Descriptor

Patterns

reval: A Python package to determine best clustering solutions with stability-based relative clustering validation

Highlights

- Implementation of a stability-based relative validation technique for clustering
- Best number of clusters selection for multiple tasks
- Automation of stable clustering labels on new data via supervised learning
- Complement or alternative to internal validation measures for clustering

Authors

Isotta Landi, Veronica Mandelli, Michael V. Lombardo

Correspondence

landi.isotta@gmail.com

In brief

In the absence of labels that help categorize data, researchers rely on clustering algorithms to find meaningful partitions. Validating such partitions is a difficult task and widespread validation methods rely highly on the structure of the data at hand. This work presents a software implementation of a stability-based relative validation method for the selection of replicable stable solutions. It can be used with multiple clustering algorithms for different tasks, and it enables the automation of cluster labeling via classification methods.

Landi et al., 2021, Patterns *2*, 100228
April 9, 2021
https://doi.org/10.1016/j.patter.2021.100228

CellPress

Patterns

CellPress
OPEN ACCESS

Descriptor

reval: A Python package to determine best clustering solutions with stability-based relative clustering validation

Isotta Landi,[1,4,5,*] Veronica Mandelli,[1,2] and Michael V. Lombardo[1,3]

[1]Laboratory for Autism and Neurodevelopmental Disorders, Center for Neuroscience and Cognitive Systems @UniTn, Istituto Italiano di Tecnologia, Rovereto, Italy

[2]Center for Mind/Brain Sciences, University of Trento, Rovereto, Italy

[3]Autism Research Centre, Department of Psychiatry, University of Cambridge, Cambridge, UK

[4]Lead contact

[5]Twitter: @IsottaLandi

*Correspondence: landi.isotta@gmail.com

https://doi.org/10.1016/j.patter.2021.100228

THE BIGGER PICTURE reval is a Python package for stability-based relative clustering validation. It works with multiple clustering and classification algorithms, and as such, it enables the selection of best clustering solutions as the ones that replicate, via supervised learning, on unseen subsets of data. It is a tool that provides measures to evaluate clustering replicability and implements the automation of the labeling process. reval can be used as a complement or an alternative to internal validation measures, which highly rely on features inherent to a specific grouping solution, hindering the validation of replicable clusters.

Proof-of-Concept: Data science output has been formulated, implemented, and tested for one domain/problem

SUMMARY

Determining the best partition for a dataset can be a challenging task because of the lack of *a priori* information within an unsupervised learning framework and the absence of a unique clustering validation approach to evaluate clustering solutions. Here we present reval: a Python package that leverages stability-based relative clustering validation methods to select best clustering solutions as the ones that replicate, via supervised learning, on unseen subsets of data. The implementation of relative validation methods can contribute to the theory of clustering by fostering new approaches for the investigation of clustering results in different situations and for different data distributions. This work aims at contributing to this effort by implementing a package that works with multiple clustering and classification algorithms, hence allowing both the automation of the labeling process and the assessment of the stability of different clustering mechanisms.

INTRODUCTION

Clustering algorithms identify intrinsic subgroups in a dataset by arranging together elements that show higher pairwise similarities relative to other subgroups.[1] While their usage is relatively widespread, the lack of *a priori* information complicates the evaluation of clustering solutions. Attempts to address this challenge usually rely on the implementation of internal validation approaches, which focus on quantities and features inherent to a grouping solution.[2] Here we present the reval Python package (pronounced 'rɛvəl), which implements an approach for stability-based validation of clustering solutions described by Lange et al.[3] that allows for the identification and evaluation of partitions that best generalize to unseen data and the automation of the labeling process. In contrast to internal validation, relative validation methods[3,4] have the potential to transform cluster analysis into a model selection problem and help evaluate the best clustering solution (i.e., best number of clusters). The way these methods are conceived also offers the possibility to determine the extent to which a clustering solution generalizes to unseen data and hence to enable the replication of the data partition chosen. While a variety of software packages contain internal cluster validation methods and

measures, open-source software to easily implement the full potential of relative validation techniques are lacking.

Many methods are available to compute internal validation measures that help in determining the best number of clusters.[5–7] For example, the elbow method[5] selects the number of clusters for which the within-cluster variability decrement is minimal. Another popular method using internal criteria is the silhouette-based approach.[7] This method maximizes cluster cohesion and separation; that is, how similar an object is to other elements of the same cluster compared with elements of other clusters. Libraries and methods for the automated selection of the best number of clusters are available in both Python and R. The yellowbrick Python visual analysis and diagnostic tool suite[8] includes the implementation of the elbow method to determine the best number of clusters. In R, NbClust[9] is a popular library that compiles 30 different internal metrics and allows for users to compute all or a subset of metrics to be used in combination with a majority vote rule to select the optimal number of clusters. For relative validation approaches there are the clValid[10] and cstab[11] libraries, which apply stability-based relative validation models. clValid was designed to work with highly correlated high-throughput genomic data and computes stability measures comparing clustering solutions based on full data and data with a single column removed. cstab implements the selection of the best number of clusters via model-based and model-free clustering instability[12] using a bootstrap approach.

The reval package contributes to this landscape by implementing a stability-based approach that can be easily applied to different datasets using multiple clustering and classification algorithms. Built on top of the stability-based algorithm[3], reval applies a classifier trained on the best clustering solution to a test set, returning classification metrics that help interpret the generalization performance, guide the clustering process, and enable labeling replication. This tool can be used with internal measures to assess the underlying structure of a dataset to help avoid the risk of overfitting. With respect to clustering errors, internal and relative indices can exhibit similar behavior, with the advantage of the former being less computationally expensive.[13] However, in the case of complex models and clusters, an approach based on the minimization of prediction error may be particularly advantageous because internal indices tend to fail to correlate well with errors.[13]

Methods

Stability-based methods return the number of clusters that minimizes the expected distance between clustering solutions obtained for different datasets. Several options are available[14] to (1) generate the datasets (e.g., random subsampling of the original dataset[15], or adding random noise[16]); (2) compare clustering solutions (e.g., overlapping subsamples[15]); and (3) compute clustering distances (e.g., the consensus index by Vinh and Epps[17]). The method proposed by Lange et al.[3] has the advantage of transforming the unsupervised setting into a classification problem and guiding selection through the minimization of prediction error. First, a dataset is split into training and validation sets and then independently partitioned into clusters. Second, training set labels are used within supervised classification methods to learn how to best predict the labels. Applying the classification model to the validation set, the model's predicted labels are then compared with the actual clustering labels derived from the validation set. This procedure is repeated using cross-validation and the optimal number of clusters is identified corresponding to the maximum number of clusters that minimizes prediction error. Prediction performance is defined by the authors as the 0-1 loss in supervised classification[3,14], namely, the normalized Hamming distance. Nevertheless, other choices are possible; for example, Tibshirani and Walther[4] used prediction strength; that is, the proportion of observation pairs in the validation set that are assigned to the same cluster by both the clustering algorithm and the partition based on the training set centers.

Stability measure

The notion of stability by Lange et al.[3] is used to assess solutions of clustering algorithms based on the rationale that true clusters are those that can always be identified by a clustering algorithm when applied to different datasets from the same generating process. Formally, let $\mathcal{A}_k$ be a clustering algorithm with k the number of clusters, ϕ a classifier, and $(\mathbf{X},\mathbf{Y})$ the training set and clustering labels, i.e., $\mathcal{A}_k(\mathbf{X}) = \mathbf{Y}$. After training ϕ on $(\mathbf{X},\mathbf{Y})$, both the clustering algorithm and trained classifier are applied to a separate dataset $\mathbf{X}'$. The distance between the two solutions is the normalized Hamming distance:

$$d_{S_k}(\phi(\mathbf{X}'),\mathbf{Y}') = \min_{\sigma\in S_k}\frac{1}{n}\sum_{i=1}^{n} 1_{\{\sigma(\phi(X'_i))\neq Y'_i\}} \qquad \text{(Equation 1)}$$

with S_k the set of all possible permutations of k elements. Supervised labels are permuted to overcome the non-uniqueness of clustering labeling and σ is the permutation that minimizes the solutions dissimilarity. Averaging out the distance between any pair of partitions $\mathbf{X}$, $\mathbf{X}'$ from Equation 1 we define the stability index of the clustering algorithm as:

$$\mathcal{S}(\mathcal{A}_k) = E_{\mathbf{X},\mathbf{X}'}\left[d_{S_k}(\phi(\mathbf{X}'),\mathbf{Y}')\right] \qquad \text{(Equation 2)}$$

The stability index ranges in $[0,1]$, with lower values indicating more stable and reproducible solutions.[3] Because this measure scales with the number of clusters, the measure suggested by the authors is the normalized stability $\overline{\mathcal{S}}_k$, i.e., the stability from Equation 2 normalized for the stability of random labeling $\mathcal{R}_k$.

Reval algorithm

The algorithm implemented in reval allows the user to (1) automatically select the number of clusters for a dataset by minimizing validation stability, via repeated cross-validation (see Algorithm 1); and (2) compute classification performance obtained when generalizing the solution to a held-out dataset (see Algorithm 2). An overview of the framework is reported in Figure 1.

A dataset $\mathbf{X}$ is first split into training $\mathbf{X}_{tr}$ and test $\mathbf{X}_{ts}$ sets and a clustering $\mathcal{A}$ and classifier ϕ are selected. Let n_{fold} be the number of folds for cross-validation and n_r the number of repetitions. In Algorithm 1, we refer to $[n_{\text{fold}}]$ and $[n_r]$ as the sets of indices corresponding to fold splits. Moreover, let n_{rnd} be the number of random labeling iterations, and k the number of clusters in set K. In Algorithm 1 we indicate with $\mathbf{X}^{ij}_{itr}$ and $\mathbf{X}^{ij}_{val}$ the internal training and validation splits of training set $\mathbf{X}_{tr}$, respectively, for cross-validation ith fold split at the jth shuffled repetition. These correspond to $\mathbf{X}$ and $\mathbf{X}'$ sets introduced in the stability measure section. With $(K\times[n_{\text{fold}}]\times[n_r])$ we indicate the Cartesian product of

Algorithm 1. Return number of clusters that minimizes normalized stability

Input: $\mathbf{X}_{tr}$, $\mathcal{A}$, ϕ, K, n_{fold}, n_r, n_{rnd}
Result: k^*, $\overline{\mathcal{S}}_{k^*}$
for $(k,i,j)\in(K\times[n_{\text{fold}}]\times[n_r])$ do
Find clustering solution $\mathcal{A}_k(\mathbf{X}^{i_j}_{itr})=\mathbf{Y}^{i_j}_{itr}$ and train ϕ on $(\mathbf{X}^{i_j}_{itr},\mathbf{Y}^{i_j}_{itr})$;
Compute $\phi_{i_j}(\mathbf{X}^{i_j}_{val})$ and $\mathcal{A}_k(\mathbf{X}^{i_j}_{val})=\mathbf{Y}^{i_j}_{val}$;
Select permutation $\overline{\sigma}_{i_j}\in S_k$ that yields to minimum dissimilarity $d_{\overline{\sigma}_{i_j}}(\phi_{i_j}(\mathbf{X}^{i_j}_{val}),\mathbf{Y}^{i_j}_{val})$;
for $r=1,\ldots,n_{rnd}$ do
Train ϕ on $(\mathbf{X}^{i_j}_{itr},\mathcal{R}_k(\mathbf{Y}^{i_j}_{itr}))$;
Compute $d_{\overline{\sigma}_{r_j}}\left(\phi_{r_j}\left(\mathbf{X}^{i_j}_{val}\right),\mathbf{Y}^{i_j}_{val}\right)$ as before;
end
Compute $d_{i_j}=d_{\overline{\sigma}_{i_j}}/\text{Avg}_{r=1}^{n_{rnd}}(d_{\overline{\sigma}_{r_j}})$
Compute normalized stability $\overline{\mathcal{S}}_k=\text{Avg}_{j=1}^{n_r}\text{Avg}_{i=1}^{n_{\text{fold}}}d_{i_j}$;
end
Return k^* s.t. max $\text{argmin}_{k\in K}\overline{\mathcal{S}}_k$.
Input parameters: training dataset ($\mathbf{X}_{tr}$), clustering algorithm ($\mathcal{A}$), classification algorithm (ϕ), set of number of clusters to evaluate (K), number of cross-validation folds (n_{fold}), number of cross-validation repetitions (n_r), and number of random labeling repetitions (n_{rnd}). With × we indicate the Cartesian product of sets, i.e., the set of all possible ordered combinations of elements in the sets, which is equivalent to a nested for-loop.

the sets of number of clusters and repeated cross-validation splits. The fitted model becomes the one trained on $\mathbf{X}_{itr}$ that returns the maximum number of clusters with minimum stability. That model can then be used within Algorithm 2 for generalization on the test set.

Among clustering methods that work within reval, hierarchical density-based spatial clustering of applications with noise (HDBSCAN)[18] does not need any assumption on the number of clusters. Hence, we do not need to iterate over different number of clusters to select the best solution. Instead, normalized stability is computed within the repeated cross-validation loops that return the same number of clusters.

Technical details

The reval package has 4 core modules and additional functions that can be found in the *utils* file.

- `relative_validation`. This module includes training and test methods that return misclassification errors obtained by comparing classification labels and clustering labels. It also includes the random labeling method, which allows users to compute the asymptotic misclassification rate.

- `best_nclust_cv`. This module implements repeated cross-validation and returns the best clustering solution together with normalized stability scores, obtained from the average of the misclassification scores divided by the asymptotic misclassification rate. Repeated cross-validation leads to unbiased stability estimates and it can also be performed stratifying the repeated randomized splits according to a desired variable. To control for the size imbalance that derives from cross-validation, we initialized the repeated cross-validation loop to a 1×2 schema as default. Users can change this configuration according to dataset size and available stratifiers, which can be useful to overcome imbalance issues. The evaluation method applies the fitted model with the returned number of clusters to the held-out dataset and returns accuracy (ACC). Other

Algorithm 2. Test best solution on unseen data

Input: $\mathbf{X}_{tr}$, $\mathbf{X}_{ts}$, k^*, $\mathcal{A}_{k*}$, ϕ
Result: classification accuracy
Find clustering solution $\mathcal{A}_{k^*}(\mathbf{X}_{tr})=\mathbf{Y}_{tr}$ and train ϕ on $(\mathbf{X}_{tr},\mathbf{Y}_{tr})$;
Compute $\phi_{k^*}(\mathbf{X}_{ts})$ and $\mathcal{A}_{k^*}(\mathbf{X}_{ts})=\mathbf{Y}_{ts}$;
Compute the accuracy (ACC) with permuted clustering labels for consistency between training and test sets:

$$\text{ACC}=\max_{\sigma\in\overline{S}_{k^*}}\text{Avg}_{i=1}^{|\mathbf{X}_{ts}|}1_{\left\{\phi\left((X_{ts})_i\right)=\sigma\left((Y_{ts})_i\right)\right\}}$$

Return ACC$\in[0,1]$.
Input parameters: training dataset ($\mathbf{X}_{tr}$), testing dataset ($\mathbf{X}_{ts}$), best number of clusters selected with cross-validation (k^*), clustering algorithm with best number of clusters fixed ($\mathcal{A}_{k^*}$), and classifier (ϕ).

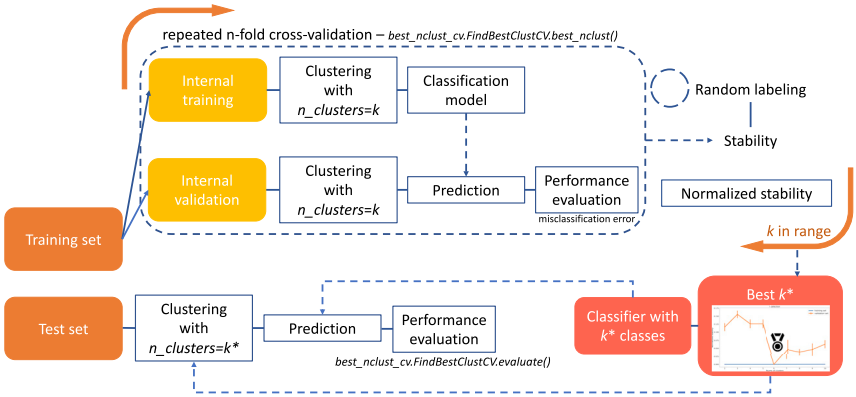

Figure 1. reval implementation overview

Repeated cross-validation procedure is included within the dashed circle and it is repeated for different numbers of clusters k as indicated by the orange arrows (Algorithm 1). The clustering algorithm with number of clusters k^*, i.e., the maximum value that minimizes normalized stability, is evaluated on a held-out dataset (Algorithm 2).

metrics, such as Matthews correlation coefficient (MCC)[19], F1 score, precision, and recall scores, can also be computed (see *utils* file).

- `param_selection`. This module enables hyperparameter tuning to select the best configuration of classifier/clustering (`SCParamSelection` class) and the parameters within clustering and classifier themselves (`ParamSelection` class). Best parameters are those that report minimum normalized stability. If the number of true classes is available, this module also returns the best solution that correctly identifies the true number of clusters, if it exists.
- `visualization`. This module includes the function to plot cross-validation performance metrics with 95% confidence intervals for a varying number of clustering solutions. The threshold of random labeling stability can be displayed to visually investigate model performance.

As suggested by Lange et al.[3], we used the Kuhn-Munkres algorithm[20,21] to obtain the label permutation that minimizes misclassification error. However, differently from the work of Lange et al.[3], reval permutes the clustering labels instead of the classification labels, i.e., the normalized Hamming distance in Equation 1 becomes:

$$d_{S_k}(\phi(\mathbf{X}'), \mathbf{Y}') = \min_{\sigma \in S_k} \frac{1}{n} \sum_{i=1}^{n} 1_{\{\phi(X_i') \neq \sigma(Y_i')\}}$$

This approach allows the test set to preserve the partition structure of the training set, to better investigate results replicability and to aid visual comparison. Figure 2 shows the rationale behind the need to permute clustering labels when training a classifier within reval. We simulated 3 Gaussian blobs and divided them into training ($N = 20$) and test ($N = 10$) sets. Figure 2A shows clustering labels for the training set and Figure 2B the clustering labels for the test set. Figure 2C shows what might happen when training a classifier on the training set labels and then predicts labels for the test set. Because 2 out of 3 classes show label discordance, the trained classifier fails to correctly predict the classes. Nevertheless, if we permute class labels 0 and 1 in the example, the trained classifier will correctly identify all 3 classes on the test set returning minimum prediction error and consistent label ordering is preserved. The `kuhn_munkres_algorithm()` function can be found in *utils* file.

A more thorough description of the code and its usage can be found at https://reval.readthedocs.io/en/latest/code_description.html. reval mainly works with the scikit-learn Python library for machine learning.[22] In particular, among clustering methods, users can select those with number of clusters parameter, i.e., k-means, hierarchical clustering, and spectral clustering, but also density-based clustering HDBSCAN[18] from the hdbscan library. Moreover, any classifier from scikit-learn can be selected.

Algorithm complexity

We report here the complexity analysis of the 2 core methods included in `best_nclust_cv` module. In particular, we focus on `best_nclustcv()`, which enables the selection of the best number of clusters via repeated cross-validation, and `evaluate()`, which implements the testing of the best solution on held-out datasets. The `best_nclustcv()` method includes sequential calls to train (`train()`), test (`test()`), and random labeling (`rndlabels_traineval()`) methods from the `relative_validation` module, whereas `evaluate()` sequentially calls the train and test functions.

The complexity of the training method is led by the sum of the costs of the chosen classification (i.e., $O(\mathrm{C})$) and clustering (i.e., $O(\mathrm{G})$) algorithms, which depend on the number of data samples and features (see Table 1). The complexity of the `test()` method mainly depends on $O(\mathrm{G})$ in that only prediction is performed for the classifier. The random labeling algorithm increases $O(\mathrm{C})$ by a factor of n_{rnd} (i.e., the number of random labeling iterations). The overall complexity to perform cross-validation and evaluation depends on the number of calls to the relative validation module functions and their intrinsic cost. To perform cross-validation and compute normalized stability, we need to set the following parameters: $n_C = |K|$ (i.e., the cardinality of the set with the different number of clusters to try); n_{fold} and n_r, which correspond to the number of cross-validation folds and repetitions, respectively; and n_{rnd}. In conclusion, the cost of `best_nclustcv()` is equal to $O(n_C \cdot n_{\mathrm{fold}} \cdot n_r \cdot (O(\mathrm{G}) + O(n_{rnd} \cdot O(\mathrm{C}))))$ and the complexity of `evaluate()` is $O(\mathrm{C}) + O(\mathrm{G})$.

We run the methods considered on simulated datasets to empirically investigate the execution times. Simulations were run on a MacBook Pro 2020 with a 2.3 GHz Quad-Core Intel Core i7 processor and 32 GB of RAM. The complexity of clustering algorithms and state-of-the-art classifiers that can be used with reval can be found in Table 1. Figures 3A and 3B show the execution times of `best_nclustcv()` and `evaluate()`, respectively, on a 2-blob dataset of 100 samples and 10 features with different combinations of classifiers and clustering algorithms. n_C is set at 5, n_{fold} and n_r are 2 and 10

Table 1. Algorithm complexity

Algorithm	Complexity	Problem
HDBSCAN	$O(N\log N)$	clustering
K-means	$O(kNT)$	clustering
Agglomerative	$O(kN^2)$	clustering
Spectral	$O(N^3)$	clustering
LR	$O(pN)$	classification
KNN	$O(pN)$	classification
Support vector machine	$[O(pN^2); O(pN^3)]$	classification
RF	$O(pN^2 n_{trees})$	classification

N = number of samples; p = number of features
K-means: k = number of clusters; T = number of iterations
Agglomerative: k = number of clusters
RF: n_{tress} = number of trees in the forest
Clustering and classification algorithms available within reval package from the scikit-learn and hdbscan libraries.

respectively, and n_{rnd} is equal to 10. Default parameters were used for all classifiers and clustering algorithms. It is straightforward to observe that execution times largely depend on the chosen algorithms, with HDBSCAN the least expensive and spectral clustering the most expensive choice among clustering techniques, irrespective of classifier, and random forest the most expensive choice among classifiers. Overall, the execution time of the `evaluate()` method is reduced compared with repeated cross-validation. The package implementation also includes a multiprocessing feature that speeds up computations for repeated cross-validation but not for the evaluation method (see Figure S1A), where 7 jobs are simultaneously run. Figure 3C shows the execution times with varying number of samples and features when sequentially running `best_nclustcv()` and `evaluate()` with KNN classifier and k-means clustering. We can observe a moderate increase for low number of samples ($<10^3$) with varying number of features and a steep growth for 10^3 samples. Corresponding execution times with multiprocessing can be observed in Figure S1C.

RESULTS

Technical validation

To investigate reval performance, first we investigate group detection in simulated non-overlapping Gaussian blobs, and the handwritten digits dataset from the Modified National Institute of Standards and Technology (MNIST) digit recognition database (http://yann.lecun.com/exdb/mnist/). Then, we leverage the `SCParamSelection` class to determine the combination of classifier and clustering algorithms that best identifies the number of classes for 19 different datasets from the University of California, Irvine (UCI) Machine Learning repository.[23]

Blobs dataset

To provide a simple example of how to use reval, we generated 5 isotropic Gaussian blobs for a total of 1,000 samples with 2 features. To run the example, refer to Code S1-S6 in the blobs dataset section of the supplemental material. The dataset was first split into training and test sets (70/30%). Then we selected KNN with the number of neighbors equal to 15 and k-means with Euclidean distance, as suggested by the experiment reported in the algorithm selection section. We then run the stability-based algorithm with a 10×2 repeated cross-validation framework, with 10 random labeling repetitions, and the number of clusters varying from 2 to 6. We evaluated the solution found on the test set and report ACC and MCC scores as metrics. Last, we report the adjusted mutual information (AMI) score to compare true labels and predicted labels on the test set. AMI external score measures the similarity of 2 labelings of the same data, irrespective of label order and ranges from [0, 1], with a perfect match equal to 1. We also determined the best number of clusters maximizing and minimizing silhouette and Davies-Bouldin internal measures, respectively. Davies-Bouldin index[6] measures clusters separation and "tightness" with Euclidean distance. Lower indices correspond to better solutions. We computed the metrics independently on the training and test sets and return the best number of clusters found with k-means.

The normalized stability for varying number of clusters in Figure 4 shows that the model perfectly identified 5 clusters with 0.0 normalized stability. The comparison between true and clustering labels can be observed in Figure S3. The ACC and MCC scores on the test set are equal to 1.0 and the AMI is equal to 1.0. The maximum silhouette score is 0.83 on both training and test sets with the correct number of clusters and AMI scores equal to 1.0. The Davies-Bouldin index results do not replicate between training and test sets. The index is equal to 0.23 with 4 clusters on the training set and to 0.23 with 5 clusters on the test set. AMI scores are 0.91 for training and 1.0 for testing.

MNIST dataset

For the real-world dataset example, we considered the handwritten digits MNIST dataset (http://yann.lecun.com/exdb/mnist/), which includes 70,000 samples corresponding to 28×28 images of digits from 0 to 9. When flattened, each sample has 784 features. The dataset has 10 classes, with ~7,000 samples for each class.

First, we split the dataset into training and test sets of 60,000 and 10,000 samples, respectively. Then, we preprocessed the dataset with uniform manifold approximation and projection (UMAP)[24] for dimensionality reduction with 2 components. We run HDBSCAN clustering, as suggested in https://umap-learn.readthedocs.io/en/latest/clustering.html, with different classifiers and selected the best configuration. In particular, we considered k-nearest neighbors (KNN) with the number of neighbors equal to 30, and support vector machines (SVMs), random forest (RF), logistic regression (LR) with default parameters from scikit-learn library. The HDBSCAN algorithm was initialized with minimum samples equal to 10 and minimum cluster size equal to 200. The relative clustering validation procedure was run with 10 repetitions of 2-fold cross-validation, and number of random labeling iterations equal to 10. Because HDBSCAN does not need the number of clusters specified *a priori*, the normalized stability is computed averaging the misclassification error over the solutions that return the same number of clusters. We selected the one that minimizes the normalized stability and the trained classifier applied to test set is the one trained on

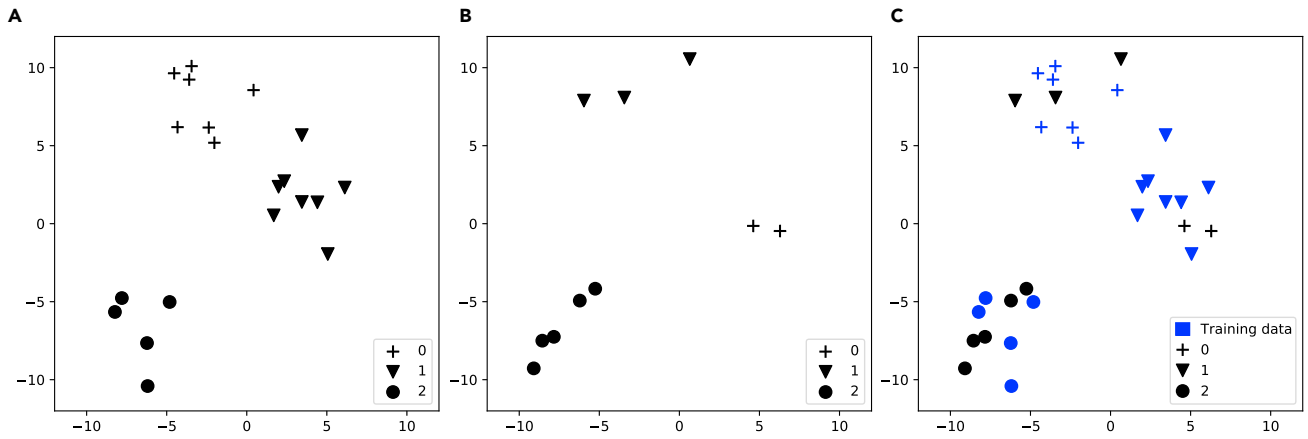

Figure 2. Relabeling practice

Clustering labels on the training (A) and test (B) sets. Differences in labeling between training and test sets are displayed in (C). The labeled points on which the classifier is trained are shown in blue, the labeled points whose classes we want to predict are in black.

the corresponding cross-validation fold. Because we are not preselecting the number of clusters, results might differ between training and test sets.

The best classifier-clustering combination is RF and HDBSCAN, which correctly identifies 10 clusters on the training set, with a misclassification error equal to 0.06 (0.06) and AMI score equal to 0.91 (see Figure 5). On the test set, it identifies 9 clusters with AMI equal to 0.87. When the experiment was run with internal validation measures, we obtained 10 clusters in training and 9 clusters during testing, with AMI scores equal to 0.88 and 0.87, respectively, for both silhouette and Davies-Bouldin measures. In Figure 6 and by the comparison of training set AMI scores between reval and internal measures, we observe that internal measures fail to detect the actual digit classes during training, whereas reval with HDBSCAN and RF successfully identifies them (see Figures 6C and 6D), with AMI score of 0.91. On test set the result is the same among all methods, as demonstrated by equal AMI scores (see Figure S4). Based on these results, we can speculate that clustering on a subset of data ($N = 35,000$) better detects the digits classes.

It is worth noting that the classifier performance highly depends on the clustering solution in that it tends to overfit to the training dataset. To guide the choice of a classifier, users should first select an appropriate clustering algorithm, then select the classifier that minimizes stability and that reports the best performance on the held-out dataset. In case of equal or very similar outcomes, algorithm complexity should guide the classifier selection.

Algorithm selection

We considered 19 datasets from the UCI Machine Learning repository,[23] including the test set of the handwritten digits dataset that can be found in scikit-learn toy datasets (https://scikit-learn.org/stable/datasets/index.html). In Table 2 we report the dataset names along with the number of samples, features, and inherent classes. We applied the relative validation algorithm with

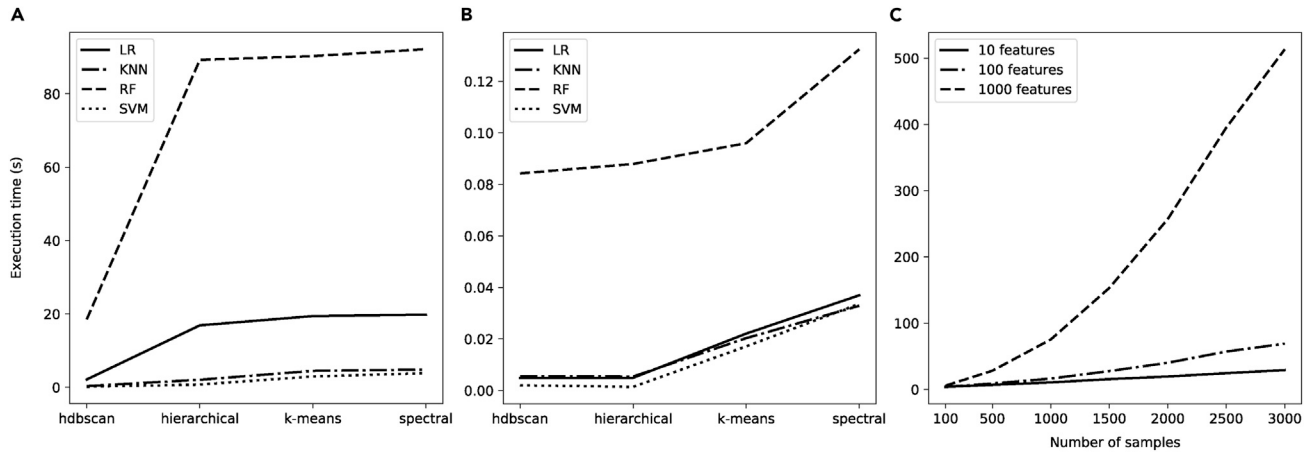

Figure 3. Execution times

Different combinations of classification and clustering algorithms applied to blobs dataset with 100 samples and 10 features. `best_nclustcv()` (A); `evaluate()` (B); sequential calls to `best_nclustcv()` and `evaluate()` (C) with KNN and k-means for blobs dataset with varying combinations of samples and features.

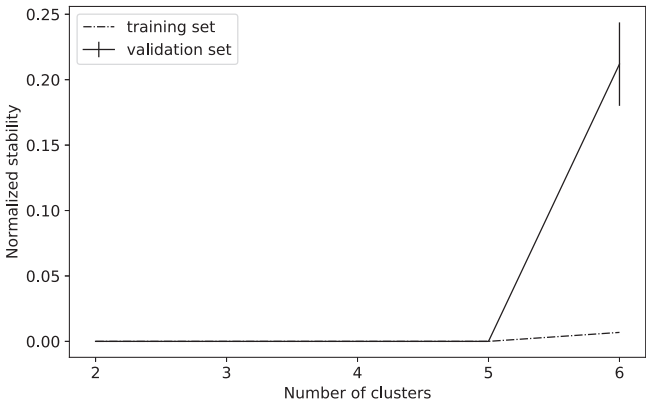

Figure 4. reval performance for blobs dataset

Solid line represents the validation normalized stability with 95% confidence intervals. Dashed line shows stability during training.

different combinations of classifier and clustering algorithms. In particular, for clustering we selected hierarchical clustering with Ward's method and Euclidean distance, k-means clustering with Euclidean distance, and HDBSCAN with minimum cluster size equal to 5, and Euclidean distance. Among classifiers, we opted for KNN with 1 neighbor and Euclidean distance, RF classifier with 100 estimators, SVM with $C = 1$ and $\gamma = \frac{1}{\text{N samples}}$, and LR. We did not consider spectral clustering because of its computational cost (see Table 1). To improve the performance, in addition to raw datasets, we repeatedly run the experiments after preprocessing with (1) standard scaler, which removes the mean and scales to unit variance; (2) UMAP algorithm[24] for dimensionality reduction; and (3) standard scaler and UMAP. UMAP parameters are chosen according to those suggested in the documentation (https://umap-learn.readthedocs.io/en/latest/clustering.html). We ran the algorithm with 10 replications of 2-fold cross-validation and 10 random labeling iterations. The number of clusters ranges from 2 to $n_{(classes)} + 2$, where n_{classes} is the number of classes for each dataset.

Table 3 reports the best solutions selected as those reporting minimum stability along with the correct number of clusters. If no experiment identified the correct number of classes, we chose, among the solutions with minimum stability for each preprocessed dataset, the one with maximum AMI on test set. For comparison, we computed the silhouette score and Davies-Bouldin index[6] internal measures independently on both training and test sets with the clustering algorithm selected by the stability-based approach and with the same varying number of clusters. We report in Table 3 the best solutions as those that maximize and minimize those measures, respectively, and AMI scores to evaluate the similarity of true and cluster labels on test sets.

We observe that the stability-based approach identified the correct number of classes in 68% of cases ($N = 13$). Moreover, 6 out 13 experiments selected k-means clustering and KNN classifier as the best choice. Of these, only 1 utilized raw data with no preprocessing, whereas the rest required either UMAP-preprocessed and/or scaled datasets. Because k-means works well with center-based spherical clusters and usually cannot find a good representation if clusters are very elongated or have complicated shapes, data preprocessing can relax this issue. From this experiment, it emerged that UMAP can be used as a preprocessing tool in this sense, in that it unwraps manifolds to find manifold boundaries. Nevertheless, because each dataset has its own intrinsic characteristics, preprocessing steps must be chosen with care to avoid breaking clusters into several erroneous small spherical clusters (see example with k-means at https://umap-learn.readthedocs.io/en/latest/clustering.html). It has been observed[14] that, if the number of clusters k is too large with respect to the true clusters, the k-means algorithm tends to be unstable. On the contrary, if k is smaller or equal to the true number of clusters, the algorithm tends to be stable. Therefore, it is argued that k-means stability depends on the number of true clusters in the dataset, which should be on the order of 10 to provide stable solutions. This seems to hold when we look at the UCI datasets with ≥ 10 classes for which k-means is often selected as the best clustering algorithm, although it returns a smaller number of clusters with respect to true classes. For this reason, despite KNN/k-means providing the best algorithm configuration in more than half the experiments, the choice of a classifier/clustering should be done carefully, taking into account the dataset dimension and the computational cost of the algorithms.

The number of clusters selected with the silhouette score and Davies-Bouldin index (see Table 3) is equal to that reported by reval. The exception here was the *iris* dataset,[25] whereby reval selects 3 clusters during training and 2 during testing with RF and HDBSCAN, whereas internal measures result in 4- and 2-cluster solutions. Nevertheless, the validation stability of the relative approach is equal to 0.33 (0.19), suggesting that the partition does not generalize well because the solution is not stable. If we could only rely on internal measures, we would have failed to acknowledge the quality of the solution found. Generally, because 2 out of 3 classes are not linearly separable and the data are displayed in 2 separate groups[25] the *iris* dataset is not a good candidate for clustering and only the relative validation approach can clearly show that.

More generally, when comparing the AMI scores, we have no ability to say how well the results are actually doing. On the contrary, we have a sense of how good the clustering is with relative

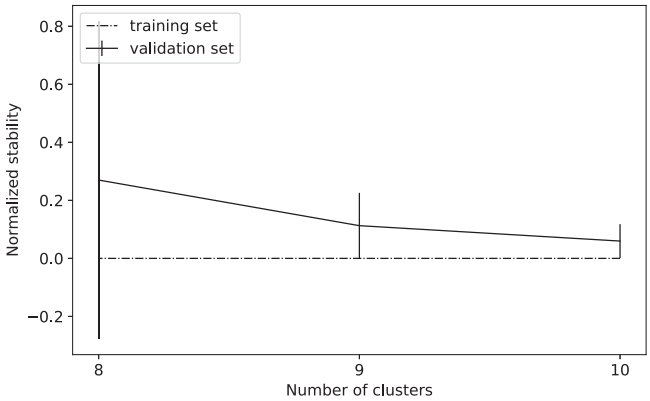

Figure 5. reval performance for MNIST dataset with RF and HDBSCAN algorithms

Solid line represents the validation normalized stability with 95% confidence intervals. Dashed line shows training stability.

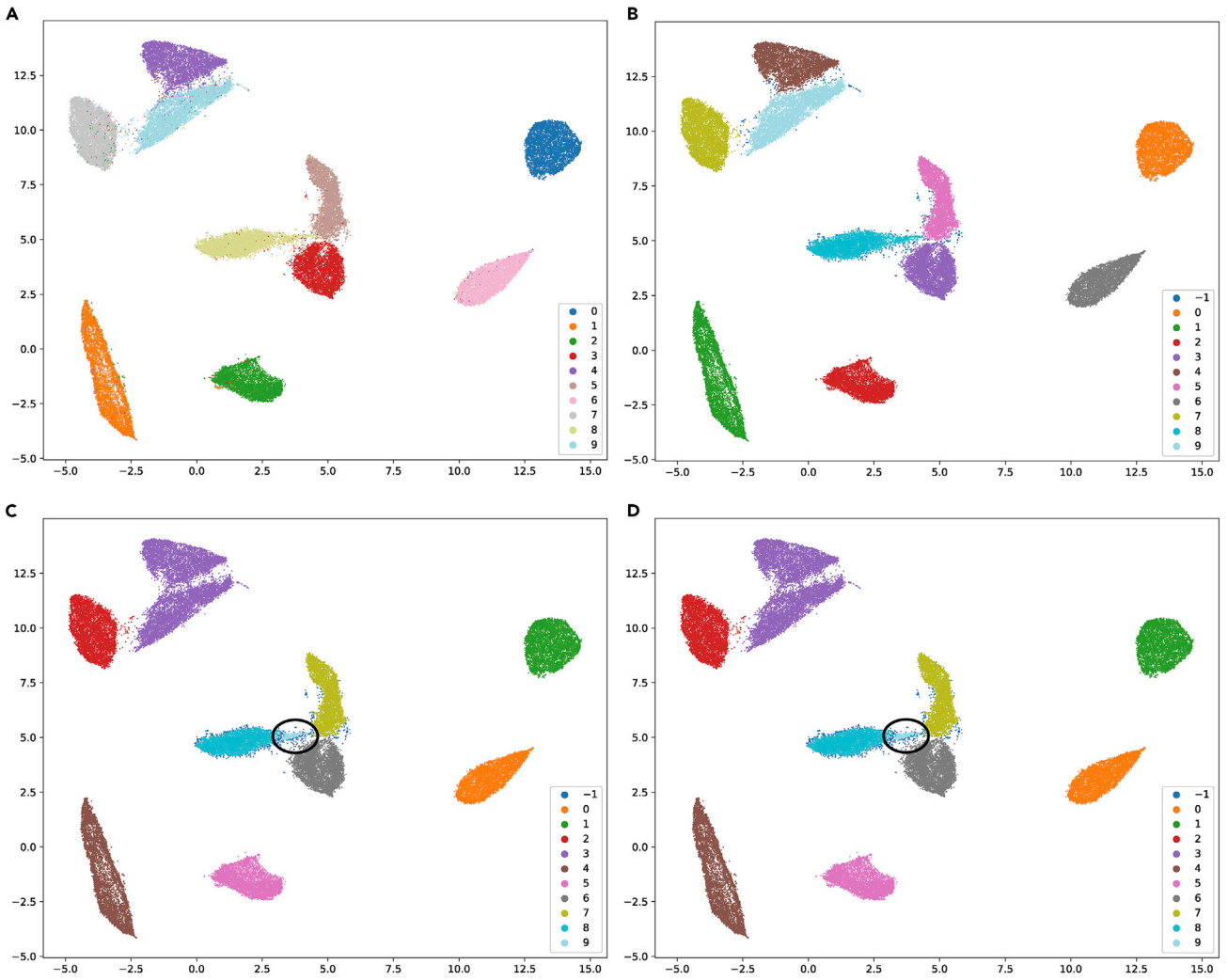

Figure 6. MNIST cluster visualization

Training set UMAP visualization for true labels (A); relative validation labels (B); silhouette score labels (C); Davies-Bouldin index labels (D). For internal measures we circled the erroneous cluster identified.

validation through the generalization process. See the example of the climate dataset, which has a validation normalized stability of 0.20 (0.04), indicating poor generalization, and a silhouette score on test set of 0.49, although both methods provide the same clustering solution on test set given that AMI score is equal to −0.005 in both cases.

In conclusion, grid search for classifiers and clustering methods is easily and effectively implemented with reval and can be handy to avoid *a priori* selection of a classifier. Furthermore, the stability-based approach helps evaluate the goodness of a clustering solution by means of its generalization process. This information about generalization is absent with internal measures.

Stability regime to guide cluster selection

Ben-Hur et al.[15] presented a stability-based method with data subsampling and reported on the risk of underestimating/overestimating the true number of clusters. Introducing prediction strength computed via repeated cross-validation, Tibshirani et al.[4] linked unsupervised to supervised learning in an attempt to overcome the best cluster estimation issue. The reval implementation moves forward adding the evaluation of the best solution on unseen data, an approach that is particularly suitable for datasets with large sample size. While large sample sizes were less common in the past, dataset size has increased substantially over time and reval can be particularly important and well suited for modern data science contexts. Solution generalizability can be leveraged to select the best number of clusters not only based on validation metrics (e.g., prediction strength greater than 0.8/0.9[4]) but also on the performance of the solution applied to a new set of data. In this way, we are able to compare test set distribution with the one on which the result was based, hence reinforcing the decision. Selecting the number of clusters that best generalizes to new data (i.e., investigation of the stability regime) holds promise for overcoming the underestimation issue. To give a better sense of how this works in practice, we

Table 2. Benchmark datasets from the UCI Machine Learning repository

Dataset	Samples	Features	Classes
Handwritten digits	1,797	64	10
Yeast	1,484	8	10
Banknote	1,372	4	2
Biodegradation	1,055	41	2
Transfusion	748	4	2
Breast cancer Wisconsin (WI)	683	9	2
Urban land cover	675	147	9
Climate	540	18	2
Forest type	523	27	4
Wholesale	440	7	3
Movement	360	90	15
Ionosphere	351	34	2
Liver disorders	345	5	2
Leaf	340	14	30
Ecoli	336	7	8
Glass	214	9	6
Seeds	210	7	3
Parkinsons	195	22	2
Iris	150	4	3

present a case study in the context of autism research, where clustering solutions within a stability regime could be further investigated.

Autism spectrum conditions (ASCs) are characterized by difficulties in social communication alongside heightened restricted and repetitive behaviors.[26] The spectrum of affected individuals with ASC is highly heterogeneous and this heterogeneity is present at multiple levels, from genome to phenome, and can coexist with differing levels of severity and comorbidities.[27,28] Given the high level of heterogeneity, data-driven clustering could be a promising approach to isolating different types of autisms. To split ASC into data-driven subtypes, we applied reval to clinical data obtained from the National Database for Autism Research (NDAR; https://nda.nih.gov/). NDAR is a database that includes a heterogeneous collection of de-identified human subjects' data for autism research. We focus here on clinical behavioral data from the Vineland Adaptive Behavior Scales (VABS).[29,30] Within the VABS, there are 3 domain total scores for communication, living skills, and socialization skills. Using these 3 domain total scores with UMAP preprocessing, we trained the stability-based model on 420 subjects (mean age 41.65 (17.91) months, female/male counts 109/311). As for analysis choices, we ran this analysis with 100×3 repeated cross-validation and 100 iterations of random labeling, with number of clusters ranging from 2 to 10, using k-means clustering, and a KNN classifier with number of neighbors equal to 15.

From the performance plot in Figure 7, we find that both a 2-cluster and 3-cluster solution results in small stabilities (2-cluster solution (error): 0.027 (0.004); 3-cluster solution [error]: 0.036 (0.003)) and thus form a stability regime whereby either solution might be a promising solution to follow up with future work. Based on the minimization of the normalized stability measure, the default behavior when using reval would be to select 2 as the best number of clusters. However, upon evaluation of these solutions on the unseen test set (n = 344 subjects; mean age 43.12 (17.04) months, female/male counts 90/254), we find that both solutions reach 94% accuracy, further confirming the presence of a stability regime whereby more than 1 solution might be a plausibly good model for follow-up work. The generalization performance alongside visual inspection (see Figure S5) indicates that the default selection of a 2-cluster solution would possibly underestimate the true number of clusters.[4] Thus, based on the stability regime present here, we could select 3 as the true number of clusters since the stability differences are likely negligible and because both 2- and 3-cluster solutions generalize equally well. In practice, we would ultimately follow up with examination of both 2- and 3-cluster solutions and utilize other datasets to better understand which solution might be most illuminating for decomposing clinical and biological heterogeneity of importance for autism research.

To contrast the reval stability-regime results here with internal measures, we would have obtained 2-cluster solutions for both silhouette (scores of 0.41 and 0.42 on training and test respectively) and Davies-Bouldin measures (0.86 on both training and test sets). If we force the number of clusters to 3, we obtain lower silhouette scores (i.e., 0.35 in both training and test sets) and higher Davies-Bouldin indices (i.e., 0.93 and 0.95, respectively). In this example, internal measures do not reveal a regime of possible cluster solutions. Given the additional lack of information about generalization from internal measures, such a regime might be easily missed. This example illustrates a real-world example in data science for how relative validation implemented with reval may reveal insights regarding regimes of clustering solutions that may be missed with internal validation approaches.

DISCUSSION

In this work, we introduce the reval package for relative clustering validation and describe how it can be utilized, as well as providing examples for how it performs in simulations and several real datasets. In many cases, reval successfully identifies the correct number of clusters and confers several other advantages over and above other internal validation approaches. In particular, from the examples reported, it is straightforward to observe that the numbers of clusters identified through reval and internal measures usually do not differ, and that the clustering solutions report the same AMI scores compared with true labels. Internal measures have the advantage of being less computationally expensive; nevertheless, they do not inform on the generalization process, nor do they grant the possibility to generate labels on unseen data. This happens because cluster labels are obtained from in-sample measures. On the contrary, reval relies on out-of-sample stability of the solutions and it allows estimation of whether the clustering used is successful in determining partitions. In conclusion, compared with internal validation measures, reval is able to report the extent to which different clustering solutions fit to the data at hand and how well those solutions may generalize or replicate on unseen data.

Moreover, because reval works with multiple clustering algorithms, it can facilitate a more thorough investigation of

Table 3. Best combinations of classifiers and clustering methods for reval with grid search applied to benchmark datasets from the UCI Machine Learning repository

		reval					Silhouette		Davies-Bouldin	
Dataset	Classes	Clusters	Best model	Preprocessing	Stability (error)	AMI	Clusters	AMI	Clusters	AMI
Handwritten digits	10	10	KNN/k-means	UMAP	0.0	0.76	10/10	0.76	10/10	0.76
Yeast	10	7	RF/k-means*	scaled + UMAP	0.05 (0.01)	0.25	4/4	0.25	4/4	0.25
Banknote	2	2	SVM/HC	scaled + UMAP	0.003 (0.006)	0.93	2/2	0.93	2/2	0.93
Biodegradation	2	2	KNN/k-means	Raw	0.03 (0.006)	− 0.002	2/2	− 0.002	2/2	− 0.002
Transfusion	2	2	KNN/k-means	UMAP	0.0	0.005	2/2	0.005	2/2	0.005
Breast cancer (WI)	2	2	SVM/k-means	raw	0.03 (0.01)	0.76	2/2	0.76	2/2	0.76
Urban land cover	9	3	KNN/k-means*	scaled + UMAP	0.006 (0.003)	0.45	3/3	0.45	3/3	0.45
Climate	2	2	KNN/k-means	scaled + UMAP	0.20 (0.04)	− 0.005	2/2	− 0.005	2/2	− 0.005
Forest type	4	4	KNN/HC	raw	0.35 (0.11)	0.55	4/4	0.55	4/4	0.55
wholesale	3	3	SVM/k-means	UMAP	0.07 (0.03)	0.002	3/3	0.002	3/3	0.002
Movement	15	6	KNN/HC*	UMAP	0.05 (0.02)	0.42	6/6	0.42	6/6	0.42
Ionosphere	2	2	SVM/k-means	raw	0.01 (0.009)	0.21	2/2	0.21	2/2	0.21
Liver disorders	2	2	KNN/k-means	UMAP	0.11 (0.04)	0.03	2/2	0.03	2/2	0.03
Leaf	30	3	RF/k-means*	scaled	0.19 (0.03)	0.22	3/3	0.22	3/3	0.22
Ecoli	8	2	KNN/k-means*	UMAP	0.0	0.48	2/2	0.48	2/2	0.48
Glass	6	3	KNN/k-means*	scaled	0.39 (0.06)	0.35	3/3	0.35	3/3	0.35
Seeds	3	3	SVM/k-means	Raw	0.05 (0.03)	0.66	3/3	0.66	3/3	0.66
Parkinsons	2	2	KNN/k-means	scaled + UMAP	0.02 (0.01)	0.09	2/2	0.09	2/2	0.09
Iris	3	3/2 (tr/ts)	RF/HDBSCAN	UMAP	0.33 (0.19)	0.73	4/2	0.73	4/2	0.73

HC, hierarchical clustering; KNN, k-nearest neighbors; RF, random forest; SVM, support vector machine; AMI, adjusted mutual information score; ACC, accuracy.

Marked with * results that failed in identifying the correct number of clusters. Number of clusters, and stability measures are reported. For comparison, the number of clusters identified based on internal measures is reported. The best clustering algorithms selected are independently applied to train and test sets, with best solution defined as the one that maximizes or minimizes silhouette and Davies-Bouldin measures, respectively. AMI on test set is reported for performance evaluation in all cases.

clustering mechanisms. In fact, although a thorough theoretical and experimental analysis of stability-based model selection with k-means clustering has been done[3,14,15], the effectiveness of such an approach needs to be further investigated with different clustering algorithms. Furthermore, reval can be included in ensemble learning pipelines[31] or integrated in ensemble clustering frameworks for the selection of the best clustering solution.[32] Last, the ability to identify stability regimes and evaluate such regimes based on generalization to unseen data can inform the underestimation issue of the best number of clusters identified in real-world datasets.

Limitations of the study

A primary caveat or limitation to the approach reval takes is primarily one of data size. reval identifies the best clustering solution within a cross-validation framework, and hence needs large sample sizes to preserve cluster distribution between training and validation sets. Moreover, a separate held-out dataset is also needed to generalize the solution found. Smaller datasets may not allow for sufficient splitting within a cross-validation framework to allow for robust clustering solutions to generalize in unseen datasets. Finally, reval does not address the possibility of finding unrealistic data partitions. Because classifiers can overfit to their training set, a stable solution does not necessarily imply the true presence of subgroups in the data. Future work will focus on the implementation of other relative validation methods, e.g., based on prediction strength,[4] with the aim to create a comprehensive library.

EXPERIMENTAL PROCEDURES

Resource availability

Lead contact

For further information, suggestions, or to contribute to the package, please reach out to the lead contact, Isotta Landi (landi.isotta@gmail.com, @IsottaLandi).

Materials availability

This study did not generate any new materials.

Data and code availability

Code and package installation instructions can be found at https://github.com/IIT-LAND/reval_clustering. Documentation with working examples is at https://reval.readthedocs.io/en/latest/. Code with manuscript experiments and simulations can be found in the script `./working_examples/manuscript_examples.py` at https://github.com/IIT-LAND/reval_clustering.

Data used for the experiments and simulations are publicly available. In particular, the MNIST handwritten digits dataset can be downloaded from http://yann.lecun.com/exdb/mnist/, whereas the handwritten digits toy dataset can be found in the scikit-learn library. Datasets from the UCI Machine Learning repository can be found at https://archive.ics.uci.edu/ml/index.php, and those used in the algorithm selection section can be created by running the `./working_examples/datasets/manuscript_builddatasets.py` script at https://github.com/IIT-LAND/reval_clustering. The VABS dataset used to present the stability-regime-based clusters selection was extracted from the NDAR database https://nda.nih.gov/, which can be accessed upon approval by the NIH Data Access Committee.

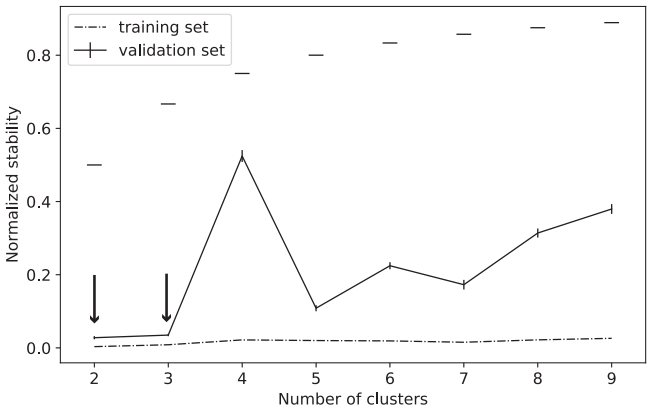

Figure 7. Stability regime for the NDAR dataset

Random label stability is displayed for performance evaluation. Dashed line shows training stability. Solid line represents validation normalized stability with 95% confidence intervals. Arrows point to the stability-regime solutions.

Computational details

The package was developed in Python 3.8 and simulations were run on a MacBook Pro 2020 with a 2.3 GHz Quad-Core Intel Core i7 processor and 32 GB RAM. All simulations presented in the technical validation section were run with reval v0.1.0.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.patter.2021.100228.

ACKNOWLEDGMENTS

This project was supported by funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program under grant agreement no. 755816 (ERC Starting Grant to M.V.L.).

AUTHOR CONTRIBUTIONS

Conceptualization, I.L. and M.V.L; methodology, I.L.; software, I.L. and V.M.; investigation, I.L. and V.M.; writing – original draft, I.L. and M.V.L.; writing – review & editing, I.L., V.M., and M.V.L.; funding acquisition, M.V.L.; supervision, M.V.L.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: August 28, 2020
Revised: January 18, 2021
Accepted: March 2, 2021
Published: April 2, 2021

REFERENCES

1. Friedman, J., Hastie, T., and Tibshirani, R. (2001). The Elements of Statistical Learning 10 (Springer series in statistics).
2. Vazirgiannis, M. (2009). Clustering validity. In Encyclopedia of Database Systems, L. Liu and M.T. Ozsu, eds. (Springer US), pp. 388–393.
3. Lange, T., Roth, V., Braun, M.L., and Buhmann, J.M. (2004). Stability-based validation of clustering solutions. Neural Comput. *16*, 1299–1323.
4. Tibshirani, R., and Walther, G. (2005). Cluster validation by prediction strength. J. Comput. Graph Stat. *14*, 511–528.
5. Thorndike, R.L. (1953). Who belongs in the family? Psychometrika *18*, 267–276.
6. Davies, D.L., and Bouldin, D.W. (1979). A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell, 224–227.
7. Rousseeuw, P.J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. *20*, 53–65.
8. Bengfort, B., and Bilbro, R. (2019). Yellowbrick: Visualizing the scikit-learn model selection process. J. Open Source Softw. *4*, 1075.
9. Charrad, M., Ghazzali, N., Boiteau, V., and Niknafs, A. (2014). NbClust: an R package for determining the relevant number of clusters in a data set. J. Stat. Softw. *61*, 1–36.
10. Brock, G., Pihur, V., Datta, S., and Datta, S. (2008). clValid: an R package for cluster validation. J. Stat. Softw. *25*, 1–22.
11. Haslbeck, J.M.B., and Wulff, D.U. (2018). Cstab: selection of number of clusters via normalized clustering instability R package version 0.2-2. https://CRAN.R-project.org/package=cstab.
12. Haslbeck, J.M.B., and Wulff, D.U.. Estimating the number of clusters via normalized cluster instability. https://arxiv.org/abs/1608.07494. 2016.
13. Brun, M., et al. (2007). Model-based evaluation of clustering validation measures. Pattern Recogn. *40*, 807824.
14. Von Luxburg, U. (2010). Clustering stability: an overview, *vol 2* (Foundations Trends® in Machine Learning), pp. 235–274, ISSN: 1935-8237.
15. Ben-Hur, A., Elisseeff, A., and Guyon, I. (2001). Biocomputing 2002 6-17 (World Scientific).
16. Moller, U. & Radke, D. A cluster validity approach based on nearest-neighbor resampling. 18th International Conference on Pattern Recognition (ICPR 06) 1 (2006). 10.1109/ICPR.2006.42.
17. Vinh, N.X. and Epps, J. A novel approach for automatic number of clusters detection in microarray data based on consensus clustering. 2009 Ninth IEEE International Conference on Bioinformatics and BioEngineering (2009). 10.1109/BIBE.2009.19.
18. Campello, R.J.G.B., Moulavi, D., and Sander, J. Density-based clustering based on hierarchical density estimates in Pacific-Asia conference on knowledge discovery and data mining (2013), 160-172.
19. Matthews, B.W. (1975). Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochim. Biophys. Acta *405*, 442–451.
20. Kuhn, H.W. (1955). The Hungarian method for the assignment problem. Naval Res. Logist. Q. *2*, 83–97.
21. Munkres, J. (1957). Algorithms for the assignment and transportation problems. J. Soc. Ind. Appl. Math. *5*, 32–38.
22. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: machine learning in Python. J. Machine Learn. Res. *12*, 2825–2830.
23. Dua, D., and Graff, C. (2017). UCI machine learning repository. http://archive.ics.uci.edu/ml.
24. McInnes, L., Healy, J., Saul, N., and Großberger, L. (2018). UMAP: uniform manifold approximation and projection. J. Open Source Softw. *3*, 861.
25. Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems. Ann. Eugen. *7*, 179–188.
26. American Psychiatric Association (2013). Diagnostic and Statistical Manual of Mental Disorders, 5th edition (American Psychiatric Association).
27. Lai, M.-C., Lombardo, M.V., and Baron-Cohen, S. (2014). Autism. Lancet *383*, 896–910, ISSN: 1474-547X.
28. Lombardo, M.V., Lai, M.-C., and Baron-Cohen, S. (2019). Big data approaches to decomposing heterogeneity across the autism spectrum. Mol. Psychiatry *24*, 1435–1450.
29. Sparrow, S.S., Balla, D.A., and Cicchetti, D.V. (2005). Vineland II: Vineland Adaptive Behavior Scales (AGS Publishing).
30. Sparrow, S.S., Cicchetti, D.V., and Saulnier, C.A. (2016). Vineland-3: Vineland Adaptive Behavior Scales (PsychCorp).
31. Rodriguez, J., Medina-Perez, M.A., Gutierrez-Rodriguez, A.E., Monroy, R., and Terashima-Marin, H. (2018). Cluster validation using an ensemble of supervised classifiers. Knowl. Based Syst. *145*, 134–144.
32. Strehl, A., and Ghosh, J. (2002). Cluster ensembles—a knowledge reuse framework for combining multiple partitions. J. Machine Learn. Res. *3*, 583–617.