Unsupervised Learning Methods and Similarity Analysis in Chemoinformatics

50

Katarzyna Odziomek, Anna Rybinska, and Tomasz Puzyn

Contents

Abstract

In this chapter, we present an overview of various chemometric methods, appropriate for analyzing and interpreting data from social media, industry, academia, medicine, and other sources. We discuss unsupervised machine-learning techniques used for grouping (hierarchical cluster analysis, *k*-means) and exploring (principal component analysis, self-organizing Kohonen maps) all types of data, both quantitative and qualitative. For each method described in this chapter, we explain the basic concepts, provide a rudimentary algorithm, and present practical applications. All the examples are based on a set of molecular descriptors calculated for a selected group of persistent organic pollutants (POPs).

K. Odziomek (✉) • A. Rybinska • T. Puzyn
Laboratory of Environmental Chemometrics, Faculty of Chemistry, University of Gdańsk, Gdańsk, Poland
e-mail: k.odziomek@qsar.eu.org; rybinska@qsar.eu.org; puzi@qsar.eu.org

J. Leszczynski et al. (eds.), *Handbook of Computational Chemistry*,
DOI 10.1007/978-3-319-27282-5_53

Introduction

The vast amount of digital information generated every day in social media, industry, and academia necessitates the use of advanced techniques appropriate for processing, analysis, and interpretation of data.

There are two main types of data analysis algorithms, namely, supervised and unsupervised machine-learning methods. A simplified schematic of machine-learning methods is presented in Fig. 1. Supervised learning is used for modeling, i.e., making predictions with the help of a calibration, discrimination, or classification model, depending on the research problem. Unsupervised learningPrincipal component analysis (PCA), used for exploring the hidden data structures and relationships between variables, helps us find groups (clusters) of objects (samples) similar to each other or, conversely, significantly dissimilar from the rest, as defined by a selected metric. The main difference between the two methods is that when constructing a model, the unsupervised learning method utilizes only the explanatory (independent) variable matrix (**X**), while the supervised learning method takes also the response (dependent) variable (y) into account (Brown et al. 2009).

This chapter discusses methods of unsupervised machine learning, or pattern recognition, which are often used in problem solving in various fields of research such as chemistry, economics, forensic science, and medicine (Skwarzec et al. 2011; Li et al. 2013; Kountchev and Iantovics 2013; Golebiowski et al. 2014; Petushkova et al. 2014; Schnegg et al. 2015). It is meant as a brief overview of the most popular methods, along with examples, so as to facilitate an easier understanding of the topic.

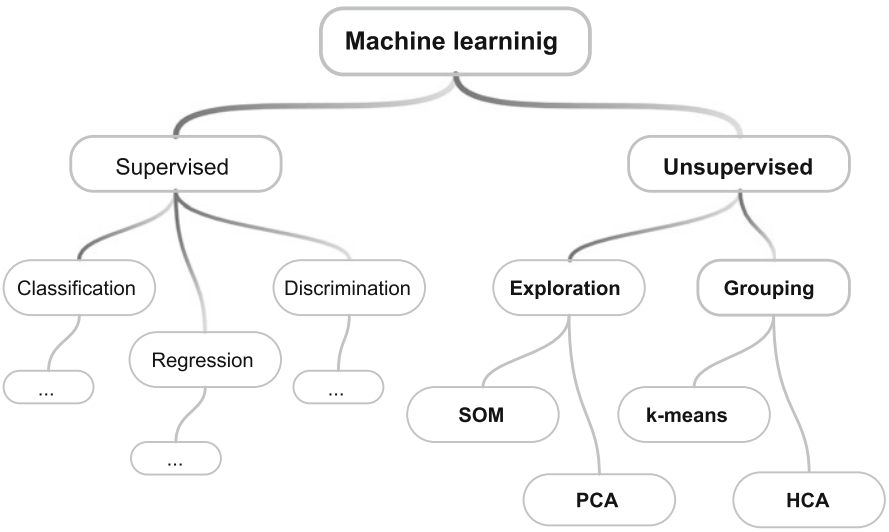

Fig. 1 An overview of data mining methods

Data

All examples given in this chapter are based on a data set on persistent organic pollutants (POPs). This data set contains 21 molecular descriptors (Table 1) calculated for 1436 chloro ("C") and bromo ("B") analogues of dibenzo-*p*-dioxins (PXDDs, where X stands either for C or B), dibenzofurans (PXDFs), biphenyls (PXBs), naphthalenes (PXNs), diphenyl ethers (PXDEs), and benzenes (XBs) (Gajewicz et al. 2010). Most of the examples in this chapter will be based on a reduced set of POPs, namely, polychlorinated naphthalenes PCNs, polychlorinated dibenzo-*p*-dioxins PCDDs, and polychlorinated dibenzofurans PCDFs (Fig. 2).

As the POP class membership of each sample is known beforehand, the authors chose to use this data as a tool for comparing the effectiveness and the performance of each clustering method presented in this chapter.

Data Preparation

In order to ensure the reliability and accuracy of the results, we must first evaluate the quality of the raw data. The purpose of such evaluation is identifying possible mistakes (errors) made during data collection and reducing the risk of propagating

Table 1 Molecular descriptors

Symbol	Definition of molecular descriptor	Unit
nAT	Number of atoms	–
nX	Number of halogen substituents	–
nO	Number of oxygen atoms	–
MW	Molecular weight	u
HOF	Standard heat of formation	$kcal \cdot mol^{-1}$
EE	Electronic energy	eV
Core	Core-core repulsion energy	eV
TE	Total energy	eV
TEp	Total energy of the corresponding cation	eV
VIP	Vertical ionization potential	eV
HOMO	Energy of the highest occupied molecular orbital	Hartree
LUMO	Energy of the lowest unoccupied molecular orbital	Hartree
D	Dipole moment	Debye
SAS	Solvent accessible surface	$Å^2$
MV	Molecular volume	$Å^3$
Qm	Lowest negative Mulliken partial charge on the molecule	–
Qp	Highest positive Mulliken partial charge on the molecule	–
P	Polarizability derived from the dipole moment	$Å^3$
EN	Mulliken electronegativity	Hartree
Hard	Parr and Pople's absolute hardness	Hartree
Shift	Schuurman MO shift alpha	Hartree

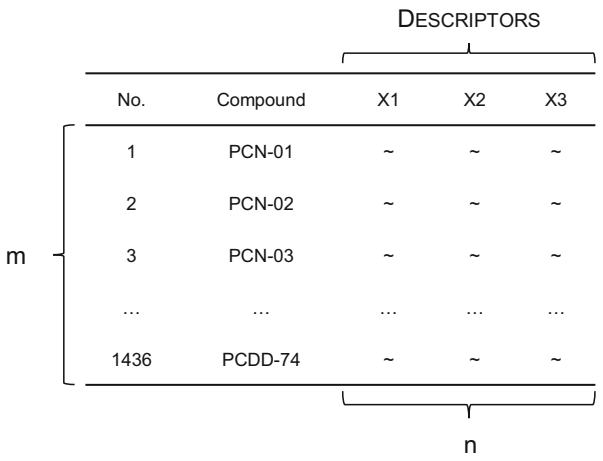

Fig. 2 A schematic representation of the example POP data set used in this chapter with m samples (molecules) and n variables (descriptors)

those errors during the next stages of analysis. Data control also allows us to assess whether any variable transformations (preprocessing) are necessary. In order to use pattern recognition methods presented in this chapter, we must first perform a specific type of data transformation, called standardization, autoscaling, or Z-transformation. Standardization is a way of centering and scaling data in such a way (Eq. 1) that the resulting variables have a mean value equal to 0 and the standard deviation equal to 1.

$$z_i = \frac{x_i - \mu}{\sigma} \tag{1}$$

where x_i is the sample value, z_i is the standardized sample value, μ is the mean value for all samples in the column, and σ is the standard deviation of all the values in the column.

It should be noted that, after this transformation, all the variables have mean value equal to 0 and variance equal to 1 (see Fig. 3). Thusly, the effect of unequal value ranges, caused by differing variable units, and has been nullified, as the standardized data are unitless. Each variable has equal significance (weight) and influence on the analysis results (Livingstone 2009).

Clustering

In order to explore the data, to identify and visualize their underlying structure, and to understand the relationships between objects (samples), we should employ clusteringClustering methods. Clustering is a way of looking for natural patterns or groups in the data, in other words, a way of determining the relative positions of

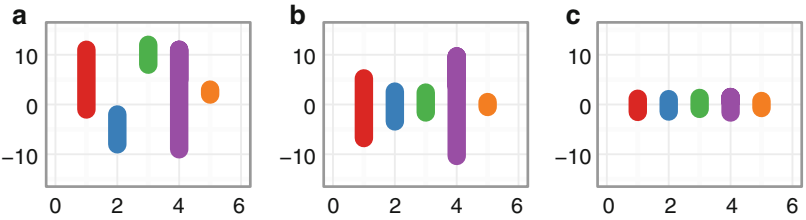

Fig. 3 Data transformation during autoscaling (**a**) raw data; (**b**) centered data, where $\mu = 0$; and (**c**) centered and scaled data, where $\mu = 0$, $\sigma = 1$

all objects in the multidimensional variable space. It is an unsupervised machine-learning method, that is to say, the size and membership of the groups are not known in advance. The groups are formed as a result of a clustering algorithm and are extracted from the data set accordingly to criteria selected by the user.

As all clustering methods are built on the concept of similarity, not only do they allow us to determine group membership but they also help us identify outliers. Ways of measuring similarity will be discussed in depth later in this chapter.

There are two main approaches to clustering. The first one, called *hierarchical*, produces levels of rank-ordered clusters. The second one, called *partitive*, sorts the data into a predefined number of clusters of equal importance (Everitt et al. 2011).

Similarity and Distance

Similarity can be expressed in terms of distance between two objects in the variable space. This space can be one-, two-, three-, or multidimensional, proportionally to the number of variables. It is quite intuitive to interpret similarity and distance as inverse concepts: the greater the distance between objects, the lesser their similarity. Two objects are considered to have similar properties and, consequently, belong to the same group, if the distance between them is sufficiently small (Brereton 2003).

The perception of similarity through distance, the way we interpret objects as similar or not, depends on the distance metric selected, variables used, and data transformation method applied.

Examples of different approaches to defining distances between objects, that is, different *similarity metrics*, are shown in Fig. 4. The *Euclidean distance* between the two objects, $i(1)$ and $j(2)$, denoted by line C, can be calculated according to Eq. 2:

$$D_{\text{Euclidean}} = \sqrt{\sum_{j=1}^{J} \left(x_{ij} - x_{kj}\right)^2} \tag{2}$$

where i, j are sample indices, n is the variable number, x_i is the i-th sample value of n-th variable, and x_i is the j-th sample value of n-th variable.

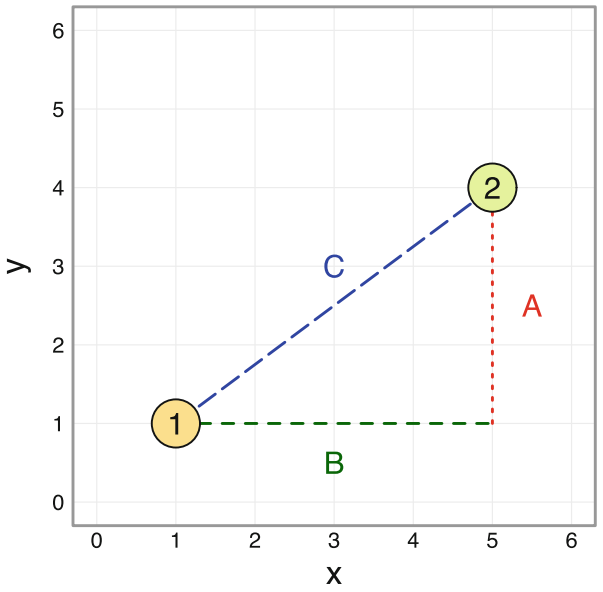

Fig. 4 Different approaches to defining distance metrics

In the presented example, the Euclidean distance between objects 1 and 2 is given as:

$$\mathrm{C} = \sqrt{\mathrm{B}^2 + \mathrm{A}^2}$$

$$\mathrm{C} = \sqrt{(5-1)^2 + (4-1)^2} = 5$$

The second distance metric represented in Fig. 4, known as *Manhattan* or *city block distance*, is calculated as follows (Eq. 3):

$$D_{\text{Manhattan}} = \sum_{j=1}^{J} \left| x_{ij} - x_{kj} \right| \tag{3}$$

In the example:

$$D_{\text{Manhatan}} = 4 + 3 = 7$$

The last distance metric, *Chebyshev*, can be determined according to Eq. 4:

$$D_{\text{Chebyshev}} = \max \left| x_{ij} - x_{kj} \right| \tag{4}$$

In our example, $|\mathrm{A}| < |\mathrm{B}|$; thus,

$$D_{\text{Chebyshev}} = |\mathrm{B}| = 4$$

Table 2 Most popular distance metrics

No.	Metric	Equation	Type	Data type
1	Euclidean	$\sqrt{\sum_{j=1}^{J}\left(x_{ij}-x_{kj}\right)^2}$	Dissimilarity	Interval/binary
2	Manhattan (city block)	$\sum_{j=1}^{J}\left\|x_{ij}-x_{kj}\right\|$	Dissimilarity	Interval
3	Chebyshev	$\max\left\|x_{ij}-x_{kj}\right\|$	Dissimilarity	Interval
4	Minkowski(order n)	$\sqrt[n]{\sum_{j=1}^{J}\left(x_{ij}-x_{kj}\right)^n}$	Dissimilarity	Interval
5	Canberra	$\sum_{j=1}^{J}\frac{\left\|x_{ij}+x_{kj}\right\|}{x_{ij}+x_{kj}}$	Dissimilarity	Interval
6	Mahalanobis	$\sqrt[n]{\sum_{j=1}^{J}\left(x_{ij}-x_{kj}\right)^n S^{-1}\left(x_{ij}-x_{kj}\right)^n}$	Dissimilarity	Interval
7	Cosine(Ochiai)	$\frac{\sum_{j=1}^{J}x_{ij}x_{kj}}{\sqrt{\sum_{j=1}^{J}x_{ij}^2\sum_{j=1}^{J}x_{kj}^2}}$	Similarity	Interval/binary
8	Pearson correlation coefficient	$\frac{\sum_{j=1}^{J}\left(x_{ij}-\overline{x}_i\right)\left(x_{kj}-\overline{x}_j\right)}{\sqrt{\sum_{j=1}^{J}\left(x_{ij}-\overline{x}_i\right)^2\sum_{j=1}^{J}\left(x_{kj}-\overline{x}_j\right)^2}}$	Similarity	Interval/binary
9	Squared Pearson correlation coefficient	$\left[\frac{\sum_{j=1}^{J}\left(x_{ij}-\overline{x}_i\right)\left(x_{kj}-\overline{x}_j\right)}{\sqrt{\sum_{j=1}^{J}\left(x_{ij}-\overline{x}_i\right)^2\sum_{j=1}^{J}\left(x_{kj}-\overline{x}_j\right)^2}}\right]^2$	Similarity	Interval/binary

Distance measures can emphasize similarity, as well as dissimilarity between the objects. The abovementioned metrics highlight the differences between samples and quantify dissimilarity. It should be noted that the choice of the distance measure depends also on the type of data available. A list of the most popular (dis)similarity metrics can be found in Table 2.

The measure of proximity between two clusters, or an object and an already formed cluster, is called *linkage* (Varmuza and Filzmoser 2009). Let us assume there are three clusters called C_1 (with n_1 objects), C_2 (with n_2 objects), and C_3 (with n_3 objects). If clusters C_2 and C_3 are aggregated to form a new single cluster called C_4

Table 3 Linkage types

Linkage type	Equation
Single [nearest neighbor]. Distance between two clusters is the minimum distance between any single observation in one cluster and any single observation in the other cluster	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \min\left(\boldsymbol{D}_{\mathrm{C_1C_2}}, \boldsymbol{D}_{\mathrm{C_1C_3}}\right)$
Complete [furthest neighbor]. Distance between two clusters is the maximum distance between any single observation in one cluster and any single observation in the other cluster	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \max\left(\boldsymbol{D}_{\mathrm{C_1C_2}}, \boldsymbol{D}_{\mathrm{C_1C_3}}\right)$
Centroid [unweighted pair group method centroid, UPGMC]. Euclidean distance between two clusters is the distance between the cluster centroids (means, middle points)	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \lVert \boldsymbol{c}_1 - \boldsymbol{c}_4 \rVert$
Average [unweighted pair group method with arithmetic mean, UPGMA]. Distance between two clusters is the mean distance between all observations in one cluster and all observations in the other cluster	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \frac{\mathrm{n_2}}{\mathrm{n_2}+\mathrm{n_3}}\left(\boldsymbol{D}_{\mathrm{C_1C_2}}\right) + \frac{\mathrm{n_3}}{\mathrm{n_2}+\mathrm{n_3}}\left(\boldsymbol{D}_{\mathrm{C_1C_3}}\right)$
McQuitty [weighted pair group method with arithmetic mean, WPGMA]. Distance between the newly formed cluster and any other cluster is the mean value of the distances from each of the two merged clusters to that cluster	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \frac{1}{2}\left(\boldsymbol{D}_{\mathrm{C_1C_2}}, \boldsymbol{D}_{\mathrm{C_1C_3}}\right)$
Ward [minimum variance]. The proximity of two clusters is calculated as the Euclidean distance between their centroids multiplied by a correction factor, thus minimizing the within-cluster sum of squares	$\boldsymbol{D}_{\mathrm{C_1C_4}} = \lVert \boldsymbol{c}_1 - \boldsymbol{c}_4 \rVert \frac{\sqrt{2\mathrm{n_1 n_4}}}{\mathrm{n_1}+\mathrm{n_4}}$

(with $\mathrm{n_2} + \mathrm{n_3} = \mathrm{n_4}$ objects), the distance between cluster $\mathrm{C_1}$ and the new cluster $\mathrm{C_4}$ is calculated according to one of the approaches listed in Table 3.

The *single linkage* method is a good choice for identifying the most homogenous (similar) groups. The *complete linkage* method tends to highlight the differences between samples – the resulting groups are similar in size but very diverse. The *furthest neighbor* method can be sensitive to outliers. While the single or complete linkage methods are based on single-pair distances, the *average* and *centroid linkage* methods use a more central measure of location.

Hierarchical Clustering

Hierarchical clustering is an iterative method of grouping objects into a tiered, ordered structure, where all individuals are assigned to a specific, mutually exclusive subgroups. There are two approaches to hierarchical clustering:

(a) *Agglomerative* (bottom-up), which starts with each of the *n* objects forming separate, single-member clusters and recursively merges the two most similar

groups or individuals into one of a higher level until all objects have been combined into a single cluster

(b) *Divisive* (top-down), which starts with all *n* objects forming one single cluster and with each iteration splits it into two smaller groups until all objects form separate, single-member clusters

The agglomerative methods are widely used in various areas of science: medicine, environmental science, computer vision, and analytical chemistry, which is why the authors choose to focus on them in this section.

Advantages
- Wide range of similarity metrics and cluster linkage techniques
- Applicability to a variety of data types (interval, binary, and count data)

Disadvantages
- Ambiguity in selecting the final number of clusters
- Irreversibility of the merge at each level

Clustering Algorithm

The workflow of agglomerative hierarchical clustering is presented below. In this example, the variables were all measured on different scales; therefore, prior to the analysis, a data transformation is necessary. In order to convert data variables into ones with comparable units, all the sample values were standardized, that is, mean centered and variance scaled (Fig. 5).

Agglomerative hierarchical clustering consists of the following steps:

1. Computing the pairwise distance matrix (Fig. 6). Using a selected metric, the distances between each point and all other objects in the data set are calculated. Here, the Euclidean distance is used.
2. Forming clusters (Fig. 7). The first step is to find two objects with smallest distance to each other (A and B). These objects will be merged into the first cluster (C_1).
3. Determining intercluster distances. Using a selected linkage technique, the relative proximity of the new cluster C_1 to all the remaining objects (C, D, E, and F) is calculated. When estimating linkage, all unmerged objects are considered to be single-member clusters or *singletons*. Here, the single linkage (nearest neighbor) approach has been used.
4. Merging the previously formed cluster (C_1) and the closest object C, into a new cluster (C_2) (Fig. 8).
5. Iterating step 3 thru 4 until all objects have been incorporated into a single cluster with nested subclusters (Fig. 9). In our example, cluster C_1 contains only objects A and B. Cluster 3 contains only objects E and D. One tier up in the hierarchy, cluster C_2 contains cluster C_1 and object C. Up another tier, Cluster C_4 contains cluster C_2 and C_3. The final tier the cluster C_5 comprises all objects, formed by merging cluster C_4 and object F.

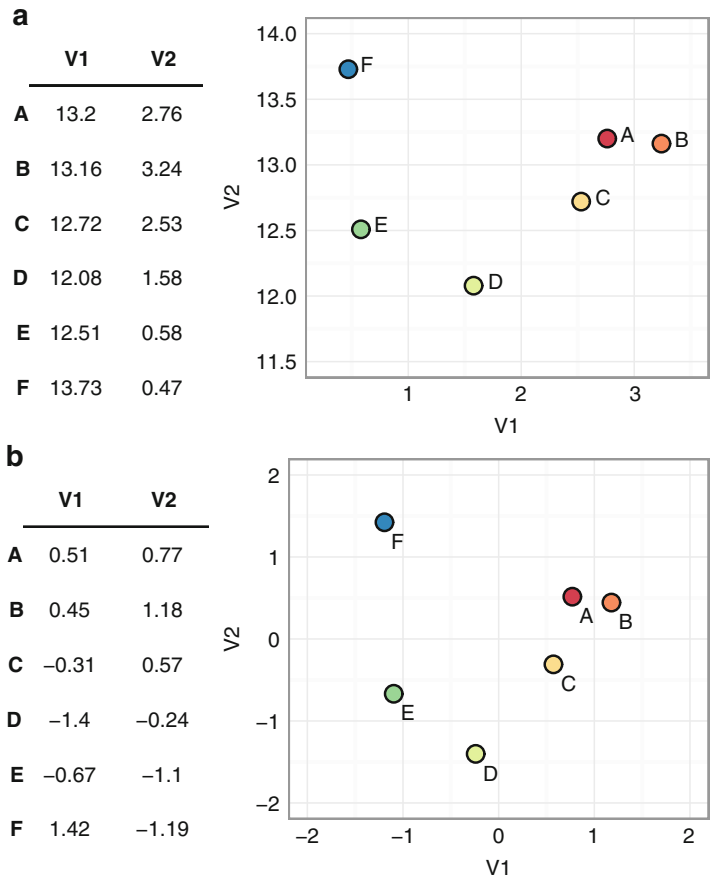

Fig. 5 The effect of autoscaling on a randomly generated data set; (**a**) original (raw) data, (**b**) autoscaled data

The hierarchical clustering results can be visualized as a tree-like diagram, called the *dendrogram* plot (Fig. 10). Typically, the y-axis indicates the distance at each successive split (or join) or branch height. On the x-axis, the objects are shown as leaves.

There are three main types of dendrograms to choose from: rectangular, triangular, or circular, although the rectangular is most popular (see Fig. 11). Regardless of the choice of the dendrogram type, the interpretation remains the same. Objects belonging to the same cluster are always more similar to each other than to the objects from other clusters.

Depending on the distance metric and linkage method used, the tree diagram will be different (Fig. 12). For example, using the Canberra distance metric instead of the

a

	A	B	C	D	E	F
A	0	**0.42**	0.85	2.17	2.21	2.16
B	**0.42**	0	0.97	2.33	2.54	2.57
C	0.85	0.97	0	1.37	1.71	2.47
D	2.17	2.33	1.37	0	1.13	2.98
E	2.21	2.54	1.71	1.13	0	2.09
F	2.16	2.57	2.47	2.98	2.09	0

b

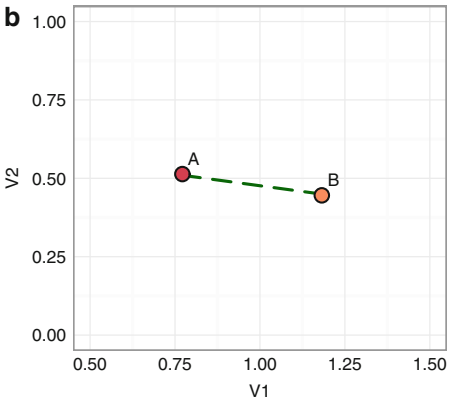

Fig. 6 Example of (**a**) Euclidean distance matrix, where the smallest distance between the two has been highlighted, (**b**) visualization of the Euclidean distance between the closest objects – *A* and *B*

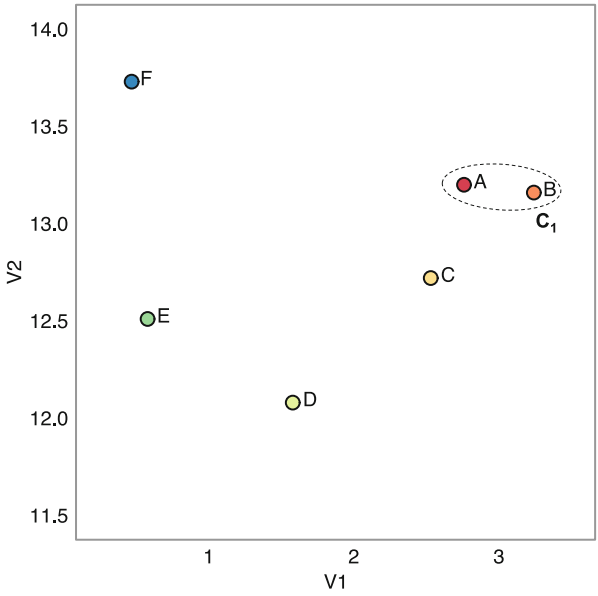

Fig. 7 Forming the first cluster C_1 by merging two closest objects A and B, based on the single linkage approach

standard Euclidean distance will result in greater distance value ranges and distinct cluster separation; see Fig. 12a.

Selecting appropriate distance and linkage parameters will help answer the questions about the data structure and facilitate the correct interpretation of results. When looking for similarities between samples, one should use the single linkage method instead of the complete linkage approach (Fig. 12b).

Let us now turn to the POP data set. As we are trying to cluster the objects into most diverse groups in order to find the common characteristics for POP samples

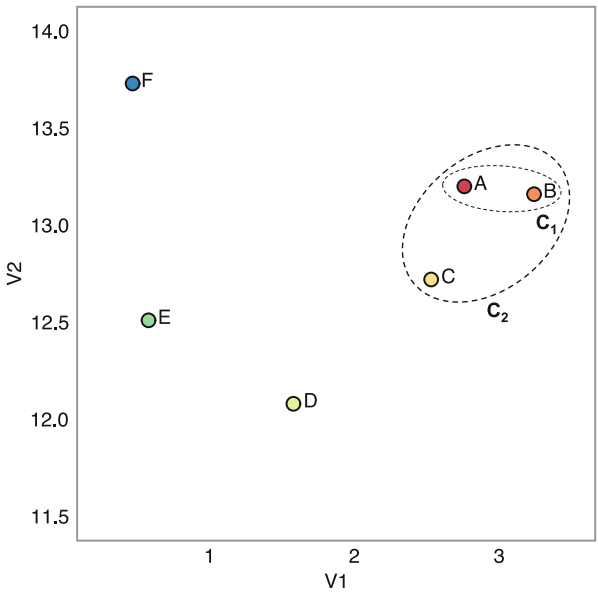

Fig. 8 Forming the second cluster by merging cluster C_1 with the closest object C, based on the single linkage approach

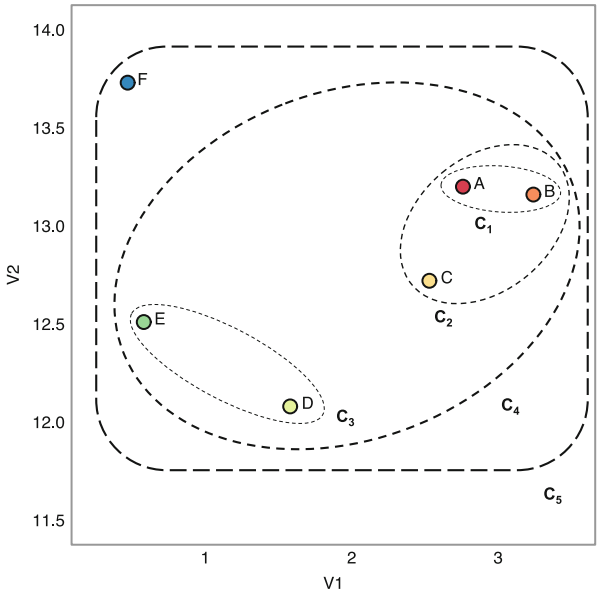

Fig. 9 Final clustering results, with showing hierarchical (nested) cluster structure

from the chemical groups of polychlorinated naphthalenes PCNs, polychlorinated dibenzo-*p*-dioxins PCDDs, and polychlorinated dibenzofurans PCDFs, we used the Euclidean distance metric. For comparison's sake, we applied the three types of linkage metrics to build a dendrogram: complete (Fig. 13a), Ward's (Fig. 13b), and single (Fig. 13c). Depending on the linkage method used, the number of main

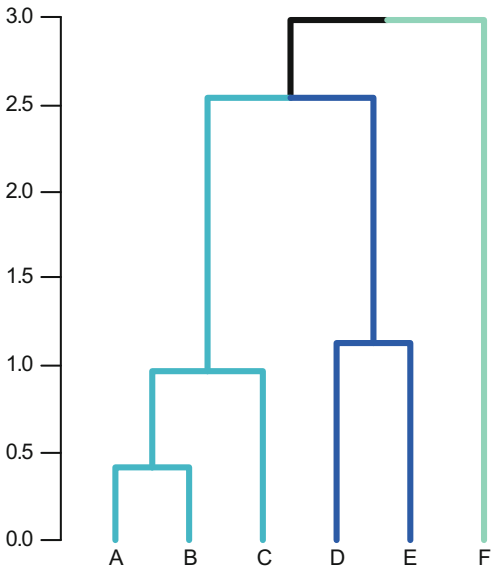

Fig. 10 An example of a rectangular dendrogram, created using Euclidean distance and complete linkage

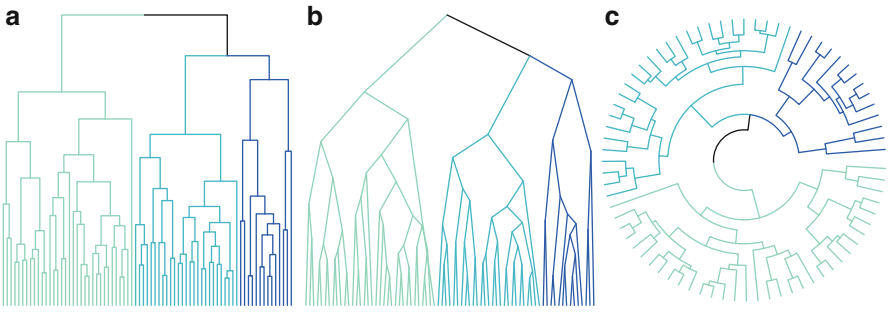

Fig. 11 Different types of dendrograms (**a**) rectangular, (**b**) triangular, and (**c**) circular

branches varies significantly, e.g., for the complete method, four main clusters can be identified; for Ward's method, three main clusters are visible; and for the single linkage method, two main cluster groups are noticeable. The preliminary assumptions on cluster number are very subjective and depend on the person analyzing the plots. Another scientist may see a completely different ramification to the one presented by the authors.

Choosing Number of Clusters

The dendrogram resulting from agglomerative hierarchical clustering does not explicitly specify group structure. Based only on the tree diagram, we are unable to identify the number of groups and their memberships or size. That information can, however, be obtained by "cutting" the dendrogram at a certain height (h) or by specifying the desired number of groups (k). Choosing the cutoff point or a default

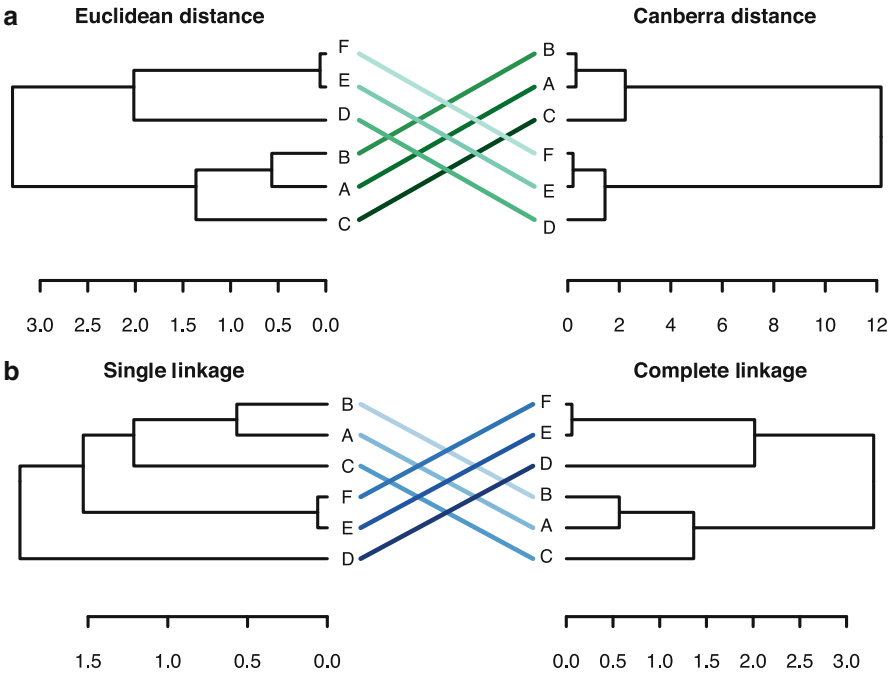

Fig. 12 Comparison of distance metrics and linkage methods: (**a**) Euclidean versus Canberra distance, complete linkage; (**b**) single versus complete linkage, Euclidean distance

cluster number is not an easy task, especially considering the variety of data types, units, and ranges, as well as differing needs and expectations of individual analysts.

A substantial number of approaches to determining the branch cut height can be found in the literature (Milligan and Cooper 1985). The simplest one, a *rule of thumb* really, is to use the square root of half of the sample number (Eq. 4):

$$g = \sqrt{\frac{n}{2}}$$

where n is the number of samples and g is the cutoff value.

In order to verify the number of main groups (clusters) identified through different linkage methods, we use the rule of thumb approach. The number of POP samples n equals to 288; therefore, the cutoff height $g = 12$. The resulting groups for all three linkage methods, colored for better identification, are shown in Fig. 14. When using the complete linkage method, after visual inspection, we identified four main clusters. This verified when using the rule of thumb approach, as shown in Fig. 14a. Curiously, the three main clusters initially identified in the second dendrogram (Ward's method) were later disproved by the rule of thumb, which showed 11 subclusters (Fig. 14b). Interestingly, while we initially identified

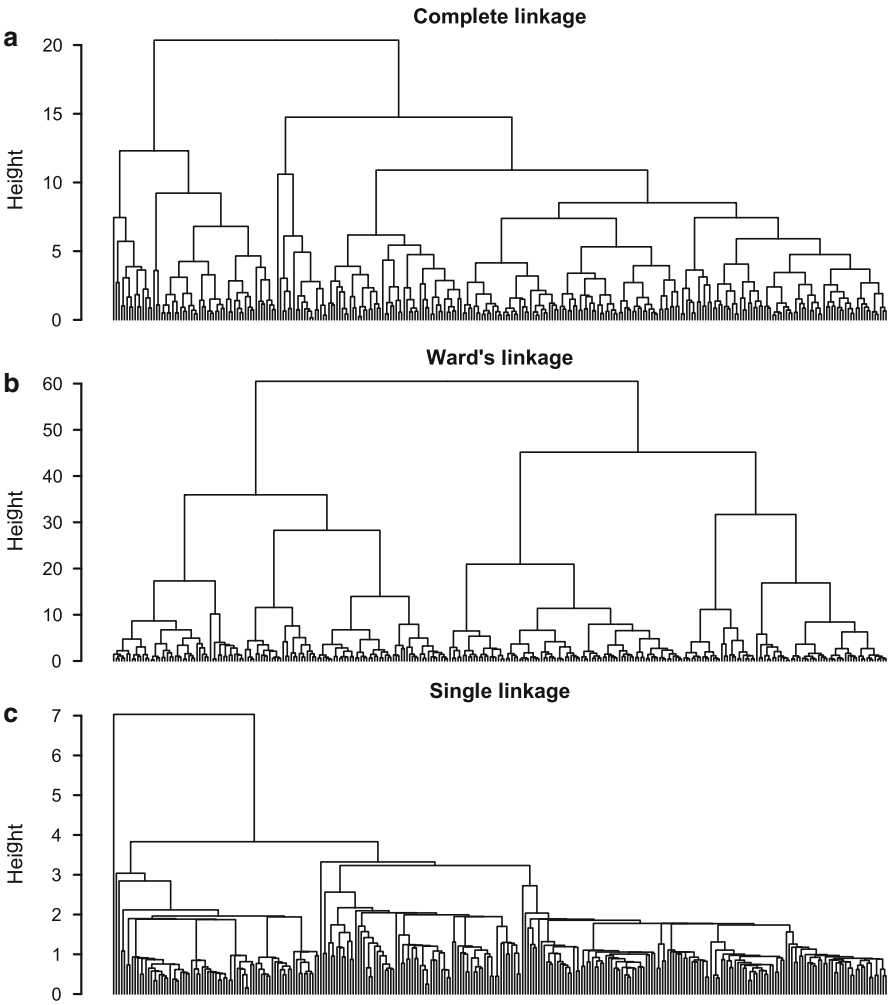

Fig. 13 Hierarchical clustering results for three POP groups (PCNs, PCDDs, and PCDFs) using the Euclidean distance: (**a**) complete linkage, (**b**) Ward's linkage method, (**c**) single linkage

two main subgroups in the third dendrogram (single linkage), according to the rule of thumb, there is only one, all-encompassing cluster (Fig. 14c).

Other, more complex means of selecting the number of clusters are also available. Most of them are automated in nature (i.e., those for which user input is not necessary), such as the C_{index} (Eq. 5):

$$C_{index} = \frac{S - S_{\min}}{S_{\max} - S_{\min}} \tag{5}$$

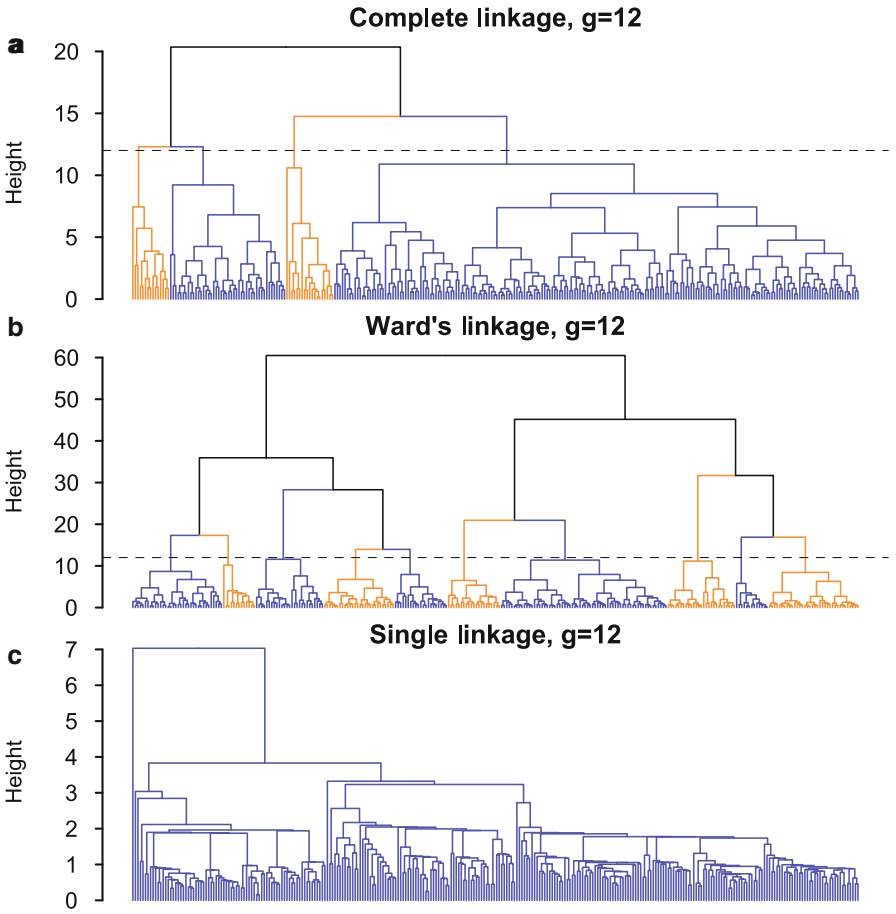

Fig. 14 Varying number of clusters depending on the linkage method used: (**a**) complete linkage, (**b**) Ward's linkage method, (**c**) single linkage. The dashed vertical line shows the cutoff tree height ($g = 12$)

where NC is the total number of pairs of objects from the same cluster, S is the sum of distances for the total number of object pairs in the same cluster (NC), $S_{\min}$ is the sum of the NC smallest distances between all the pairs of points in the entire data set, and $S_{\max}$ is the sum of the NC largest distances between all the pairs of points in the entire data set.

The C_{index} takes values between 0 and 1, where a small value indicates good clustering. In the case of the POP data set, there are 287 cluster sets ($k = n\text{-}1$ values) possible, ranging from 2 to 287. When comparing C_{index} values for all the possibilities (Fig. 15), we can see that the lowest ${C_{index}}^{287}$ value is 0, for a grouping consisting practically only of 287 one-element clusters. Such a grouping is, of course, meaningless and unusable. Therefore, in practice, we look for a global

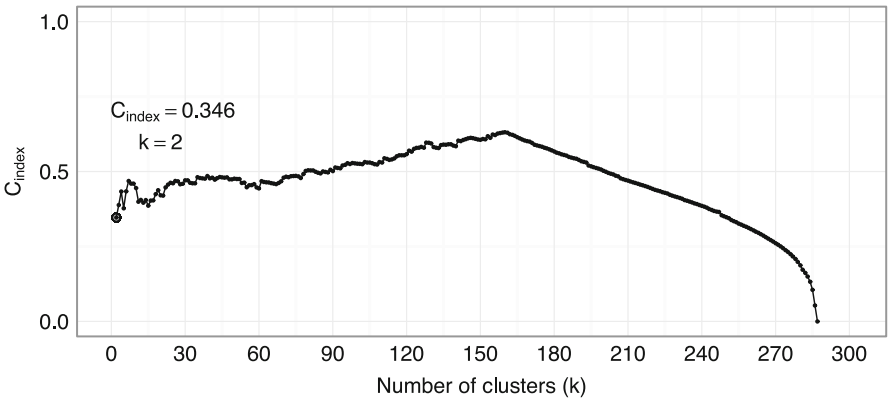

Fig. 15 Number of clusters versus C_{index} value

minimum and the lowest possible number of clusters at the same time. In our case, when taking both of those condition into account, we arrive at ${C_{index}}^2 = 0.346$, found for $k = 2$. Therefore, two is the optimal cluster number.

Another numerical technique to determine the optimal cluster number is the silhouette index S_i, which is calculated individually for each sample (Eq. 6):

$$S_i = \frac{b_i - a_i}{\max\{a_i, a_i\}} \tag{6}$$

where a_i – is the average dissimilarity of i-th object to all other objects in the same cluster, and b_i – is the minimum of average dissimilarity of i-th object to all objects in other, closest cluster.

The silhouette width S_i can take values between -1 and 1. For objects with high S_i value (close to 1), the cluster assignment is correct and accurate. For samples with silhouette value close to zero, the cluster assignment is ambiguous and imprecise. That is, the sample may have very well been assigned to another cluster instead, as it is equidistant from both clusters. Objects with S_i, value close to -1, were wrongly classified (assigned cluster membership) and lie somewhere in between all the possible clusters.

The overall silhouette width is simply the average S_i value over all objects in the data set. When assessing the optimal number of clusters, the one with the maximum average silhouette width should be chosen. The silhouette index should be visualized in the form of a bar graph (silhouette plot), which, in addition to S_i values, shows cluster membership for each object (Fig. 16).

We used the silhouette plot to verify and compare the clustering for two different linkage methods: complete (Fig. 16a) and Ward's (Fig. 16b). In the case of complete linkage, 71 out of 288 POP samples (24.65 % of the data) have S_i values below 0, meaning they could be considered as "misclustered" samples. Such a high number

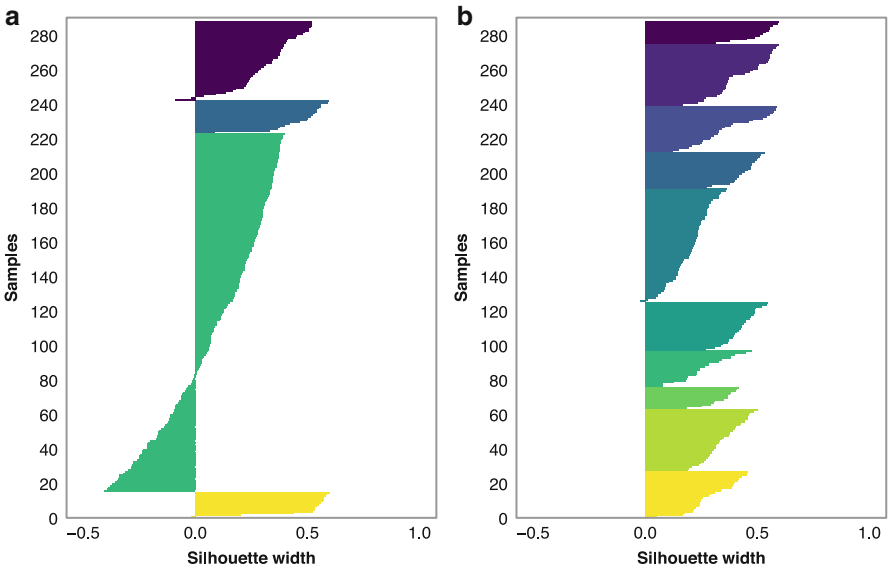

Fig. 16 Silhouette plots for three POP groups (PCN, PDDD, and PCDF). Comparison of the cluster number resulting from cutting dendrograms at height $g = 12$; (**a**) complete linkage, (**b**) Ward's linkage method. Groups have been color-coded for easier interpretation

of samples with negative silhouette values may suggest that the selected linkage method may not be the most suitable for the data at hand. This conclusion is further confirmed when looking at the silhouette plot for Ward's method (Fig. 16b). There is only one POP sample with the S_i value below 0, suggesting that this linkage method gives a more accurate separation of samples (grouping). It is worth noting that, if a data set contains a significant number of samples with S_i values below 0, it may also suggest that the selected features (variable) do not describe the underlying data structure in an appropriate manner.

There is also another way of comparing the cluster membership with the original POP group, illustrated in Fig. 17. Cluster 1 consists of several PCNs and two PCDFs. Cluster 2 contains the bulk of the data set, that is, the majority of samples from all three POP groups (PCNs, PCDDs, and PCDFs). Cluster 3 and 4 contain a small number of PCDDs and PCDFs. We see now why the silhouette analysis yielded 71 "misclustered" samples. The reason for such misclustering might be the use of too many variables (in this case, all 21 molecular descriptors), some of which might be redundant.

Despite the great number of cluster selection methods available, there is no single, universally applicable technique regarded as a golden standard. The silhouette index is a potential candidate for an effective and simple way of verifying the quality and performance of the selected clustering algorithm. Sadly though, even the most sophisticated cluster selection and identification method needs to be augmented by

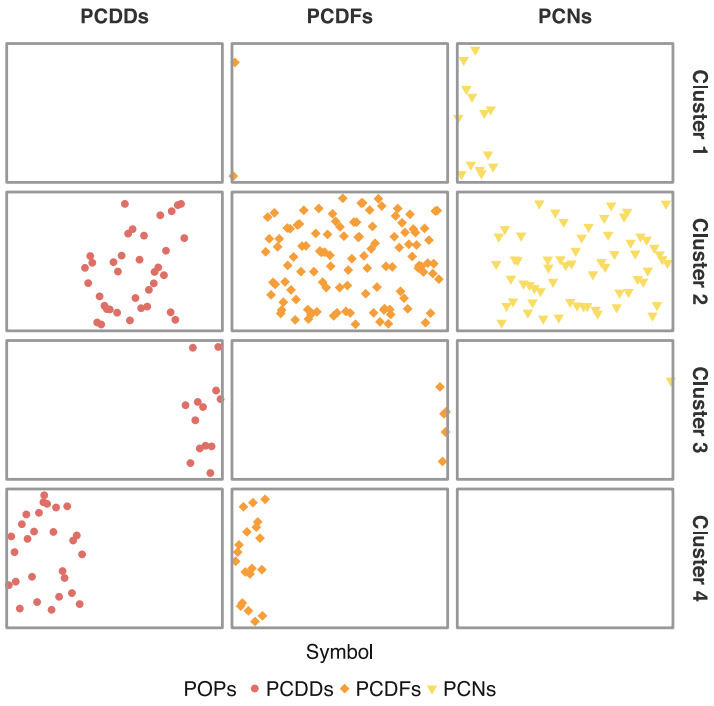

Fig. 17 Hierarchical clustering grouping versus the original chemical groups. Euclidean distance and complete linkage method were used

user expertise and experience. No two research problems are alike; therefore, no universal, unambiguous approach to all of them can be found.

k-means

The *k*-means algorithm is one of the most popular partitioning methods, mainly because of its ease of use. It groups data into a user-defined number of clusters, *k*. Each observation is assigned to the nearest cluster, the center of which is defined by the arithmetic mean (average) value of its points, the centroid (Myatt 2007). Among many available distance metrics (see Table 1) used in *k*-means, the Euclidean distance is the most popular one. The main advantage of this method is its clarity and simplicity, making its implementation an easy task.

Partitioning Algorithm
The algorithm consists of the following steps:

1. Setting up initial positions of the *k* cluster centers. This can be by supplying a set of predefined centroid coordinates or by random generation of cluster centers.

2. Calculating distances from each i-th element to each of the initial cluster centers using a selected distance metric.
3. Based on sample-center distances, assigning each i-th element to the nearest cluster.
4. Calculation of new cluster means (centroids) after object reassignment.
5. If the distance from the sample to the new cluster center is smaller than the distance to the prior cluster center, assignment of the sample to a new cluster with the closer centroid.
6. Iterating steps 2 thru 5 until permanent cluster assignment for all elements, i.e., until no element changes its cluster membership. It is possible to define the maximum number of iterations regardless of group reassignment.

When applying the k-means method to the POP data set, we observed some cluster overlap, as was the case with HCA results (Fig. 18).

In order to compare the clustering results from both the hierarchical and partitioning method, we calculated the silhouette for each sample (Fig. 19). We found that the overall performance of the k-means algorithm was better in terms of cluster homogeneity; only 7 out of all 288 samples (2.43 % of all data) have S_i below 0, which means only seven of them have been "misclustered." It is a great improvement over the HCA clustering, which yielded 71 "misclustered" samples.

All partitioning methods divide the data into a set number clusters in a way that optimizes an objective function, that is, meets specific criteria set for the results prior

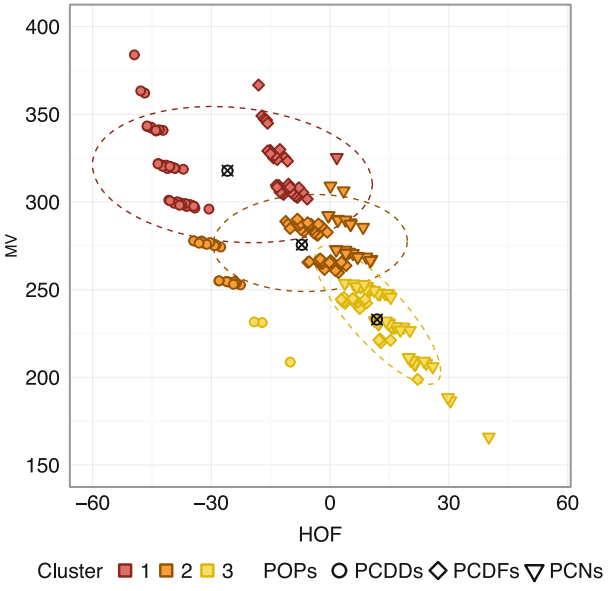

Fig. 18 The results of k-means clustering for selected POP groups (PCNs, PCDDs, PCDFs)

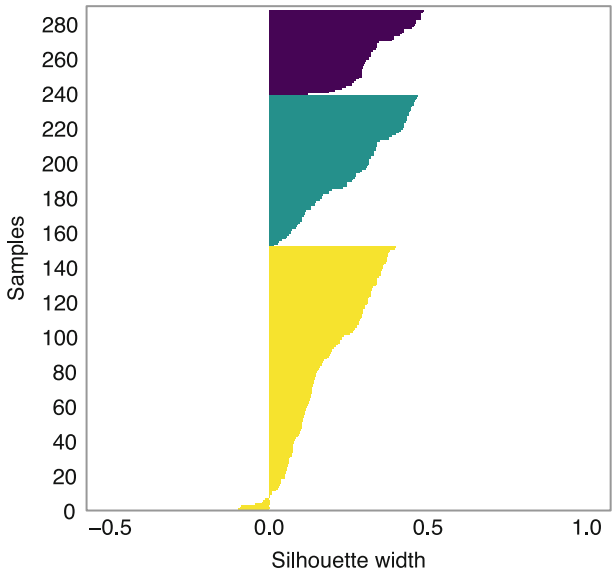

Fig. 19 The silhouette plot of k-means clustering results for the selected POP groups (PCNs, PCDDs, PCDFs)

to the analysis. In case of k-means, the algorithm minimizes the sum of squared errors, SSE (Han et al. 2012), the within-cluster sum of squared distances between each sample and cluster center (Eq. 7).

$$\text{SSE} = \sum_{k=1}^{K} \sum_{x_i \in C_k} \|x_i - c_k\|^2 \tag{7}$$

where k is the number of clusters, C_k is the k-th cluster, c_k is the centroid of the k-th cluster, and x_i is the sample.

It is important to note that a different choice of initial centroid positions may result in a different final assignment of points to clusters. Therefore, it advisable to run the k-means algorithm a number of times, each with a different set of starting parameters, and then compare the results. This will ensure that the algorithm will reach the global optimal cluster configuration and will not get "stuck" at a local minimum of the objective function.

Advantages

- Effective when working with well-separated, compact clusters
- Computationally faster and more appropriate for large data sets than agglomerative hierarchical clustering
- Intuitive use and ease of interpretation, simplicity of implementation

Disadvantages
- The number of groups must be prespecified prior to running the algorithm
- Not robust to noisy data and outliers – might create nonoptimal clusters (solution: use *k*-medoids)
- Not applicable for categorical data (solution: use *k*-modes)

k-medoids

While the *k*-means algorithm is a very simple and useful tool, it is quite susceptible to extreme values or noise, which distort the mean value of the cluster. The *k*-medoids algorithm is a modified version of the *k*-means approach, where rather than averaging all objects in the cluster to find the center, the middle point, or medoid, is selected from among the samples in the cluster (Everitt et al. 2011). Thusly, the clusters are represented by their most centric element, instead of a numerical value that may not belong to the cluster at all. Medoids are defined as data points, whose average dissimilarity to all the objects in a cluster is minimal.

Both the *k*-means and *k*-medoids algorithms minimize the distance between all points inside a cluster and its designated center. The difference being that, unlike the *k*-means algorithm, *k*-medoids work with pairwise dissimilarities instead of Euclidean distances.

The *k*-medoids algorithm is analogical to that of the *k*-means method:

1. Selection of an initial set of *k*-medoids among the *n* objects in the data set.
2. Assignment of each object in the data set to the nearest (least dissimilar) medoid.
3. Searching for more optimal medoids. Iteratively for every medoid *m*, swapping (replacing) it with each one of the non-medoid objects o_i to check if it improves the total distance of the resulting clustering.

   The suitability of the medoid "candidate" o_i is assessed by computing its average dissimilarity to all remaining non-medoid data points $o_{n \neq i}$. The objective function is the total cost of the new configuration, which is the sum of candidate-no-medoid differences *E* (Eq. 8).

$$E = \sum_{i=1}^{n} d\left(o_i, m\right) \tag{8}$$

   where *d* is the dissimilarity measure, o_i is the non-medoid object, *m* is the medoid, and *n* is the number of objects in the data set.
4. Selection of new medoids o_i with the lowest cost of the configuration (Fig. 20).

k-modes

The *k*-means clustering algorithm cannot be applied to categorical data, as it relies on means to represent cluster centers, and nonnumerical data have no defined mean value. The *k*-modes, a variant of *k*-means, enables grouping categorical data by using the mode as the cluster center point. Moreover, it replaces the Euclidean

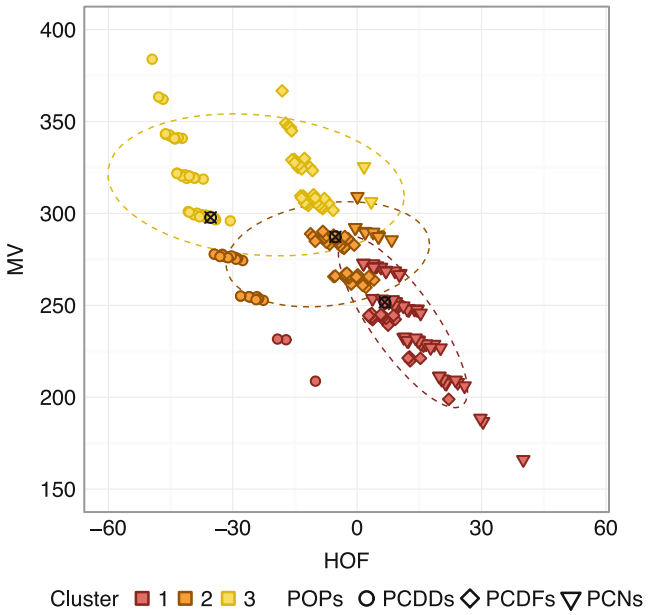

Fig. 20 The results of *k*-medoid clustering for the selected POP groups (PCNs, PCDDs, PCDFs)

distance metric from *k*-means with matching dissimilarity measure and a frequency-based approach to update cluster modes.

The *k*-modes algorithm consists of the following steps (Khan and Kant 2007):

1. Selecting k initial modes, one for each of the cluster.
2. Assigning every data object to the cluster whose mode is nearest to it. The proximity (dissimilarity) measure is based on the total number of mismatches d (Eq. 9). The smaller the number of mismatches, the more similar the two elements are.

$$d(X,Y) = \sum_{j=1}^{m} \delta(x_i, y_i)$$
$$\begin{cases} 0, & x_j = y_i \\ 1, & x_j \neq y_i \end{cases} \tag{9}$$

where X and Y are two categorical objects and m is the number of categorical attributes; x_j, y_j are categories for attribute j.

3. Computing new modes for all clusters.
4. Repeating step 2 thru step 3 until no data object has changed its cluster membership (Fig. 21).

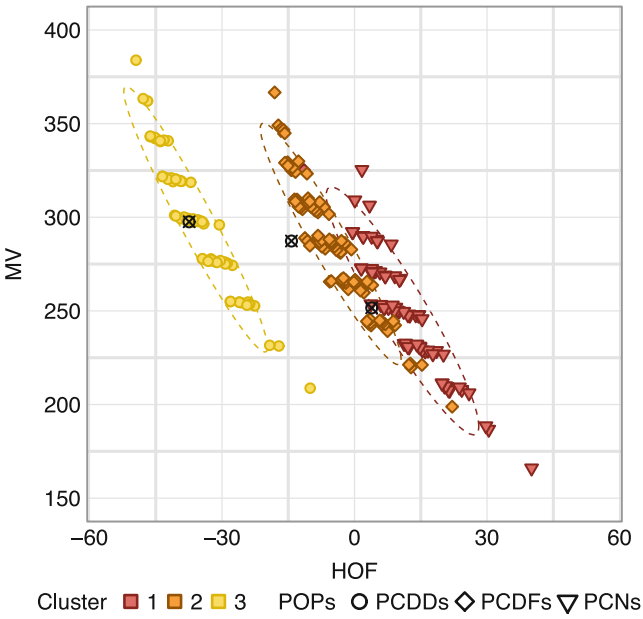

Fig. 21 The results of *k*-modes clustering for the selected POP groups (PCNs, PCDDs, CDFs)

PCA

When faced with a large, multidimensional set of variables, the sheer volume may make it difficult to see patterns and relationships hidden in its structure. Principal component analysis allows us to visualize and describe this structure by means of reducing the dimensionality of the data. This method is based on principal components (PCs), which are linear combinations of the original variables (PCs) and form a new multidimensional space onto which the original data set is projected.

The main applications of principal component analysis are:

- Projecting multidimensional (multivariate) data onto 2D and 3D scatterplots
- Winnowing out important information from data noise
- Converting data with high inter-variable correlation into a set of uncorrelated latent variables suitable for predictive modeling

In data mining, information content is measured in terms of data variance (Eq. 10).

$$\sigma^2(x) = \frac{\sum_{i=1}^{m} (x_i - \mu)^2}{n-1} \tag{10}$$

where σ is the variance of variable x, μ is the mean value of variable x, n is the number of samples, and x_i is the i-th sample of variable x.

Variables with very little difference in values (low variability) are in fact non-informative and have little bearing on the clusters present in the data set. These variables are inadequate to the task of describing and explaining the underlying data structure, as they have very little influence on it.

In order to describe the linear relationship of two variables, we can calculate their covariance:

$$\operatorname{cov}(x_k, x_l) = \frac{\sum_{j=1}^{n} (x_{ik} - \mu_k)(x_{il} - \mu_l)}{n - 1}$$

where x_k and x_l are k-th and l-th variables, μ_k and μ_l are variable means, and n is the number of samples.

The covariance parameter is the measure of how much the values of two variables change together. If values of one variable change (e.g., increase) in the same direction as values of the other, i.e., the variables tend to show similar behavior, the covariance value is positive. Although, the covariance parameter, while a good indicator of whether two variables correspond to each other, is not suitable for a pairwise comparison of multiple variables with differing units and value ranges. In other words, the same numerical value of covariance might mean a highly proportional relationship for one pair of variables, whereas for another pair, it might signify a completely random one.

In order to unambiguously assess and compare the relationships between variables, we must calculate the correlation coefficient r (Eq. 11):

$$r(x_k, x_l) = \frac{\operatorname{cov}(x_k, x_l)}{\sqrt{\operatorname{var}(x_k)}\sqrt{\operatorname{var}(x_l)}} \tag{11}$$

where x_k, x_l are k-th and l-th variables, $\sigma^2{}_k$, $\sigma^2{}_l$ are variable variances, and cov is variable covariance.

The Workflow

Prior to the analysis, the data matrix **X** requires preprocessing and the preferred data transformation methods is standardization (autoscaling). In the resulting autoscaled matrix **Z**, all the variables have mean value μ equal to 0 and the standard deviation value σ equal to 1. Thus, any distorting effects caused by differing variable units have been negated.

The principal component analysis consists of the following steps (Jolliffe 2002; Brereton 2009):

1. Calculating the correlation-covariance (*corr-cov*) matrix **C**. The diagonal elements of a covariance matrix are the variable variances, and the non-diagonal

elements are their covariance. For standardized (autoscaled) data, the covariance matrix is equivalent to the correlation matrix, and the diagonal elements are in fact the Pearson's correlation coefficients (Eq. 12):

$$\begin{aligned} \mathrm{var}\,(x_k) &= 1 \\ \mathrm{var}\,(x_l) &= 1 \\ \boldsymbol{r}\,(x_k, x_l) &= \mathrm{cov}\,(x_k, x_l) \end{aligned} \tag{12}$$

where x_k, x_l are k-th and l-th variables, cov is the variable covariance, and r is the variable correlation coefficient.

2. Computing the eigenvalues and eigenvectors of the correlation-covariance matrix. The eigenvalues λ indicate the amount of variance in data explained by their corresponding principal component. Each eigenvector contains a set of variable coefficients, i.e., loadings, **P**. The resulting eigenvectors **P** are arranged in descending order of their eigenvalues. The loadings express the variables' contributions to the principal components.
3. Calculating the scores. In order to obtain the positions of all data objects in the new principal component space, we must calculate the principal component scores, **T**. The scores matrix is a product of autoscaled matrix **Z** and the eigenvector matrix **P**.
4. Determining the number of significant principal components, k. All principal components are mutually orthogonal – each one contains unique information that none of the others represent. Yet not all of the PCs are equally important. Considering that they are sorted in a descending order of information content, only a first selected few principal components will be useful in representing the data set.

The essence of the mathematical operations performed in order to obtain the principal components can be distilled into the scheme shown in Fig. 22. The rows in matrix **Z** represent samples and the columns represent variables. The product **TP** is an approximation of to the original data set, i.e., a model, the error of which is being represented by matrix **E**.

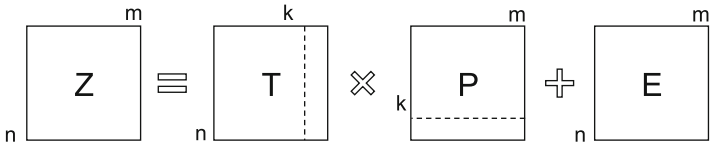

Fig. 22 A schematic of PCA decomposition. k is the number of significant principal components, $\boldsymbol{Z}$ is the $n \times m$ standardized data matrix, $\boldsymbol{T}$ is the $n \times k$ scores matrix, $\boldsymbol{P}$ is the $k \times m$ loadings matrix, and $\boldsymbol{E}$ is the $n \times m$ error matrix

Table 4 Eigenvalues and explained variance of the principal components – POP data set

PC	Eigenvalue	Explained variance [%]	Cumulative explained variance [%]
1	11.36	54.07	54.07
2	5.75	27.40	81.48
3	2.25	10.70	92.18
4	0.93	4.42	96.59
5	0.44	2.07	98.67
6	0.11	0.53	99.20
7	0.08	0.40	99.60
8	0.06	0.28	99.88
9	0.02	0.07	99.96
10	0.01	0.02	99.98
11	0.00	0.01	99.99
12	0.00	0.00	100.00
13	0.00	0.00	100.00
14	0.00	0.00	100.00
15	0.00	0.00	100.00
16	0.00	0.00	100.00
17	0.00	0.00	100.00
18	0.00	0.00	100.00
19	0.00	0.00	100.00
20	0.00	0.00	100.00
21	0.00	0.00	100.00

Let us review the results of the principal component analysis of the POP data set. There are various methods of selecting the significant number of principal components (Jolliffe 2002):

1. The Kaiser criterion, which states that the significant PCs have eigenvalue greater than or equal to 1. In our case (Table 4), we would choose the first three principal components (PC1 thru PC3).
2. The minimum cumulative variance criterion is PC significance only for values over a certain arbitrary threshold, here 90 %. Again, according to this rule, we should use the first three principal components (PC1 thru PC3, Table 4).
3. The "elbow method" based on the scree plot (Fig. 23a). The "elbow" in question is the point of the plot where the line reaches a plateau. We would make use of the first four principal components (PC1 thru PC4).

We choose to obey the Kaiser criterion, that is, to consider PC1, PC2, and PC3 as the most significant.

The next step is to visualize the data set in the space created by the principal components (Fig. 24), i.e., the *scores*. The principal component scores show the position of all samples projected onto the PC hyperspace. When analyzing the scores for the first three principal components, we noticed that PC1 and PC2 separate the

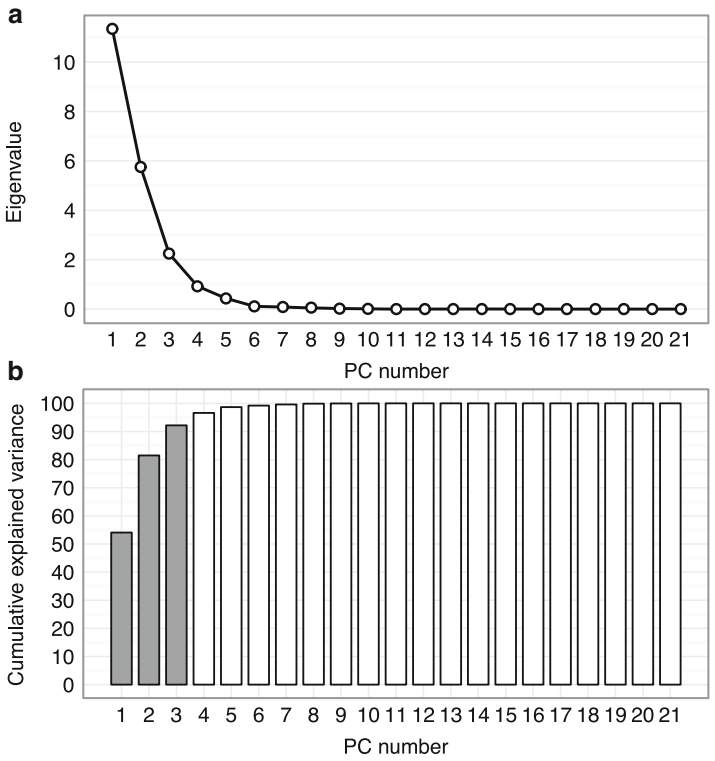

Fig. 23 Explained variance of principal components for selected POP groups (PCNs, PCDDs, PCDFs): (**a**) scree plot of the principal components (**b**) cumulative explained variance

three selected POP groups (PCNs, PCDDs, PCDFs) quite well (Fig. 24a), whereas PC1 in combination with PC3 (Fig. 24b) seems to highlight possible outliers. When juxtaposing PC2 and PC3, we see highly condensed group made up of PCN and PCDF samples as well as highlight one possible outlier (Fig. 24c).

Knowing that each principal component is a linear combination of all variables (descriptors) in the data set, we can use loadings to investigate which variables influence them most. A variable is considered to be highly influential on the principal component if the absolute value of its standardized loading is greater than 0.7 (Jolliffe 2002). What this means is that in reality, only the "highly influential" variables dictate the internal data structure.

According to PC1's loading vectors presented in Fig. 25, the first principal component is determined by the size and bulk of the molecules. The contributing structural descriptors (nX, MW, MV, and SAS) are all correlated with PC1, that, is the greater the PC1 values, the higher the number of chlorine atoms, molecular mass, volume, and surface. The polarizability (P) of a molecule and the core-core repulsion energy (Core) between two atoms are proportional to molecular size, which is expressed by the high loading values of these descriptors. Another characteristic dependent on the size of the molecular is its energy. Here, the proportion is inverse,

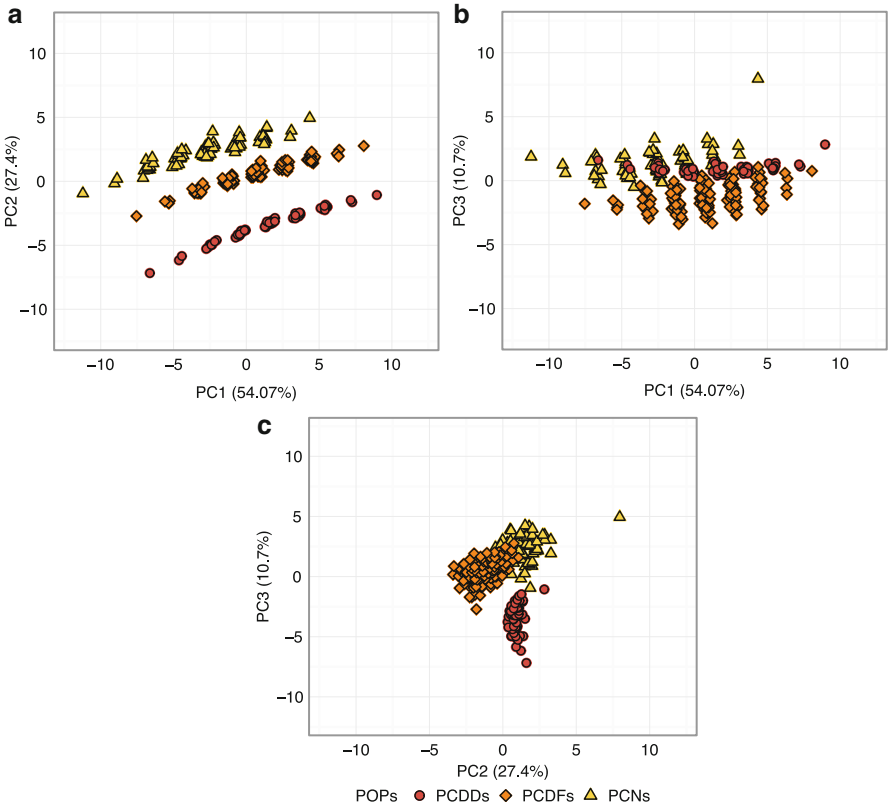

Fig. 24 Scatterplots of the first three principal component scores (PC1, PC2, PC3) – POP data set

as values of energy and heat are expressed in negative numbers, so the bigger the molecule, the lower ("more negative") the energy value. This is expressed through a set of descriptors: the heat of formation (HoF), the electronic energy (EE), the total energy (TE) and the total energy of the corresponding cation (TEp), and the LUMO energy and Parr and Pople's absolute hardness (Hard). All of them have negative loading values and are inversely correlated with PC1.

PC2 is determined by the molecule's ability to donate and accept electrons. The higher the number of oxygen atoms (nO) and the overall number of atoms in the molecule (nAT), the more difficult it is to detach an electron, which is expressed by the inverse proportion between those two descriptors and the vertical ionization potential (VIP). VIP itself is calculated as the difference between the energy of a neutral molecule and the energy of the corresponding cation. According to the Koopmans' theorem, the negative value of HOMO can be used as an approximation of the first ionization energy, so in fact, both descriptors express the same molecular feature. Conversely, the bigger the number of electronegative atoms (i.e., oxygen) in the molecule, the greater the overall electronegativity (EN). The chemical shift (Shift) is the equivalent of the electronegativity but with an opposite sign.

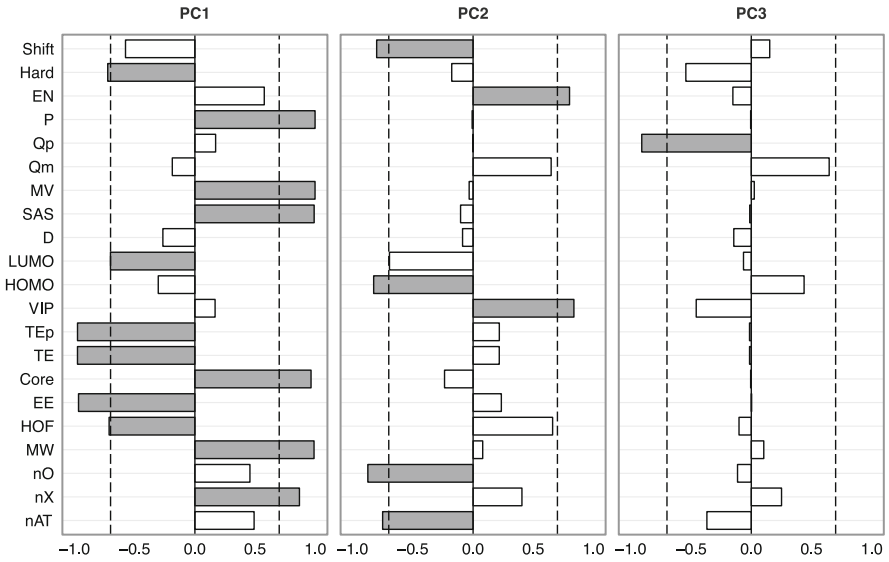

Fig. 25 First three principal component loadings (PC1, PC2, PC3) – POP data set

PC3 represents the symmetry of a molecule, expressed through the highest positive Mulliken charge on the molecule (Qp). The more symmetrical the structure of a molecule, the lower the partial charge.

SOM

Self-organizing (or Kohonen) maps, SOMs (Kohonen 2001), are a type of unsupervised partitive clustering methods. They are a quick and easy way of simultaneous data grouping and visualization, inspired by the biological information transport and processing system – neurons and synapses. SOMs are an ideal tool for analyzing large, complex data sets for which the standard scatterplot would be unreadable due to overlapping data points, the so-called overplotting.

SOMs preserve the topology, i.e., the relative distances between objects, of a high-dimensional variable space while mapping it onto a low-dimensional representation, usually a two-dimensional grid or plane.

Building SOMs

Determining the Structure

A Kohonen neural network consists of two layers (Kohonen 2001; Vesanto and Alhoniemi 2000):

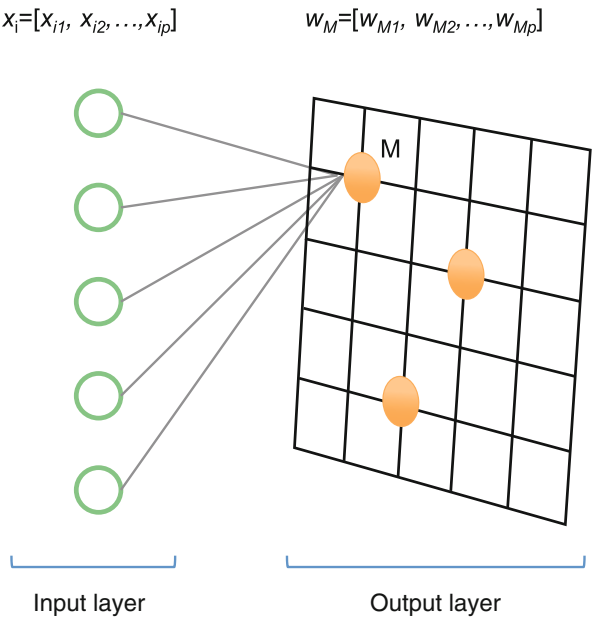

Fig. 26 A schematic representation of a SOM network

1. *Input layer* – an array of input vectors, each of length *p*, based on the original data matrix; every sample (object) is represented by an input vector containing its coordinates in the original *p*-dimensional variable space (Fig. 26).
2. *Output layer* – a grid with M nodes (neurons), where M is a number of nodes determined by the user; each node is represented by a *weight vector*, or *codebook vectors*, of length *p*, containing *weights*, i.e., values defining its location in the original *p*-dimensional variable space (Fig. 26).

In order to set up the computations, one must first define and select a number of parameters describing the shape, topology, and size of the network:

(i) Number of neurons in the output layer
(ii) Network shape: planar (1D line, 2D square or rectangle), cylindrical, and toroidal
(iii) Neighborhood type: square, rectangular, or hexagonal
(iv) Neighborhood radius
(v) Neighborhood function: rectangular, Gaussian, cut Gaussian, triangular, and exponential
(vi) Learning rate
(vii) Number of iterations during the training process

All of these parameters influence the network's learning capabilities, quality of grouping, and computation time.

The *number of neurons* (*i*) defines the number of grid nodes present in the output layer. The greater the number of neurons in the network, the more complex problems it can solve and sharper the divide between resulting groups (clusters) – at the cost of computational resources. Choosing too many neurons may mean that a part of them will go unused when applied to low-complexity problems. Conversely, employing an insufficient number of neurons produces overlapping clusters and loss of information about class structure (Gemperline 2006).

The *network shape* (*ii*) describes the arrangement of the nodes in the network. The simplest possible shape resembles a 1D chain, where each link is a node. The more complex form can be a 2D map (arranged in a square or rectangular grid), 3D cylinder to a toroid. To ensure the equal number of neighbors for every neuron, it is best to use infinite, edgeless, toroidal form (Brown et al. 2009).

The *neighborhood type* (*iii*) is a method of counting the neighboring nodes. A square neighborhood means that a neuron has four nearest neighbors, a hexagonal neighborhood means that the neuron has six nearest neighbors, and a rectangular neighborhood means that a neuron has nine nearest neighbors.

The *neighborhood radius* (*iv*) is the number of surrounding nodes associated with the *winner neuron* or the *best matching unit*, *BMU*. This group of neurons, along with the BMU, is called a *neighborhood set*. The radius value decreases during each training cycle and is ultimately limited only to the winning neuron (Maimon and Rokach 2005).

The *neighborhood function* (*v*) controls the intensity of change of the weights in the codebook vectors from the neighborhood set, that is, the vectors closest to the winner neuron. The most popular is the Gaussian function, through which the winner neuron's weight adjustment is greater than that of the remaining neurons (Kohonen 2001; Brereton 2009; Brown et al. 2009).

The *learning rate* (*vi*) is responsible for the network's performance. It's a coefficient which can take values between 0 and 1 and determines how similar the neuron weight vectors will be to the *input pattern* or *sample vector*. The sample vector is a randomly chosen object (sample) from the original data matrix, serving as a reference point for the training vector at each iteration. The initial value of the learning rate decreases with each iteration, the increment determined by a linear or exponential function. Choosing a too small number may result in an insufficient weight correction and therefore limited adjustment of codebook vectors in the *p*-dimensional space. As a consequence, some of the map units may never get a chance to fully learn the input pattern and sufficiently represent all the samples (Brereton 2009; Brown et al. 2009; Hastie et al. 2009).

The *number of iterations* (*vii*) defines the amount of repetitions in the training cycle. Too small a number may result in an undertrained network, which in turn leads to incorrect clustering. On the other hand, a too large number may lead to waste of computational time and resources on redundant cycles when an optimal solution had been found a few repetitions prior. The recommended number of iteration samples is the number of map units multiplied by a factor of 500 (Brereton 2009).

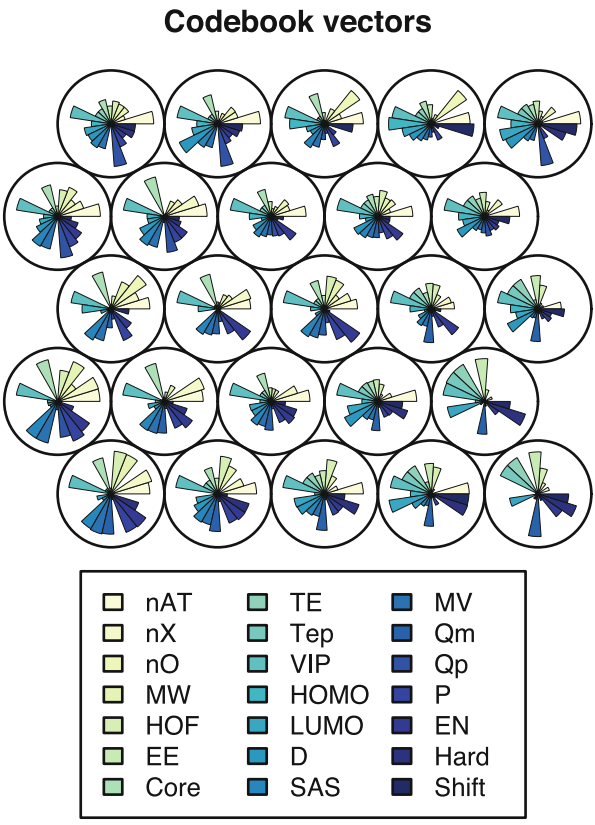

Fig. 27 Optimal weight vectors of the output neurons (*codebook vectors*) for the POP data set

Training

After establishing all the necessary parameters, it is time to begin training the network, that is, performing an iterative search of optimal weight vectors of the output neurons (Fig. 27). The steps of this iterative process are as follows (Vesanto and Alhoniemi 2000; Brereton 2009; Brown et al. 2009):

1. Initializing the network – selecting the initial weights for M neuron. The most common method is a random assignment of a value from a min-max range. The number of weights in the codebook vector is equal to the number of variables (features) in the original data set.
2. Selecting, usually at random, a single input vector.
3. Calculating the distances between the input vector and the neurons described by the codebook vector in the p-dimensional variable space. The choice of distance metric is left to the user – a list of the most commonly used one can be found elsewhere in this chapter.

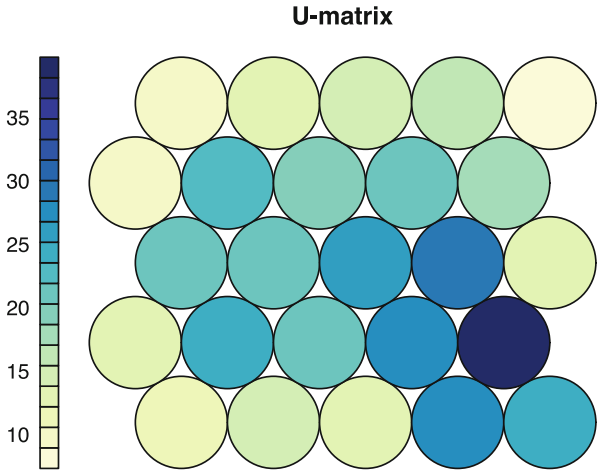

Fig. 28 SOM U-matrix for the POP data set

4. Identifying the winner neuron, or best matching unit, the most similar, i.e., the closest, to the input vector.
5. Adjustment of neuron positions. There are two methods of repositioning the nodes in the variable space:
   (a) *Winner takes all*, *WTA* – where only the winner neuron is relocated in the direction of the input vector
   (b) *Winner takes most*, *WTM* – where the winner neuron and the associated neurons are repositioned in the vector space
6. Until the user-defined number of cycles has been reached, repeat the cycle starting from step 2.

Visualization

Once we have determined the optimal weights of the output layer neurons, we can graphically represent the relationships between them and the original data. There are several aspects of SOMs to visualize (Brereton 2009):

1. Unified distance matrix, or U-matrix, which illustrates the similarity between neurons (map units), color-coded by distance (Fig. 28). Based on the U-matrix, it is possible to identify outliers as well as clusters, as neurons representing similar samples are positioned at adjacent regions of the map.
2. Hit histogram, where the size of the map unit is proportional to the number of points (objects) represented by that neuron (Fig. 29).
3. Component planes, which demonstrate how a specific variable influences the SOMs (Fig. 30). Each variable from the data set can be represented on a separate component plane.

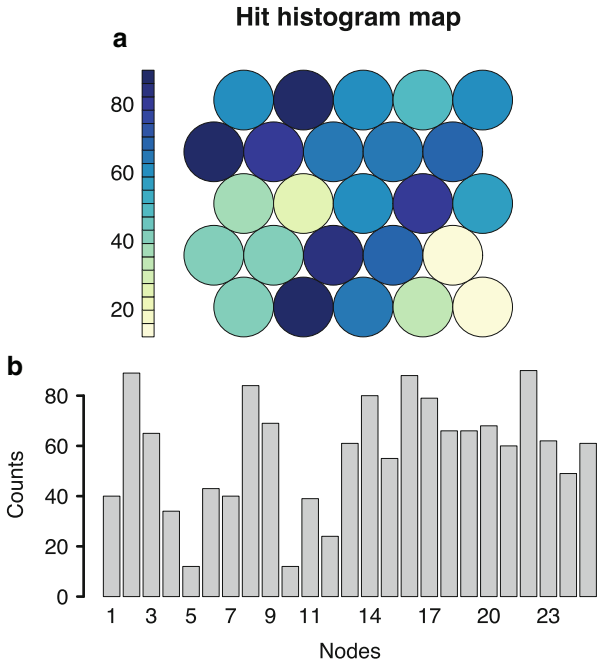

Fig. 29 Two ways of representing SOM hit histogram: (**a**) a 2D histogram, (**b**) a "classical" barplot

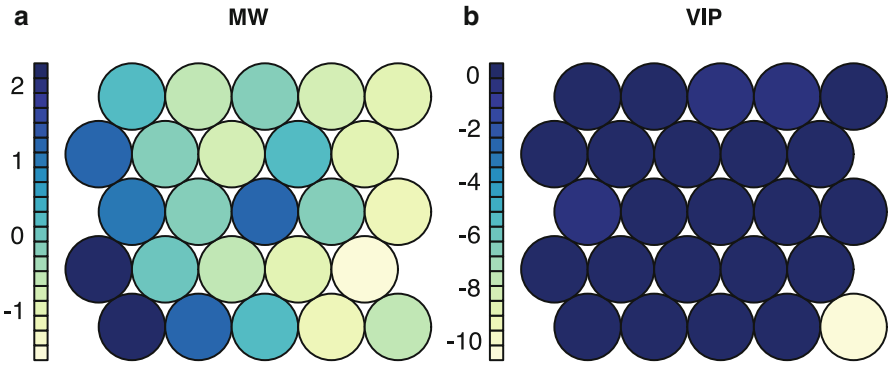

Fig. 30 SOM component planes: (**a**) molecular weight, MW, (**b**) vertical ionization potential, VIP

Quality Assessment

In order to assess the quality of the SOM training, there are a number of measures that can be employed. Among the most popular are the (Brereton 2009):

- MQE, mean quantization error. The average distance between each input vector and its best matching unit, calculated according to Eq. 13:

$$\mathrm{MQE} = \frac{\sum_{i=1}^{n} d\left(x_i, w_c\right)}{n} \tag{13}$$

where $d(x_i,\ w_c)$ is the distance between input vector (x_i) and weight vector of the winner neuron; n is a number of input patterns.

- TE, topographic error. The ratio of all input vectors for which the first and second BMUs are not adjacent, calculated according to Eq. 14:

$$\mathrm{TE} = \frac{\sum_{i=1}^{n} l\left(x_i\right)}{n} \tag{14}$$

where $l(x_i)$ is a function equal to 1 when the first and second most similar codebook vector of a particular sample is from adjacent units; otherwise, the function is 0.

Summary

In this chapter, we present selected unsupervised methods of data grouping and exploration. Each of them allows the user to focus on different aspects of the data. Hierarchical cluster analysis enables the initial exploration of the data structure and determining which objects are similar to each other in the context of various complexity levels. The *k*-means approach allows the user to more precisely identify group membership by an iterative search of the variable space and real-time improvement of clustering results. Thanks to modified versions of *k*-means, such as *k*-medoids and *k*-modes, we are able to handle outliers and categorical data with ease. With the help of robust exploratory methods, such as the principal component analysis and self-organizing Kohonen maps, it is possible to examine large data sets and determine which of the variable is crucial in describing underlying data structure.

Each of the presented techniques has some drawbacks and limitations, which is why user expertise and experience, as well as extensive knowledge on the inner workings of these approaches, are essential for effective data analysis. We hope that this chapter will serve as an overview and a starting point for further pursuit of knowledge in this field.

Bibliography

Brereton, R. G. (2003). *Chemometrics: Data analysis for the laboratory and chemical plant*. Chichester/Hoboken: Wiley.

Brereton, R. G. (2009). *Chemometrics for pattern recognition*. Chichester: Wiley.

Brown, S. D., TauleriFerre, R., & Walczak, B. (2009). *Comprehensive chemometrics: Chemical and biochemical data analysis*. Amsterdam/London: Elsevier.

Everitt, B., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis* (5th ed.). Oxford: Wiley-Blackwell.

Gajewicz, A., Haranczyk, M., & Puzyn, T. (2010). Predicting logarithmic values of the subcooled liquid vapor pressure of halogenated persistent organic pollutants with QSPR: How different are chlorinated and brominated congeners? *Atmospheric Environment, 44*(11), 1428–1436.

Gemperline, P. (2006). *Practical guide to chemometrics* (2nd ed.). Boca Raton: CRC/Taylor & Francis.

Golebiowski, M., Sosnowska, A., Puzyn, T., Bogus, M. I., Wieloch, W., Włóka, E., & Stepnowski, P. (2014). Application of two-way hierarchical cluster analysis for the identification of similarities between the individual lipid fractions of Lucilia sericata. *Chemistry and Biodiversity, 11*, 733–748.

Han, J., Kamber, M., & Pei, J. P. D. (2012). *Data mining: Concepts and techniques* (3rd ed.). Waltham/Oxford: Morgan Kaufmann/Elsevier Science, distributor.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). New York: Springer.

Jolliffe, I. T. (2002). *Principal component analysis* (Springer series in statistics 2nd ed.). New York: Springer.

Khan, S. S., & Kant, S. (2007). Computation of initial modes for K-modes clustering algorithm using evidence accumulation. Paper presented at the Proceedings of the 20th international joint conference on artificial intelligence, Hyderabad.

Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Berlin/London: Springer.

Kountchev, R., & Iantovics, B. (2013). *Advances in intelligent analysis of medical data and decision support systems* (Studies in Computational Intelligence, Vol. 473). Springer International Publishing Switzerland.

Li, Y., Pang, G.-F., Fan, C.-L., & Chen, X. (2013). Hierarchical cluster analysis of matrix effects on 110 pesticide residues in 28 tea matrixes. *Journal of AOAC International, 96*(6), 1453–1465.

Livingstone, D. (2009). *A practical guide to scientific data analysis*. Chichester: Wiley.

Maimon, O. Z., & Rokach, L. (2005). *Data mining and knowledge discovery handbook*. Ramat-Aviv: Springer.

Milligan, G., & Cooper, M. (1985). An examination of procedures for determining the number of clusters in a data set. *Psychometrika, 50*(2), 159–179.

Myatt, G. J. (2007). *Making sense of data: A practical guide to exploratory data analysis and data mining*. Hoboken: Wiley-Interscience.

Petushkova, N. A., Pyatnitskiy, M. A., Rudenko, V. A., Larina, O. V., Trifonova, O. P., Kisrieva, J. S., Samenkova, N. F., Kuznetsova, G. P., Karuzina, I. I., & Lisitsa, A. V. (2014). Applying of hierarchical clustering to analysis of protein patterns in the human cancer-associated liver. *PloS One, 9*(8), e103950.

Schnegg, M., Massonnet, G., & Gueissaz, L. (2015). Motorcycle helmets: What about their coating? *Forensic Science International, 252*, 114–126.

Skwarzec, B., Kabat, K., Puzyn, T., & Astel, A. (2011). Inflow of polonium, uranium and plutonium radionuclides in Odra River catchment area assessment by environmetric expertise. *Journal of Radioanalytical and Nuclear Chemistry, 292*(2), 519–529.

Varmuza, K., & Filzmoser, P. (2009). *Introduction to multivariate statistical analysis in chemometrics*. CRC Press: Boca Raton, p xiii, 321 p.

Vesanto, J., & Alhoniemi, E. (2000). Clustering of the self-organizing map. *IEEE Transactions on Neural Networks/A Publication of the IEEE Neural Networks Council, 11*(3), 586–600.