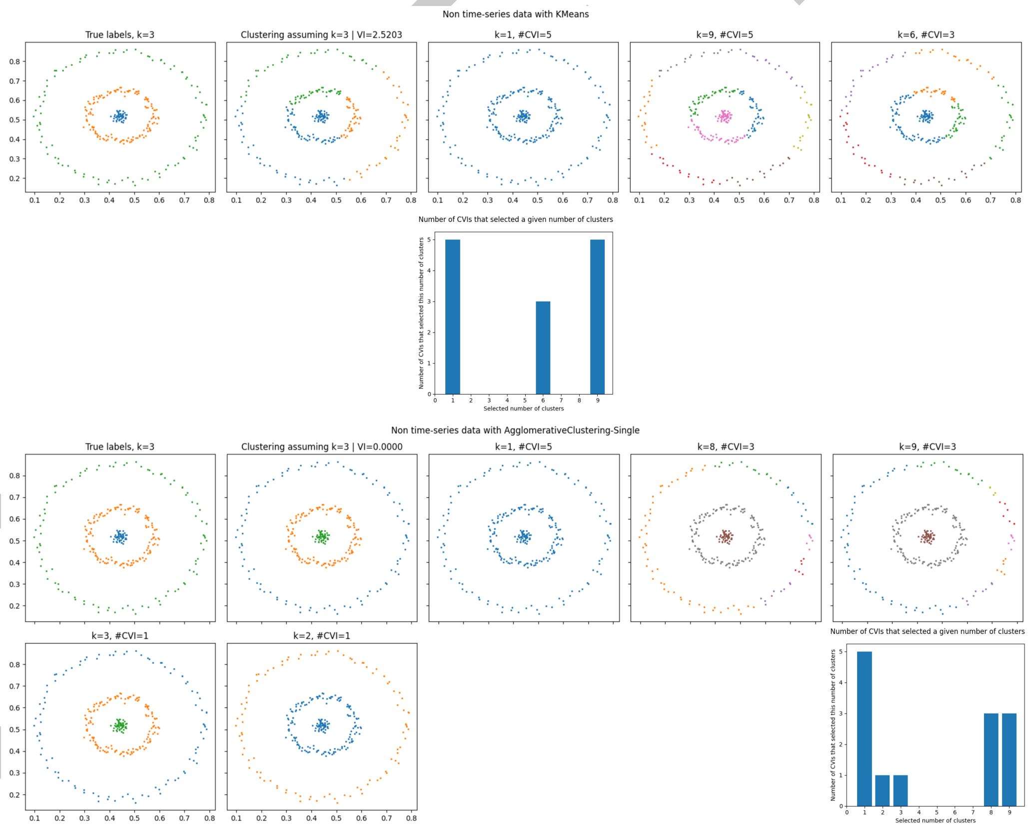

Figure 1: KMeans and AgglomerativeClustering on static data. Selected clusterings according to each implemented CVIs.

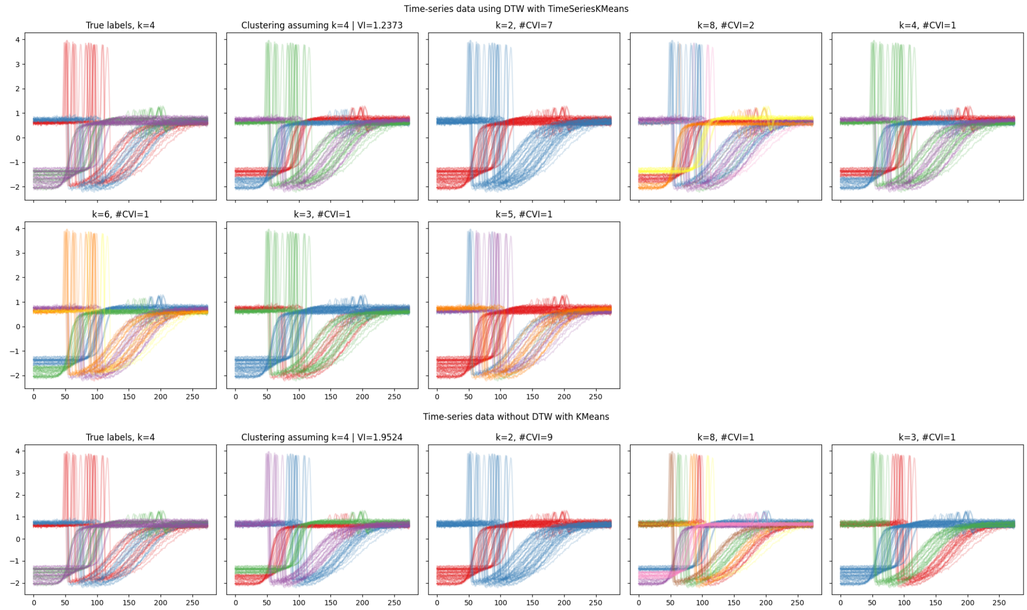

Figure 2: KMeans on time-series data, with and without DTW.

We experimented 3 cases: static data (Barton, 2015), time-series data (Dau et al., 2018) with euclidean distance and then with DTW as distance measure and and DBA as center of clusters. In addition, we used different clustering methods from different libraries: KMeans (Lloyd, 1982) and AgglomerativeClustering (Ward, 1963) from scikit-learn, TimeSeriesKMeans from aeon and KMedoids ("Partitioning Around Medoids (Program PAM)," 1990) from scikit-learn-extra to showcase PyCVI integration with other clustering libraries.

As a first example, we individually ran all CVIs implemented in PyCVI, selected the best clustering according to each CVI and plotted the selected clustering. In addition, we computed the variation of information (VI) between each selected clustering and the true clustering. High VI values mean large distances between the true clustering and the computed clusterings, meaning computed clusterings of poor quality. In Figure 1, we can see the difference of quality when assuming the correct number of clusters between the AgglomerativeClustering and the KMeans clustering method on static data. This is independent of the CVI used, meaning that a poor clustering quality will be due to the clustering method.

In Figure 1, since the quality of clusterings generated by KMeans is bad due to the clustering method, the poor selection results gives us no information about the correct clustering, nor about the quality of the CVIs used. This motivates further research on clustering methods. However, using AgglomerativeClustering, the quality of the clustering is excellent, as indicated by a null VI. The corresponding selection results shown in the corresponding histogram tells us that the CVIs used here are not adapted to this dataset. This was expected since most CVIs rely on the cluster center to compute a good separation between clusters. The dataset here consisting of concentric circles, most CVIs fail to measure how well separated the clusters actually are. This illustrates the need of further research on CVIs, which is facilitated by PyCVI, notably in the case of concentric subgroups.

Similarly, with time-series data in Figure 2, the quality of the clustering assuming the correct number of clusters varies although the same clustering method is used on the same dataset. This illustrates the difference between using DTW as a distance measure compared to using the euclidean distance, and between using DBA to compute the average of a group of time series and using the usual average.

In a second example, we demonstrate cases of successful clustering and clustering selection, while showcasing an additional feature of PyCVI: `CVIAggregator`. `CVIAggregator` selects the best clustering by combining several CVIs and by using the majority vote among the clusterings individually selected by the combined CVI.

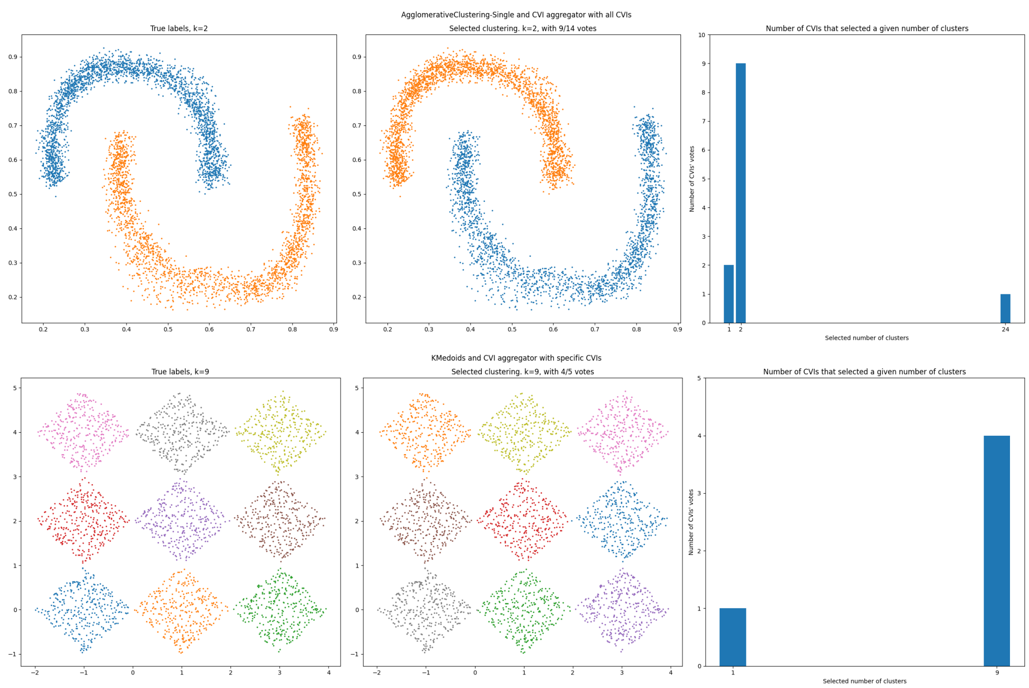

Figure 3: Selection done by a CVIAggregator using all implemented CVIs first and then with specific CVIs (GapStatistic, Silhouette, Dunn, CalinskiHarabasz and XieBeni).

In Figure 3, we used `CVIAggregator` with first all CVIs implemented in PyCVI and then only with some of the implemented CVIs, as it could be done in practice when known characteristics of the dataset can help identify unadapted CVIs. We see that in both cases, the data was correctly clustered by the clustering method and the best clustering correctly selected. This is in spite of clusters of non-convex shapes in the first case and clusters "touching" each other in the second

The code of these examples is available on the GitHub repository of the package, and its documentation.

Acknowledgements

We thank the climate and energy transition strategy of the University of Bergen in Norway (UiB) for funding this work.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., … Zheng, X. (2015). *TensorFlow: Large-scale machine learning on heterogeneous systems*. https://www.tensorflow.org/

Aeon. (n.d.). "https://github.com/aeon-toolkit/aeon"; Github.

Arbelaitz, O., Gurrutxaga, I., Muguerza, J., Pérez, J. M., & Perona, I. (2013). An extensive comparative study of cluster validity indices. *Pattern Recognition*, *46*(1), 243–256. https://doi.org/10.1016/j.patcog.2012.07.021

Barton, T. (2015). *Clustering benchmarks*. "https://github.com/deric/clusteringbenchmark"; Github.

Berndt, D. J., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, 359–370.

Calinski, T., & Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics - Theory and Methods*, *3*(1), 1–27. https://doi.org/10.1080/03610927408827101

Charrad, M., Ghazzali, N., Boiteau, V., & Niknafs, A. (2014). NbClust: An r package for determining the relevant number of clusters in a data set. *Journal of Statistical Software*, *61*(6). https://doi.org/10.18637/jss.v061.i06

Dau, H. A., Keogh, E., Kamgar, K., Yeh, C.-C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., Yanping, Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, Gustavo, & Hexagon-ML. (2018). *The UCR time series classification archive*.

Davies, D. L., & Bouldin, D. W. (1979). A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-1*(2), 224–227. https://doi.org/10.1109/tpami.1979.4766909

Dunn, J. C. (1974). Well-separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, *4*(1), 95–104. https://doi.org/10.1080/01969727408546059

Gagolewski, M., Bartoszuk, M., & Cena, A. (2021). Are cluster validity measures (in) valid? *Information Sciences*, *581*, 620–636. https://doi.org/10.1016/j.ins.2021.10.004

Gurrutxaga, I., Muguerza, J., Arbelaitz, O., Pérez, J. M., & Martín, J. I. (2011). Towards a standard methodology to evaluate internal cluster validity indices. *Pattern Recognition Letters*, *32*(3), 505–515. https://doi.org/10.1016/j.patrec.2010.11.006

Halkidi, M., & Vazirgiannis, M. (2001). Clustering validity assessment: Finding the optimal partitioning of a data set. *Proceedings 2001 IEEE International Conference on Data Mining*, 187–194. https://doi.org/10.1109/icdm.2001.989517

Halkidi, M., Vazirgiannis, M., & Batistakis, Y. (2000). Quality scheme assessment in the clustering process. In *Principles of data mining and knowledge discovery* (pp. 265–276). Springer Berlin Heidelberg. https://doi.org/10.1007/3-540-45372-5_26

Kim, M., & Ramakrishna, R. S. (2005). New indices for cluster validity assessment. *Pattern Recognition Letters*, *26*(15), 2353–2363. https://doi.org/10.1016/j.patrec.2005.04.007

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Löning, M., Bagnall, A., Ganesh, S., Kazakov, V., Lines, J., & Király, F. J. (2019). *Sktime: A unified interface for machine learning with time series*. arXiv. https://doi.org/10.48550/ARXIV.1909.07872

Maulik, U., & Bandyopadhyay, S. (2002). Performance evaluation of some clustering algorithms and validity indices. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(12), 1650–1654. https://doi.org/10.1109/tpami.2002.1114856

Meilă, M. (2003). Comparing clusterings by the variation of information. In *Lecture notes in computer science* (pp. 173–187). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-45167-9_14

Partitioning around medoids (program PAM). (1990). In *Finding groups in data* (pp. 68–125).

John Wiley & Sons, Ltd. https://doi.org/10.1002/9780470316801.ch2

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. https://doi.org/10.48550/ARXIV.1912.01703

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.

Petitjean, F., Ketterlin, A., & Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, *44*(3), 678–693. https://doi.org/10.1016/j.patcog.2010.09.013

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, *20*, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7

Saitta, S., Raphael, B., & Smith, I. F. C. (2007). A bounded index for cluster validity. In *Machine learning and data mining in pattern recognition* (pp. 174–187). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-73499-4_14

Scikit-learn Extra. (n.d.). https://scikit-learn-extra.readthedocs.io/en/stable/.

Strauss, D. J., & Hartigan, J. A. (1975). Clustering algorithms. *Biometrics*, *31*(3), 793. https://doi.org/10.2307/2529577

Tavenard, R., Faouzi, J., Vandewiele, G., Divo, F., Androz, G., Holtz, C., Payne, M., Yurchak, R., Rußwurm, M., Kolar, K., & Woods, E. (2020). Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research*, *21*(118), 1–6. http://jmlr.org/papers/v21/20-091.html

Theodoridis, S., & Koutroumbas, K. (2009). Chapter 16 - cluster validity. In S. Theodoridis & K. Koutroumbas (Eds.), *Pattern recognition (fourth edition)* (Fourth Edition, pp. 863–913). Academic Press. https://doi.org/10.1016/B978-1-59749-272-0.50018-9

Tibshirani, R., Walther, G., & Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, *63*(2), 411–423. https://doi.org/10.1111/1467-9868.00293

Ward, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*(301), 236–244. https://doi.org/10.1080/01621459.1963.10500845

Xie, X. L., & Beni, G. (1991). A validity measure for fuzzy clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *13*(8), 841–847. https://doi.org/10.1109/34.85677