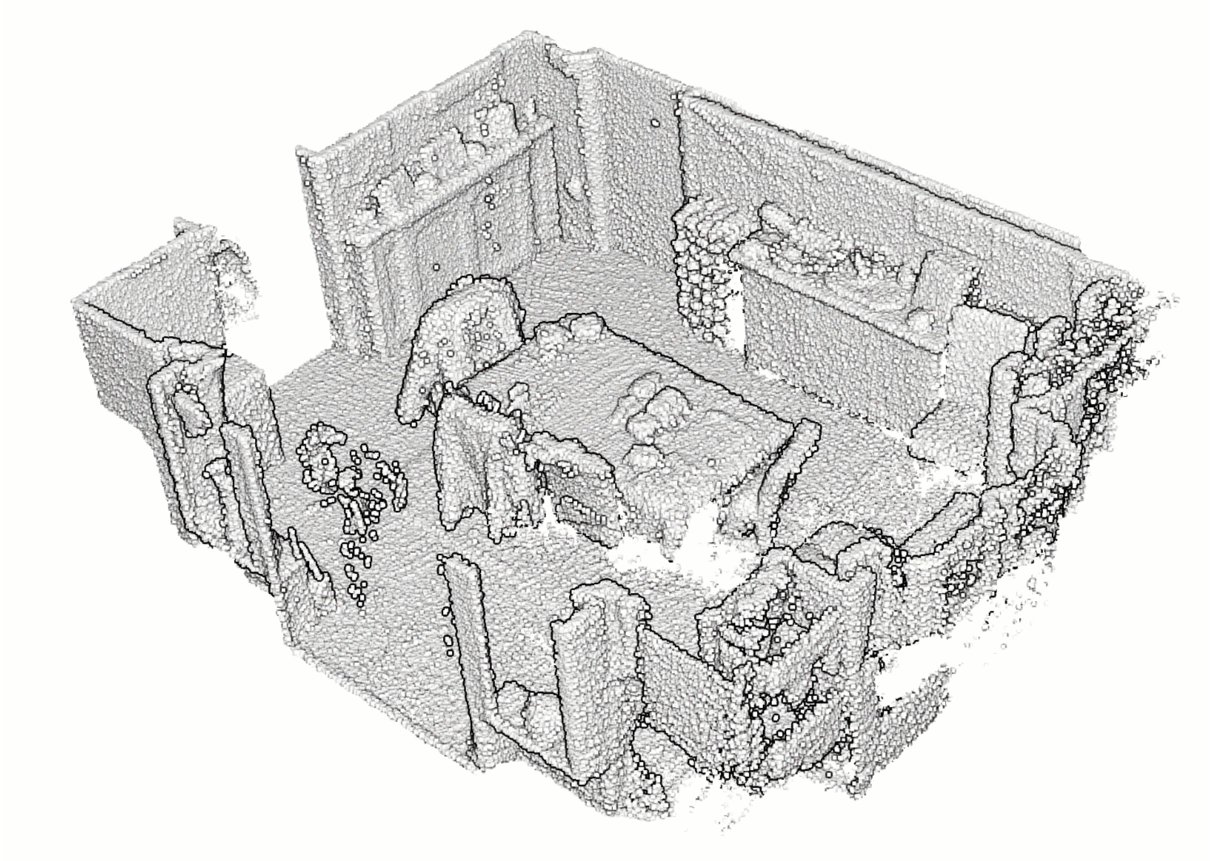
Fundamentals to clustering high-dimensional data (3D point clouds)

Why unsupervised segmentation & clustering is the “bulk of AI”? What to look for when using them? How to evaluate performances? Explications and Illustration over 3D point cloud data.

Clustering algorithms allow data to be partitioned into subgroups, or clusters, in an unsupervised manner. Intuitively, these segments group similar observations together. Clustering algorithms are therefore highly dependent on how one defines this notion of similarity, which is often specific to the field of application.

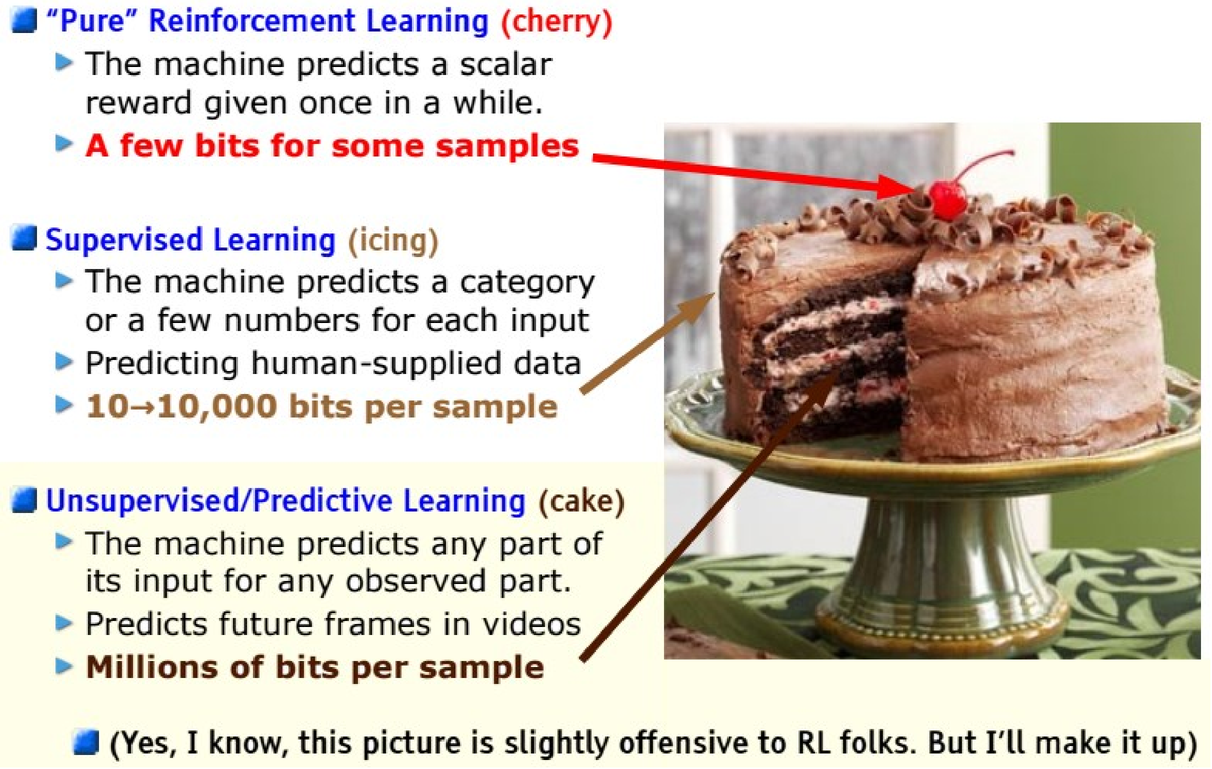


Different clustering strategies applied to this noisy point cloud of a room. One can see that spatial proximities seems a choice criterion to define this similarity to constitute segments.

What is clustering?

Clustering algorithms are often used for exploratory data analysis. They also constitute the bulk of the processes in AI classification pipelines to create nicely labeled datasets in an unsupervised/self-learning fashion.

Published in Towards Data Science



- **"Pure" Reinforcement Learning (cherry)**
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning (icing)**
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning (cake)**
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**
- (Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)

Original LeCun cake analogy slide presented at NIPS 2016, the highlighted area has now been updated.

Within the scope of 3D Geodata, clustering algorithms (also defined as unsupervised segmentation) permit to obtain a segment soup that becomes the backbone of several processes such as feature extraction, classification or 3D modeling as illustrated below.

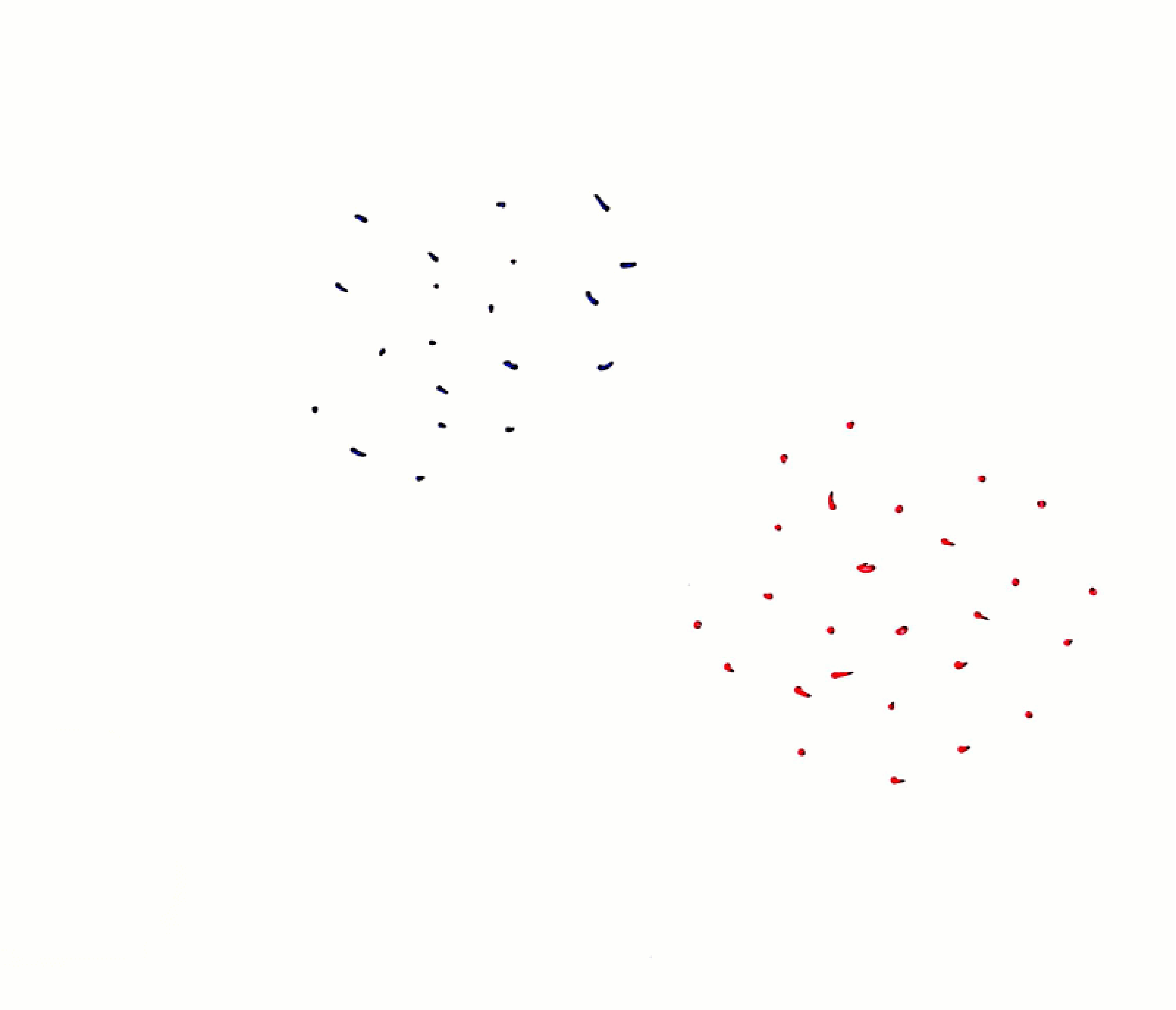
Here you can see an automatic modeling process that leverages the segment information for a nice extraction of a 3D mesh. © Florent Poux, Ph.D.

Published in Towards Data Science

Aside from Geodata applications, they are used to identify :

- customers with similar behaviors (market segmentation);
- users who have similar uses of a tool;
- communities in social networks;
- recurring patterns in financial transactions.

They most often act in addition to a dimensionality reduction algorithm that allows the different attributes (called dimensions) to be viewed in two or three dimensions. If a “view” presents sufficient decorrelation, a clustering algorithm can be used to form sub-groups of these points—the clusters—as illustrated below.

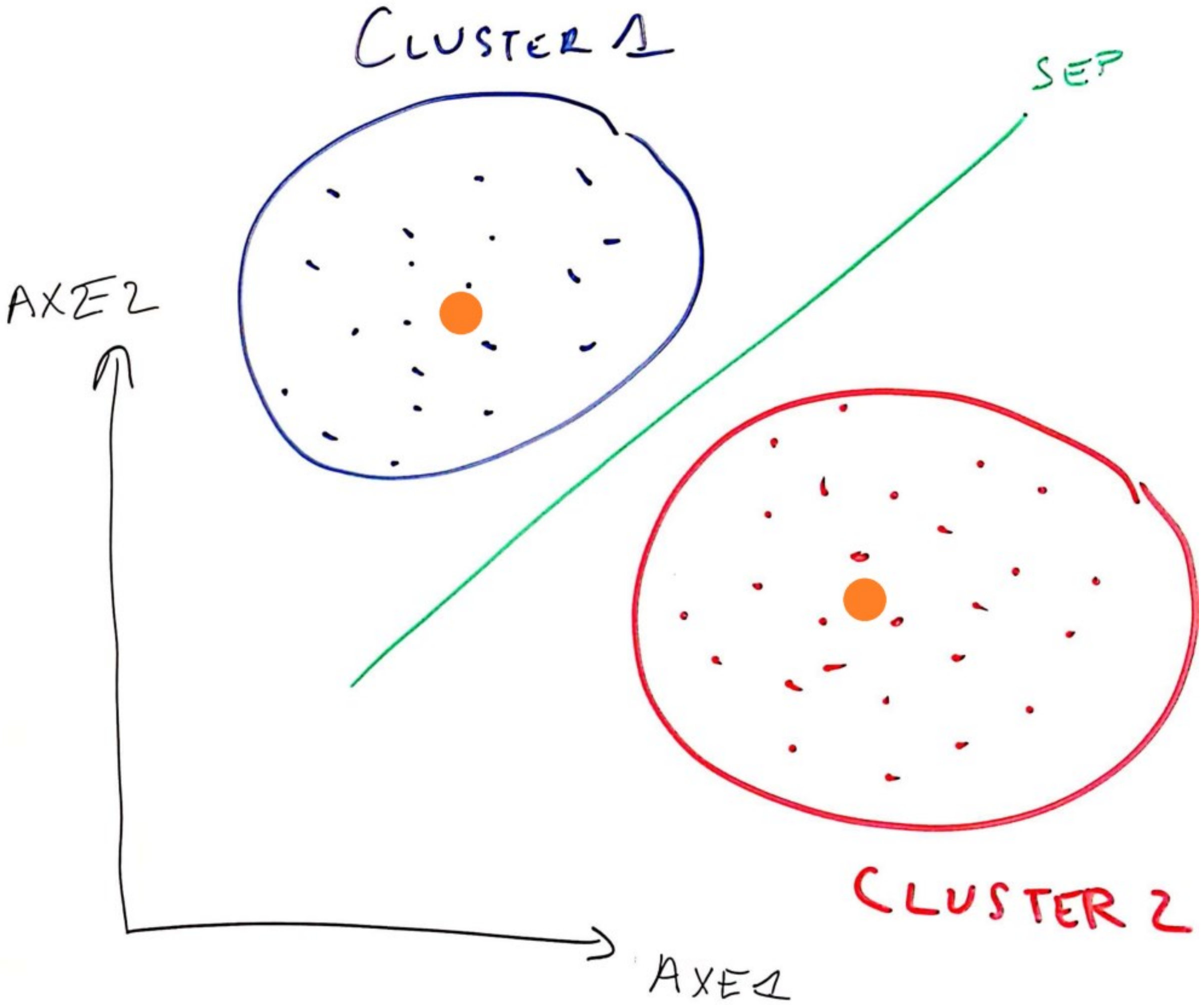


A simple illustration of finding two clusters by creating a line to separate the dataset in two sub-groups.

In this way, the relationships between the points can be visually represented. Alternatively, instead of representing the entire data, only one representative point per cluster can be displayed.

Published in Towards Data Science

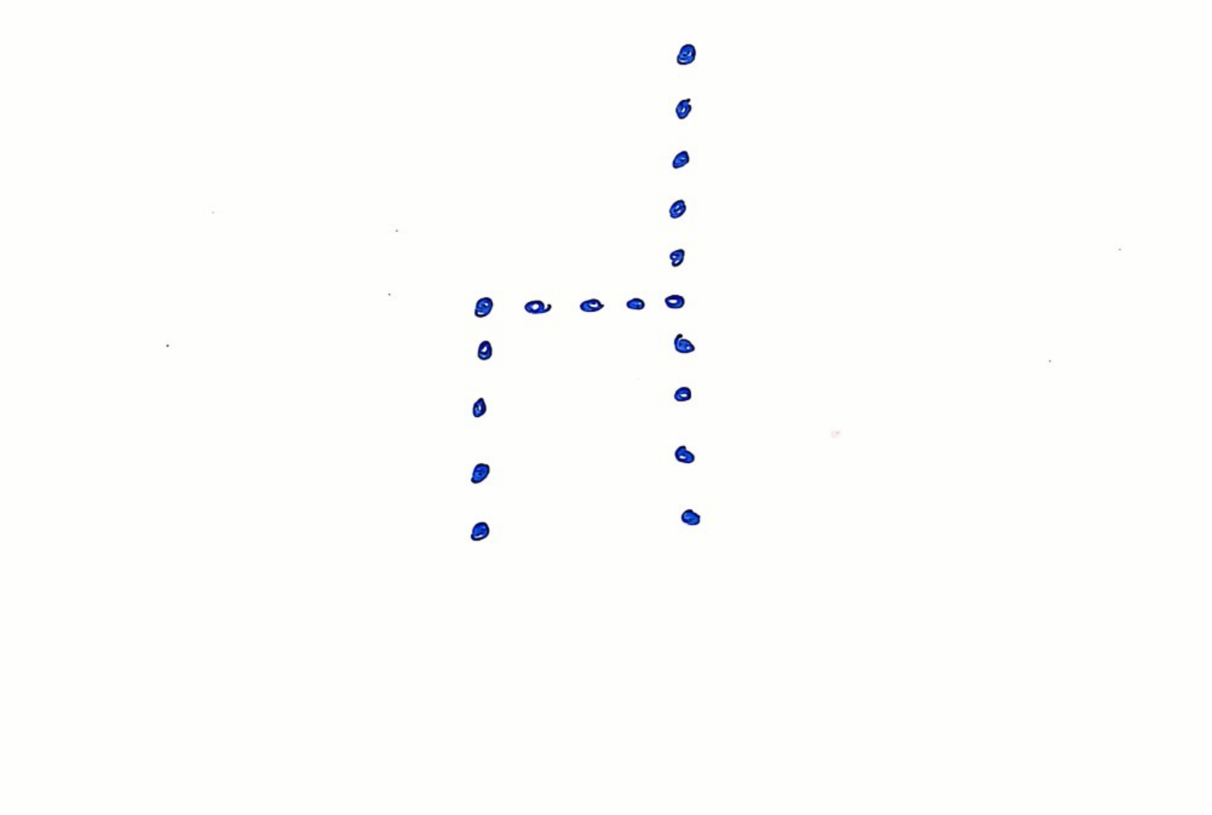
Once clusters have been identified, data can also be viewed using only one representative per cluster and discarding the others.



Determination of centroids of two clusters to act as the new base data. © Florent Poux, Ph.D.

Why is this useful?

Clustering algorithms are particularly useful in the frequent cases where it is expensive to label data. Take the example of annotating a large point cloud. Annotating each point by what it represents can be a long and tedious job, to the point that the people doing it can unintentionally introduce errors through inattention or fatigue. It is cheaper and perhaps even more efficient to let a clustering algorithm group similar points together and then only involve a human operator when assigning a label to the cluster.



Simple illustration over a chair of one advantage within semantic segmentation workflows. © Florent Poux, Ph.D.

Thus, clustering algorithms can be used to extend a property of one of the points in the same cluster to all the points in the same cluster (in the previous example, the represented chair object.).

Outside the geodata scope, inferring data properties is useful for:

- finding similar images, likely to represent the same object, the same animal or the same person;
- extracting similar texts, likely to speak about the same subject;
- searching in an image the pixels that belong to the same object (which is called segmentation).

In the examples above, the subjects (image, text, pixels) are represented as 2D/3D/nD points and are then grouped into clusters. Then, it is sufficient to infer that if one image in a cluster represents a duck, all the images part of the cluster are likely to represent ducks.

We will define several criteria to be optimized to define an interesting partition of the data. These are then used to derive some of the best-known clustering algorithms, and will be covered in another article as else the read would be a bit dense.

Published in Towards Data Science

How to know if the clustering is representative?

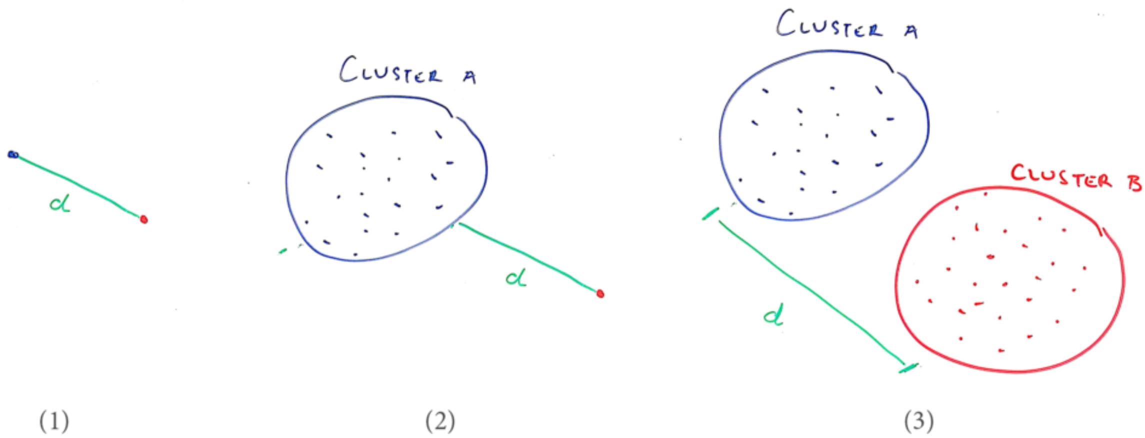
In the case of unsupervised algorithms, the purpose of the algorithm is less obvious to define than in the case of supervised algorithms, where there is a clear task to accomplish (E.g. classification or regression). The success of the model is therefore more subjective. The fact that the task is more difficult to define does not prevent a wide range of measures of the performance which I will detail below.

Distances and similarities

Clustering means grouping together the closest or most similar points. The concept of clustering relies heavily on the concepts of distance and similarity.

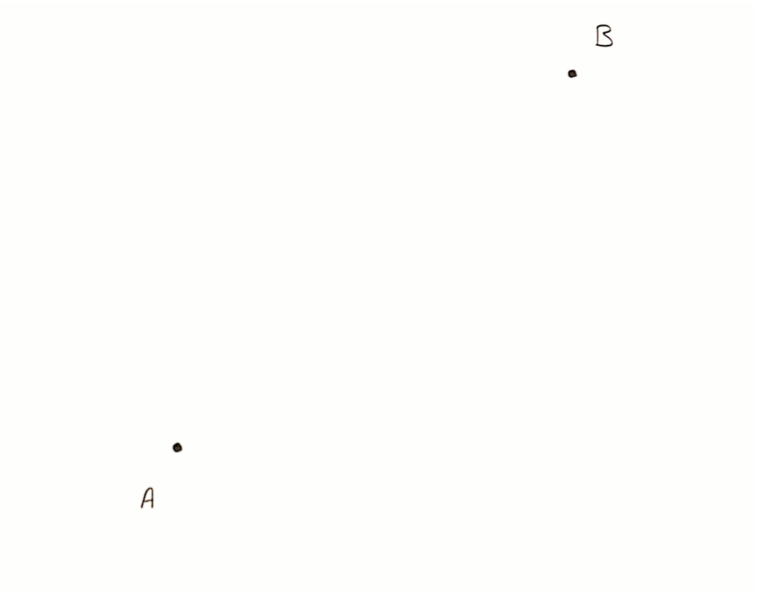
These concepts will be very useful to formalize:

- (1) How close two observations are to each other;
- (2) How close an observation is to a cluster;
- (3) How close two clusters are to each other.



Simple illustration of some distances between two observations (1), one observation and a cluster (2), two clusters (3). © Florent Poux, Ph.D.

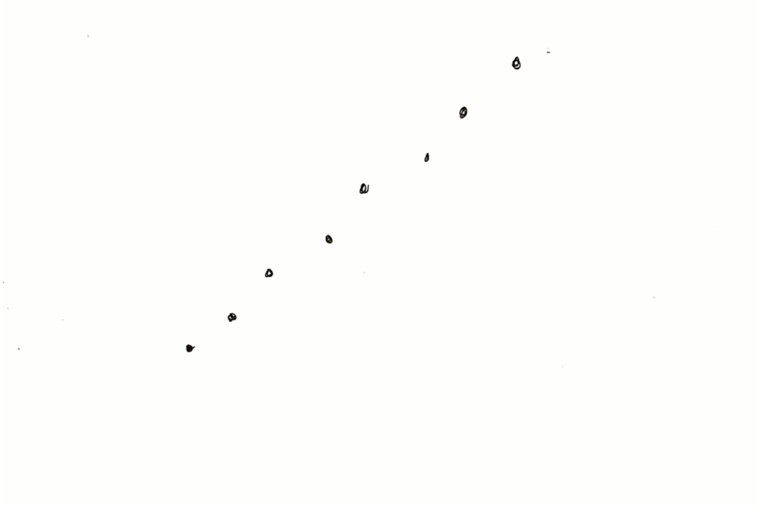
The most commonly used examples of distances are the Euclidean distance, and the Manhattan distance. The Euclidean distance is the “ordinary” straight-line distance between two points in Euclidean space. The Manhattan distance is so-called because it corresponds in two dimensions to the distance traveled by a taxi on the streets of Manhattan, which are all either parallel or perpendicular to each other.



Simple illustration of the Euclidean and Manhattan distances. © Florent Poux, Ph.D.

Thus, a distance can be used to define a similarity: the further apart two points are, the less similar they are, and vice versa. For injecting a very tiny bit of math, we can transform a distance d between x and y into a similarity measure s very simply such as: s(x,y)=1/1+d(x,y).

Another common way to define similarity is to use the Pearson correlation which measures the cosine of the angle formed by vectors x and y when the underlying data is centered.



Simple illustration of the Pearson Correlation coefficient. © Florent Poux, Ph.D.

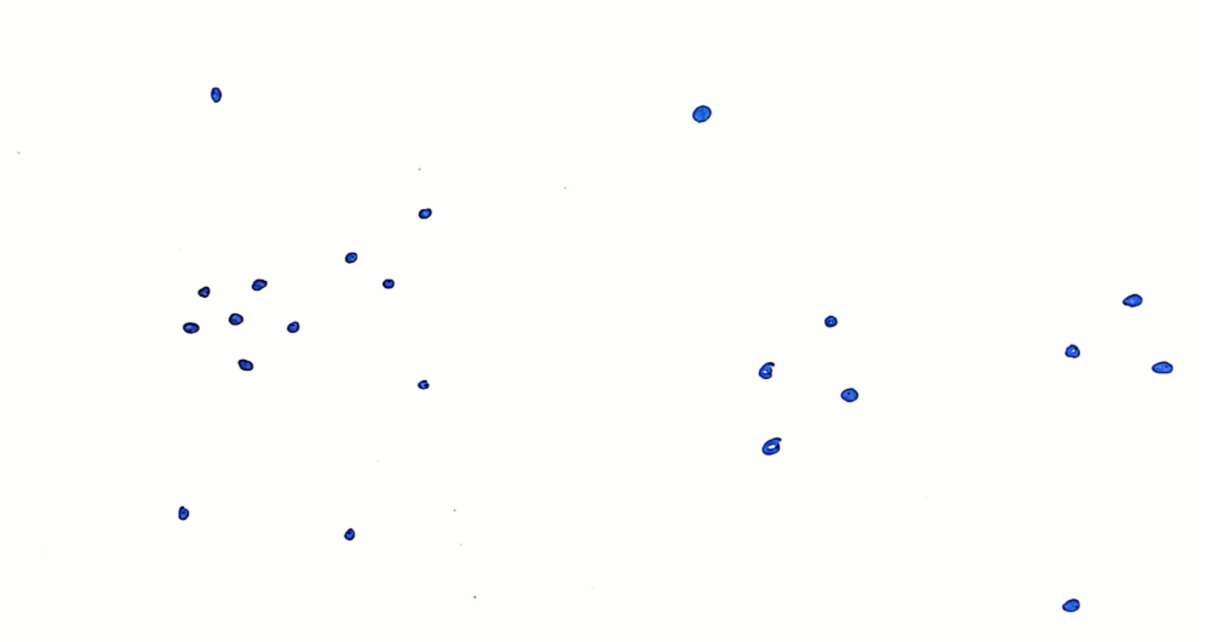
But without going too deep, it is important to note that the Pearson's correlation will take into account the shape of the distribution rather than their amplitude, which the Euclidean distance mainly takes into account. The choice of the distance measure is therefore important.

Published in Towards Data Science

Cluster shape

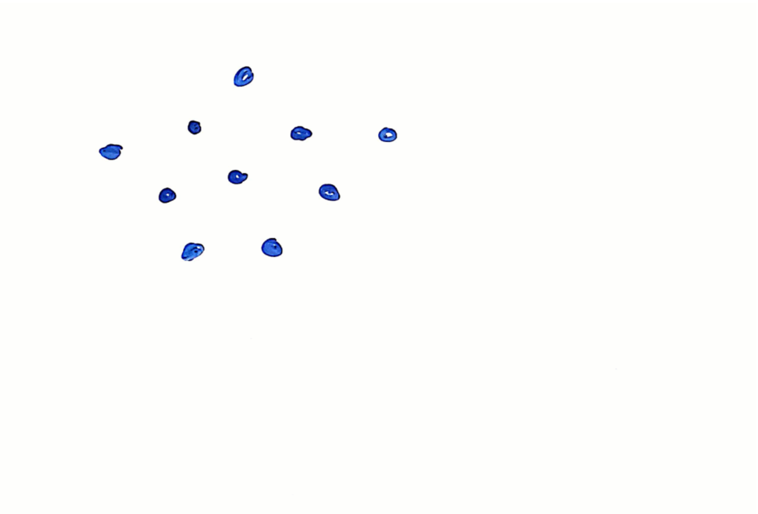
The shape of a cluster is an important element that we initially describe as:

- (1) Tightened on themselves: two close points must belong to the same cluster
- (2) far from each other: two points that are far apart must belong to different clusters.



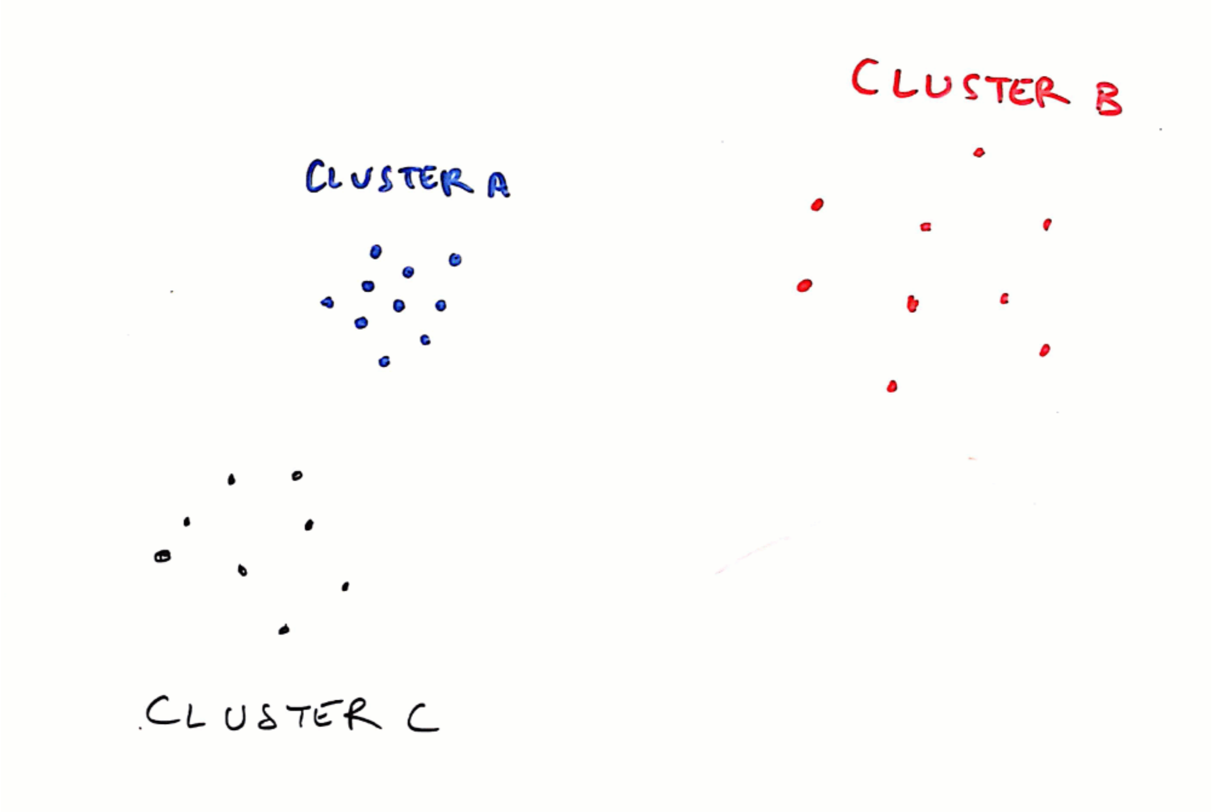
How tighten can give a hint as to the formation of coherent clusters.

Often, we search for clusters tighten on themselves. Let us translate these properties with an example, using the Euclidean distance. First, we can compute the centroid of a cluster (the barycentre of the points of this cluster) pretty easily. The homogeneity of a cluster can then be defined as the average of the distances of each of the points contained in this cluster to the centroid. In this way, a tightened cluster will have a lower heterogeneity than a cluster of scattered points. Then, to characterize not one cluster, but all clusters in our dataset, we can calculate the average of the homogeneity of each cluster.



Simple illustration about how homogeneity gives intuitive sense to better characterize clusters.

Secondly, we want the clusters to be far from each other. To quantify this, we usually define the separation of two clusters as the distance between their centroids. Once again, we can calculate the average of these quantities on all the pairs of clusters obtained.



A simple illustration to show how separation can be used to get a nice clustering.

We now have two criteria to optimize: homogeneity and separation. To make it easier for us, we can group them into a single criterion, the **Davies-Bouldin index**. The idea of this index is to compare intra-cluster distances (homogeneity)—which we want to be low—to intercluster distances (separation), which we want to be high. For a given cluster, this index is all the weaker as all the clusters are homogeneous and well separated.

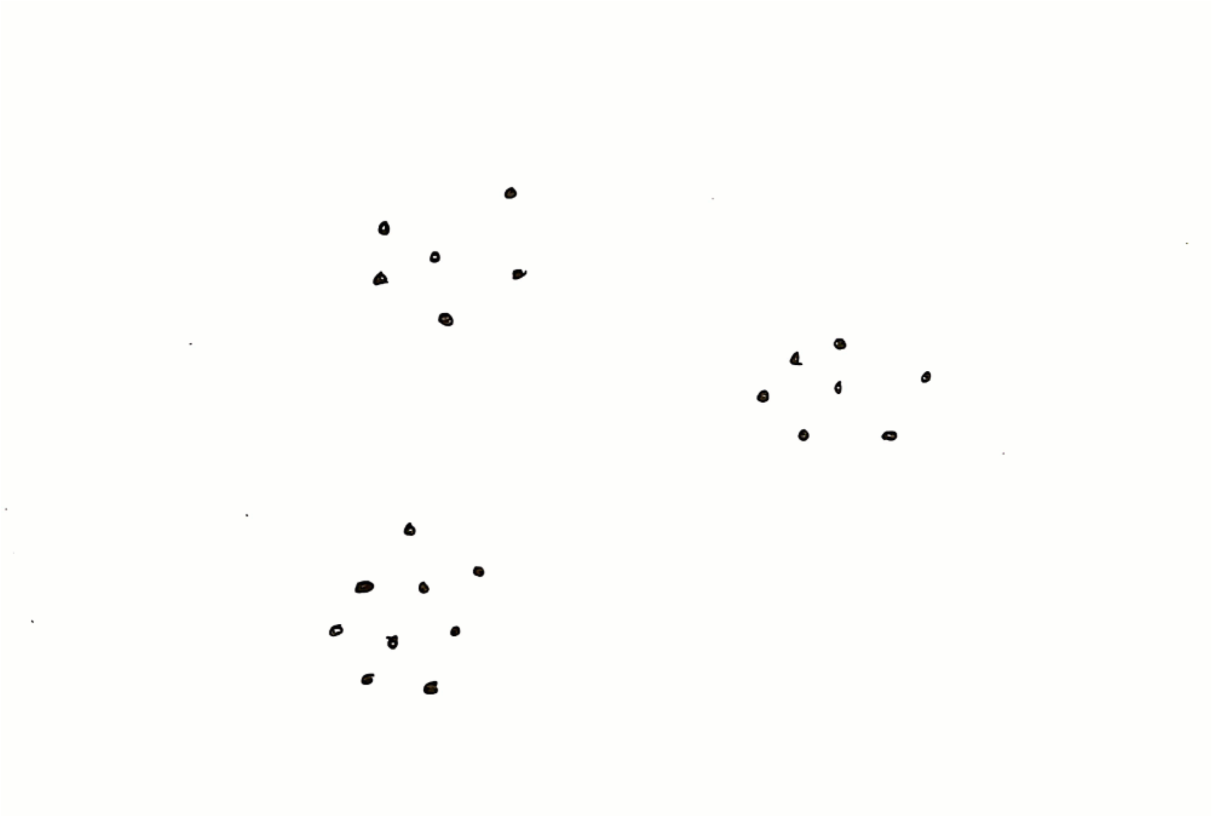
Another way to quantify how well a clustering meets these two requirements (homogeneity and separation) is to measure the so-called **silhouette coefficient**. For a given point p, the silhouette coefficient s(p) is used to assess whether this point belongs to the “right” cluster. For this, we try and answer two questions:

- Is p close to the points of the cluster to which it belongs? We can calculate the average distance a(p) of p from all the other points of the cluster to which it belongs.
- Is the point far from the other points? We calculate the smallest value b(p) that a(p) could take if p were assigned to another cluster. If p has been correctly assigned, then $a(x) < b(x)$. The silhouette coefficient is given by $s(x)=b(x)-a(x)/max(a(x),b(x))$, and ranges between -1 and 1. The closer to 1 it is, the more the assignment of p to its cluster is satisfactory.

Hint: To evaluate a clustering, its mean silhouette coefficient can be calculated, for example using scikit-learn and the command sklearn.metrics.silhouette_score.

Cluster stability

Another important criterion is the stability of the clustering: if I run the algorithm several times on the same data with a different initialization, or on different subsets of the data, or on the same slightly noisy data, do I get the same results? This criterion is particularly relevant when choosing the number of clusters: if the number of clusters chosen corresponds to the natural structure of the data, the clustering will be more stable than if it does not.

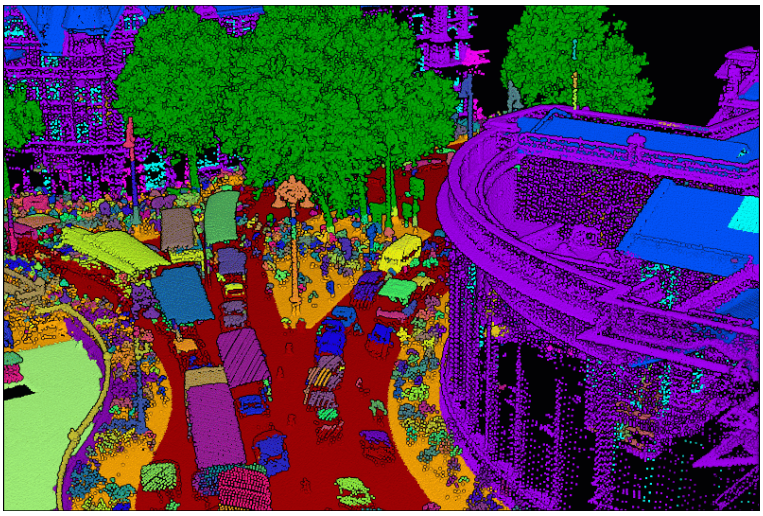


An example of the “parameter supervision” for finding clusters and its impact.

On the example above, an algorithm that tries to determine 3 clusters will reasonably find the three clusters we see. But if it is asked to determine 4 clusters, the distribution in these 4 clusters will be more random and will not necessarily be twice the same. This is one way to determine that 3 is a better number of clusters than 4.

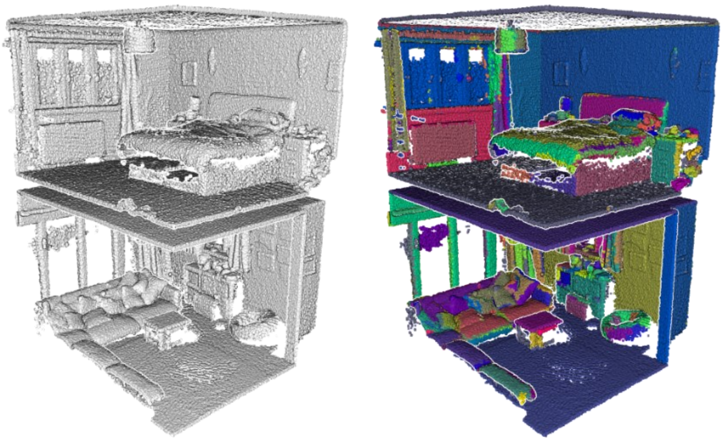
Compatibility with domain-specific knowledge

Very often, we will also evaluate a clustering algorithm “by eye”, and see if the proposed clusters make sense. Do the points grouped in this cluster all represent the same object? Do the points in these two clusters represent different objects?



Pay attention to the various clusters in the following illustration. Do they make intuitive sense? Should the central lamp post be described by one cluster? 3 clusters? more?

To do this more neatly, we can work on a dataset on which we know a reasonable partition of the data. We will then compare this partition with the one returned by our clustering algorithm. For example, we can work with a point cloud partitioned by planar shapes. The next step is to evaluate whether the groups formed by the clustering algorithm correspond to those defined a priori.



Example of taking a portion of a point cloud, and creating a “planar-labeled” dataset to compare to the clustering results.

It’s easy! It’s like evaluating a multi-class classification algorithm. But not so fast: if we are interested in whether the same objects belong to the same cluster, it doesn’t matter whether this cluster is the first, the second, or the k-th cluster. Therefore, specific performance metrics must be used to evaluate the concordance of two partitions of the dataset.

Hint: A list of these can be found in sklearn.metrics.

An example of these measures is the Rand index. The Rand index is the proportion of pairs of points (p1,p2) that are grouped in the same way in both partitions: either because, in both cases, p_1 and p_2 belong to the same cluster, or because, in both cases, p_1 and p_2 belong to different clusters.

The Rand index can be artificially inflated by predicting a lot of clusters: the pairs of points belonging to different clusters will be numerous, and there will be a good chance that two points labeled differently will be in two different clusters. The Adjusted Rand Index (ARI) corrects for this effect by normalizing the Rand Index (RI): ARI=RI-E(RI)/max(RI)-E(RI), where E(RI) is the expected value of the Rand index, i.e. the index obtained by partitioning the data at random. This adjusted index is close to 0 for random clustering and equal to 1 only when the clustering corresponds exactly to the initial partition.

Hint: *In* scikit-learn *it can be calculated thanks to* sklearn.metrics.adjusted_rand_score

Conclusion

Unsupervised and self-learning methods are very important for solving automation challenges. Particularly, in the era of deep learning, creating labeled datasets manually is tedious, and ways to alleviate this process are more than welcome. Clustering algorithms provide crucial solutions for this, and are used to partition a dataset into sub-groups of similar observations:

- They can be used to better understand the data;
- They can be used to facilitate data visualization;
- They can be used to infer data properties.

Then, to evaluate a clustering algorithm, we can look at :

- the shape of the clusters it produces (are they dense, well separated). The silhouette coefficient is often used here;
- the stability of the algorithm;
- the compatibility of the results with domain-specific knowledge, which can be evaluated using enrichment measures.

Published in Towards Data Science

References

1. Poux, F.; Neuville, R.; Hallot, P.; Billen, R. MODEL FOR SEMANTICALLY RICH POINT CLOUD DATA. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2017**, *IV-4/W5*, 107–115, doi:10.5194/isprs-annals-IV-4-W5-107-2017.

2. Poux, F. The Smart Point Cloud: Structuring 3D intelligent point data, Liège, 2019.

3. Poux, F.; Billen, R. Smart point cloud: Toward an intelligent documentation of our world. In *PCON*; Liège, 2015; p. 11.

4. Poux, F.; Neuville, R.; Nys, G.-A.; Billen, R. 3D Point Cloud Semantic Modelling: Integrated Framework for Indoor Spaces and Furniture. *Remote Sens.* **2018**, *10*, 1412, doi:10.3390/rs10091412.

5. Poux, F.; Neuville, R.; Van Wersch, L.; Nys, G.-A.; Billen, R. 3D Point Clouds in Archaeology: Advances in Acquisition, Processing and Knowledge Integration Applied to Quasi-Planar Objects. *Geosciences* **2017**, *7*, 96, doi:10.3390/geosciences7040096.

6. Poux, F.; Billen, R. Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring vs deep learning methods. *ISPRS Int. J. Geo-Information* **2019**, *8*, 213, doi:10.3390/ijgi8050213.

7. Poux, F.; Billen, R. Laser Scanning. In *Laser scanning: an emerging technology in structural engineering*; Riveiro, B., Lindenbergh, R., Eds.; ISPRS Book Series; CRC Press: London, UK, 2019; pp. 127–149 ISBN 9781351018869.

8. Poux, F.; Ponciano, J. J. SELF-LEARNING ONTOLOGY FOR INSTANCE SEGMENTATION OF 3D INDOOR POINT CLOUD. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*; ISPRS, Ed.; Copernicus Publications: Nice, 2020; Vol. XLIII, pp. 309–316.

9. Poux, F.; Valembois, Q.; Mattes, C.; Kobbelt, L.; Billen, R. Initial User-Centered Design of a Virtual Reality Heritage System: Applications for Digital Tourism. *Remote Sens.* **2020**, *12*, 2583, doi:10.3390/rs12162583.

10. Tabkha, A.; Hajji, R.; Billen, R.; Poux, F. SEMANTIC ENRICHMENT OF POINT CLOUD BY AUTOMATIC EXTRACTION

AND ENHANCEMENT OF 360° PANORAMAS. *ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *XLII-2/W17*, 355–362, doi:10.5194/isprs-archives-XLII-2-W17-355-2019.

11. Kharroubi, A.; Hajji, R.; Billen, R.; Poux, F. CLASSIFICATION AND INTEGRATION OF MASSIVE 3D POINTS CLOUDS IN A VIRTUAL REALITY (VR) ENVIRONMENT. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2019**, *42*, 165–171, doi:10.5194/isprs-archives-XLII-2-W17-165-2019.

12. Poux, F.; Neuville, R.; Hallot, P.; Billen, R. Point clouds as an efficient multiscale layered spatial representation. In *Eurographics Workshop on Urban Data Modelling and Visualisation*; Vincent, T., Biljecki, F., Eds.; The Eurographics Association: Liège, Belgium, 2016.