PLOS ONE

RESEARCH ARTICLE

Clustering algorithms: A comparative approach

Mayra Z. Rodriguez[1], Cesar H. Comin[2]*, Dalcimar Casanova[3], Odemir M. Bruno[4], Diego R. Amancio[1], Luciano da F. Costa[4], Francisco A. Rodrigues[1]

1 Institute of Mathematics and Computer Science, University of São Paulo, São Carlos, São Paulo, Brazil, **2** Department of Computer Science, Federal University of São Carlos, São Carlos, São Paulo, Brazil, **3** Federal University of Technology, Paraná, Paraná, Brazil, **4** São Carlos Institute of Physics, University of São Paulo, São Carlos, São Paulo, Brazil

* chcomin@gmail.com

Citation: Rodriguez MZ, Comin CH, Casanova D, Bruno OM, Amancio DR, Costa LdF, et al. (2019) Clustering algorithms: A comparative approach. PLoS ONE 14(1): e0210236. https://doi.org/10.1371/journal.pone.0210236

Editor: Hans A Kestler, University of Ulm, GERMANY

Received: December 26, 2016

Accepted: December 19, 2018

Published: January 15, 2019

Data Availability Statement: All datasets used for evaluating the algorithms can be obtained from Figshare: https://figshare.com/s/29005b491a418a667b22.

Funding: This work has been supported by FAPESP - Fundação de Amparo à Pesquisa do Estado de São Paulo (grant nos. 15/18942-8 and 18/09125-4 for CHC, 14/20830-0 and 16/19069-9 for DRA, 14/08026-1 for OMB and 11/50761-2 and 15/22308-2 for LdFC), CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico (grant nos. 307797/2014-7 for OMB and 307333/2013-2

Abstract

Many real-world systems can be studied in terms of pattern recognition tasks, so that proper use (and understanding) of machine learning methods in practical applications becomes essential. While many classification methods have been proposed, there is no consensus on which methods are more suitable for a given dataset. As a consequence, it is important to comprehensively compare methods in many possible scenarios. In this context, we performed a systematic comparison of 9 well-known clustering methods available in the R language assuming normally distributed data. In order to account for the many possible variations of data, we considered artificial datasets with several tunable properties (number of classes, separation between classes, etc). In addition, we also evaluated the sensitivity of the clustering methods with regard to their parameters configuration. The results revealed that, when considering the default configurations of the adopted methods, the spectral approach tended to present particularly good performance. We also found that the default configuration of the adopted implementations was not always accurate. In these cases, a simple approach based on random selection of parameters values proved to be a good alternative to improve the performance. All in all, the reported approach provides subsidies guiding the choice of clustering algorithms.

Introduction

In recent years, the automation of data collection and recording implied a deluge of information about many different kinds of systems [1–8]. As a consequence, many methodologies aimed at organizing and modeling data have been developed [9]. Such methodologies are motivated by their widespread application in diagnosis [10], education [11], forecasting [12], and many other domains [13]. The definition, evaluation and application of these methodologies are all part of the machine learning field [14], which became a major subarea of computer science and statistics due to their crucial role in the modern world.

Machine learning encompasses different topics such as regression analysis [15], feature selection methods [16], and classification [14]. The latter involves assigning classes to the

for LdFC), Núcleo de Apoio à Pesquisa (LdFC) and CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Finance Code 001).

Competing interests: The authors have declared that no competing interests exist.

objects in a dataset. Three main approaches can be considered for classification: supervised, semi-supervised and unsupervised classification. In the former case, the classes, or labels, of some objects are known beforehand, defining the training set, and an algorithm is used to obtain the classification criteria. Semi-supervised classification deals with training the algorithm using both labeled and unlabeled data. They are commonly used when manually labeling a dataset becomes costly. Lastly, unsupervised classification, henceforth referred as *clustering*, deals with *defining* classes from the data without knowledge of the class labels. The purpose of clustering algorithms is to identify groups of objects, or clusters, that are more similar to each other than to other clusters. Such an approach to data analysis is closely related to the task of creating a model of the data, that is, defining a simplified set of properties that can provide intuitive explanation about relevant aspects of a dataset. Clustering methods are generally more demanding than supervised approaches, but provide more insights about complex data. This type of classifiers constitute the main object of the current work.

Because clustering algorithms involve several parameters, often operate in high dimensional spaces, and have to cope with noisy, incomplete and sampled data, their performance can vary substantially for different applications and types of data. For such reasons, several different approaches to clustering have been proposed in the literature (e.g. [17–19]). In practice, it becomes a difficult endeavor, given a dataset or problem, to choose a suitable clustering approach. Nevertheless, much can be learned by comparing different clustering methods. Several previous efforts for comparing clustering algorithms have been reported in the literature [20–29]. Here, we focus on generating a diversified and comprehensive set of artificial, normally distributed data containing not only distinct number of classes, features, number of objects and separation between classes, but also a varied structure of the involved groups (e.g. possessing predefined correlation distributions between features). The purpose of using artificial data is the possibility to obtain an unlimited number of samples and to systematically change any of the aforementioned properties of a dataset. Such features allow the clustering algorithms to be comprehensive and strictly evaluated in a vast number of circumstances, and also grants the possibility of quantifying the sensitivity of the performance with respect to small changes in the data. It should be observed, nevertheless, that the performance results reported in this work are therefore respective and limited to normally distributed data, and other results could be expected for other types of data following other statistical behavior. Here we associate performance with the similarity between the known labels of the objects and those found by the algorithm. Many measurements have been defined for quantifying such similarity [30], we compare the Jaccard index [31], Adjusted Rand index [32], Fowlkes-Mallows index [33] and Normalized mutual information [34]. A modified version of the procedure developed by [35] was used to create 400 distinct datasets, which were used in order to quantify the performance of the clustering algorithms. We describe the adopted procedure and the respective parameters used for data generation. Related approaches include [36].

Each clustering algorithm relies on a set of parameters that needs to be adjusted in order to achieve viable performance, which corresponds to an important point to be addressed while comparing clustering algorithms. A long standing problem in machine learning is the definition of a proper procedure for setting the parameter values [37]. In principle, one can apply an optimization procedure (e.g., simulated annealing [38] or genetic algorithms [39]) to find the parameter configuration providing the best performance of a given algorithm. Nevertheless, there are two major problems with such an approach. First, adjusting parameters to a given dataset may lead to overfitting [40]. That is, the specific values found to provide good performance may lead to lower performance when new data is considered. Second, parameter optimization can be unfeasible in some cases, given the time complexity of many algorithms, combined with their typically large number of parameters. Ultimately, many researchers resort

to applying classifier or clustering algorithms using the default parameters provided by the software. Therefore, efforts are required for evaluating and comparing the performance of clustering algorithms in the optimization and default situations. In the following, we consider some representative examples of algorithms applied in the literature [37, 41].

Clustering algorithms have been implemented in several programming languages and packages. During the development and implementation of such codes, it is common to implement changes or optimizations, leading to new versions of the original methods. The current work focuses on the comparative analysis of several clustering algorithm found in popular packages available in the R programming language [42]. This choice was motivated by the popularity of the R language in the data mining field, and by virtue of the well-established clustering packages it contains. This study is intended to assist researchers who have programming skills in R language, but with little experience in clustering of data.

The algorithms are evaluated on three distinct situations. First, we consider their performance when using the default parameters provided by the packages. Then, we consider the performance variation when single parameters of the algorithms are changed, while the rest are kept at their default values. Finally, we consider the simultaneous variation of all parameters by means of a random sampling procedure. We compare the results obtained for the latter two situations with those achieved by the default parameters, in such a way as to investigate the possible improvements in performance which could be achieved by modifying the algorithms.

The algorithms were evaluated on 400 artificial, normally distributed, datasets generated by a robust methodology previously described in [36]. The number of features, number of classes, number of objects for each class and average distance between classes can be systematically changed among the datasets.

The text is divided as follows. We start by revising some of the main approaches to clustering algorithms comparison. Next, we describe the clustering methods considered in the analysis, we also present the R packages implementing such methods. The data generation method and the performance measurements used to compare the algorithms are presented, followed by the presentation of the performance results obtained for the default parameters, for single parameter variation and for random parameter sampling.

Related works

Previous approaches for comparing the performance of clustering algorithms can be divided according to the nature of used datasets. While some studies use either real-world or artificial data, others employ both types of datasets to compare the performance of several clustering methods.

A comparative analysis using real world dataset is presented in several works [20, 21, 24, 25, 43, 44]. Some of these works are reviewed briefly in the following. In [43], the authors propose an evaluation approach based in a multiple criteria decision making in the domain of financial risk analysis over three real world credit risk and bankruptcy risk datasets. More specifically, clustering algorithms are evaluated in terms of a combination of clustering measurements, which includes a collection of external and internal validity indexes. Their results show that no algorithm can achieve the best performance on all measurements for any dataset and, for this reason, it is mandatory to use more than one performance measure to evaluate clustering algorithms.

In [21], a comparative analysis of clustering methods was performed in the context of text-independent speaker verification task, using three dataset of documents. Two approaches were considered: clustering algorithms focused in minimizing a distance based objective function

and a Gaussian models-based approach. The following algorithms were compared: k-means, random swap, expectation-maximization, hierarchical clustering, self-organized maps (SOM) and fuzzy c-means. The authors found that the most important factor for the success of the algorithms is the model order, which represents the number of centroid or Gaussian components (for Gaussian models-based approaches) considered. Overall, the recognition accuracy was similar for clustering algorithms focused in minimizing a distance based objective function. When the number of clusters was small, SOM and hierarchical methods provided lower accuracy than the other methods. Finally, a comparison of the computational efficiency of the methods revealed that the split hierarchical method is the fastest clustering algorithm in the considered dataset.

In [25], five clustering methods were studied: k-means, multivariate Gaussian mixture, hierarchical clustering, spectral and nearest neighbor methods. Four proximity measures were used in the experiments: Pearson and Spearman correlation coefficient, cosine similarity and the euclidean distance. The algorithms were evaluated in the context of 35 gene expression data from either Affymetrix or cDNA chip platforms, using the adjusted rand index for performance evaluation. The multivariate Gaussian mixture method provided the best performance in recovering the actual number of clusters of the datasets. The k-means method displayed similar performance. In this same analysis, the hierarchical method led to limited performance, while the spectral method showed to be particularly sensitive to the proximity measure employed.

In [24], experiments were performed to compare five different types of clustering algorithms: CLICK, self organized mapping-based method (SOM), k-means, hierarchical and dynamical clustering. Data sets of gene expression time series of the *Saccharomyces cerevisiae* yeast were used. A k-fold cross-validation procedure was considered to compare different algorithms. The authors found that k-means, dynamical clustering and SOM tended to yield high accuracy in all experiments. On the other hand, hierarchical clustering presented a more limited performance in clustering larger datasets, yielding low accuracy in some experiments.

A comparative analysis using artificial data is presented in [45–47]. In [47], two subspace clustering methods were compared: MAFIA (Adaptive Grids for Clustering Massive Data Sets) [48] and FINDIT (A Fast and Intelligent Subspace Clustering Algorithm Using Dimension Voting) [49]. The artificial data, modeled according to a normal distribution, allowed the control of the number of dimensions and instances. The methods were evaluated in terms of both scalability and accuracy. In the former, the running time of both algorithms were compared for different number of instances and features. In addition, the authors assessed the ability of the methods in finding adequate subspaces for each cluster. They found that MAFIA discovered all relevant clusters, but one significant dimension was left out in most cases. Conversely, the FINDIT method performed better in the task of identifying the most relevant dimensions. Both algorithms were found to scale linearly with the number of instances, however MAFIA outperformed FINDIT in most of the tests.

Another common approach for comparing clustering algorithms considers using a mixture of real world and artificial data (e.g. [23, 26–28, 50]). In [28], the performance of k-means, single linkage and simulated annealing (SA) was evaluated, considering different partitions obtained by validation indexes. The authors used two real world datasets obtained from [51] and three artificial datasets (having two dimensions and 10 clusters). The authors proposed a new validation index called *I* index that measures the separation based on the maximum distance between clusters and compactness based on the sum of distances between objects and their respective centroids. They found that such an index was the most reliable among other considered indices, reaching its maximum value when the number of clusters is properly chosen.

A systematic quantitative evaluation of four graph-based clustering methods was performed in [27]. The compared methods were: markov clustering (MCL), restricted neighborhood search clustering (RNSC), super paramagnetic clustering (SPC), and molecular complex detection (MCODE). Six datasets modeling protein interactions in the *Saccharomyces cerevisiae* and 84 random graphs were used for the comparison. For each algorithm, the robustness of the methods was measured in a twofold fashion: the variation of performance was quantified in terms of changes in the (i) methods parameters and (ii) dataset properties. In the latter, connections were included and removed to reflect uncertainties in the relationship between proteins. The restricted neighborhood search clustering method turned out to be particularly robust to variations in the choice of method parameters, whereas the other algorithms were found to be more robust to dataset alterations. In [52] the authors report a brief comparison of clustering algorithms using the Fundamental clustering problem suite (FPC) as dataset. The FPC contains artificial and real datasets for testing clustering algorithms. Each dataset represents a particular challenge that the clustering algorithm has to handle, for example, in the Hepta and LSum datasets the clusters can be separated by a linear decision boundary, but have different densities and variances. On the other hand, the ChainLink and Atom datasets cannot be separated by linear decision boundaries. Likewise, the Target dataset contains outliers. Lower performance was obtained by the single linkage clustering algorithm for the Tetra, EngyTime, Twodiamonds and Wingnut datasets. Although the datasets are quite versatile, it is not possible to control and evaluate how some of its characteristics, such as dimensions or number of features, affect the clustering accuracy.

Clustering methods

Many different types of clustering methods have been proposed in the literature [53–56]. Despite such a diversity, some methods are more frequently used [57]. Also, many of the commonly employed methods are defined in terms of similar assumptions about the data (e.g., k-means and k-medoids) or consider analogous mathematical concepts (e.g, similarity matrices for spectral or graph clustering) and, consequently, should provide similar performance in typical usage scenarios. Therefore, in the following we consider a choice of clustering algorithms from different families of methods. Several taxonomies have been proposed to organize the many different types of clustering algorithms into families [29, 58]. While some taxonomies categorize the algorithms based on their objective functions [58], others aim at the specific structures desired for the obtained clusters (e.g. hierarchical) [29]. Here we consider the algorithms indicated in Table 1 as examples of the categories indicated in the same table. The

Table 1. Clustering methods considered in our analysis and the respective libraries and functions in R employing the methods. The first column shows the name of the algorithms used throughout the text. The second column indicates the category of the algorithms. The third and fourth columns contain, respectively, the function name and R library of each algorithm.

Algorithm name	Category	Function in R	Library in R
k-means	Partitional	*k-means*	stats
clara	Partitional	*clara*	cluster
hierarchical	Linkage	*agnes*	cluster
EM	Model-based	*mstep, estep*	mclust
hcmodel	Model-based	*hc*	mclust
spectral	Spectral methods	*specc*	kernlab
subspace	Based on subspaces	*hddc*	HDclassif
optics	Density	*optics*	dbscan
dbscan	Density	*dbscan*	dbscan

https://doi.org/10.1371/journal.pone.0210236.t001

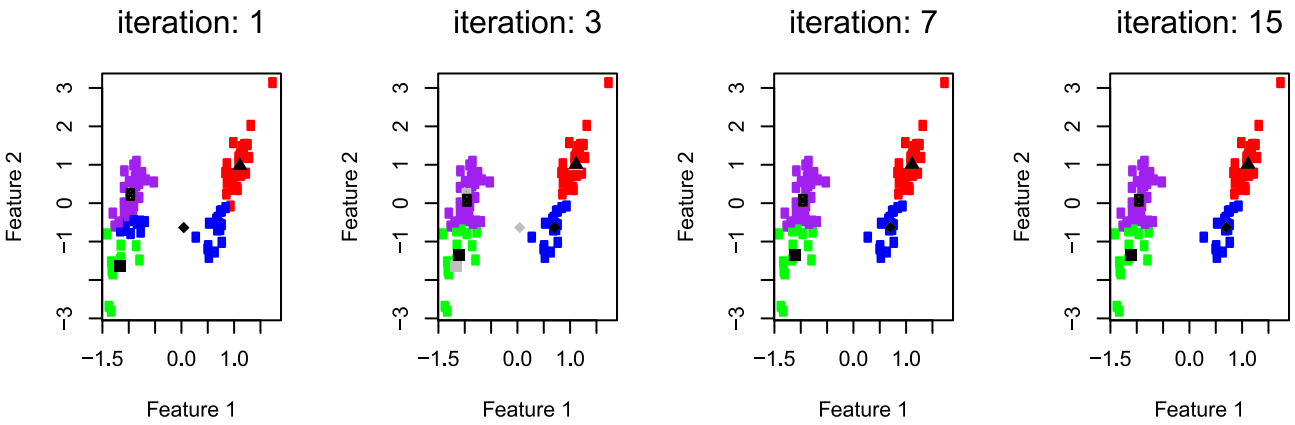

Fig 1. Illustration of the k-means clustering method. Each plot shows the partition obtained after specific iterations of the algorithm. The centroids of the clusters are shown as a black marker. Points are colored according to their assigned clusters. Gray markers indicate the position of the centroids in the previous iteration. The dataset contains 2 clusters, but $k = 4$ seeds were used in the algorithm.

https://doi.org/10.1371/journal.pone.0210236.g001

algorithms represent some of the main types of methods in the literature. Note that some algorithms are from the same family, but in these cases they posses notable differences in their applications (e.g., treating very large datasets using clara). A short description about the parameters of each considered algorithm is provided in S1 File of the supplementary material.

Regarding partitional approaches, the k-means [68] algorithm has been widely used by researchers [57]. This method requires as input parameters the number of groups (k) and a distance metric. Initially, each data point is associated with one of the k clusters according to its distance to the centroids (clusters centers) of each cluster. An example is shown in Fig 1(a), where black points correspond to centroids and the remaining points have the same color if the centroid that is closest to them is the same. Then, new centroids are calculated, and the classification of the data points is repeated for the new centroids, as indicated in Fig 1(b), where gray points indicate the position of the centroids in the previous iteration. The process is repeated until no significant changes of the centroids positions is observed at each new step, as shown in Fig 1(c) and 1(d).

The *a priori* setting of the number of clusters is the main limitation of the k-means algorithm. This is so because the final classification can strongly depend on the choice of the number of centroids [68]. In addition, the k-means is not particularly recommended in cases where the clusters do not show convex distribution or have very different sizes [59, 60]. Moreover, the k-means algorithm is sensitive to the initial seed selection [41]. Given these limitations, many modifications of this algorithm have been proposed [61–63], such as the k-medoid [64] and k-means++ [65]. Nevertheless, this algorithm, besides having low computational cost, can provide good results in many practical situations such as in anomaly detection [66] and data segmentation [67]. The R routine used for k-means clustering was the *k-means* from the *stats* package, which contains the implementation of the algorithms proposed by Macqueen [68], Hartigan and Wong [69]. The algorithm of Hartigan and Wong is employed by the *stats* package when setting the parameters to their default values, while the algorithm proposed by Macqueen is used for all other cases. Another interesting example of partitional clustering algorithms is the clustering for large applications (clara) [70]. This method takes into account multiple fixed samples of the dataset to minimize sampling bias and, subsequently, select the best medoids among the chosen samples, where a medoid is defined as the object i for which the average dissimilarity to all other objects in its cluster is minimal. This method tends to be efficient for large amounts of data because it does not explore the whole neighborhood of the

data points [71], although the quality of the results have been found to strongly depend on the number of objects in the sample data [62]. The clara algorithm employed in our analysis was provided by the *clara* function contained in the *cluster* package. This function implements the method developed by Kaufman and Rousseeuw [70].

The Ordering Points To Identify the Clustering Structure (OPTICS) [72, 73] is a density-based cluster ordering based on the concept of maximal density-reachability [72]. The algorithm starts with a data point and expands its neighborhood using a similar procedure as in the dbscan algorithm [74], with the difference that the neighborhood is first expanded to points with low core-distance. The core distance of an object p is defined as the m-th smallest distance between p and the objects in its ϵ-neighborhood (i.e., objects having distance less than or equal to ϵ from p), where m is a parameter of the algorithm indicating the smallest number of points that can form a cluster. The optics algorithm can detect clusters having large density variations and irregular shapes. The R routine used for optics clustering was the *optics* from the *dbscan* package. This function considers the original algorithm developed by Ankerst et al. [72]. An hierarchical clustering structure from the output of the optics algorithm can be constructed using the function *extractXi* from the *dbscan* package. We note that the function *extractDBSCAN*, from the same package, provides a clustering from an optics ordering that is similar to what the dbscan algorithm would generate.

Clustering methods that take into account the linkage between data points, traditionally known as hierarchical methods, can be subdivided into two groups: agglomerative and divisive [59]. In an agglomerative hierarchical clustering algorithm, initially, each object belongs to a respective individual cluster. Then, after successive iterations, groups are merged until stop conditions are reached. On the other hand, a divisive hierarchical clustering method starts with all objects in a single cluster and, after successive iterations, objects are separated into clusters. There are two main packages in the R language that provide routines for performing hierarchical clustering, they are the *stats* and *cluster*. Here we consider the *agnes* routine from the *cluster* package which implements the algorithm proposed by Kaufman and Rousseeuw [70]. Four well-known linkage criteria are available in *agnes*, namely single linkage, complete linkage, Ward's method, and weighted average linkage [75].

Model-based methods can be regarded as a general framework for estimating the maximum likelihood of the parameters of an underlying distribution to a given dataset. A well-known instance of model-based methods is the expectation-maximization (EM) algorithm. Most commonly, one considers that the data from each class can be modeled by multivariate normal distributions, and, therefore, the distribution observed for the whole data can be seen as a mixture of such normal distributions. A maximum likelihood approach is then applied for finding the most probable parameters of the normal distributions of each class. The EM approach for clustering is particularly suitable when the dataset is incomplete [76, 77]. On the other hand, the clusters obtained from the method may strongly depend on the initial conditions [54]. In addition, the algorithm may fail to find very small clusters [29, 78]. In the R language, the package *mclust* [79, 80]. provides iterative EM (Expectation-Maximization) methods for maximum likelihood estimation using parameterized Gaussian mixture models. Functions *estep* and *mstep* implement the individual steps of an EM iteration. A related algorithm that is also analyzed in the current study is the hcmodel, which can be found in the *hc* function of the *mclust* package. The hcmodel algorithm, which is also based on Gaussian-mixtures, was proposed by Fraley [81]. The algorithm contains many additional steps compared to traditional EM methods, such as an agglomerative procedure and the adjustment of model parameters through a Bayes factor selection using the BIC aproximation [82].

Another class of methods considered in our analyses is spectral clustering. These methods emerged as an alternative to traditional clustering approaches that were not able to define

nonlinear discriminative hypersurfaces [83]. The main advantage of spectral methods lies on the definition of an adjacency structure from the original dataset, which avoids imposing a prefixed shape for the clusters [84]. The first step of the method is to construct an affinity matrix $A \in \mathbb{R}^{NxN}$, where the value in the j-th row and k-th column indicates the similarity between points j and k. This matrix can be regarded as a weighted graph representation of the data. Then, the eigenvalues and eigenvectors of the matrix are used for partitioning the data according to a given criterion. Many different types of similarity matrices can be used, a popular choice being the Laplacian matrix [85]. Spectral methods involve the potentially demanding process of calculating the eigenvectors of the similarity matrix [54]. The function *specc* from the *kernlab* R package implements the algorithm of Jordan and Weiss [86], which employs a kernel function to compute the affinity matrix from the data. The function is defined as $A_{ij} = \exp(-||x_i - x_j||^2/2\sigma^2)$, where x_i and x_j are points to be clustered into k subsets and σ^2 controls how rapidly the affinity matrix A_{ij} falls off with the distance between x_i and x_j. A recent discussion about the relationship between the spectral and kmeans algorithms can be found in [87]. The weighted kernel kmeans is a generalization of the kmeans method. It can be used to locally optimize the graph partitioning objectives into k disjoint partitions or clusters. Usually, this task would be performed by some spectral algorithm using eigenvectors to help determine the partitions. However, depending on the size of the dataset, computing a large number of eigenvectors can become computationally expensive. The authors show that the weighted kernel kmeans algorithm can be used to aid in optimizing a number of graph partitioning objectives.

In recent years, the efficient handling of high dimensional data has become of paramount importance and, for this reason, this feature has been desired when choosing the most appropriate method for obtaining accurate partitions. To tackle high dimensional data, subspace clustering was proposed [49]. This method works by considering the similarity between objects with respect to distinct subsets of the attributes [88]. The motivation for doing so is that different subsets of the attributes might define distinct separations between the data. Therefore, the algorithm can identify clusters that exist in multiple, possibly overlapping, subspaces [49]. Subspace algorithms can be categorized into four main families [89], namely: lattice, statistical, approximation and hybrid. The *hddc* function from package *HDclassif* implements the subspace clustering method of Bouveyron [90] in the R language. The algorithm is based on statistical models, with the assumption that all attributes may be relevant for clustering [91]. Some parameters of the algorithm, such as the number of clusters or model to be used, are estimated using an EM procedure.

So far, we have discussed the application of clustering algorithms on static data. Nevertheless, when analyzing data, it is important to take into account whether the data are dynamic or static. Dynamic data, unlike static data, undergo changes over time. Some kinds of data, like the network packets received by a router and credit card transaction streams, are transient in nature and they are known as *data stream*. Another example of dynamic data are time series because its values change over time [92]. Dynamic data usually include a large number of features and the amount of objects is potentially unbounded [59]. This requires the application of novel approaches to quickly process the entire volume of continuously incoming data [93], the detection of new clusters that are formed and the identification of outliers [94].

Materials and methods

Artificial datasets

The proper comparison of clustering algorithms requires a robust artificial data generation method to produce a variety of datasets. For such a task, we apply a methodology based on a

previous work by Hirschberger et al. [35]. The procedure can be used to generate normally distributed samples characterized by F features and separated into C classes. In addition, the method can control both the variance and correlation distributions among the features for each class. The artificial dataset can also be generated by varying the number of objects per class, N_e, and the expected separation, α, between the classes.

The main difficult in generating datasets with the aforementioned properties is the definition of a proper covariance matrix R for the considered features. A valid covariance matrix must be positive semi-definite [95], which is hard to ensure. However, for a given matrix $\mathbf{G} \in \mathbb{R}^{n \times m}$, the matrix $R = \mathbf{G}\mathbf{G}^T$ is guaranteed to be positive semi-definite [95]. Thus any random matrix $\mathbf{G}$ can define a valid respective covariance matrix. As a consequence, additional constraints on matrix $\mathbf{G}$ can be imposed for the generation of datasets with the required properties. Hirschberger et al. [35] developed a robust approach to generate such a matrix given the first two statistical moments of the co-variance distribution of a set of F artificial features. The resulting covariance matrix contains variances and co-variances drawn from such distribution. Here we consider a normal distribution to represent the elements of $\mathbf{R}$.

For each class i in the dataset, a covariance matrix $\mathbf{R}_i$ of size $F \times F$ is created, and this matrix is used for generating N_e objects for the classes. This means that pairs of features can have distinct correlation for each generated class. Then, the generated class values are divided by α and translated by s_i, where s_i is a random variable described by a uniform random distribution defined in the interval $[-1, 1]$. Parameter α is associated with the expected distances between classes. Such distances can have different impacts on clusterization depending on the number of objects and features used in the dataset. The features in the generated data have a multivariate normal distribution. In addition, the covariance among the features also have a normal distribution. Notice that such a procedure for the generation of artificial datasets was previously used in [36].

In Fig 2, we show some examples of artificially generated data. For visualization purposes, all considered cases contain $F = 2$ features. The parameters used for each case are described in the caption of the figure. Note that the methodology can generate a variety of dataset configurations, including variations in features correlation for each class.

In this study, we considered the following values for the artificial dataset parameters:

- **Number of classes** (C): The generated datasets are divided into $C = \{2, 10, 50\}$ classes.
- **Number of features** (F): The number of features to characterize the objects is $F = \{2, 5, 10, 50, 200\}$.
- **Number of object per class** (N_e): we considered $Ne = \{5, 50, 100, 500, 5000\}$ objects per class. In our experiments, in a given generated dataset, the number of instances for each class is constant.
- **Mixing parameter** (α): This parameter has a non-trivial dependence on the number of classes and features. Therefore, for each dataset, the value of this parameter was tuned so that no algorithm would achieve an accuracy of 0% or 100%.

We refer to datasets containing 2, 10, 50 and 200 features as DB2F, DB10F, DB50F, DB200F respectively. Such datasets are composed of all considered number of classes, C = {2, 10, 50}, and 50 elements for each class (i.e., Ne = 50). In some cases, we also indicate the number of classes considered for the dataset. For example, dataset DB2C10F contains 2 classes, 10 features and 50 elements per class.

For each case, we consider 10 realizations of the dataset. Therefore, 400 datasets were generated in total.

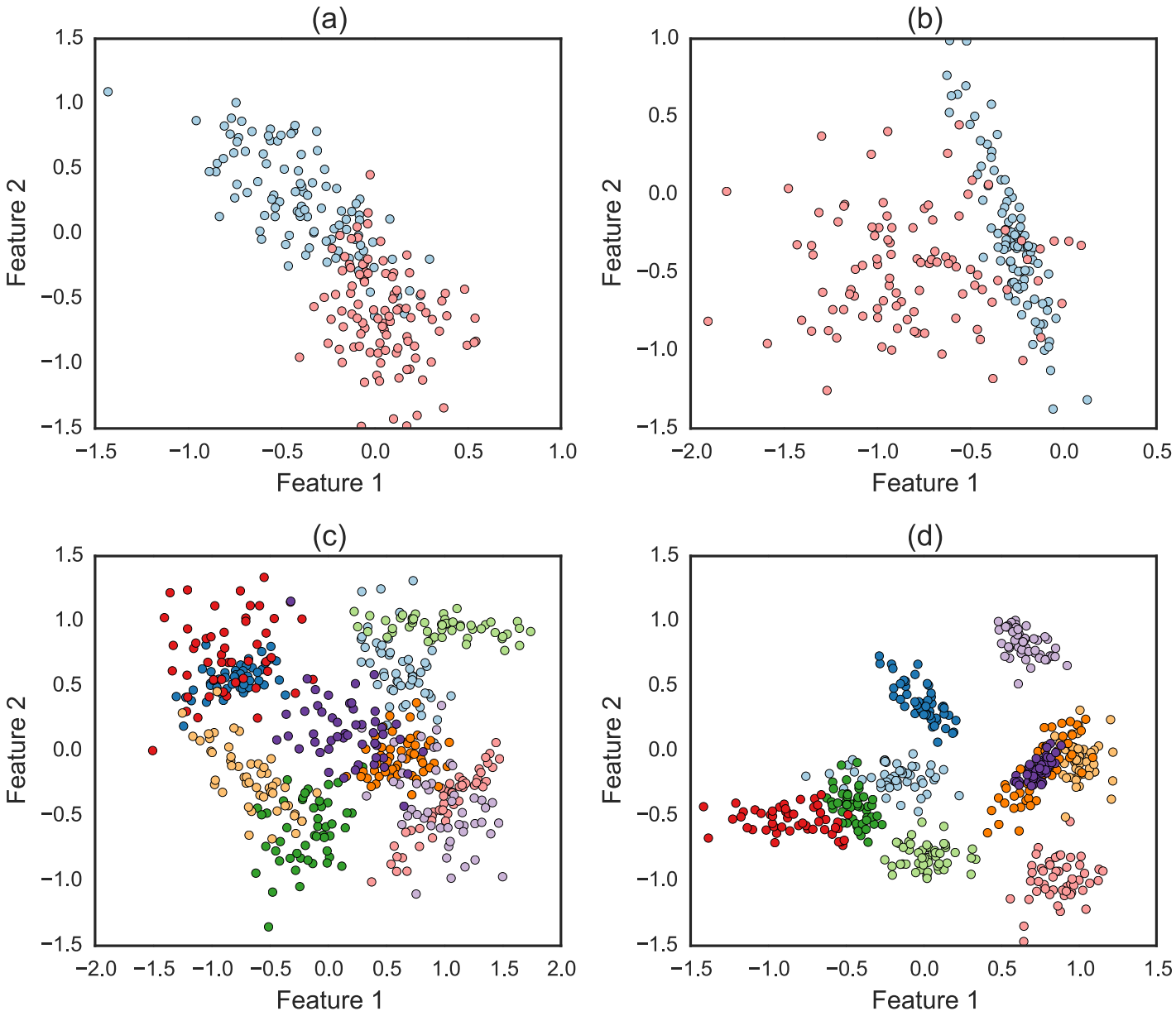

Fig 2. Examples of artificial datasets generated by the methodology. The parameters used for each case are (a) C = 2, Ne = 100 and α = 3.3. (b) C = 2, Ne = 100 and α = 2.3. (c) C = 10, Ne = 50 and α = 4.3. (d) C = 10, Ne = 50 and α = 6.3. Note that each class can present highly distinct properties due to differences in correlation between their features.

https://doi.org/10.1371/journal.pone.0210236.g002

Evaluating the performance of clustering algorithms

The evaluation of the quality of the generated partitions is one of the most important issues in cluster analysis [30]. Indices used for measuring the quality of a partition can be categorized into two classes, internal and external indices. Internal validation indices are based on information intrinsic to the data, and evaluates the goodness of a clustering structure without external information. When the correct partition is not available it is possible to estimate the quality of a partition measuring how closely each instance is related to the cluster and how well-separated a cluster is from other clusters. They are mainly used for choosing an optimal clustering

algorithm to be applied on a specific dataset [96]. On the other hand, external validation indices measure the similarity between the output of the clustering algorithm and the correct partitioning of the dataset. The Jaccard, Fowlkes-Mallows and adjusted rand index belong to the same pair counting category, making them closely related. Some differences include the fact that they can exhibit biasing with respect to the number of clusters or the distribution of class sizes in a partition. Normalization helps prevent this unwanted effect. In [97] the authors discuss several types of bias that may affect external cluster validity indices. A total of 26 pair-counting based external cluster validity indices were used to identify the bias generated by the number of clusters. It was shown that the Fowlkes Mallows and Jaccard index monotonically decrease as the number of clusters increases, favoring partitions with smaller number of clusters, while the Adjusted Rand Index tends to be indifferent to the number of clusters.

Here, we adopt the most traditional external indexes, which are the Jaccard Index (J) [31], Adjusted Rand Index (ARI) [32], Fowlkes Mallows Index (FM) [33] and Normalized Mutual Information (NMI) [34]. In order to define the cluster external index, we consider the following concepts. Let $U = \{u_1, u_2 \ldots u_R\}$ represent the original partition of a dataset, where u_i denote a subset of the objects associated with cluster i. Equivalently, let $V = \{v_1, v_2 \ldots v_C\}$ represent the partition found by a cluster algorithm. We denote as a the number of pairs of objects that are placed in the same group in both U and V. Mathematically, a can be computed by

$$a = \sum_{i,j} \binom{n_{ij}}{2}, \tag{1}$$

where n_{ij} is the number of objects belonging to both subset u_i and v_j.

Let b indicate the number of pairs of objects belonging to the same group in U but different groups in V, i.e.

$$b = \sum_{i} \binom{n_{i.}}{2} - \sum_{i,j} \binom{n_{ij}}{2}, \tag{2}$$

where $n_{i.} = \sum_j n_{ij}$. Let c be the number of pairs of objects belonging to different groups in U and to the same group in V, which can be written as

$$c = \sum_{j} \binom{n_{.j}}{2} - \sum_{i,j} \binom{n_{ij}}{2}, \tag{3}$$

where $n_{.j} = \sum_i n_{ij}$.

The Jaccard Index (J), Adjusted Rand Index (ARI) and Fowlkes Mallows (FM) index can then be defined based on a, b, c:

$$J = \frac{a}{a+b+c}, \tag{4}$$

$$\mathrm{ARI} = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}{1/2\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}, \tag{5}$$

$$\mathrm{FM} = a\frac{\sqrt{a+b}\sqrt{a+c}}{(a+b)(a+c)}. \tag{6}$$

We also consider the normalized mutual information (NMI) as a quality metric because it quantifies the mutual dependence between two random variables based on well-established

concepts of information theory [98]. The NMI measure is defined as [99]

$$\mathrm{NMI}(C,T) = \frac{I(C,T)}{\sqrt{[H(C),H(T)]}}. \tag{7}$$

where C is the random variable denoting the cluster assignments of the points, and T is the random variable denoting the underlying class labels on the points. $I(C, T) = H(C) - H(C|T)$ is the mutual information between the random variables C and T. $H(C)$ is the Shannon entropy of C. $H(C|T)$ is the conditional entropy of C given T.

Note that when the two sets of labels have a perfect one-to-one correspondence, the quality measures are all equal to unity.

Previous works have shown that there is no single internal cluster validation index that outperforms the other indices [100, 101]. In [101] the authors compare a set of internal cluster validation indices in many distinct scenarios, indicating that the Silhouette index yielded the best results in most cases.

The Dunn's validation index quantifies not only the degree of compactness of clusters, but also the degree of separation between individual clusters. The goal is therefore to maximize the inter-cluster distance while minimizing the intra-cluster distance, where c_i represents the i-th cluster of the partition. The index is calculated as

$$DU = \min_{1\le i\le k}\left\{\min_{i+1\le j\le k}\left\{\frac{dist(c_i,c_j)}{\max\limits_{1\le l\le k} diam(c_l)}\right\}\right\} \tag{8}$$

where $dist(c_i, c_j)$ is the minimal distance between clusters c_i and c_j, and $diam(c_l) = \max_{x,y\in c_l}\|x-y\|$. A high value of this measure indicates that a compact and well-separated cluster.

The Silhouette index (SI) computes for each point a width depending on its membership inside a cluster.

$$SI_k = \frac{1}{n}\sum_{i=1}^{n}\frac{(b_i - a_i)}{max(a_i,b_i)} \tag{9}$$

where n is the total number of points, a_i is the average distance between point i and all other points in its own cluster, and b_i is the minimum of the average dissimilarities between i and points in other clusters. The interval of the silhouette index values is $-1 \le SI \le 1$. The partition with the highest SI is taken to be optimal.

Results and discussion

The accuracy of each considered clustering algorithm was evaluated using three methodologies. In the first methodology, we consider the default parameters of the algorithms provided by the R package. The reason for measuring performance using the default parameters is to consider the case where a researcher applies the classifier to a dataset without any parameter adjustment. This is a common scenario when the researcher is not a machine learning expert. In the second methodology, we quantify the influence of the algorithms parameters on the accuracy. This is done by varying a single parameter of an algorithm while keeping the others at their default values. The third methodology consists in analyzing the performance by randomly varying all parameters of a classifier. This procedure allows the quantification of certain properties such as the maximum accuracy attained and the sensibility of the algorithm to parameter variation.

Performance when using default parameters

In this experiment, we evaluated the performance of the classifiers for all datasets described in Section *Artificial datasets*. All unsupervised algorithms were set with their default configuration of parameters. For each algorithm, we divide the results according to the number of features contained in the dataset. In other words, for a given number of features, F, we used datasets with $C = \{2, 10, 50, 200\}$ classes, and $N_e = \{5, 50, 100\}$ objects for each class. Thus, the performance results obtained for each F corresponds to the performance averaged over distinct number of classes and objects per class. We note that the algorithm based on subspaces cannot be applied to datasets containing 2 features, and therefore its accuracy was not quantified for such datasets.

In Fig 3, we show the obtained values for the four considered performance metrics. The results indicate that all performance metrics provide similar results. Also, the hierarchical method seems to be strongly affected by the number of features in the dataset. In fact, when using 50 and 200 features the hierarchical method provided lower accuracy. The k-means, spectral, optics and dbscan methods benefit from an increment in the number of features. Interestingly, the hcmodel has a better performance in the datasets containing 10 features than in those containing 2, 50 and 200 features, which suggests an optimum performance for this algorithm for datasets containing around 10 features. It is also clear that for 2 features the performance of the algorithms tend to be similar, with the exception of the optics and dbscan methods. On the other hand a larger number of features induce marked differences in performance. In particular, for 200 features, the spectral algorithm provides the best results among all classifiers.

We use the Kruskal-Wallis test [102], a nonparametric test, to explore the statistical differences in performance when considering distinct number of features in clustering methods. First, we test if the difference in performance is significant for 2 features. For this case, the Kruskal-Wallis test returns a p-value of $p = 6.48 \times 10^{-7}$, with a chi-squared distance of $\chi^2 = 41.50$. Therefore, the difference in performance is statistically significant when considering all algorithms. For datasets containing 10 features, a p-value of $p = 1.53 \times 10^{-8}$ is returned by the Kruskal-Wallis test, with a chi-squared distance of $\chi^2 = 52.20$). For 50 features, the test returns a p-value of $p = 1.56 \times 10^{-6}$, with a chi-squared distance of $\chi^2 = 41.67$). For 200 features, the test returns a p-value of $p = 2.49 \times 10^{-6}$, with a chi-squared distance of $\chi^2 = 40.58$). Therefore, the null hypothesis of the Kruskal–Wallis test is rejected. This means that the algorithms indeed have significant differences in performance for 2, 10, 50 and 200 features, as indicated in Fig 3.

In order to verify the influence of the number of objects used for classification, we also calculated the average accuracy for datasets separated according to the number of objects N_e. The result is shown in Fig 4. We observe that the impact that changing N_e has on the accuracy depends on the algorithm. Surprisingly, the hierarchical, k-means and clara methods attain lower accuracy when more data is used. The result indicates that these algorithms tend to be less robust with respect to the larger overlap between the clusters due to an increase in the number of objects. We also observe that a larger N_e enhances the performance of the hcmodel, optics and dbscan algorithms. This results is in agreement with [90].

In most clustering algorithms, the size of the data has an effect on the clustering quality. In order to quantify this effect, we considered a scenario where the data has a high number of instances. Datasets with F = 5, C = 10 and Ne = {5, 50, 500, 5000} instances per class were created. This dataset will be referenced as DB10C5F. In Fig 5 we can observe that the subspace and spectral methods lead to improved accuracy when the number of instances increases. On the other hand, the size of the dataset does not seem to influence the accuracy of the kmeans,

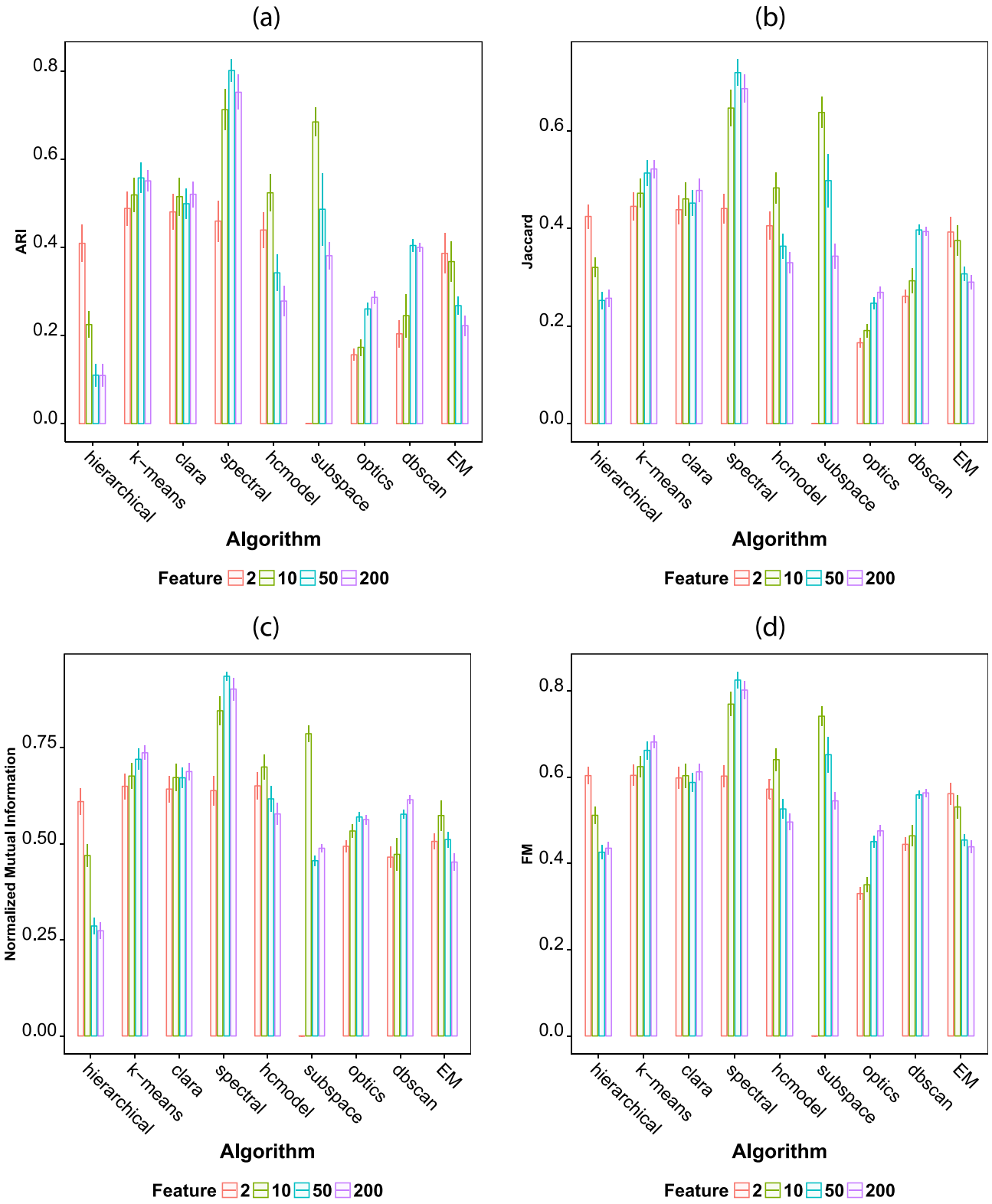

Fig 3. Average performance of the seven considered clustering algorithms according to the number of features in the dataset. All artificial datasets were used for evaluation. The averages were calculated separately for datasets containing 2, 10 and 50 features. The considered performance indexes are (a) adjusted Rand, (b) Jaccard, (c) normalized mutual information and (d) Fowlkes Mallows.

https://doi.org/10.1371/journal.pone.0210236.g003

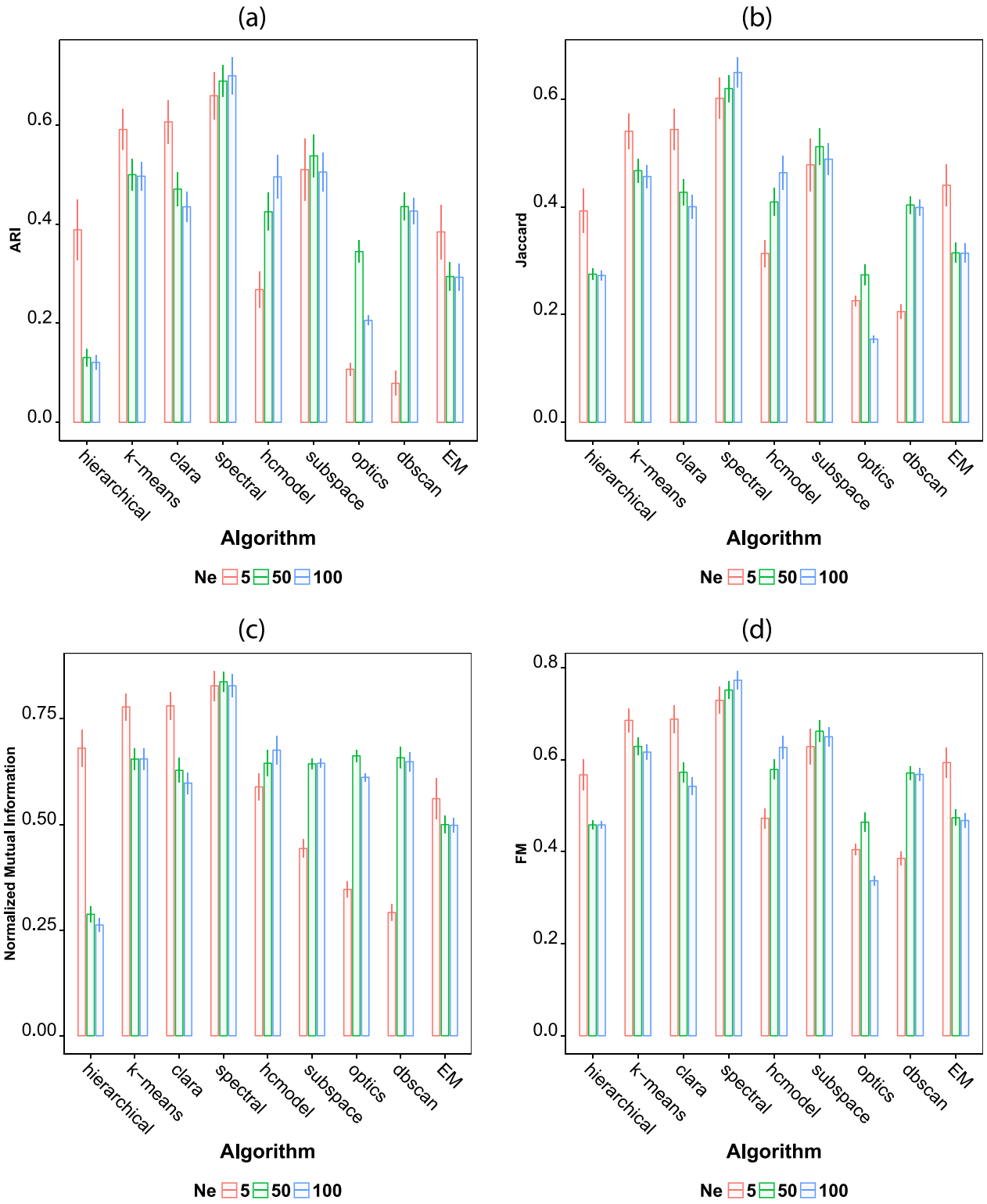

Fig 4. Average performance of the seven considered clustering algorithms according to the number of objects per class in the dataset. All artificial datasets were used for evaluation. The averages were calculated separately for datasets containing 5, 50 and 100 objects per class. The considered performance indexes are (a) adjusted Rand, (b) Jaccard, (c) normalized mutual information and (d) Fowlkes Mallows.

https://doi.org/10.1371/journal.pone.0210236.g004

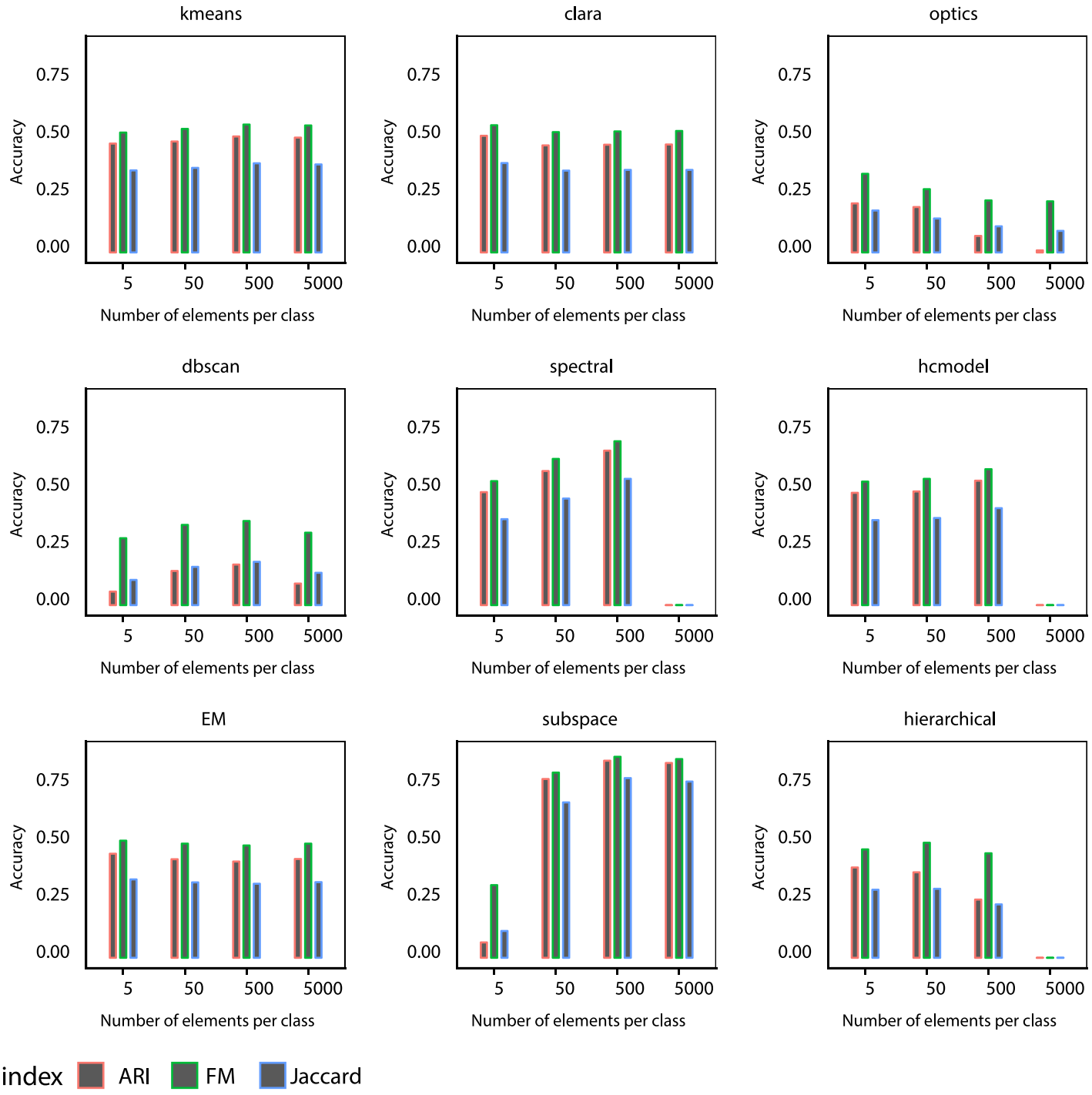

Fig 5. Performance of the algorithms when the number of elements by class correspond to *Ne* = 5, 50, 500, 5000. The plots correspond to the ARI, Jaccard and FM indexes averaged for all datasets containing 10 classes and 5 features (DB10C5F).

https://doi.org/10.1371/journal.pone.0210236.g005

clara, hcmodel and EM algorithms. For the spectral, hierarchical and hcmodel algorithms, the accuracy could not be calculated when 5000 instances per class was used due to the amount of memory used by these methods. For example, in the case of the spectral algorithm method, a lot of processing power is required to compute and store the kernel matrix when the algorithm

is executed. When the size of the dataset is too small, we see that the subspace algorithm results in low accuracy.

It is also interesting to verify the performance of the clustering algorithms when setting distinct values for the expected number of classes K in the dataset. Such a value is usually not known beforehand in real datasets. For instance, one might expect the data to contain 10 classes, and, as a consequence, set $K = 10$ in the algorithm, but the objects may actually be better accommodated into 12 classes. An accurate algorithm should still provide reasonable results even when a wrong number of classes is assumed. Thus, we varied K for each algorithm and verified the resulting variation in accuracy. Observe that the optics and dbscan methods were not considered in this analysis as they do not have a parameter for setting the number of classes. In order to simplify the analysis, we only considered datasets comprising objects described by 10 features and divided into 10 classes (DB10C10F). The results are shown in Fig 6. The top figures correspond to the average ARI and Jaccard indexes calculated for DB10C10F, while the Silhoute and Dunn indexes are shown at the bottom of the figure. The results indicate that setting $K < 10$ leads to a worse performance than obtained for the cases where $K > 10$, which suggests that a slight overestimation of the number of classes has smaller effect on the performance. Therefore, a good strategy for choosing K seems to be setting it to values that are slightly larger than the number of expected classes. An interesting behavior is observed for hierarchical clustering. The accuracy improves as the number of expected classes increases. This behavior is due to the default value of the *method* parameter, which is set as "average". The "average" value means that the unweighted pair group method with arithmetic mean

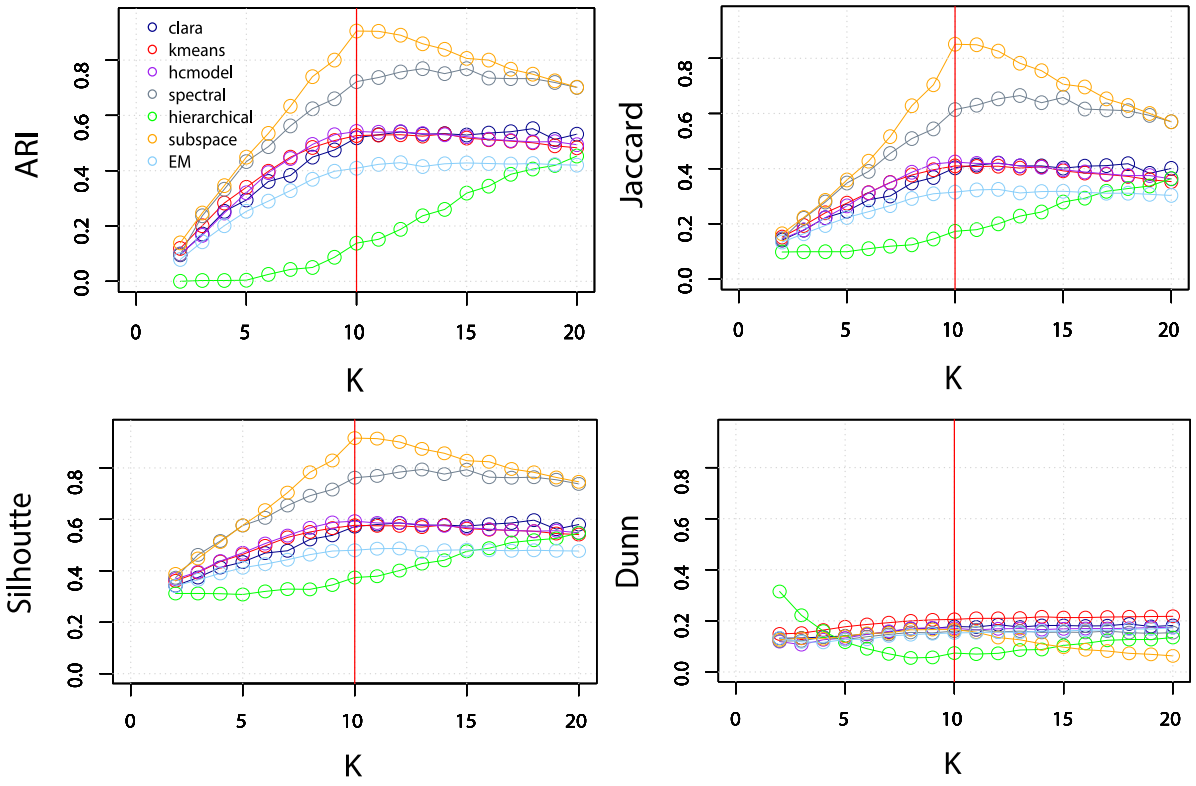

Fig 6. Performance of the algorithms when changing the expected number of clusters K in the dataset. The upper plots correspond to the ARI and Jaccard indices averaged for all datasets containing 10 classes and 10 features (DB10C10F). The lower plots correspond to the Silhouette and Dunn indices for the same dataset. The red line indicates the actual number of clusters in the dataset.

https://doi.org/10.1371/journal.pone.0210236.g006

(UPGMA) is used to agglomerate the points. UPGMA is the average of the dissimilarities between the points in one cluster and the points in the other cluster. The moderate performance of UPGMA in recovering the original groups, even with high subgroup differentiation, is probably a consequence of the fact that UPGMA tends to result in more unbalanced clusters, that is, the majority of the objects are assigned to a few clusters while many other clusters contain only one or two objects.

The external validation indices show that most of the clustering algorithms correctly identify the 10 main clusters in the dataset. Naturally, this knowledge would not be available in a real life cluster analysis. For this reason, we also consider internal validation indices, which provides feedback on the partition quality. Two internal validation indices were considered, the Silhouette index (defined in the range [−1,1]) and the Dunn index (defined in the range $[0, \infty]$). These indices were applied to the DB10C10F and DB10C2F dataset while varying the expected number of clusters K. The results are presented in Figs 6 and 7. In Fig 6 we can see that the results obtained for the different algorithms are mostly similar. The results for the Silhouette index indicate high accuracy around k = 10. The Dunn index displays a slightly lower performance, misestimating the correct number of clusters for the hierarchical algorithm. In Fig 7 Silhouette and Dunn show similar behavior.

The results obtained for the default parameters are summarized in Table 2. The table is divided into four parts, each part corresponds to a performance metric. For each performance metric, the value in row i and column j of the table represents the average performance of the method in row i minus the average performance of the method in column j. The last column

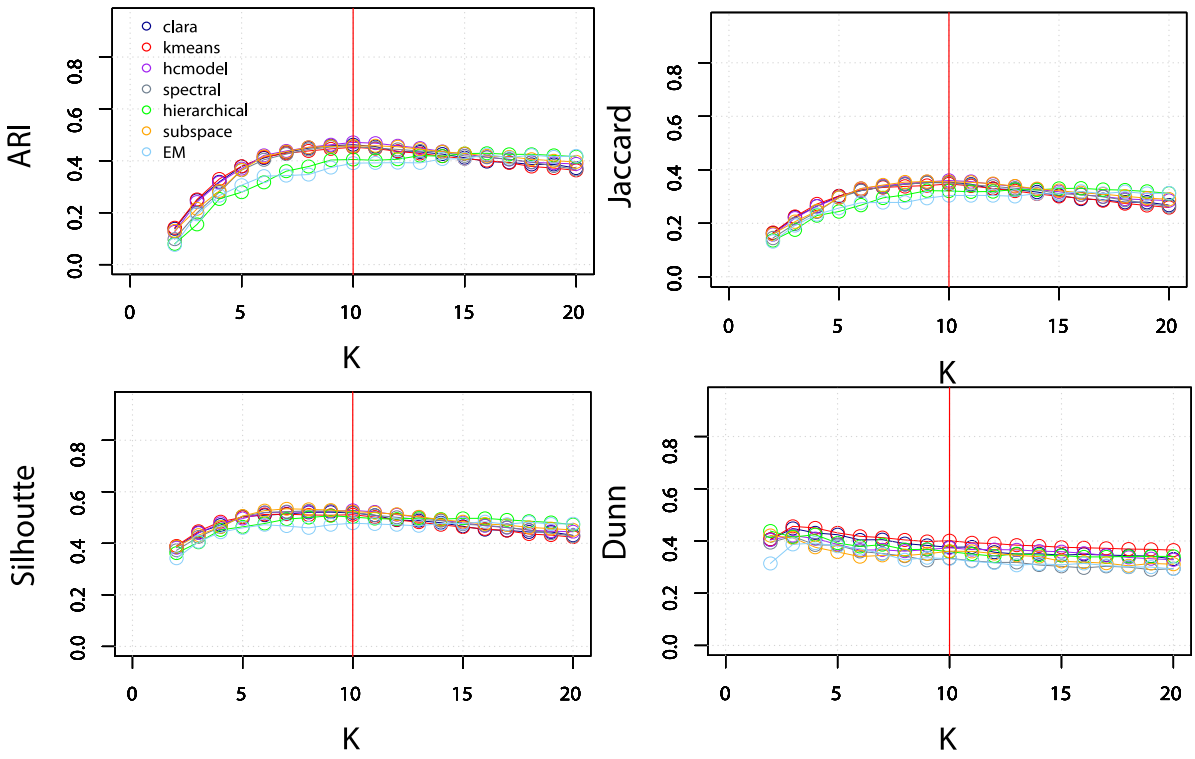

Fig 7. Performance of the algorithms when changing the expected number of clusters *K* in the dataset. The upper plots correspond to the ARI and Jaccard indices averaged for all datasets containing 10 classes and 2 features (DB10C2F). The lower plots correspond to the Silhouette and Dunn indices for the same dataset. The red line indicates the actual number of clusters in the dataset.

https://doi.org/10.1371/journal.pone.0210236.g007

Table 2. Average difference of accuracies obtained when clustering algorithms are used with their default configuration of parameters. In general, the spectral algorithm provides the highest accuracy rate among all evaluated methods.

	Algorithm	hierarchical	k-means	clara	spectral	hcmodel	subspace	optics	dbscan	EM	MAcc
NMI	hierarchical	-	-28.59%	-25.89%	-42.10%	-22.65%	-16.72%	-13.01%	-9.59%	-10.29%	40.99%
	k-means	28.59%	-	2.70%	-13.51%	5.94%	11.87%	15.58%	19.00%	18.30%	69.58%
	clara	25.89%	-2.70%	-	-16.21%	3.24%	9.17%	12.88%	16.30%	15.60%	66.88%
	spectral	42.10%	13.51%	16.21%	-	19.45%	25.38%	29.09%	32.51%	31.81%	83.09%
	hcmodel	22.65%	-5.94%	-3.24%	-19.45%	-	5.93%	9.64%	13.06%	12.36%	63.64%
	subspace	16.72%	-11.87%	-9.17%	-25.38%	-5.93%	-	3.71%	7.13%	6.43%	57.71%
	optics	13.01%	-15.58%	-12.88%	-29.09%	-9.64%	-3.71%	-	3.42%	2.72%	54.00%
	dbscan	9.59%	-19.00%	-16.30%	-32.51%	-13.06%	-7.13%	-3.42%	-	-0.70%	50.58%
	EM	10.29%	-18.30%	-15.60%	-31.81%	-12.36%	-6.43%	-2.72%	0.70%	-	51.28%
ARI	hierarchical	-	-31.58%	-29.06%	-46.82%	-18.27%	-30.42%	-0.58%	-9.14%	-10.07%	21.34%
	k-means	31.58%	-	2.52%	-15.24%	13.31%	1.16%	31.00%	22.44%	21.51%	52.92%
	clara	29.06%	-2.52%	-	-17.76%	10.79%	-1.36%	28.48%	19.92%	18.99%	50.40%
	spectral	46.82%	15.24%	17.76%	-	28.55%	16.40%	46.24%	37.68%	36.75%	68.16%
	hcmodel	18.27%	-13.31%	-10.79%	-28.55%	-	-12.15%	17.69%	9.13%	8.20%	39.61%
	subspace	30.42%	-1.16%	1.36%	-16.40%	12.15%	-	29.84%	21.28%	20.35%	51.76%
	optics	0.58%	-31.00%	-28.48%	-46.24%	-17.69%	-29.84%	-	-8.56%	-9.49%	21.92%
	dbscan	9.14%	-22.44%	-19.92%	-37.68%	-9.13%	-21.28%	8.56%	-	-0.93%	30.48%
	EM	10.07%	-21.51%	-18.99%	-36.75%	-8.20%	-20.35%	9.49%	0.93%	-	31.41%
Jaccard	hierarchical	-	-17.46%	-14.36%	-30.97%	-8.20%	-17.94%	9.54%	-1.62%	-2.93%	31.95%
	k-means	17.46%	-	3.10%	-13.51%	9.26%	-0.48%	27.00%	15.84%	14.53%	48.81%
	clara	14.36%	-3.10%	-	-16.61%	6.16%	-3.58%	23.90%	12.74%	11.43%	45.71%
	spectral	30.97%	13.51%	16.61%	-	22.77%	13.03%	40.51%	29.35%	28.04%	62.32%
	hcmodel	8.20%	-9.26%	-6.16%	-22.77%	-	-9.74%	17.74%	6.58%	5.27%	39.55%
	subspace	17.94%	0.48%	3.58%	-13.03%	9.74%	-	27.48%	16.32%	15.01%	49.24%
	optics	-9.54%	-27.00%	-23.90%	-40.51%	-17.74%	-27.48%	-	-11.16%	-12.47%	21.81%
	dbscan	1.62%	-15.84%	-12.74	-29.35%	-6.58%	-16.32%	11.16%	-	-1.31%	32.97%
	EM	2.93%	-14.53%	-11.43%	-28.04%	-5.27%	-15.01%	12.47%	1.31%	-	34.28%
FM	hierarchical	-	-14.98%	-10.64%	-25.57%	-6.47%	-15.22%	9.25%	-0.53%	-0.41%	49.44%
	k-means	14.90%	-	4.26%	-10.67%	8.43%	-0.32%	24.15%	14.37%	14.49%	64.34%
	clara	10.64%	-4.26%	-	-14.93%	4.17%	-4.58%	19.89%	10.11%	10.23%	60.08%
	spectral	25.57%	10.67%	14.93%	-	19.10%	10.35%	34.82%	25.04%	25.16%	75.01%
	hcmodel	6.47%	-8.43%	-4.17%	-19.10%	-	-8.75%	15.72%	5.94%	6.06%	55.91%
	subspace	15.22%	0.32%	4.58%	-10.35%	8.75	-	24.47%	14.69%	14.81%	64.66%
	optics	-9.25%	-24.15%	-19.89%	-34.82%	-15.72%	-24.47%	-	-9.78%	-9.66%	40.19%
	dbscan	0.53%	-14.37%	-10.11%	-25.04%	-5.94%	-14.69%	9.78%	-	0.12%	49.97%
	EM	0.41%	-14.49%	-10.23%	-25.16%	-6.06%	-14.81%	9.66%	-0.12%	-	49.85%

https://doi.org/10.1371/journal.pone.0210236.t002

of the table indicates the average performance of each algorithm. We note that the averages were taken over all generated datasets.

The results shown in Table 2 indicate that the spectral algorithm tends to outperform the other algorithms by at least 10%. On the other hand, the hierarchical method attained lower performance in most of the considered cases. Another interesting result is that the k-means and clara provided equivalent performance when considering all datasets. In the light of the results, the spectral method could be preferred when no optimitization of parameters values is performed.

One-dimensional analysis

The objective of the one-dimensional analysis is to verify how sensitive the accuracy of the clustering algorithms is to the variation of a single parameter. In addition, this analysis is also useful to verify if a very simple optimization strategy can lead to significant improvements in performance. For the one-dimensional analysis, we considered the databases DB2C2F (with $\alpha = 2.5$), DB10C2F (with $\alpha = 4.3$), DB2C10F (with $\alpha = 1.16$), DB10C10F (with $\alpha = 1.75$), DB2C200F (with $\alpha = 0.87$) and DB10C200F (with $\alpha = 1.09$). For each parameter, we varied its values while keeping the other parameters at their default configuration. The effect of varying the values of a single parameter P was quantified by comparing the obtained accuracy $\Gamma(x)$ when the parameter takes the value x and the accuracy Γ_{def} achieved with the default configuration of parameters. The improvement in performance was quantified in terms of the average ($\langle S \rangle$) and maximum value (max S), given by

$$\langle S \rangle = \frac{1}{n_P} \sum_x (\Gamma(x) - \Gamma_{\text{def}}), \tag{10}$$

$$\max S = \max_x (\Gamma(x) - \Gamma_{\text{def}}), \tag{11}$$

where n_P is the cardinality of all possible values taken by the parameter P in our experiments. We also measured the sensitivity of varying the values of P using the standard deviation ΔS:

$$\Delta S = \left[\frac{1}{n_P} \sum_x (\Gamma(x) - \Gamma_{\text{def}} - \langle S \rangle)^2 \right]^{1/2}. \tag{12}$$

In addition to the aforementioned quantities, we also measured, for each dataset, the maximum accuracy obtained when varying each single parameter of the algorithm. We then calculate the average of maximum accuracies, $\langle \max \text{Acc} \rangle$, obtained over all considered datasets. In Table 3, we show the values of $\langle S \rangle$, max S, ΔS and $\langle \max \text{Acc} \rangle$ for datasets containing two features. When considering a two-class problem (DB2C2F), a significant improvement in performance ($\langle S \rangle = 10.75\%$ and $\langle S \rangle = 13.35\%$) was observed when varying parameter *modelName*, *minPts* and *kpar* of, respectively, the EM, optics and spectral methods. For all other cases, only minor average gain in performance was observed. For the 10-class problem, we notice that an inadequate value for parameter *method* of the hierarchical algorithm can lead to substantial loss of accuracy (16.15% on average). In most cases, however, the average variation in performance was small.

In Table 4, we show the values of $\langle S \rangle$, max S, ΔS and $\langle \max \text{Acc} \rangle$ for datasets described by 10 features. For the the two-class clustering problem, a moderate improvement can be observed for the k-means, hierarchical and optics algorithm through the variation of, respectively, parameter *nstart*, *method* and *minPts*. A large increase in accuracy was observed when varying parameter *modelName* of the EM method. Changing the *modelName* used by the algorithm led to, on average, an improvement of 18.8%. A similar behavior was obtained when the number of classes was set to $C = 10$. For 10 classes, the variation of *method* in the hierarchical algorithm provided an average improvement of 6.72%. A high improvement was also observed when varying parameter *modelName* of the EM algorithm, with an average improvement of 13.63%.

Differently from the parameters discussed so far, the variation of some parameters plays a minor role in the discriminative power of the clustering algorithms. This is the case, for instance, of parameters *kernel* and *iter* of the spectral clustering algorithm and parameter *iter.max* of the kmeans clustering. In some cases, the effect of a unidimensional variation of

Table 3. One-parameter analysis performed in DB2C2F and DB10C2F. This analysis is based on the performance (measured through the ARI index) obtained when varying a single parameter of the clustering algorithm, while maintaining the others in their default configuration. $\langle S \rangle$, max S, ΔS are associated with the average, standard deviation and maximum difference between the performance obtained when varying a single parameter and the performance obtained for the default parameter values. We also measure $\langle$max Acc$\rangle$, the average of best ARI values obtained when varying each parameter, where the average is calculated over all considered datasets.

		DB2C2F				DB10C2F			
Algorithm	**Parameter**	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)
k-means	iter.max	0.05	2.37	14.46	51.5	0.04	0.91	4.49	47.3
k-means	nstart	1.98	5.62	16.73	51.9	1.24	1.98	6.80	47.9
k-means	algorithm	0.29	2.46	6.63	49.8	-0.92	1.29	0.65	45.0
clara	metric	-1.52	8.10	11.27	49.6	-3.66	5.36	5.10	42.5
clara	samples	-0.10	3.82	6.39	52.3	-0.21	3.03	7.48	47.5
clara	sampsize	-2.78	12.96	27.31	54.0	-0.54	2.92	4.88	47.1
clara	rngR	-0.16	3.19	4.19	51.0	-4.53	4.04	-0.03	41.7
hierarchical	metric	5.27	22.28	63.65	23.3	1.83	3.64	9.26	42.3
hierarchical	method	2.07	36.90	100.0	57.2	-16.15	21.26	15.89	46.5
hierarchical	par.method	0.0	0.0	0.0	18.0	0.0	0.0	0.0	40.5
spectral	kernel	-0.61	10.42	39.45	43.7	-0.3	2.84	6.78	48.0
spectral	kpar	7.36	16.78	33.3	44.6	-1.83	3.16	3.35	43.5
spectral	iter	1.14	19.19	85.34	54.1	0.06	2.62	5.84	47.9
spectral	mod.simple	-2.11	9.7	33.32	43.0	0.54	2.02	4.5	47.7
hcmodel	modelName	-2.41	19.48	29.89	60.0	-0.56	3.26	6.44	48.4
hcmodel	use	-2.14	10.14	12.58	57.4	-0.50	1.11	2.19	47.5
EM	z	1.71	8.77	19.34	33.9	7.04	8.30	28.17	45.4
EM	modelName	10.75	26.18	66.64	64.4	0.14	6.25	16.20	45.0
optics	eps	0.0	0.03	0.04	16.6	-10.06	8.67	0.02	5.4
optics	minPts	11.35	18.42	72.97	45.7	-4.97	15.5	30.32	13.7
optics	Xi	-0.16	1.7	3.05	17.4	-10.07	8.67	1.52	5.6
dbscan	minPts	3.68	11.68	53.0	35.4	-0.9	6.92	21.05	26.7
dbscan	eps	0.49	14.91	54.42	33.9	-2.81	5.7	18.91	27.1

https://doi.org/10.1371/journal.pone.0210236.t003

parameter resulted in reduction of performance. For instance, the variation of *min.individuals* and *models* of the subspace algorithm provided an average loss of accuracy on the order of $\langle S \rangle$ = 20%, depending on the dataset. Similar behavior is observed for the dbscan method, for which the variation of *minPts* causes and average loss of accuracy of 20.32%. Parameters *metric* and *rngR* of the clara algorithm also led to marked decrease in performance.

In Table 5, we show the values of $\langle S \rangle$, max S, ΔS and $\langle$max Acc$\rangle$ for datasets described by 200 features. For the two-class clustering problem, a significant improvement in performance was observed when varying *nstart* in the k-means method, *method* in the hierarchical algorithm, *modelName* in the hcmodel method and *modelName* in the EM method. On the other hand, when varying *metric*, *min.individuals* and *use* in, respectively, the clara, subspace, and hcmodel methods an average loss of accuracy larger than 10% was verified. The largest loss of accuracy happens with parameter *minPts* (49.47%) of the dbscan method. For the 10-class problem, similar results were observed, with the exception of the clara method, for which any parameter change resulted in a large loss of accuracy.

Multi-dimensional analysis

A complete analysis of the performance of a clustering algorithm requires the simultaneous variation of all of its parameters. Nevertheless, such a task is difficult to do in practice, given

Table 4. One-parameter analysis performed in DB2C10F and DB10C10F. This analysis is based on the performance obtained when varying a single parameter, while maintaining the others in their default configuration. $\langle S \rangle$, max S, ΔS are associated with the average, standard deviation and maximum difference between the performance obtained when varying a single parameter and the performance obtained for the default parameter values. We also measure $\langle$max Acc$\rangle$, the average of best ARI values obtained when varying each parameter, where the average is calculated over all considered datasets.

		DB2C10F				DB10C10F			
Algorithm	**Parameter**	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)
k-means	iter.max	0.30	8.13	36.36	53.2	0.14	1.92	6.41	56.6
k-means	nstart	5.26	12.0	36.36	53.5	2.68	2.65	9.43	57.8
k-means	algorithm	-0.35	6.72	25.5	42.3	-2.11	3.3	2.71	52.7
clara	metric	-10.9	22.31	25.05	51.8	-16.63	6.84	-5.1	37.6
clara	samples	1.04	8.94	25.05	60.4	-4.83	8.96	10.26	51.9
clara	sampsize	0.44	13.94	37.31	61.0	-4.46	9.97	14.18	57.6
clara	rngR	-2.89	15.08	25.05	56.9	-14.75	6.29	-5.21	39.3
hierarchical	metric	4.82	21.46	96.0	9.7	1.15	8.52	27.18	19.2
hierarchical	method	8.76	21.93	100.0	43.7	6.72	25.52	71.1	61.5
hierarchical	par.method	0.00	0.00	0.00	0.00	0.0	0.0	0.0	13.8
spectral	kernel	0.64	15.91	50.56	87.9	1.3	7.13	15.81	82.3
spectral	kpar	-1.08	16.88	50.56	88.1	-2.25	5.81	6.03	71.7
spectral	iter	-0.96	15.91	50.56	87.9	0.45	7.27	20.01	79.8
spectral	mod.simple	3.36	15.72	50.56	87.9	-1.35	7.55	14.24	78.7
subspace	models	-1.77	36.80	97.4	100.0	-22.44	8.92	-6.7	69.6
subspace	init	-0.78	23.47	97.4	99.5	-0.57	9.29	11.13	87.4
subspace	algo	-1.32	1.99	0.27	88.9	0.7	1.1	1.9	87.4
subspace	min.individuals	-26.9	43.17	10.73	90.9	-12.32	16.6	7.78	89.1
hcmodel	modelName	3.70	24.23	75.6	51.3	3.63	4.89	14.4	61.5
hcmodel	use	-0.92	17.68	51.47	49.1	-1.86	6.09	10.69	55.9
EM	z	1.68	8.62	18.99	29.9	-0.35	5.49	15.06	43.3
EM	modelName	18.80	31.93	96.62	100.0	13.63	16.09	64.52	91.6
optics	eps	0.0	0.03	0.04	32.1	-0.01	0.02	0.03	35.8
optics	minPts	6.96	15.35	52.38	55.1	7.16	11.72	31.5	53.2
optics	Xi	-1.11	3.44	7.23	33.5	0.43	1.51	3.9	37.6
dbscan	minPts	-19.61	30.78	11.19	45.2	-20.32	17.66	9.67	43.0
dbscan	eps	-1.53	17.65	59.26	55.2	-4.72	14.32	35.4	52.0

https://doi.org/10.1371/journal.pone.0210236.t004

the large number of parameter combinations that need to be taken into account. Therefore, here we consider a random variation of parameters aimed at obtaining a sampling of each algorithm performance for its complete multi-dimensional parameter space.

The methodology is applied as follows. Considering the one-dimensional variation of parameters, presented in the previous section, we identify the parameter bounds, $[P_{\min}, P_{\max}]$, where the classification either does not significantly change anymore or provides substantially smaller performance when compared to the default parameter value. Such bounds define the interval where the parameter will be randomly sampled. In order to generate the values for a given set of parameters $P^{(1)}, P^{(2)}, \ldots, P^{(n)}$ of an algorithm, we randomly sample each parameter $P^{(i)}$ according to a uniform distribution defined in the interval $[P^{(i)}_{\min}, P^{(i)}_{\max}]$. This procedure generates sets of parameter values, which are then used to evaluate the performance of the algorithms. For each algorithm, 500 sets of parameters were generated.

The performance of the algorithms for the different sets of parameters was evaluated according to the following procedure. Consider the histogram of ARI values obtained for the

Table 5. One-parameter analysis performed in DB2C200F and DB10C200F. This analysis is based on the performance obtained when varying a single parameter, while maintaining the others in their default configuration. $\langle S \rangle$, max S, ΔS are associated with the average, standard deviation and maximum difference between the performance obtained when varying a single parameter and the performance obtained for the default parameter values. We also measure $\langle$max Acc$\rangle$, the average of best ARI values obtained when varying each parameter.

		DB2C200F				DB10C200F			
Algorithm	**Parameter**	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)	$\langle S \rangle$ (%)	ΔS (%)	max S (%)	$\langle$max Acc$\rangle$ (%)
k-means	iter.max	0.08	8.26	33.51	62.4	-0.01	3.98	13.59	71.7
k-means	nstart	10.66	28.51	71.14	77.9	15.05	8.42	31.49	90.7
k-means	algorithm	-9.01	8.52	8.07	38.5	-5.39	4.4	6.45	60.4
clara	metric	-17.6	26.37	20.66	54.8	-31.72	12.38	-10.84	25.5
clara	samples	6.96	17.77	31.62	77.1	-26.54	10.57	-3.01	35.2
clara	sampsize	-1.17	24.21	34.23	79.3	-10.34	21.99	30.35	69.5
clara	rngR	-14.6	23.73	20.66	62.0	-35.55	11.33	-10.84	24.1
hierarchical	metric	0.00	0.00	0.00	0.0	0.0	0.0	0.0	0.0
hierarchical	method	12.63	31.36	100.0	71.9	17.25	35.22	99.98	92.1
hierarchical	par.method	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
spectral	kernel	0.19	29.84	40.2	94.0	-0.33	4.6	10.99	82.4
spectral	kpar	4.32	23.57	40.2	100.0	-3.61	5.59	10.99	76.5
spectral	iter	0.23	30.01	40.2	93.0	0.08	4.08	10.99	81.5
spectral	mod.simple	-1.0	33.26	40.2	93.0	0.43	4.58	10.99	83.0
subspace	models	-1.46	22.6	68.96	82.0	-12.26	25.34	63.42	79.0
subspace	init	0.54	22.22	45.52	55.7	-0.07	16.83	34.49	46.9
subspace	algo	1.41	13.69	37.79	45.7	10.44	19.3	37.09	57.8
subspace	min.individuals	-12.74	24.61	65.52	53.4	-3.4	18.16	31.59	55.3
hcmodel	modelName	20.12	35.47	92.61	74.7	30.54	31.34	71.77	95.10
hcmodel	use	-11.46	24.1	4.65	32.4	-6.23	13.70	18.31	36.90
EM	z	4.79	15.7	49.48	31.8	-3.38	4.44	1.76	33.6
EM	modelName	17.3	21.93	73.35	69.3	14.93	25.26	51.29	77.5
optics	eps	0.01	13.60	16.00	38.0	0.0	5.77	12.36	54.9
optics	minPts	1.71	19.83	42.3	63.5	9.92	14.84	33.76	81.4
optics	Xi	-0.59	13.72	21.80	38.9	0.17	5.89	12.46	57.8
dbscan	minPts	-49.47	30.8	18.81	70.7	-33.16	34.3	12.35	78.4
dbscan	eps	-13.4	31.09	24.41	77.2	6.67	8.81	24.59	87.6

https://doi.org/10.1371/journal.pone.0210236.t005

random sampling of parameters for the k-means algorithm, shown in Fig 8. The red dashed line indicates the ARI value obtained for the default parameters of the algorithm. The light blue shaded region indicates the parameters configurations where the performance of the algorithm improved. From this result we calculated four main measures. The first, which we call p-value, is given by the area of the blue region divided by the total histogram area, multiplied by 100 in order to result in a percentage value. The p-value represents the percentage of parameter configurations where the algorithm performance improved when compared to the default parameters configuration. The second, third and fourth measures are given by the mean, $\langle R \rangle$, standard deviation, ΔR, and maximum value, max R, of the relative performance for all cases where the performance is improved (e.g. the blue shaded region in Fig 8). The relative performance is calculated as the difference in performance between a given realization of parameter values and the default parameters. The mean indicates the expected improvement of the algorithm for the random variation of parameters. The standard deviation represents the stability of such improvement, that is, how certain one is that the performance will be improved when

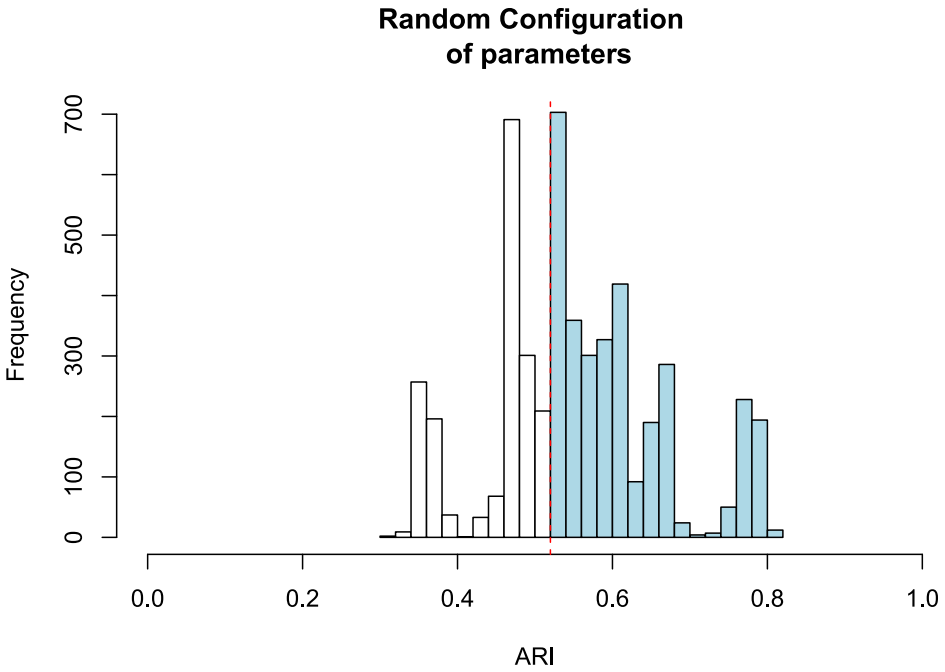

Fig 8. Distribution of ARI values obtained for the random sampling of the k-means parameters. The algorithm was applied to dataset DB10C10F, and 500 sets of parameters were drawn.

https://doi.org/10.1371/journal.pone.0210236.g008

doing such random variation. The maximum value indicates the largest improvement obtained when random parameters are considered. We also measured the average of the maximum accuracies ⟨max ARI⟩ obtained for each dataset when randomly selecting the parameters. In the S2 File of the supplementary material we show the distribution of ARI values obtained for the random sampling of parameters for all clustering algorithms considered in our analysis.

In Table 6 we show the performance (ARI) of the algorithms for dataset DB2C2F when applying the aforementioned random selection of parameters. The optics and EM methods are the only algorithms with a p-value larger than 50%. Also, a high average gain in performance was observed for the EM (22.1%) and hierarchical (30.6%) methods. Moderate improvement was observed for the hcmodel, kmeans, spectral, optics and dbscan algorithms.

Table 6. Multi-parameter analysis performed in dataset DB2C2F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column ⟨max ARI⟩ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	⟨max ARI⟩ (%)
EM	68.1	22.1	21.7	69.6	69.0
hierarchical	43.9	30.6	33.6	100.0	63.0
clara	29.2	4.9	4.7	27.3	60.0
hcmodel	25.8	13.3	8.2	29.9	63.0
k-means	21.7	13.2	3.9	21.4	55.0
spectral	47.0	14.7	13.9	85.3	59.0
optics	71.1	18.8	16.6	73.0	46.1
dbscan	39.8	17.0	15.9	51.5	41.9

https://doi.org/10.1371/journal.pone.0210236.t006

Table 7. Multi-parameter analysis performed in dataset DB10C2F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column $\langle$max ARI$\rangle$ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	$\langle$max ARI$\rangle$ (%)
EM	76.5	7.5	8.0	35.8	51.4
clara	54.7	2.3	1.8	9.0	51.0
k-means	77.7	2.2	1.7	6.9	49.0
hcmodel	28.4	2.7	2.5	6.8	49.0
hierarchical	36.6	5.9	4.2	21.7	49.0
spectral	40.0	2.3	1.6	8.0	52.0
optics	96.6	15.9	9.2	39.2	39.9
dbscan	30.6	3.7	3.6	21.9	29.1

https://doi.org/10.1371/journal.pone.0210236.t007

The performance of the algorithms for dataset DB10C2F is presented in Table 7. A high p-value was obtained for the optics (96.6%), EM (76.5%) and k-means (77.7%). Nevertheless, the average improvement in performance was relatively low for most algorithms, with the exception of the optics method, which led to an average improvement of 15.9%.

A more marked variation in performance was observed for dataset DB2C10F, with results shown in Table 8. The EM, kmeans, hierarchical and optics clustering algorithms resulted in a p-value larger than 50%. In such cases, when the performance was improved, the average gain in performance was, respectively, 30.1%, 18.0%, 25.9% and 15.5%. This means that the random variation of parameters might represent a valid approach for improving these algorithms. Actually, with the exception of clara and dbscan, all methods display significant average improvement in performance for this dataset. The results also show that a maximum accuracy of 100% can be achieved for the EM and subspace algorithms.

In Table 9 we show the performance of the algorithms for dataset DB10C10F. The p-values for the EM, clara, k-means and optics indicate that the random selection of parameters usually improves the performance of these algorithms. The hierarchical algorithm can be significantly improved by the considered random selection of parameters. This is a consequence of the

Table 8. Multi-parameter analysis performed in dataset DB2C10F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column $\langle$max ARI$\rangle$ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	$\langle$max ARI$\rangle$ (%)
EM	70.8	30.1	29.9	96.6	100.0
hierarchical	52.0	25.9	31.4	100.0	80.0
subspace	11.1	43.1	45.4	97.4	100.0
clara	44.9	6.5	6.3	37.3	70.0
hcmodel	38.4	31.8	25.3	81.2	70.0
k-means	50.1	18.0	7.1	62.4	60.0
spectral	48.9	9.9	18.5	31.5	90.0
optics	62.1	15.5	11.2	52.4	56.6
dbscan	43.3	5.0	6.5	23.8	50.9

https://doi.org/10.1371/journal.pone.0210236.t008

Table 9. Multi-parameter analysis performed in dataset DB10C10F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column $\langle$max ARI$\rangle$ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	$\langle$max ARI$\rangle$ (%)
EM	86.0	17.1	15.5	69.1	100.0
clara	72.1	7.1	4.4	22.8	68.0
k-means	83.0	4.3	2.3	12.0	60.0
hcmodel	53.4	7.4	4.6	17.5	64.0
hierarchical	51.9	32.1	19.4	72.9	68.0
spectral	49.1	5.6	4.1	19.7	87.3
subspace	10.7	7.5	4.7	21.4	99.3
optics	75.3	13.2	8.0	32.1	56.0
dbscan	7.4	6.1	4.7	20.9	46.2

https://doi.org/10.1371/journal.pone.0210236.t009

Table 10. Multi-parameter analysis performed in dataset DB2C200F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column $\langle$max ARI$\rangle$ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	$\langle$max ARI$\rangle$ (%)
EM	65.1	39.1	29.3	91.2	100.0
hierarchical	40.8	50.6	44.3	100.0	100.0
clara	44.9	23.0	8.9	34.2	81.3
hcmodel	35.3	62.3	27.7	92.6	75.5
k-means	65.6	35.4	24.6	71.1	89.1
spectral	15.9	28.4	36.3	75.4	100.0
Subspace	15.9	27.6	21.5	73.5	97.2
optics	56.8	16.1	13.6	53.5	64.4
dbscan	11.7	14.0	9.6	34.0	77.8

https://doi.org/10.1371/journal.pone.0210236.t010

default value of parameter *method*, which, as discussed in the previous section, is not appropriate for this dataset.

The performance of the algorithms for the dataset DB2C200F is presented in Table 10. A high p-value was obtained for the EM (65.1%) and k-means (65.6%) algorithms. The average

Table 11. Multi-parameter analysis performed in dataset DB10C200F. The p-value represents the probability that the classifier set with a random configuration of parameters outperform the same classifier set with its default parameters. $\langle R \rangle$, ΔR and max R represent the average, standard deviation and maximum value of the improvement obtained when random parameters are considered. Column $\langle$max ARI$\rangle$ indicates the average of the best accuracies obtained for each dataset.

Algorithm	p-value (%)	$\langle R \rangle$ (%)	ΔR (%)	max R (%)	$\langle$max ARI$\rangle$ (%)
EM	75.6	31.7	18.3	71.7	100.0
clara	73.8	22.8	11.9	58.0	93.1
k-means	96.8	18.9	8.1	42.1	99.5
hcmodel	51.8	60.0	12.0	76.2	100.0
hierarchical	68.6	36.8	43.2	100.0	99.2
spectral	46.0	10.0	6.0	26.0	100.0
optics	78.5	17.7	8.4	37.9	83.1
dbscan	24.9	10.7	4.9	24.6	88.5

https://doi.org/10.1371/journal.pone.0210236.t011

gain in performance in such cases was 39.1% and 35.4%, respectively. On the other hand, only in approximately 16% of the cases the Spectral and Subspace methods resulted in an improved ARI. Interestingly, the random variation of parameters led to, on average, large performance improvements for all algorithms.

In Table 11 we show the performance of the algorithms for dataset DB10C200F. A high p-value was obtained for all methods. On the other hand, the average improvement in accuracy tended to be lower than in the case of the dataset DB2C200F.

Conclusions

Clustering data is a complex task involving the choice between many different methods, parameters and performance metrics, with implications in many real-world problems [63, 103–108]. Consequently, the analysis of the advantages and pitfalls of clustering algorithms is also a difficult task that has been received much attention. Here, we approached this task focusing on a comprehensive methodology for generating a large diversity of heterogeneous datasets with precisely defined properties such as the distances between classes and correlations between features. Using packages in the R language, we developed a comparison of the performance of nine popular clustering methods applied to 400 artificial datasets. Three situations were considered: default parameters, single parameter variation and random variation of parameters. It should be nevertheless be borne in mind that all results reported in this work are respective to specific configurations of normally distributed data and algorithmic implementations, so that different performance can be obtained in other situations. Besides serving as a practical guidance to the application of clustering methods when the researcher is not an expert in data mining techniques, a number of interesting results regarding the considered clustering methods were obtained.

Regarding the default parameters, the difference in performance of clustering methods was not significant for low-dimensional datasets. Specifically, the Kruskal-Wallis test on the differences in performance when 2 features were considered resulted in a p-value of $p = 6.48 \times 10^{-7}$ (with a chi-squared distance of $\chi^2 = 41.50$). For 10 features, a p-value of $p = 1.53 \times 10^{-8}$ ($\chi^2 = 52.20$) was obtained. Considering 50 features resulted in a p-value of $p = 1.56 \times 10^{-6}$ for the Kruskal-Wallis test ($\chi^2 = 41.67$). For 200 features, the obtained p-value was $p = 2.49 \times 10^{-6}$ ($\chi^2 = 40.58$).

The Spectral method provided the best performance when using default parameters, with an Adjusted Rand Index (ARI) of 68.16%, as indicated in Table 2. In contrast, the hierarchical method yielded an ARI of 21.34%. It is also interesting that underestimating the number of classes in the dataset led to worse performance than in overestimation situations. This was observed for all algorithms and is in accordance with previous results [44].

Regarding single parameter variations, for datasets containing 2 features, the hierarchical, optics and EM methods showed significant performance variation. On the other hand, for datasets containing 10 or more features, most methods could be readily improved through changes on selected parameters.

With respect to the multidimensional analysis for datasets containing ten classes and two features, the performance of the algorithms for the multidimensional selection of parameters was similar to that using the default parameters. This suggests that the algorithms are not sensitive to parameter variations for this dataset. For datasets containing two classes and ten features, the EM, hcmodel, subspace and hierarchical algorithm showed significant gain in performance. The EM algorithm also resulted in a high p-value (70.8%), which indicates that many parameter values for this algorithm can provide better results than the default configuration. For datasets containing ten classes and ten features, the improvement was significantly

Table 12. Summary table for the performance of clustering algorithms in datasets DB2C2F and DB10C2F. ARI_{def} represents the average accuracy obtained when considering the default parameters of the algorithms. ARI_{best_p} represents the average of the best accuracies obtained when varying a single parameter. ARI_{best_r} represents the average of the best accuracies obtained when parameters are randomly selected.

		DB2C2F			DB10C2F		
#	Algorithm	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)
3	EM	32.2	64.4	69.0	38.4	45.4	51.4
2	spectral	37.2	54.1	59.0	45.4	48.0	52.0
4	clara	51.1	54.0	60.0	46.2	47.5	51.0
5	hcmodel	54.0	60.0	63.0	47.1	48.4	49.0
6	k-means	48.1	51.9	55.0	45.2	47.9	49.0
7	hierarchical	18.0	57.2	63.0	40.5	46.5	49.0
8	optics	16.6	45.7	46.1	15.5	13.7	39.9
9	dbscan	22.5	35.4	41.9	22.7	27.1	29.1

https://doi.org/10.1371/journal.pone.0210236.t012

lower for almost all the algorithms, with the exception of the hierarchical clustering. When a large number of features was considered, such as in the case of the datasets containing 200 features, large gains in performance were observed for all methods.

In Tables 12, 13 and 14 we show a summary of the best accuracies obtained during our analysis. The tables contain the best performance, measured as the ARI of the resulting partitions, achieved by each algorithm in the three considered situations (default, one- and multi-dimensional adjustment of parameters). The results are respective to datasets DB2C2F, DB10C2F, DB2C10F, DB10C10F, DB2C200F and DB10C200F. We observe that, for datasets containing 2 features, the algorithms tend to show similar performance, specially when the number of classes is increased. For datasets containing 10 features or more, the spectral algorithm seems to consistently provide the best performance, although the EM, hierarchical, k-means and subspace algorithms can also achieve similar performance with some parameter tuning. It should be observed that several clustering algorithms, such as optics and dbscan, aim at other data distributions such as elongated or S-shaped [72, 74]. Therefore, different results could be obtained for non-normally distributed data.

Table 13. Summary table for the performance of clustering algorithms in datasets DB2C10F and DB10C10F. ARI_{def} represents the average accuracy obtained when considering the default parameters of the algorithms. ARI_{best_p} represents the average of the best accuracies obtained when varying a single parameter. ARI_{best_r} represents the average of the best accuracies obtained when parameters are randomly selected.

		DB2C10F			DB10C10F		
#	Algorithm	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)
1	subspace	89.9	100.0	100.0	86.1	89.1	99.3
3	EM	23.4	100.0	100.0	40.9	91.6	100.0
2	spectral	82.4	88.1	90.0	70.9	82.3	87.3
4	clara	53.0	61.0	70.0	51.9	57.6	68.0
5	hcmodel	34.2	51.3	70.0	54.2	61.5	64.0
6	k-means	36.6	53.5	60.0	52.0	57.8	60.0
7	hierarchical	0.0	43.7	80.0	13.8	61.5	68.0
8	optics	32.1	55.1	56.6	35.8	53.2	56.0
9	dbscan	39.4	55.2	50.9	41.1	52.0	46.2

https://doi.org/10.1371/journal.pone.0210236.t013

Table 14. Summary table for the performance of clustering algorithms in datasets DB2C200F and DB10C200F. ARI_{def} represents the average accuracy obtained when considering the default parameters of the algorithms. ARI_{best_p} represents the average of the best accuracies obtained when varying a single parameter. ARI_{best_r} represents the average of the best accuracies obtained when parameters are randomly selected.

		DB2C200F			DB10C200F		
#	Algorithm	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)	ARI_{def} (%)	ARI_{best_p} (%)	ARI_{best_r} (%)
1	subspace	36.4	82.0	97.2	33.1	79.0	-
3	EM	17.6	69.3	100.0	33.6	77.5	100.0
2	spectral	92.4	100.0	100.0	76.7	83.0	100.0
4	clara	66.4	79.3	81.3	56.1	69.5	93.1
5	hcmodel	31.7	74.7	75.5	34.1	95.1	100.0
6	k-means	40.0	77.9	89.1	60.9	90.7	99.5
7	hierarchical	0.2	71.9	100.0	0.04	92.1	100.0
8	optics	38.0	63.5	64.4	54.9	81.4	83.1
9	dbscan	66.5	77.2	77.8	69.0	87.6	88.5

https://doi.org/10.1371/journal.pone.0210236.t014

Other algorithms could be compared in future extensions of this work. An important aspect that could also be explored is to consider other statistical distributions for modeling the data. In addition, an analogous approach could be applied to semi-supervised classification.

Supporting information

S1 File. Description of the clustering algorithms' parameters. We provide a brief description about the parameters of the clustering algorithms considered in the main text.
(PDF)

S2 File. Clustering performance obtained for the random selection of parameters. The file contains figures showing the histograms of ARI values obtained for identifying the clusters of, respectively, datasets DB10C10F and DB2C10F using a random selection of parameters. Each plot corresponds to a clustering method considered in the main text.
(PDF)

Author Contributions

Conceptualization: Cesar H. Comin, Luciano da F. Costa.

Data curation: Mayra Z. Rodriguez, Cesar H. Comin.

Formal analysis: Mayra Z. Rodriguez, Cesar H. Comin, Luciano da F. Costa.

Funding acquisition: Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, Francisco A. Rodrigues.

Investigation: Mayra Z. Rodriguez, Cesar H. Comin.

Methodology: Mayra Z. Rodriguez, Cesar H. Comin, Luciano da F. Costa.

Project administration: Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, Francisco A. Rodrigues.

Resources: Luciano da F. Costa.

Software: Mayra Z. Rodriguez.

Supervision: Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, Francisco A. Rodrigues.

Validation: Mayra Z. Rodriguez, Cesar H. Comin, Dalcimar Casanova, Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, Francisco A. Rodrigues.

Visualization: Mayra Z. Rodriguez.

Writing – original draft: Mayra Z. Rodriguez, Cesar H. Comin, Dalcimar Casanova, Diego R. Amancio, Luciano da F. Costa.

Writing – review & editing: Mayra Z. Rodriguez, Cesar H. Comin, Dalcimar Casanova, Odemir M. Bruno, Diego R. Amancio, Luciano da F. Costa, Francisco A. Rodrigues.

References

1. Golder SA, Macy MW. Diurnal and Seasonal Mood Vary with Work, Sleep, and Daylength Across Diverse Cultures. Science. 2011; 333(6051):1878–1881. https://doi.org/10.1126/science.1202775 PMID: 21960633
2. Michel JB, Shen YK, Aiden AP, Veres A, Gray MK, Pickett JP, et al. Quantitative Analysis of Culture Using Millions of Digitized Books. Science. 2011; 331(6014):176–182. https://doi.org/10.1126/science.1199644 PMID: 21163965
3. Bollen J, Van de Sompel H, Hagberg A, Chute R. A Principal Component Analysis of 39 Scientific Impact Measures. PLoS ONE. 2009; 4(6):1–11. https://doi.org/10.1371/journal.pone.0006022
4. Amancio DR, Oliveira ON Jr, Costa LF. Three-feature model to reproduce the topology of citation networks and the effects from authors' visibility on their h-index. Journal of Informetrics. 2012; 6(3): 427—434. https://doi.org/10.1016/j.joi.2012.02.005
5. Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters. Commun ACM. 2008; 51(1):107–113. https://doi.org/10.1145/1327452.1327492
6. Viana MP, Amancio DR, Costa LF. On time-varying collaboration networks. Journal of Informetrics. 2013; 7(2):371—378. https://doi.org/10.1016/j.joi.2012.12.005
7. Aggarwal CC, Zhai C. In: Aggarwal CC, Zhai C, editors. A Survey of Text Clustering Algorithms. Boston, MA: Springer US; 2012. p. 77–128.
8. Ridgeway G, Madigan D. A Sequential Monte Carlo Method for Bayesian Analysis of Massive Datasets. Data Mining and Knowledge Discovery. 2003; 7(3):301–319. https://doi.org/10.1023/A:1024084221803 PMID: 19789656
9. Fayyad U, Piatetsky-Shapiro G, Smyth P. From data mining to knowledge discovery in databases. AI magazine. 1996; 17(3):37.
10. Bellazzi R, Zupan B. Predictive data mining in clinical medicine: current issues and guidelines. International journal of medical informatics. 2008; 77(2):81–97. https://doi.org/10.1016/j.ijmedinf.2006.11.006 PMID: 17188928
11. Abdullah Z, Herawan T, Ahmad N, Deris MM. Extracting highly positive association rules from students' enrollment data. Procedia-Social and Behavioral Sciences. 2011; 28:107–111. https://doi.org/10.1016/j.sbspro.2011.11.022
12. Khashei M, Bijari M. An artificial neural network (p, d, q) model for timeseries forecasting. Expert Systems with applications. 2010; 37(1):479–489. https://doi.org/10.1016/j.eswa.2009.05.044
13. Joachims T. Text categorization with support vector machines: Learning with many relevant features. In: European conference on machine learning. Springer; 1998. p. 137–142.
14. Witten IH, Frank E. Data Mining: Practical Machine Learning Tools and Techniques. 2nd ed. San Francisco: Morgan Kaufmann; 2005.
15. Wang Y, Fan Y, Bhatt P, Davatzikos C. High-dimensional pattern regression using machine learning: from medical images to continuous clinical variables. Neuroimage. 2010; 50(4):1519–1535. https://doi.org/10.1016/j.neuroimage.2009.12.092 PMID: 20056158
16. Blum AL, Langley P. Selection of relevant features and examples in machine learning. Artificial intelligence. 1997; 97(1):245–271. https://doi.org/10.1016/S0004-3702(97)00063-5
17. Jing L, Ng MK, Huang JZ. An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. IEEE Transactions on knowledge and data engineering. 2007; 19(8): 1026–1041. https://doi.org/10.1109/TKDE.2007.1048

18. Suzuki R, Shimodaira H. Pvclust: an R package for assessing the uncertainty in hierarchical clustering. Bioinformatics. 2006; 22(12):1540–1542. https://doi.org/10.1093/bioinformatics/btl117 PMID: 16595560
19. Camastra F, Verri A. A novel kernel method for clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2005; 27(5):801–805. https://doi.org/10.1109/TPAMI.2005.88 PMID: 15875800
20. Jung YG, Kang MS, Heo J. Clustering performance comparison using K-means and expectation maximization algorithms. Biotechnology & Biotechnological Equipment. 2014; 28(sup1):S44–S48. https://doi.org/10.1080/13102818.2014.949045
21. Kinnunen T, Sidoroff I, Tuononen M, Fränti P. Comparison of clustering methods: A case study of text-independent speaker modeling. Pattern Recognition Letters. 2011; 32(13):1604–1617. https://doi.org/10.1016/j.patrec.2011.06.023
22. Abbas OA. Comparisons Between Data Clustering Algorithms. Int Arab J Inf Technol. 2008; 5(3): 320–325.
23. Pirim H, Ekşioğlu B, Perkins AD, Yüceer Ç. Clustering of high throughput gene expression data. Computers & operations research. 2012; 39(12):3046–3061. https://doi.org/10.1016/j.cor.2012.03.008
24. Costa IG, Carvalho FdATd, Souto MAICPd. Comparative analysis of clustering methods for gene expression time course data. Genetics and Molecular Biology. 2004; 27:623—631. https://doi.org/10.1590/S1415-47572004000400025
25. de Souto MC, Costa IG, de Araujo DS, Ludermir TB, Schliep A. Clustering cancer gene expression data: a comparative study. BMC bioinformatics. 2008; 9(1):497. https://doi.org/10.1186/1471-2105-9-497 PMID: 19038021
26. Dougherty ER, Barrera J, Brun M, Kim S, Cesar RM, Chen Y, et al. Inference from clustering with application to gene-expression microarrays. Journal of Computational Biology. 2002; 9(1):105–126. https://doi.org/10.1089/10665270252833217 PMID: 11911797
27. Brohée S, van Helden J. Evaluation of clustering algorithms for protein-protein interaction networks. BMC Bioinformatics. 2006; 7(1):1–19.
28. Maulik U, Bandyopadhyay S. Performance evaluation of some clustering algorithms and validity indices. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2002; 24(12):1650–1654. https://doi.org/10.1109/TPAMI.2002.1114856
29. Fraley C, Raftery AE. How many clusters? Which clustering method? Answers via model-based cluster analysis. The computer journal. 1998; 41(8):578–588. https://doi.org/10.1093/comjnl/41.8.578
30. Halkidi M, Batistakis Y, Vazirgiannis M. On clustering validation techniques. Journal of intelligent information systems. 2001; 17(2-3):107–145. https://doi.org/10.1023/A:1012801612483
31. Jaccard P. Nouvelles recherches sur la distribution florale. Bulletin de la Société Vaudense des Sciences Naturelles. 1908; 44:223–270.
32. Lawrence H, Arabie P. Comparing partitions. Journal of Classification. 1985; 2(1):193–218. https://doi.org/10.1007/BF01908075
33. Fowlkes E B, Mallows C L. A Method for Comparing Two Hierarchical Clusterings. Journal of the American Statistical Association. 1983; 78(383):553–569.
34. Strehl A, Ghosh J, Cardie C. Cluster Ensembles—A Knowledge Reuse Framework for Combining Multiple Partitions. Journal of Machine Learning Research. 2002; 3:583–617.
35. Hirschberger M, Qi Y, Steuer RE. Randomly generating portfolio-selection covariance matrices with specified distributional characteristics. European Journal of Operational Research. 2007; 177(3): 1610–1625. https://doi.org/10.1016/j.ejor.2005.10.014
36. Amancio DR, Comin CH, Casanova D, Travieso G, Bruno OM, Rodrigues FA, et al. A systematic comparison of supervised classifiers. PloS one. 2014; 9(4):e94137. https://doi.org/10.1371/journal.pone.0094137 PMID: 24763312
37. Berkhin P. In: Kogan J, Nicholas C, Teboulle M, editors. A Survey of Clustering Data Mining Techniques. Berlin, Heidelberg: Springer Berlin Heidelberg; 2006. p. 25–71.
38. Hwang CR. Simulated annealing: theory and applications. Acta Applicandae Mathematicae. 1988; 12(1):108–111.
39. Goldberg DE, Holland JH. Genetic algorithms and machine learning. Machine learning. 1988; 3(2): 95–99. https://doi.org/10.1023/A:1022602019183
40. Hawkins DM. The problem of overfitting. Journal of chemical information and computer sciences. 2004; 44(1):1–12. https://doi.org/10.1021/ci0342472 PMID: 14741005
41. Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM computing surveys. 1999; 31(3): 264–323. https://doi.org/10.1145/331499.331504

42. R Development Core Team. R: A Language and Environment for Statistical Computing; 2006. Available from: http://www.R-project.org.
43. Kou G, Peng Y, Wang G. Evaluation of clustering algorithms for financial risk analysis using MCDM methods. Information Sciences. 2014; 275:1–12. https://doi.org/10.1016/j.ins.2014.02.137
44. Erman J, Arlitt M, Mahanti A. Traffic classification using clustering algorithms. In: Proceedings of the 2006 SIGCOMM workshop on mining network data. ACM; 2006. p. 281–286.
45. Mingoti SA, Lima JO. Comparing SOM neural network with Fuzzy c-means, K-means and traditional hierarchical clustering algorithms. European Journal of Operational Research. 2006; 174(3): 1742–1759. https://doi.org/10.1016/j.ejor.2005.03.039
46. Mangiameli P, Chen SK, West D. A comparison of SOM neural network and hierarchical clustering methods. European Journal of Operational Research. 1996; 93(2):402–417. https://doi.org/10.1016/0377-2217(96)00038-0
47. Parsons L, Haque E, Liu H. Evaluating subspace clustering algorithms. In: Workshop on Clustering High Dimensional Data and its Applications, SIAM Int. Conf. on Data Mining. Citeseer; 2004. p. 48–56.
48. Burdick D, Calimlim M, Gehrke J. MAFIA: A Maximal Frequent Itemset Algorithm for Transactional Databases. In: Proceedings of the 17th International Conference on Data Engineering. Washington, DC, USA: IEEE Computer Society; 2001. p. 443–452.
49. Parsons L, Haque E, Liu H. Subspace clustering for high dimensional data: a review. ACM SIGKDD Explorations Newsletter. 2004; 6(1):90–105. https://doi.org/10.1145/1007730.1007731
50. Verma D, Meila M. A comparison of spectral clustering algorithms. University of Washington Tech Rep UWCSE030501. 2003; 1:1–18.
51. UCI. breast-cancer-wisconsin;. Available from: https://http://archive.ics.uci.edu/ml/machine-learning-databases/breast-cancer-wisconsin/.
52. Ultsch A. Clustering wih som: U* c. In: Proceedings of the 5th Workshop on Self-Organizing Maps. vol. 2; 2005. p. 75–82.
53. Guha S, Rastogi R, Shim K. Cure: an efficient clustering algorithm for large databases. Information Systems. 2001; 26(1):35—58. https://doi.org/10.1016/S0306-4379(01)00008-4
54. Aggarwal CC, Reddy CK. Data Clustering: Algorithms and Applications. vol. 2. 1st ed. Chapman & Hall/CRC; 2013.
55. Karypis G, Han EH, Kumar V. Chameleon: Hierarchical clustering using dynamic modeling. Computer. 1999; 32(8):68–75. https://doi.org/10.1109/2.781637
56. Huang J, Sun H, Kang J, Qi J, Deng H, Song Q. ESC: An efficient synchronization-based clustering algorithm. Knowledge-Based Systems. 2013; 40:111–122. https://doi.org/10.1016/j.knosys.2012.11.015
57. Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, et al. Top 10 algorithms in data mining. Knowledge and information systems. 2008; 14(1):1–37. https://doi.org/10.1007/s10115-007-0114-2
58. Jain AK, Topchy A, Law MH, Buhmann JM. Landscape of clustering algorithms. In: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. vol. 1. IEEE; 2004. p. 260–263.
59. Jain AK. Data clustering: 50 years beyond K-means. Pattern Recognition Letters. 2010; 31(8): 651–666. https://doi.org/10.1016/j.patrec.2009.09.011
60. Steinley D. K-means clustering: a half-century synthesis. British Journal of Mathematical and Statistical Psychology. 2006; 59(1):1–34. https://doi.org/10.1348/000711005X48266 PMID: 16709277
61. Dunn JC. A fuzzy relative of the ISODATA process and its use in detecting compact well separated clusters. Cybernetics. 1973; 3:32–57. https://doi.org/10.1080/01969727308546046
62. Huang Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. Data mining and knowledge discovery. 1998; 2(3):283–304. https://doi.org/10.1023/A:1009769707641
63. Raykov YP, Boukouvalas A, Baig F, Little MA. What to Do When K-Means Clustering Fails: A Simple yet Principled Alternative Algorithm. PLoS ONE. 2016; 11(9):1–28. https://doi.org/10.1371/journal.pone.0162259
64. Kaufman L, Rousseeuw PJ. Finding groups in data: an introduction to cluster analysis. Series in Probability& Mathematical Statistics. 2009;.
65. Arthur D, Vassilvitskii S. k-means++: The advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics; 2007. p. 1027–1035.

66. Sequeira K, Zaki M. ADMIT: anomaly-based data mining for intrusions. In: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 2002. p. 386–395.
67. Williams GJ, Huang Z. Mining the knowledge mine. In: Australian Joint Conference on Artificial Intelligence. Springer; 1997. p. 340–348.
68. MacQueen J. Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. Berkeley, Calif: University of California Press; 1967. p. 281–297.
69. Hartigan JA, Wong MA. Algorithm AS 136: A k-means clustering algorithm. Journal of the Royal Statistical Society Series C. 1979; 28(1):100–108.
70. Kaufman L, Rousseeuw PJ. Finding Groups in Data: an introduction to cluster analysis. John Wiley & Sons; 1990.
71. Han J, Kamber M. Data Mining. Concepts and Techniques. vol. 2. 2nd ed. -: Morgan Kaufmann; 2006.
72. Ankerst M, Breunig MM, Kriegel HP, Sander J. OPTICS: Ordering Points to Identify the Clustering Structure. SIGMOD. 1999; 28(2):49–60. https://doi.org/10.1145/304182.304187
73. Ankerst M, Breunig MM, Kriegel HP, Sander J. OPTICS: Ordering Points To Identify the Clustering Structure. ACM Press; 1999. p. 49–60.
74. Ester M, Kriegel HP, Sander J, Xu X. A Density-based Algorithm for Discovering Clusters a Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. KDD'96. AAAI Press; 1996. p. 226–231.
75. Lance GN, Williams WT. A general theory of classificatory sorting strategies II. Clustering systems. The computer journal. 1967; 10(3):271–277. https://doi.org/10.1093/comjnl/10.3.271
76. Redner R, Walker H. Mixture densities, maximum likelihood and the em algorithm. SIAM Review. 1984; 26(6).
77. Dempster AP, Laird NM, Rubin DB. Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society Series B. 1977; 39(6).
78. Fraley C, Raftery AE. Model-based clustering, discriminant analysis, and density estimation. Journal of the American statistical Association. 2002; 97(458):611–631. https://doi.org/10.1198/016214502760047131
79. Fraley C, Raftery AE. MCLUST: Software for model-based cluster analysis. Journal of Classification. 1999; 16(2):297–306. https://doi.org/10.1007/s003579900058
80. Fraley C, Raftery EA. Enhanced Model-Based Clustering, Density Estimation, and Discriminant Analysis Software: MCLUST". Journal of Classification. 2003; 20(2):263–286. https://doi.org/10.1007/s00357-003-0015-3
81. Fraley C. Algorithms for model-based Gaussian hierarchical clustering. SIAM Journal on Scientific Computing. 1998; 20(1):270–281. https://doi.org/10.1137/S1064827596311451
82. Schwarz G. Estimating the dimension of a model. The annals of statistics. 1978; 6(2):461–464. https://doi.org/10.1214/aos/1176344136
83. Nascimento MC, De Carvalho AC. Spectral methods for graph clustering–a survey. European Journal of Operational Research. 2011; 211(2):221–231. https://doi.org/10.1016/j.ejor.2010.08.012
84. Filippone M, Camastra F, Masulli F, Rovetta S. A survey of kernel and spectral methods for clustering. Pattern recognition. 2008; 41(1):176–190. https://doi.org/10.1016/j.patcog.2007.05.018
85. Von Luxburg U. A tutorial on spectral clustering. Statistics and computing. 2007; 17(4):395–416. https://doi.org/10.1007/s11222-007-9033-z
86. Ng AY, Jordan MI, Weiss Y. On spectral clustering: Analysis and an algorithm. In: Advances in Neural Information Processing Systems 14. MIT Press; 2001. p. 849–856.
87. Dhillon IS, Guan Y, Kulis B. A unified view of kernel k-means, spectral clustering and graph cuts. Citeseer; 2004.
88. Kriegel HP, Kröger P, Zimek A. Subspace clustering. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. 2012; 2(4):351–364.
89. Sim K, Gopalkrishnan V, Zimek A, Cong G. A survey on enhanced subspace clustering. Data mining and knowledge discovery. 2013; 26(2):332–397. https://doi.org/10.1007/s10618-012-0258-x
90. Bergé L, Bouveyron C, Girard S. HDclassif: an R Package for Model-Based Clustering and Discriminant Analysis of High-Dimensional Data. Journal of Statistical Software. 2012; 46(6):1–29.

91. Bouveyron C, Girard S, Schmid C. High-dimensional data clustering. Computational Statistics & Data Analysis. 2007; 52(1):502–519. https://doi.org/10.1016/j.csda.2007.02.009

92. Łuczak M. Combining raw and normalized data in multivariate time series classification with dynamic time warping. Journal of Intelligent & Fuzzy Systems. 2018; 34(1):373–380. https://doi.org/10.3233/JIFS-171393

93. Guha S, Meyerson A, Mishra N, Motwani R, O'Callaghan L. Clustering data streams: Theory and practice. IEEE transactions on knowledge and data engineering. 2003; 15(3):515–528. https://doi.org/10.1109/TKDE.2003.1198387

94. Silva JA, Faria ER, Barros RC, Hruschka ER, De Carvalho AC, Gama J. Data stream clustering: A survey. ACM Computing Surveys. 2013; 46(1):13. https://doi.org/10.1145/2522968.2522981

95. Horn RA, Johnson CR. Matrix Analysis. 2nd ed. New York, NY, USA: Cambridge University Press; 2012.

96. Liu Y, Li Z, Xiong H, Gao X, Wu J. Understanding of internal clustering validation measures. In: Data Mining (ICDM), 2010 IEEE 10th International Conference on. IEEE; 2010. p. 911–916.

97. Lei Y, Bezdek JC, Romano S, Vinh NX, Chan J, Bailey J. Ground truth bias in external cluster validity indices. Pattern Recognition. 2017; 65:58–70. https://doi.org/10.1016/j.patcog.2016.12.003

98. Cover TM, Thomas JA. Elements of Information Theory. vol. 2. Wiley; 2012.

99. Strehl A, Ghosh J. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. Journal of machine learning research. 2002; 3:583–617.

100. Brun M, Sima C, Hua J, Lowey J, Carroll B, Suh E, et al. Model-based evaluation of clustering validation measures. Pattern recognition. 2007; 40(3):807–824. https://doi.org/10.1016/j.patcog.2006.06.026

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

102. McKight PE, Najab J. Kruskal-Wallis Test. Corsini Encyclopedia of Psychology. 2010;.

103. Arruda GF, Costa LF, Rodrigues FA. A complex networks approach for data clustering. Physica A: Statistical Mechanics and its Applications. 2012; 391(23):6174—6183. https://doi.org/10.1016/j.physa.2012.07.007

104. Naeni LM, Craig H, Berretta R, Moscato P. A Novel Clustering Methodology Based on Modularity Optimisation for Detecting Authorship Affinities in Shakespearean Era Plays. PLOS ONE. 2016; 11(8):1–27. https://doi.org/10.1371/journal.pone.0157988

105. Amancio DR. Authorship recognition via fluctuation analysis of network topology and word intermittency. Journal of Statistical Mechanics: Theory and Experiment. 2015; 2015(3):P03005. https://doi.org/10.1088/1742-5468/2015/03/P03005

106. Garcia C. BoCluSt: Bootstrap Clustering Stability Algorithm for Community Detection. PLOS ONE. 2016; 11(6):1–15. https://doi.org/10.1371/journal.pone.0156576

107. Colavizza G, Franceschet M. Clustering citation histories in the Physical Review. Journal of Informetrics. 2016; 10(4):1037—1051. https://doi.org/10.1016/j.joi.2016.07.009

108. Benaim M. A Stochastic Model of Neural Network for Unsupervised Learning. Europhysics Letters. 1992; 19(3):241. https://doi.org/10.1209/0295-5075/19/3/015