Unsupervised Learning: k-means, PAM, Hierarchical Clustering

Ahmed Eleish

Data Analytics ITWS/CSCI/MGMT - 4600/6600 BCBP - 4600

February 10th , 2026

Tetherless World Constellation
Rensselaer Polytechnic Institute

Contents

- Unsupervised Learning (Clustering)
- k-Means Clustering
- Partitioning Around Medoids (PAM)
- Hierarchical Clustering

Unsupervised Learning

- Machine learning paradigm where algorithms learn patterns from unlabeled data.

- **Clustering** is a type of unsupervised learning with the goal of finding structure in a dataset by identifying natural clusters of data points based on a similarity criterion (usually distance).

- The observations in these clusters are generally more similar (closer) to each other than they are to points in other clusters.

k-means

k-means

- k-means clustering is an unsupervised learning algorithm that, as the name hints, finds a fixed number (*k*) of clusters in a set of data.

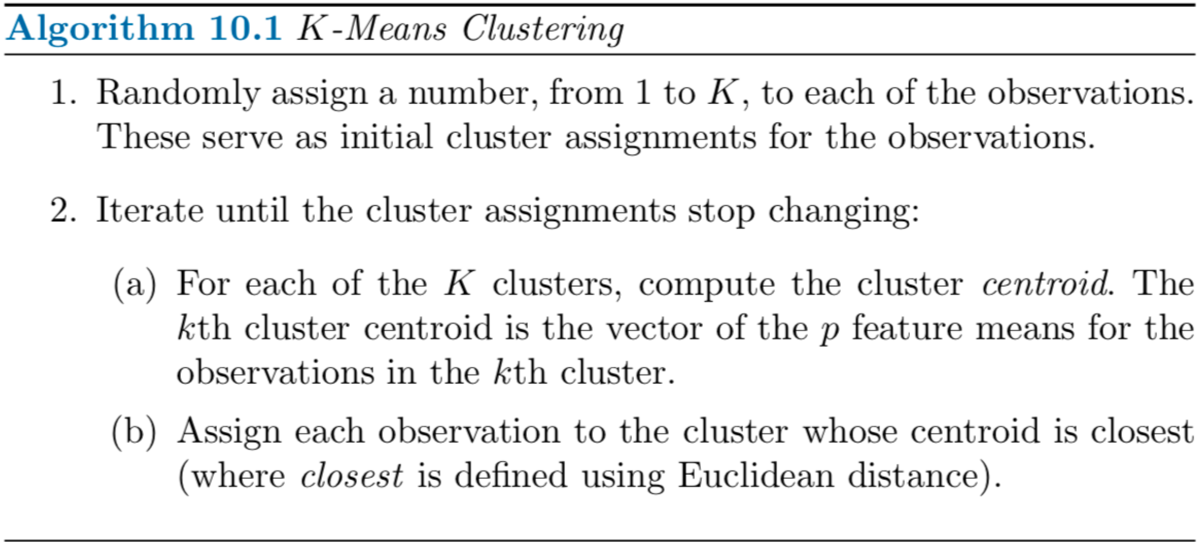
- A *cluster* is a group of data points that are grouped together due to similarities in their features. When using a k-means algorithm, a cluster is defined by a *centroid*, which is a calculated point at the center of a cluster.

- Every point in a data set is part of the cluster whose centroid is most closely located in feature space. To put it simply, k-means finds *k* number of centroids, and then assigns all data points to the closest cluster while minimizing the sum of squared distances from the points to their assigned centroid.

- k-means assumes spherical clusters.

- k-means is sensitive to outliers.

k-means Algorithm



Algorithm 10.1 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

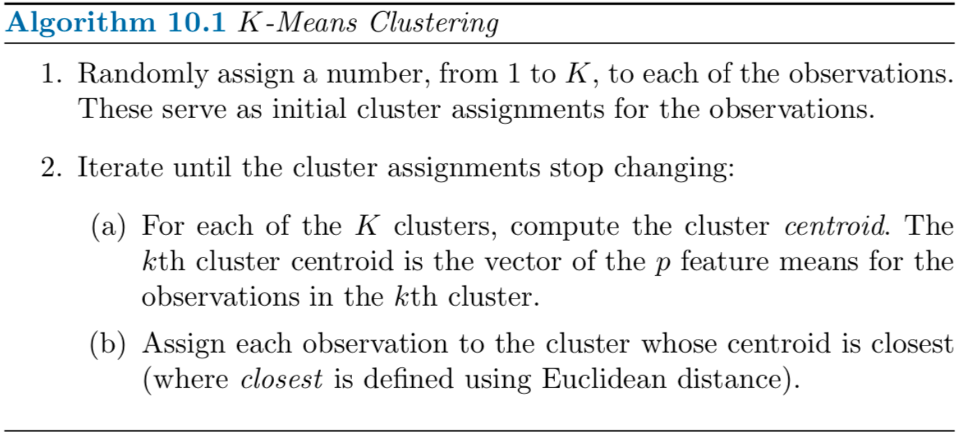
2. Iterate until the cluster assignments stop changing:

   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

k-means Algorithm



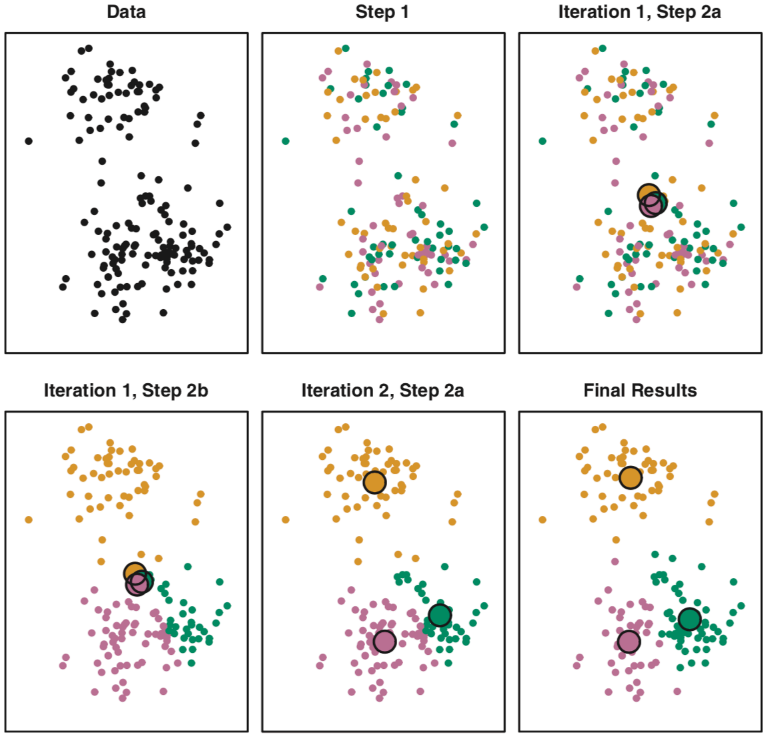
Algorithm 10.1 *K-Means Clustering*

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.

2. Iterate until the cluster assignments stop changing:

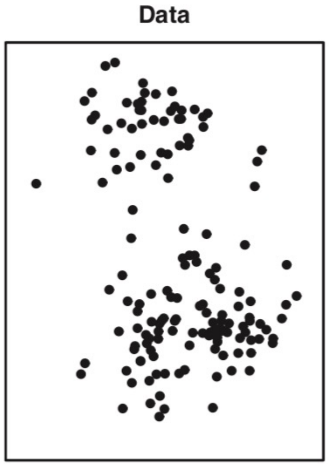
   (a) For each of the K clusters, compute the cluster *centroid*. The kth cluster centroid is the vector of the p feature means for the observations in the kth cluster.

   (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).

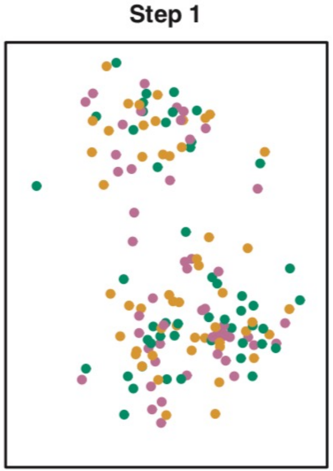


Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

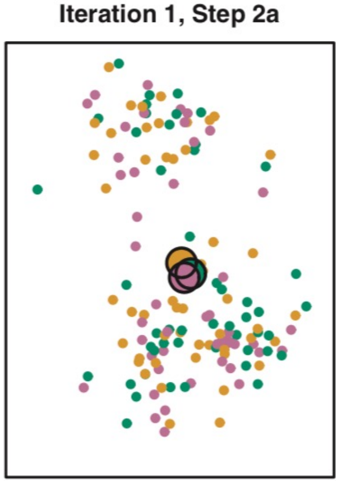
k-means Algorithm


Data

Observations (data) is shown


Step 1

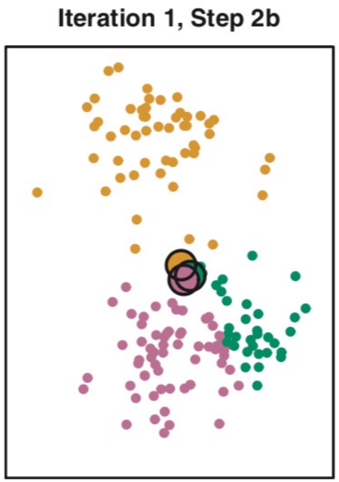
Step 1 of the algorithm: each observation is randomly assigned to a cluster


Iteration 1, Step 2a

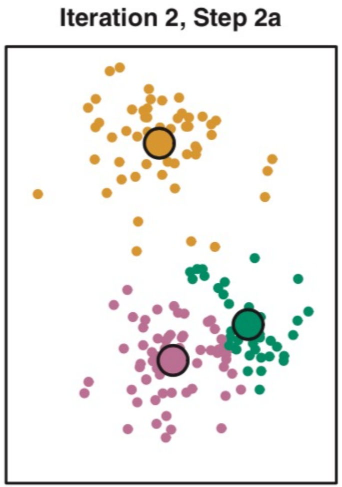
Iteration1 Step 2(a): The cluster centroids are computed; these are shown in large colored disks. Initially centroids are almost completely overlapping because the initial cluster assignment were chosen at random

Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

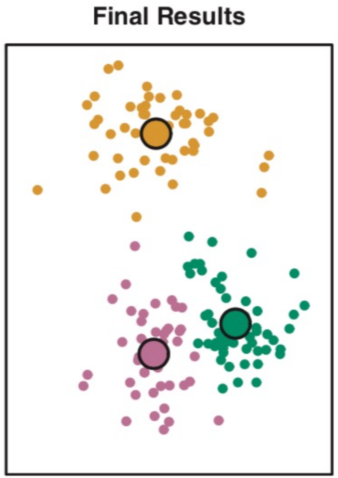
k-means Algorithm



Iteration 1 Step 2(b) : each observation is assigned to the nearest centroid



Iteration 2, Step 2(a): the step 2(a) is once again performed, leading to new cluster centroids.



Final Results: the results obtained after ten iterations. You can see the distinct clusters with their centroids.

Image/Photo Credit: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans
Reference: Introduction to Statistical Learning with Applications in R, 7th Edition, Chapter 10 – KMeans

- k-means clustering Animation

http://shabal.in/visuals/kmeans/6.html

Evaluating k-means models

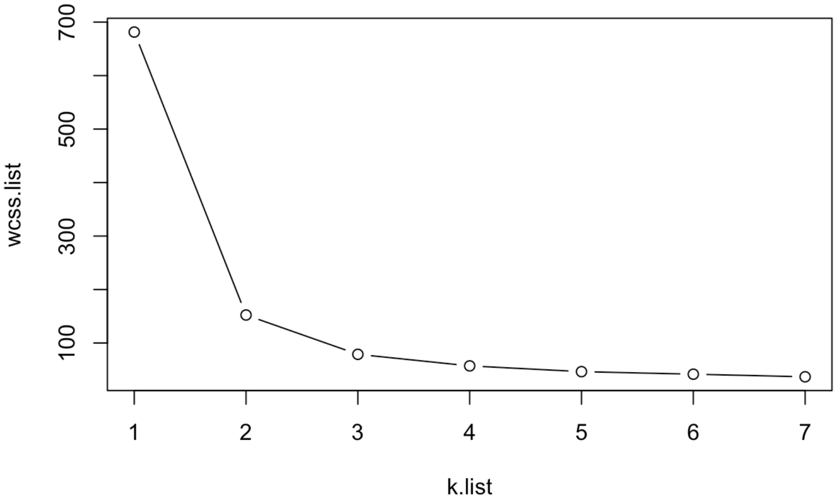
- The parameter *k* represents the number of clusters to be identified by the algorithm
- Depends on background knowledge/research question
- Finding the optimal value for *k*
    - For a range of *k* values, train a k-means model and calculate within cluster sum of squares (WCSS) or other measure.
    - Select *k* value where after which the change in the measure diminishes.
    - This is called the elbow method

Evaluating k-means models (elbow Method)

Within-Cluster Sum of Squares: sum of squared Euclidean distances between all points in a cluster and the cluster's centroid.

$$WCSS = \sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{x} - \mathbf{c}_i||^2$$

- The elbow method is a heuristic that can be subjective and unreliable.



Plot of total within cluster sum of squares with values of *k*

Evaluating k-means models (silhouette method)

Silhouette value: a measure of similarity between a point and its cluster

$$a(i) = \frac{1}{|C_i| - 1} \sum_{j \in C_i, i \neq j} d(i, j)$$

Mean distance between *i* and other points in the cluster

$$b(i) = \min_{J \neq I} \frac{1}{|C_j|} \sum_{j \in C_j} d(i, j)$$

Minimum mean distance between *i* and all points in any other clsuter

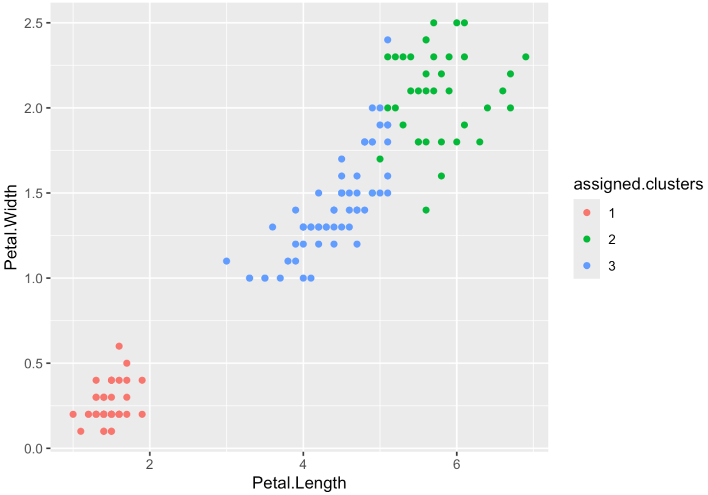
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}, \text{ if } |C_i| > 1$$

$$s(i) = 0, \text{ if } |C_i| = 1$$

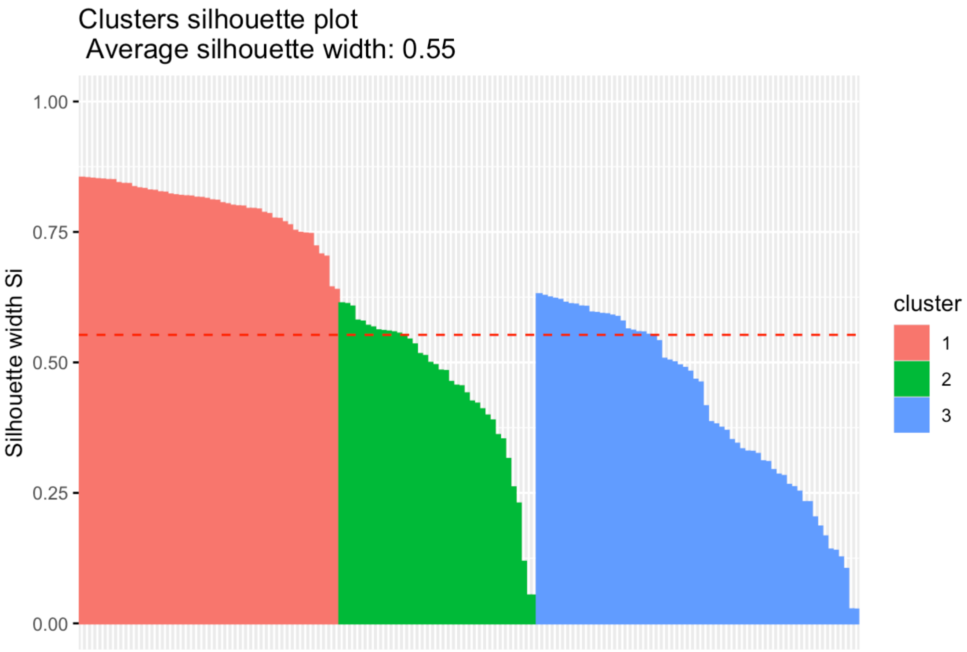
https://en.wikipedia.org/wiki/Silhouette_(clustering)

Evaluating k-means models (silhouette method)

Silhouette value: a measure of similarity between a point and its cluster



https://en.wikipedia.org/wiki/Silhouette_(clustering)



Partitioning Around Medoids (PAM)

Partitioning Around Medoids (PAM)

- PAM, also called k-medoids, is similar to k-means but instead of calculating centroids, this algorithm selects actual data points (medoids) as cluster centers

- PAM is more robust to outliers and noise.

- The algorithm can be computationally expensive because it calculates pairwise distances between all data points.

Sadeghi, B. (2025). Clustering in geo-data science: Navigating uncertainty to select the most reliable method. *Ore Geology Reviews*, 106591. https://doi.org/10.1016/j.oregeorev.2025.106591

Partition Around Medoids (PAM)

Algorithm:

- **Build Phase** (find initial *k* medoids):

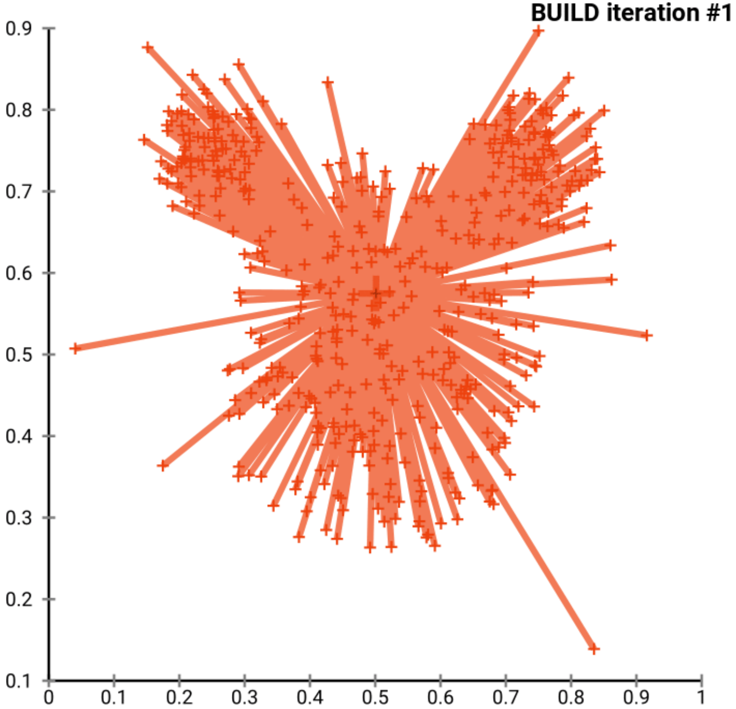
  - Select *k* points with the least cost, i.e. sum of distances to all other points.
  - Assign all non-medoid points to the cluster whose medoid is closest.

- **Swap Phase** (find best *k* medoids):

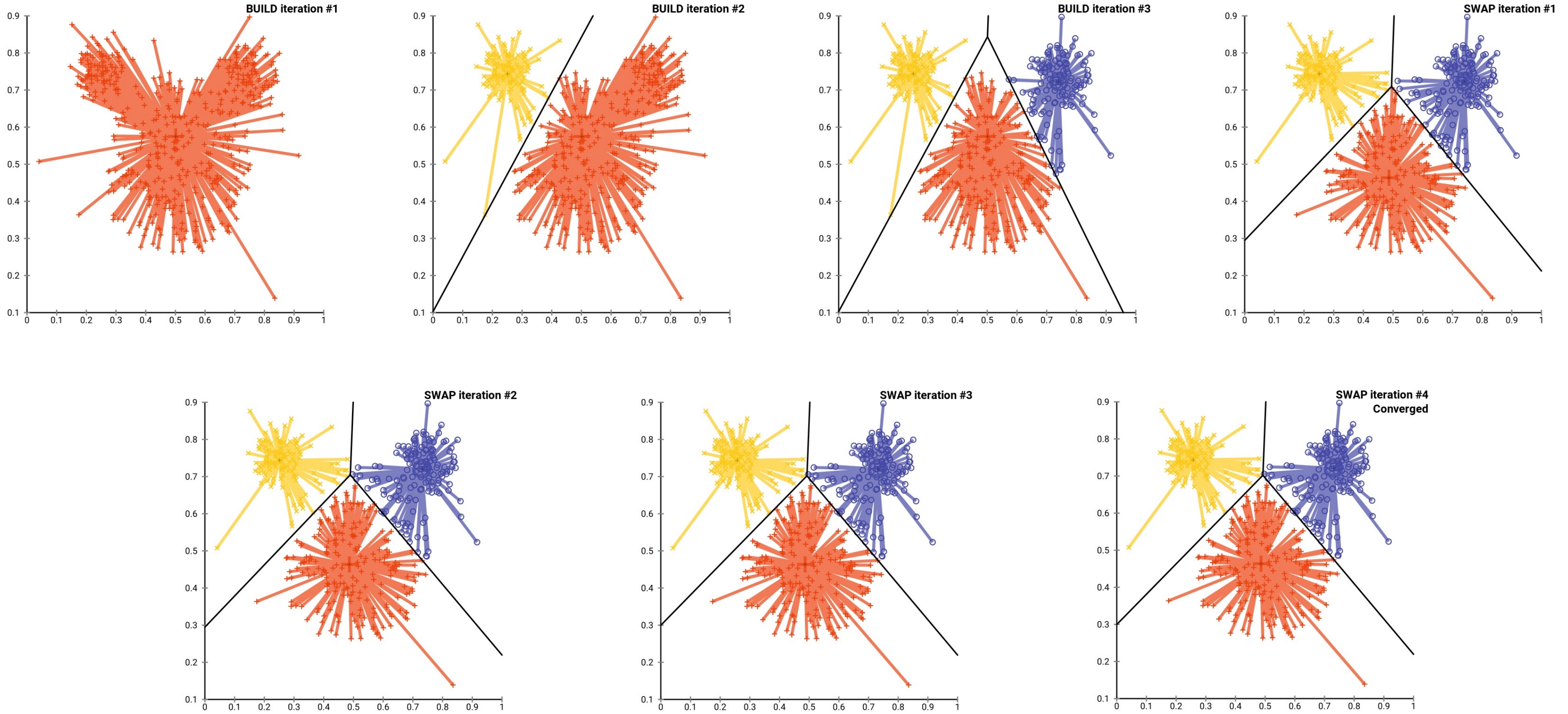
```
For each medoid m:
        For each non-medoid o:

            - Consider swapping the points, calculate the cost (summed
              distances between new m and all other points)
            - If the cost change is the best so far, store the pair (m,o)

- Make the best swap that decreases the cost function, otherwise STOP.
```



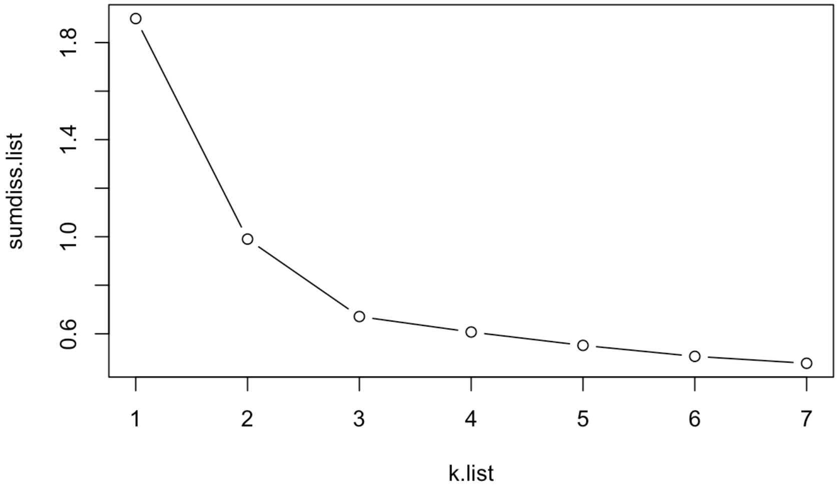
mage credit: Chire; CC BY-SA 4.0 (unchanged)


BUILD iteration #1
BUILD iteration #2
BUILD iteration #3
SWAP iteration #1
SWAP iteration #2
SWAP iteration #3
SWAP iteration #4
Converged
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Evaluating PAM models (elbow method)

Objective (Cost) Function: sum of distances between all points and their closest medoid.

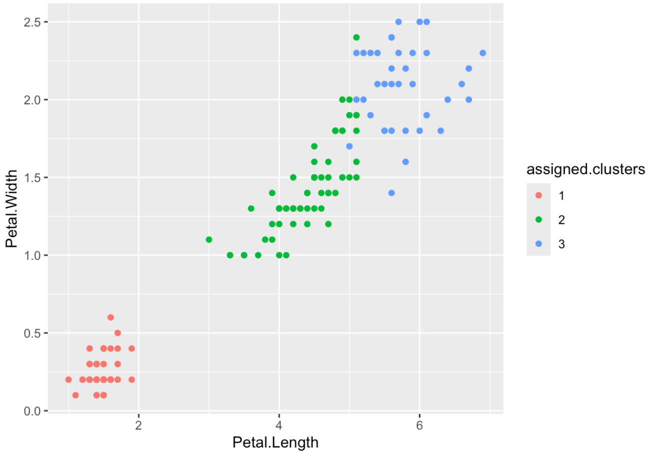
$$\sum_{i=1}^{K} \sum_{x \in C_i} d(x, m_i)$$

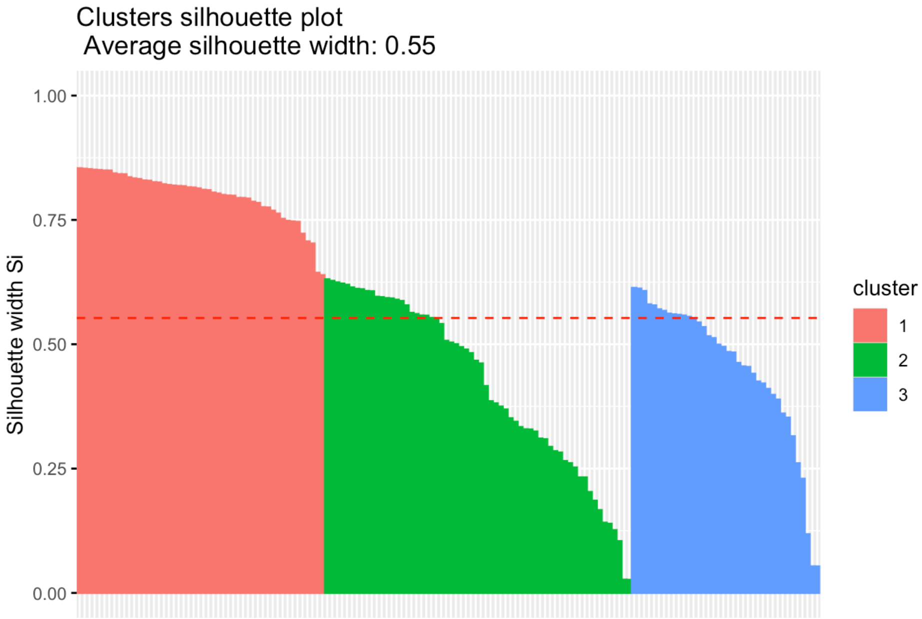


Plot of PAM cost function with values of *k*

Evaluating k-medoids models (silhouette method)

Silhouette value: a measure of similarity between a point and its cluster





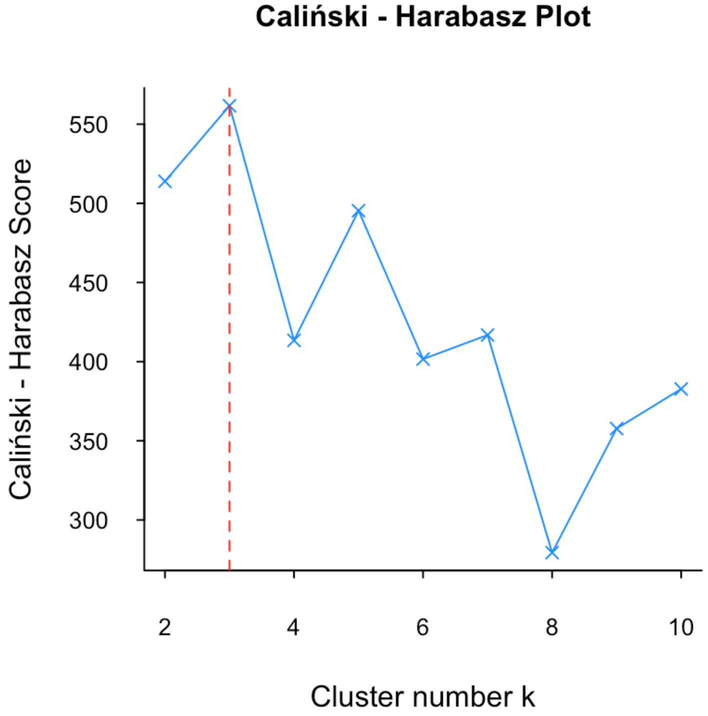
https://en.wikipedia.org/wiki/Silhouette_(clustering)

Calinski–Harabasz index (CHI)

$$CH = \frac{BCSS/(k-1)}{WCSS/(n-k)}$$

BCSS (Between-Cluster Sum of Squares) is the weighted sum of squared Euclidean distances between each cluster centroid (mean) and the overall data centroid (mean)

$$BCSS = \sum_{i=1}^{k} n_i ||\mathbf{c}_i - \mathbf{c}||^2$$



Davies — Bouldin Index (DBI)

- Indicates increased separation between clusters and decreased variation within clusters
- Lower index value -> better clustering

DBI = (1/k) Σ max(R_{ij}) for $i \neq j$

- where:

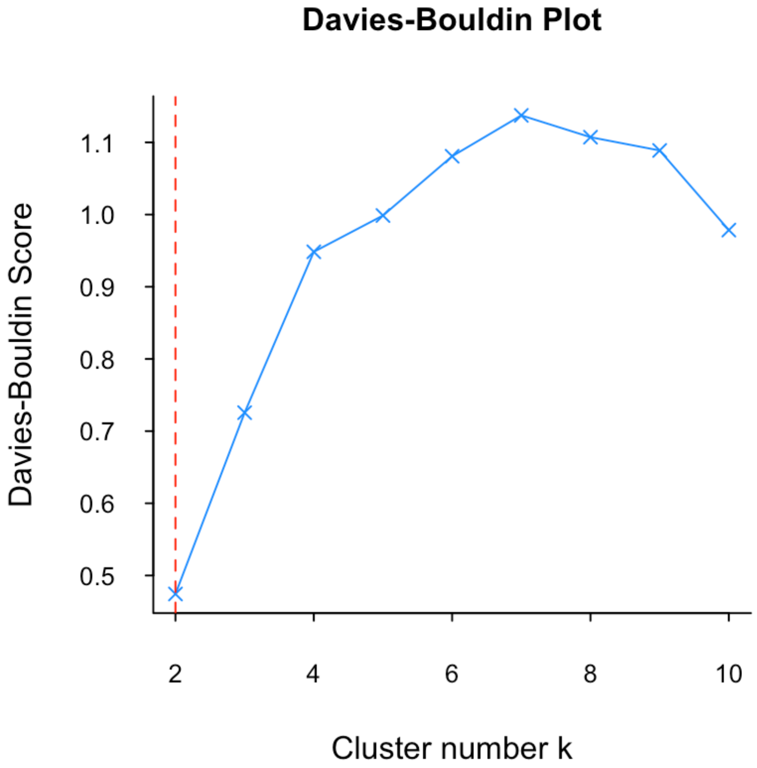
k = number of clusters
R_{ij} = similarity ratio between cluster *i* and cluster *j*

$$R_{ij} = (S_i + S_j) / M_{ij}$$

- where:

S_i = average distance of points in cluster *i* to cluster centroid *i* (compactness)
S_j = average distance of points in cluster *j* to cluster centroid *j* (compactness)
M_{ij} = distance between centroids of cluster *i* and cluster *j* (separation)

https://en.wikipedia.org/wiki/Davies%E2%80%93Bouldin_index#Definition



Hierarchical Clustering

- Contrary to partitional clustering, hierarchical clustering creates hierarchies of clusters through a bottom-up (agglomerative) or top-down (divisive) approach.

- Agglomerative clustering starts with each point in its own cluster, then iteratively merges clusters until it ends with a single cluster.

- Divisive clustering starts with all points in a single cluster, then recursively splits into multiple cluster.

- Hierarchical clustering offers flexibility and interpretability.

- Hierarchical clustering produces a dendrogram.

Sadeghi, B. (2025). Clustering in geo-data science: Navigating uncertainty to select the most reliable method. *Ore Geology Reviews*, 106591. https://doi.org/10.1016/j.oregeorev.2025.106591

Hierarchical Clustering

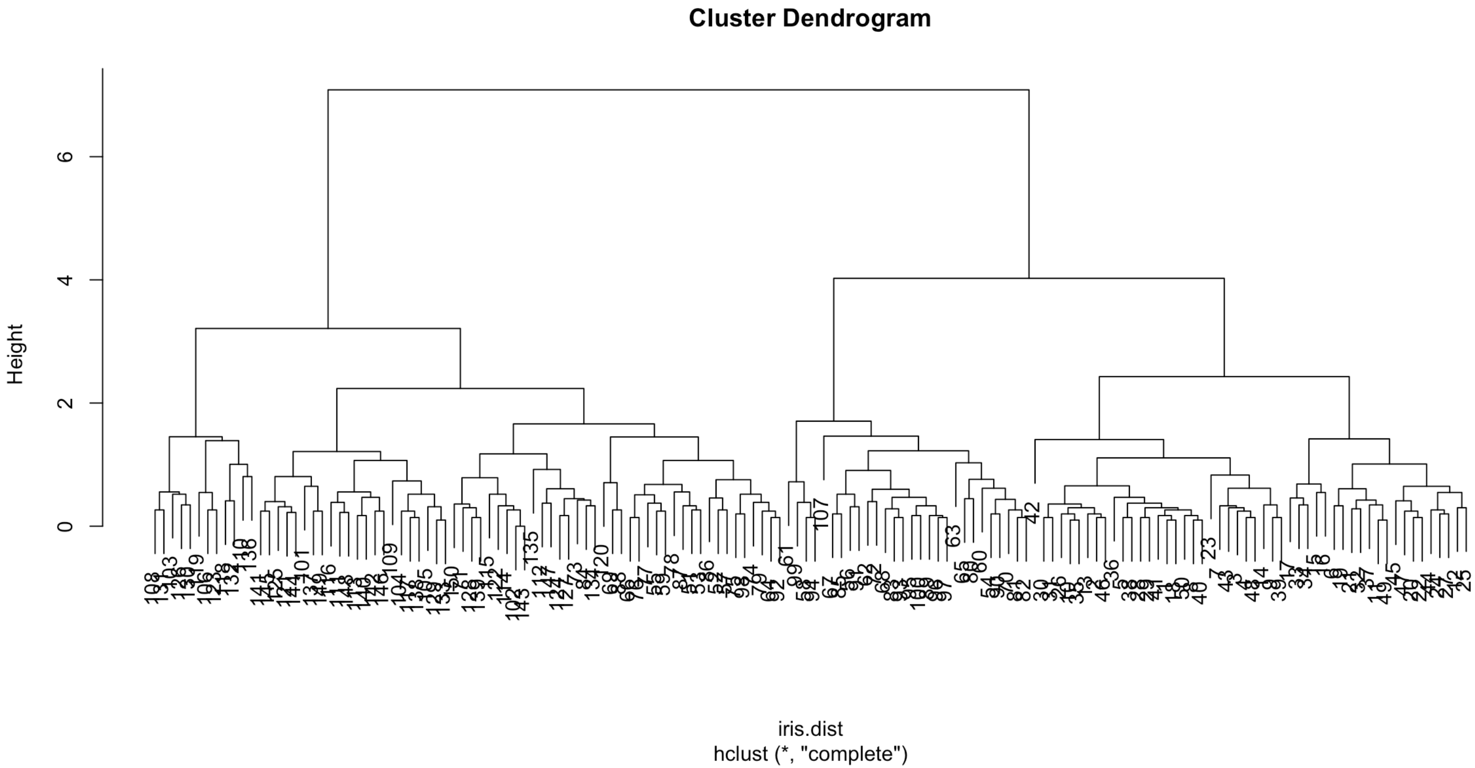
Algorithm (agglomerative clustering)

1. Assign each point to a separate cluster (*n* cluster)
2. Merge the two closest* clusters
3. Repeat (2) Until all points are in a single cluster

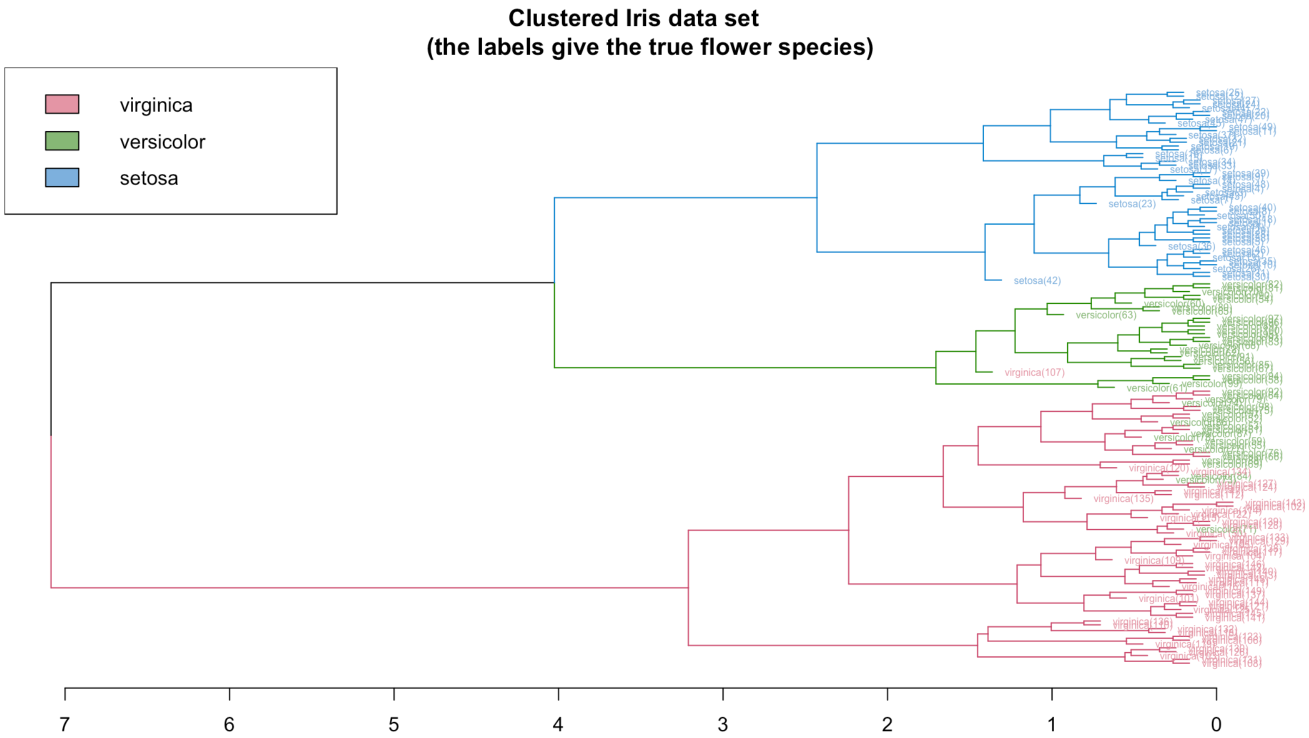
* Distance between clusters:

- Complete linkage: maximum distance between points in two clusters
- Single linkage: minimum distance between points in two clusters
- Average linkage: average distance between points in two clusters

Examining hierarchical clustering (dendrogram)



Examining hierarchical clustering (dendrogram)



https://cran.r-project.org/web/packages/dendextend/vignettes/Cluster_Analysis.html

In class exercise code:
https://rpi.box.com/s/brbvls9dpt0a4gy85z9ywl8toc15raxj

Thanks!