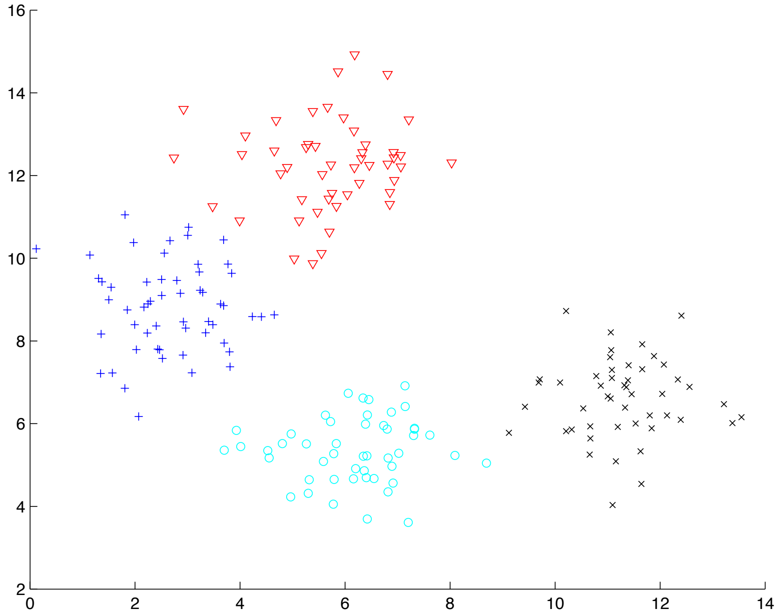
Clustering

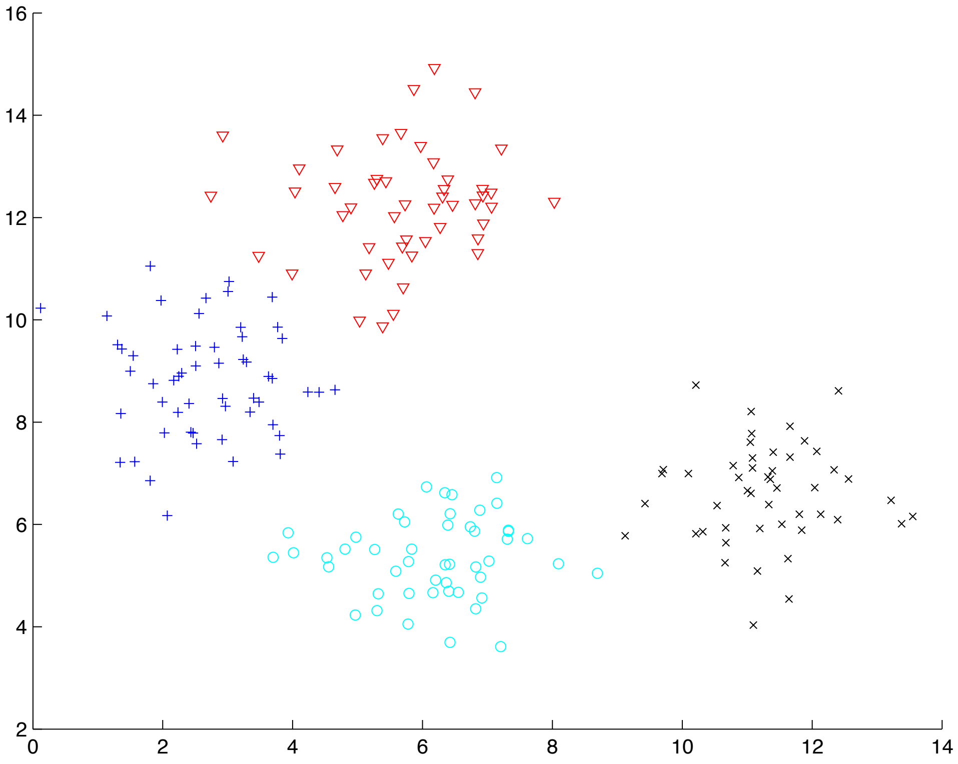
Chapter 10



Clustering

- Clustering is the art of finding groups in data (Kaufman and Rousseeuw, 1990).

- What is a cluster?
  - Group of objects separated from other clusters



Digression: means and medians

The mean is the minimizer of

$$\underset{\mathbf{y}}{\operatorname{argmin}} \sum_{\mathbf{x} \in D} ||\mathbf{x} - \mathbf{y}||^2$$

Using

$$\underset{\mathbf{y}}{\operatorname{argmin}} \sum_{\mathbf{x} \in D} ||\mathbf{x} - \mathbf{y}||$$

Gives rise to the geometric median, which is more robust to outliers.

Issue: no closed form solution.

Means, medians, medoids

It may be useful to restrict exemplars to be one of the given data points.

Medoid: representative object of a data set or a cluster whose average distance to all the objects in the cluster is minimal

How would we compute the medoid for a set of points?

Clustering

A clustering is a partition of the elements in your dataset into K subsets such that the following holds:

Each observation belongs to a cluster:

$$C_1 \cup C_2 \cup \ldots \cup C_K = \{1, \ldots, n\}$$

The clusters are non-overlapping:

$$C_k \cap C_{k'} = \emptyset \text{ for all } k \neq k'$$

A plausible objective

We would like a clustering to minimize the within-cluster variation.

Let's assume it is measured via a function W(C_k). So the overall objective is:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

A possible definition of W(C_k):

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,j \in C_k} ||\mathbf{x}_i - \mathbf{x}_j||^2$$

A plausible objective

We would like a clustering to minimize the within-cluster variation.

Let's assume it is measured via a function W(C$_k$). So the overall objective is:

$$\underset{C_1,\ldots,C_K}{\text{minimize}} \left\{ \sum_{k=1}^{K} W(C_k) \right\}$$

A possible definition of W(C$_k$):

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,j \in C_k} ||\mathbf{x}_i - \mathbf{x}_j||^2$$

$\boldsymbol{\mu}_k$ the centroid for cluster k

$$= 2 \sum_{i \in C_k} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

The k-means objective function

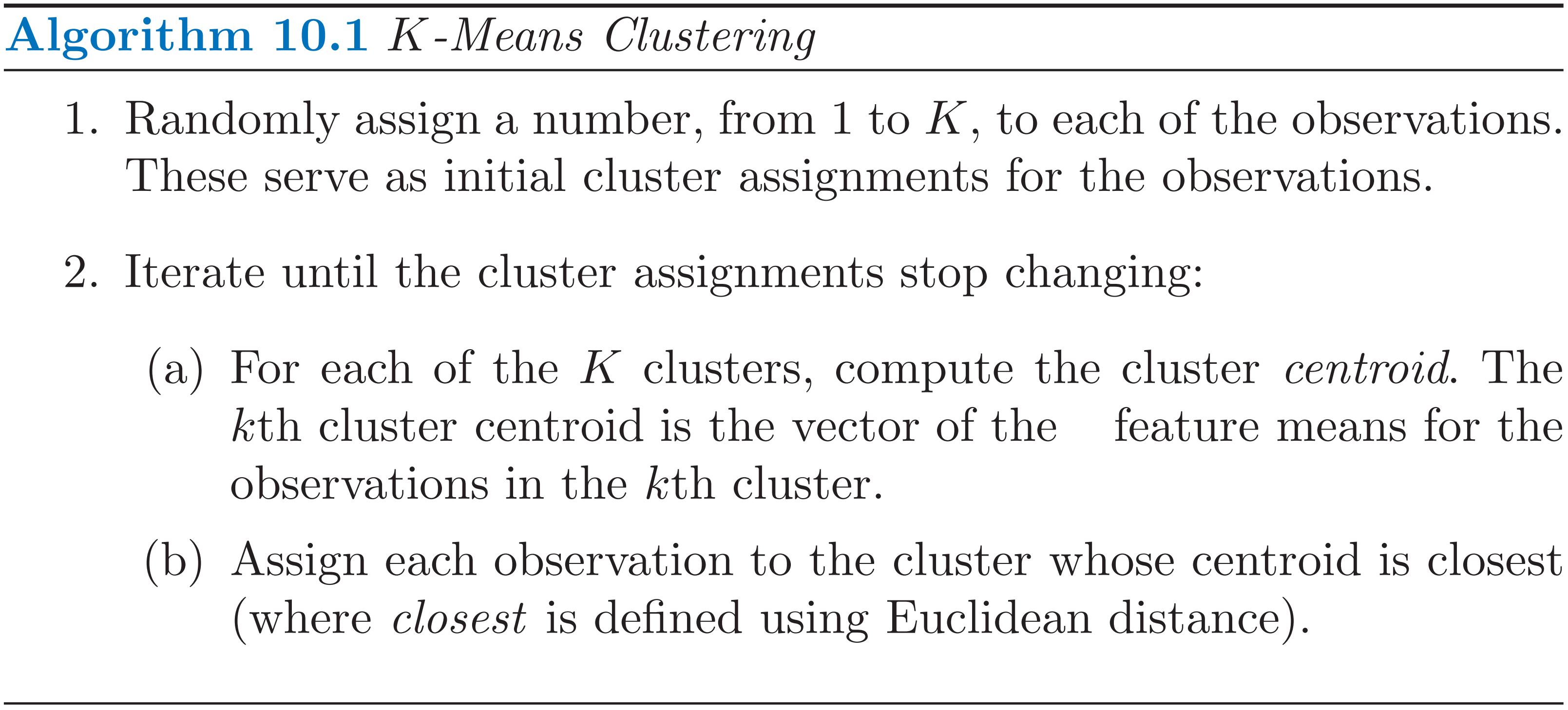
Putting it all together:

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} \sum_{i \in C_k} ||\mathbf{x}_i - \boldsymbol{\mu}_k||^2$$

A problem with this problem: NP-complete.

The k-means algorithm

A heuristic algorithm for solving the problem:



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 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).

Example

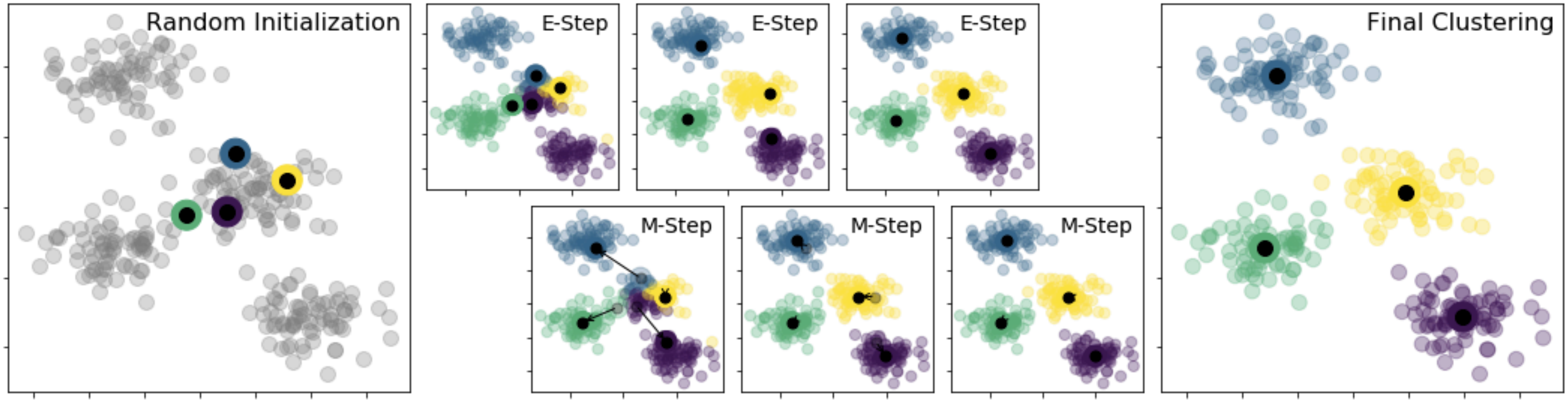
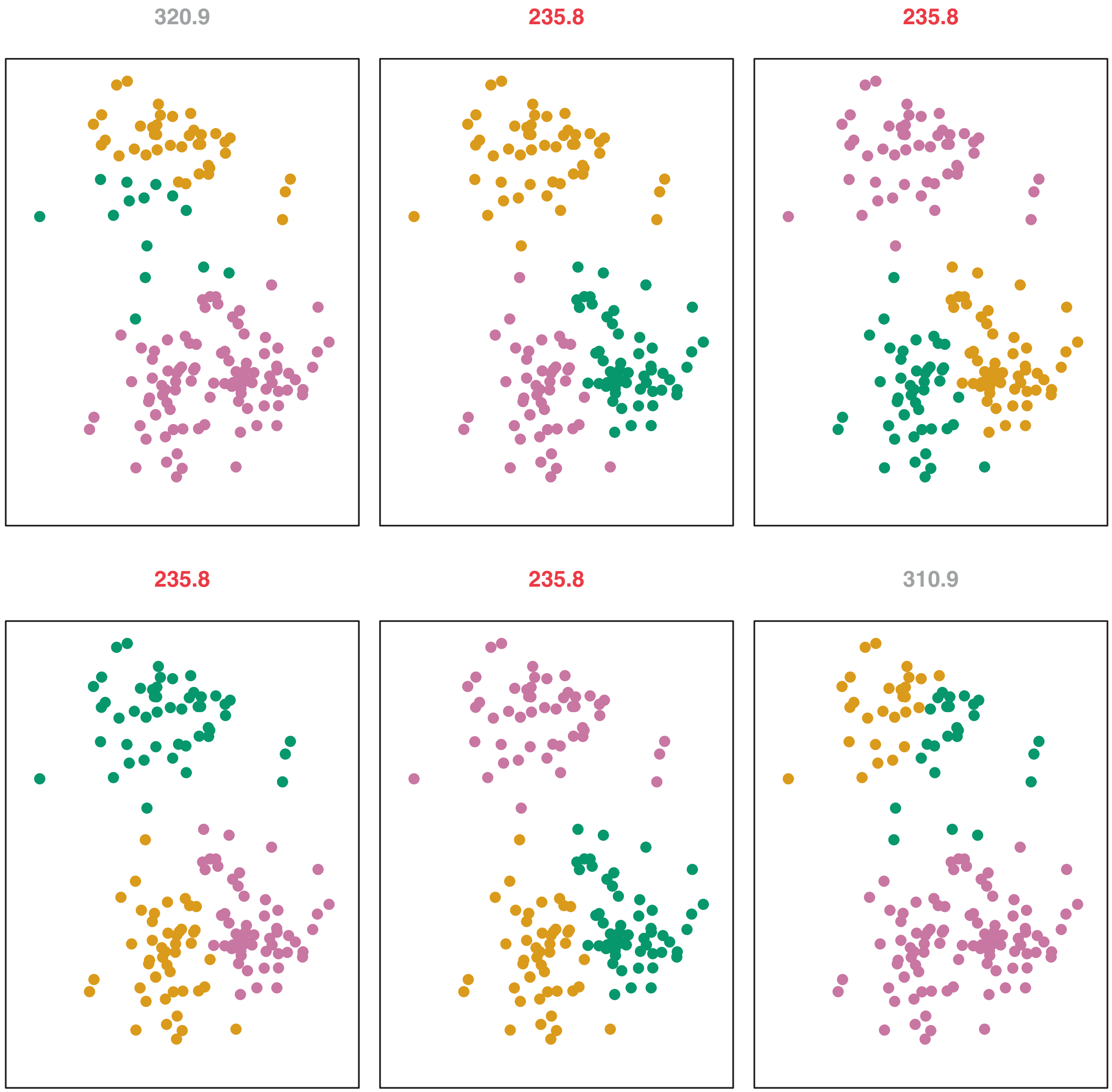


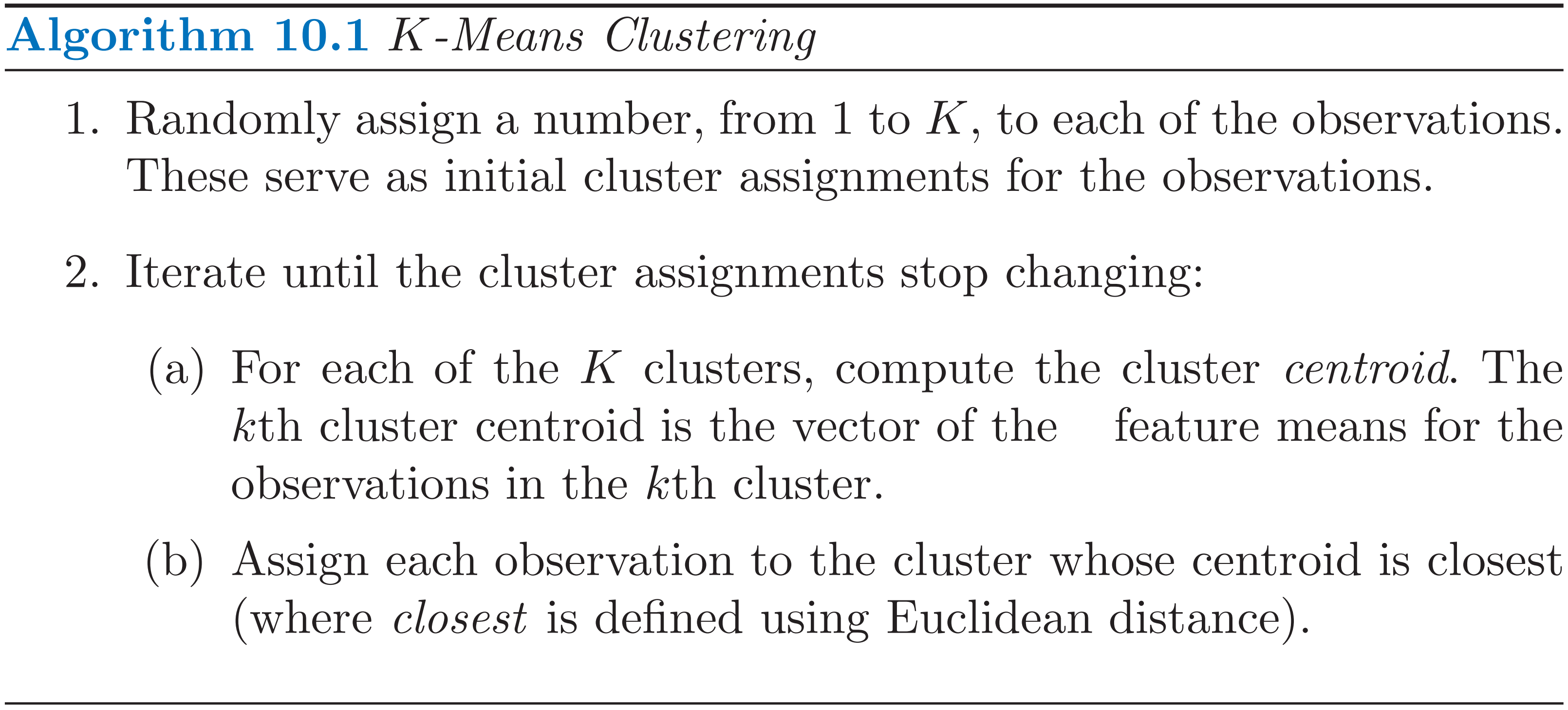
figure generated using code from
https://jakevdp.github.io/PythonDataScienceHandbook/06.00-figure-code.html#K-Means

Local minima



Running time?

What is the running time per iteration?

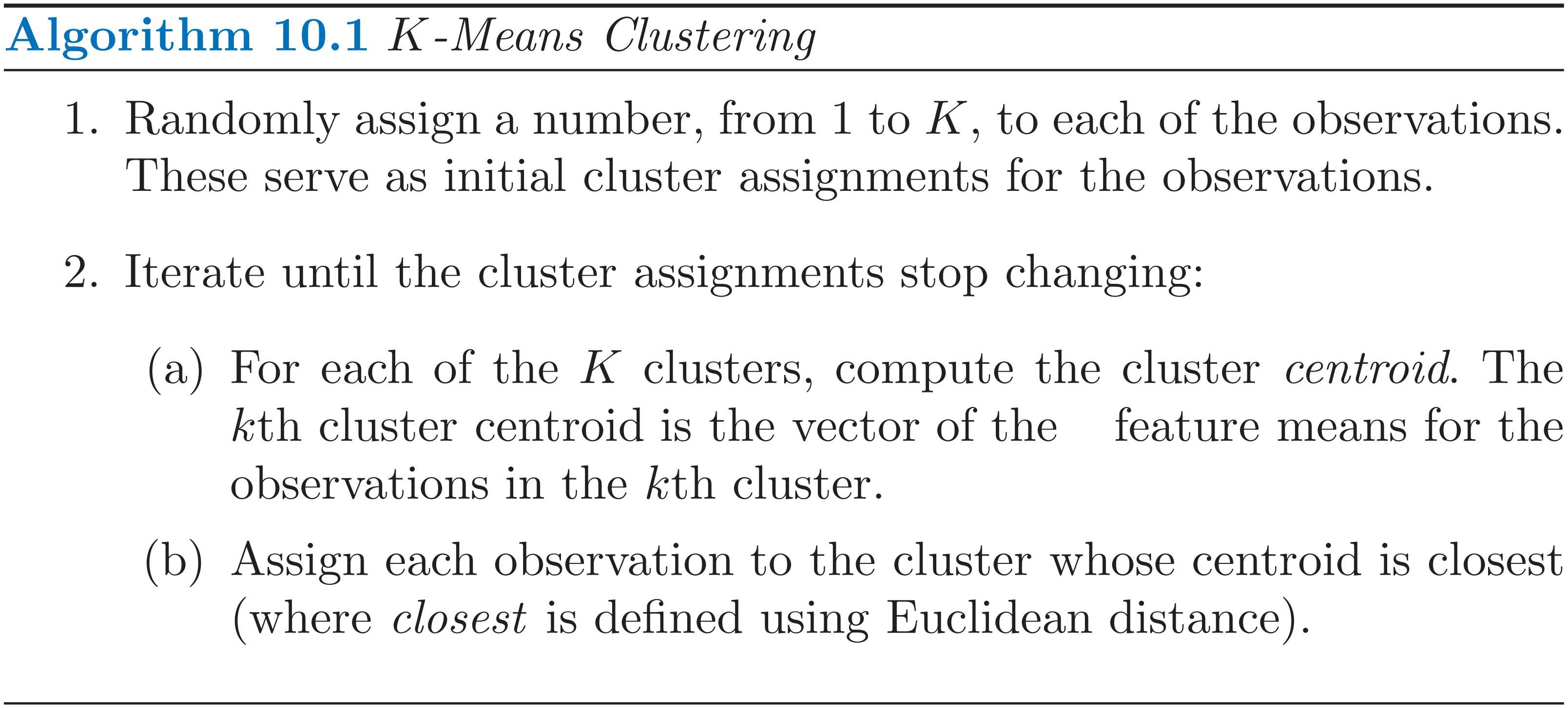


Algorithm 10.1 *K-Means Clustering*

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 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).

Running time?

What is the running time per iteration?

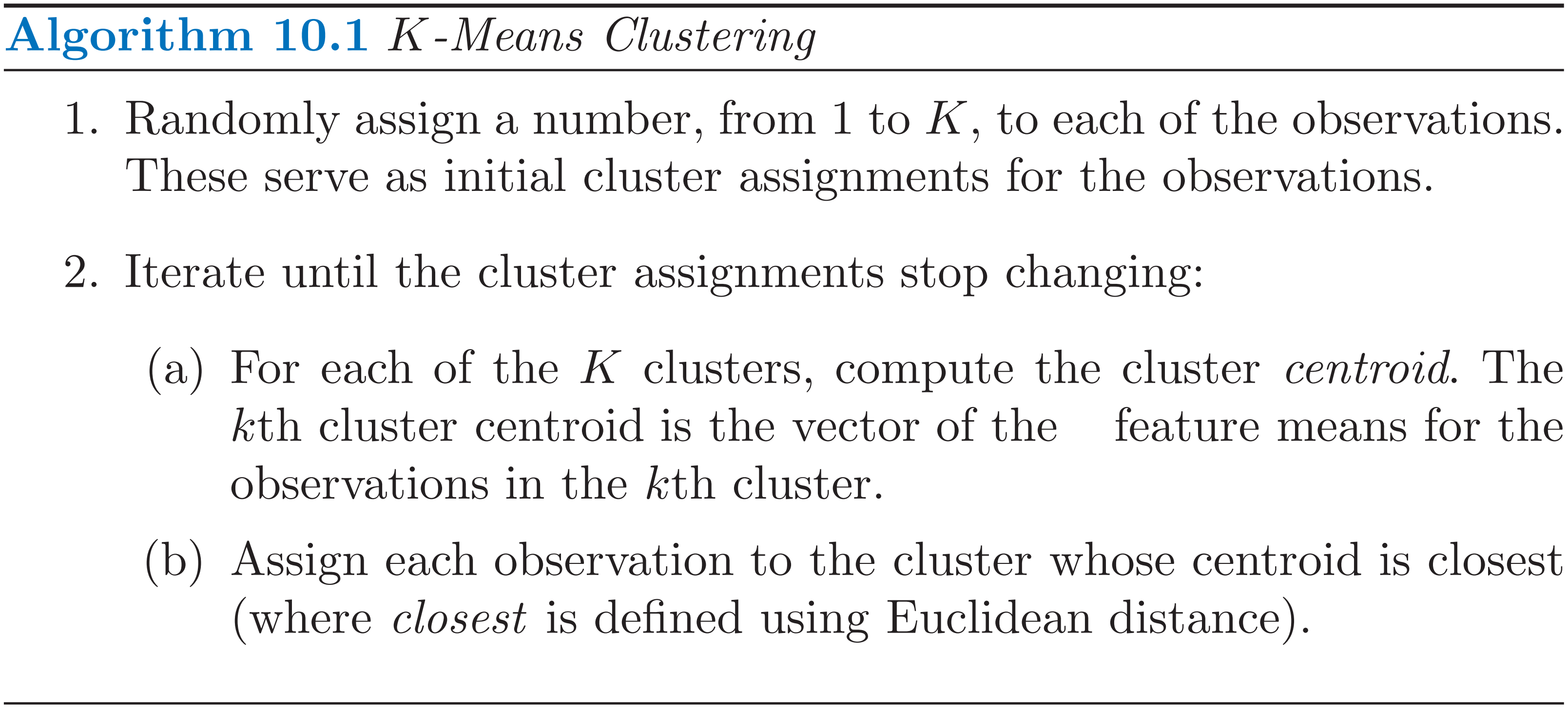


Algorithm 10.1 *K-Means Clustering*

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 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).

Typically, converges very quickly (and in fact, guaranteed to converge in a finite number of iterations)

Running time?



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 $k$th cluster centroid is the vector of the   feature means for the observations in the $k$th cluster.
    (b) Assign each observation to the cluster whose centroid is closest (where *closest* is defined using Euclidean distance).
```

Can easily be kernelized.

Dealing with local minima

Run the algorithm multiple times with different initializations.

Initialization

A good initialization can lead to faster convergence to a better optimal solution.

The standard choice: k random data points

More sophisticated approaches:

- Create a collection of subsamples of the data. Cluster the resulting cluster centers using k-means and use for initialization.

  P.S. Bradley, and Usama M. Fayyad. Refining Initial Points for K-Means Clustering. Proceedings of the Fifteenth International Conference on Machine Learning ICML '98

- k-means++: Choose the first center randomly; subsequent centers are chosen with probability proportional to their distance to the closest center. The default in scikit-learn

  Arthur, D. and Vassilvitskii, S. (2007). "k-means++: the advantages of careful seeding". Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms.

Related algorithms

Related algorithms:

K-medoids

Partitioning around medoids (PAM)

Assumptions behind the model

K-means assumes spherical clusters.

There are probabilistic extensions that address this to some extent.

Probably the most widely used clustering algorithm because of its simplicity, speed, and easy implementation

k-means demo

How many clusters in my data?

Choosing a "good" clustering is more challenging than in supervised learning because we don't have ground truth.

Most methods quantify some geometrical aspect of the clustering.

How many clusters in my data?

Let's try to quantify the quality of a clustering solution.

The Silhouette coefficient is defined for each example as:

$$s(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(b(\mathbf{x}), a(\mathbf{x}))}$$

a: the mean distance between the example and all other points in the same cluster.

b: the mean distance between the example and all other points in the next nearest cluster.

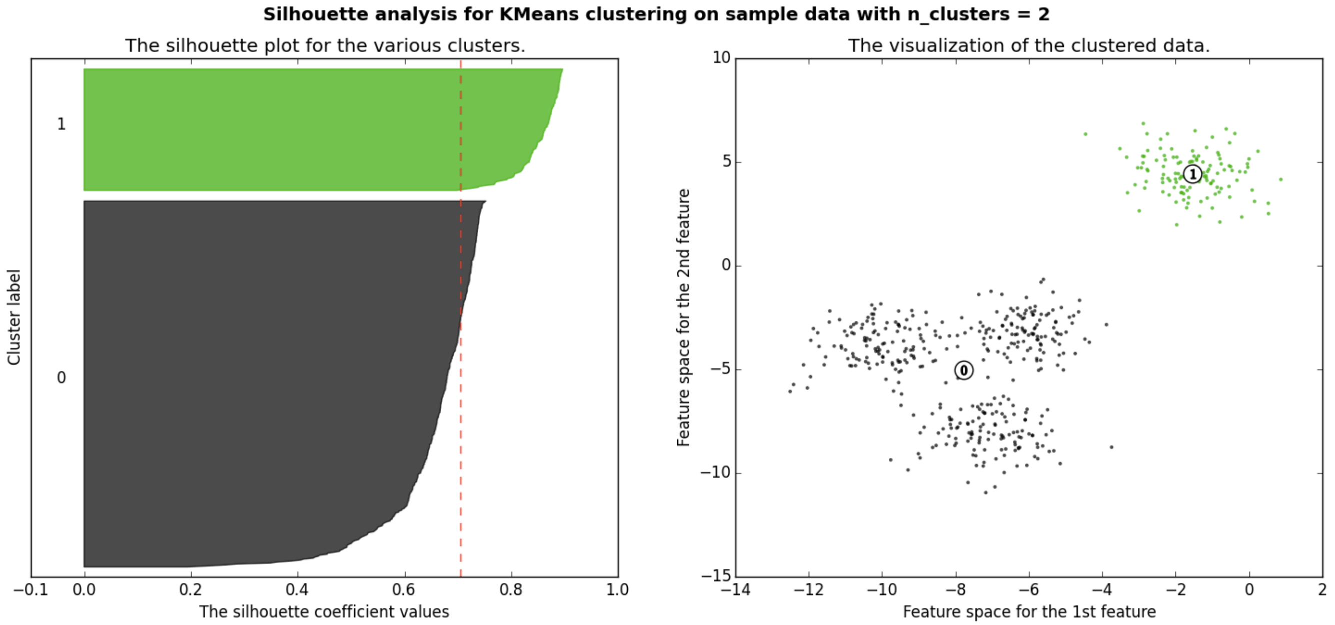
Peter J. Rousseeuw (1987). "Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis". Computational and Applied Mathematics 20: 53–65.

Silhouettes

The Silhouette coefficient is defined for each example

$$s(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(b(\mathbf{x}), a(\mathbf{x}))}$$

Sort s(x) and group by cluster:



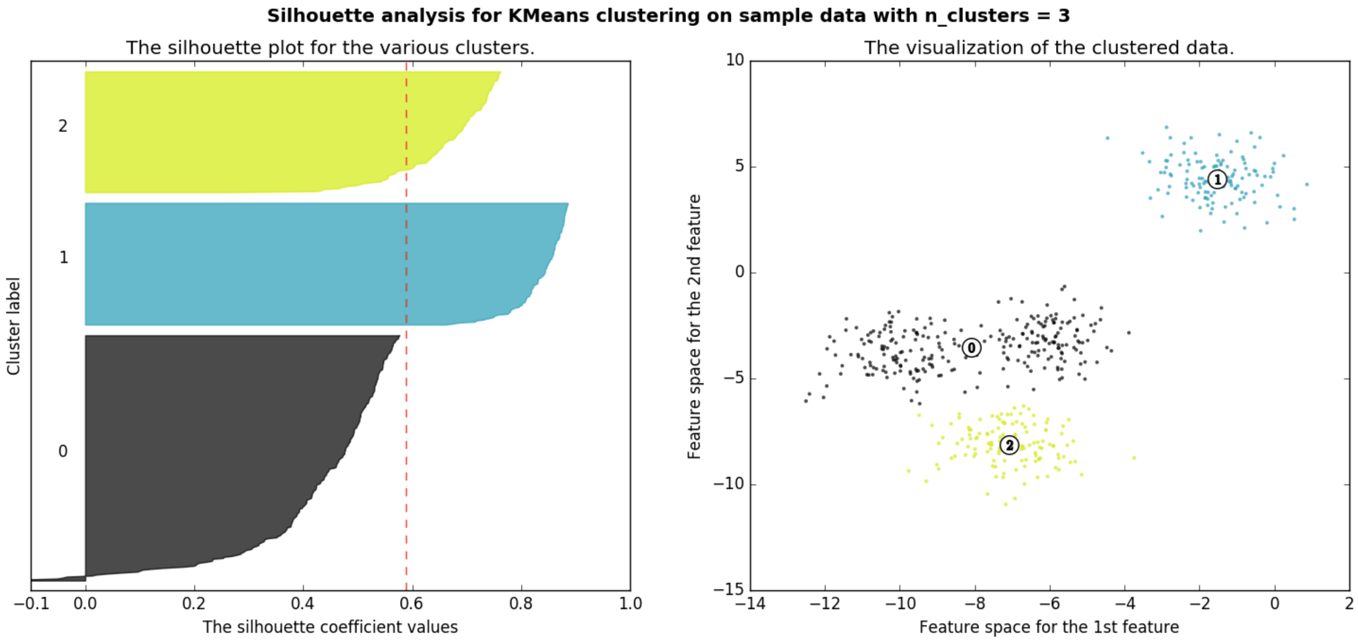
Figures generated using the scikit-learn silhoutte method
https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html

Silhouettes

The Silhouette coefficient is defined for each example

$$s(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(b(\mathbf{x}), a(\mathbf{x}))}$$

Sort s(x) and group by cluster:



Silhouette analysis for KMeans clustering on sample data with n_clusters = 3

The silhouette plot for the various clusters.

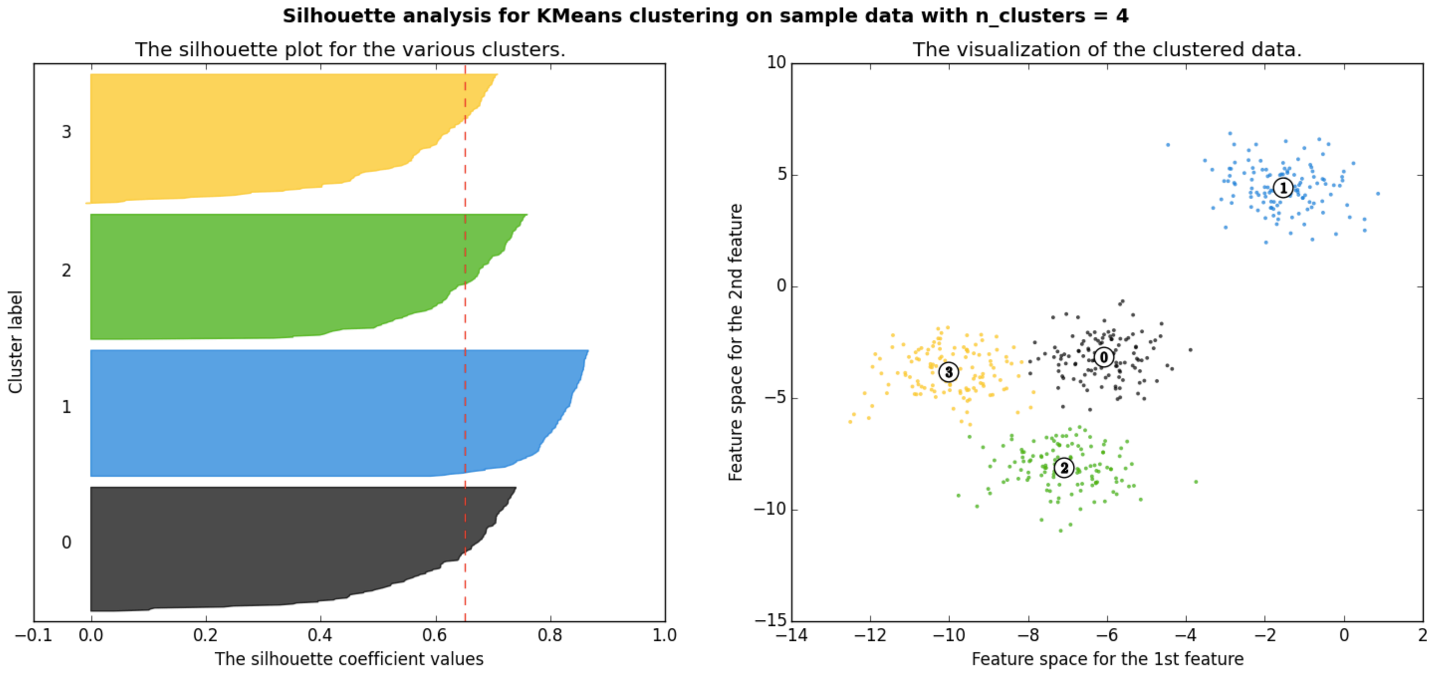
The visualization of the clustered data.

Silhouettes

The Silhouette coefficient is defined for each example

$$s(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(b(\mathbf{x}), a(\mathbf{x}))}$$

Sort s(x) and group by cluster:

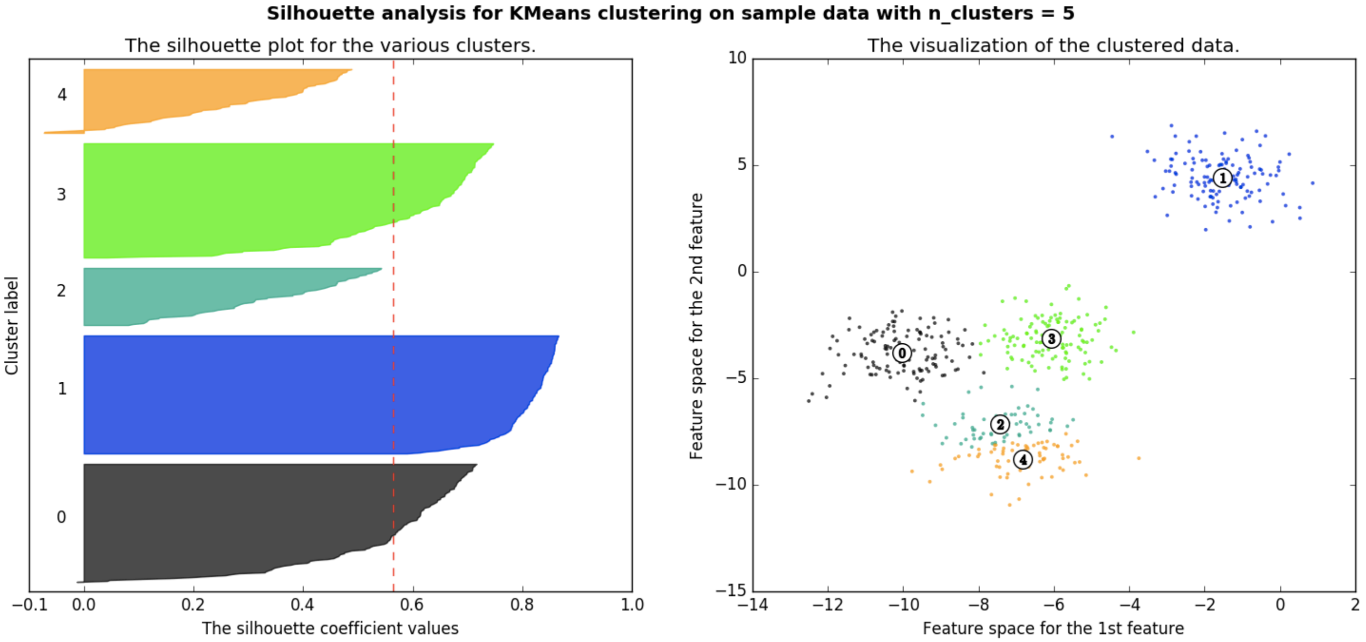


Silhouettes

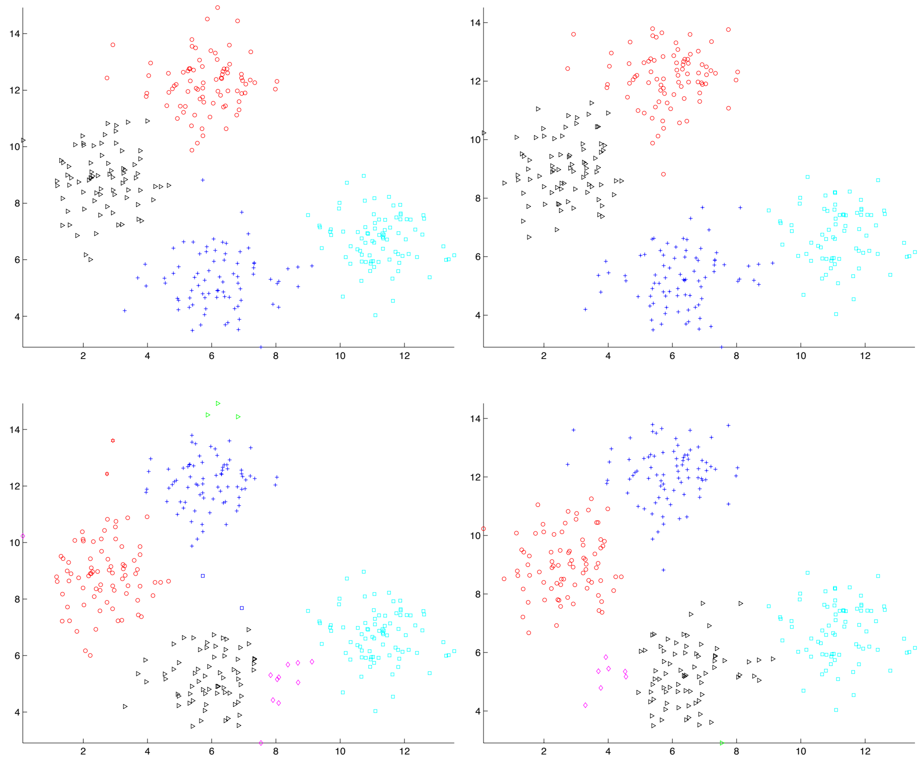
The Silhouette coefficient is defined for each example

$$s(\mathbf{x}) = \frac{b(\mathbf{x}) - a(\mathbf{x})}{\max(b(\mathbf{x}), a(\mathbf{x}))}$$

Sort s(x) and group by cluster:



alternative: stability under sampling



Premise: a clustering algorithm has captured some of the structure in a dataset if clustering solutions over different subsamples are similar.

resources

Sources:

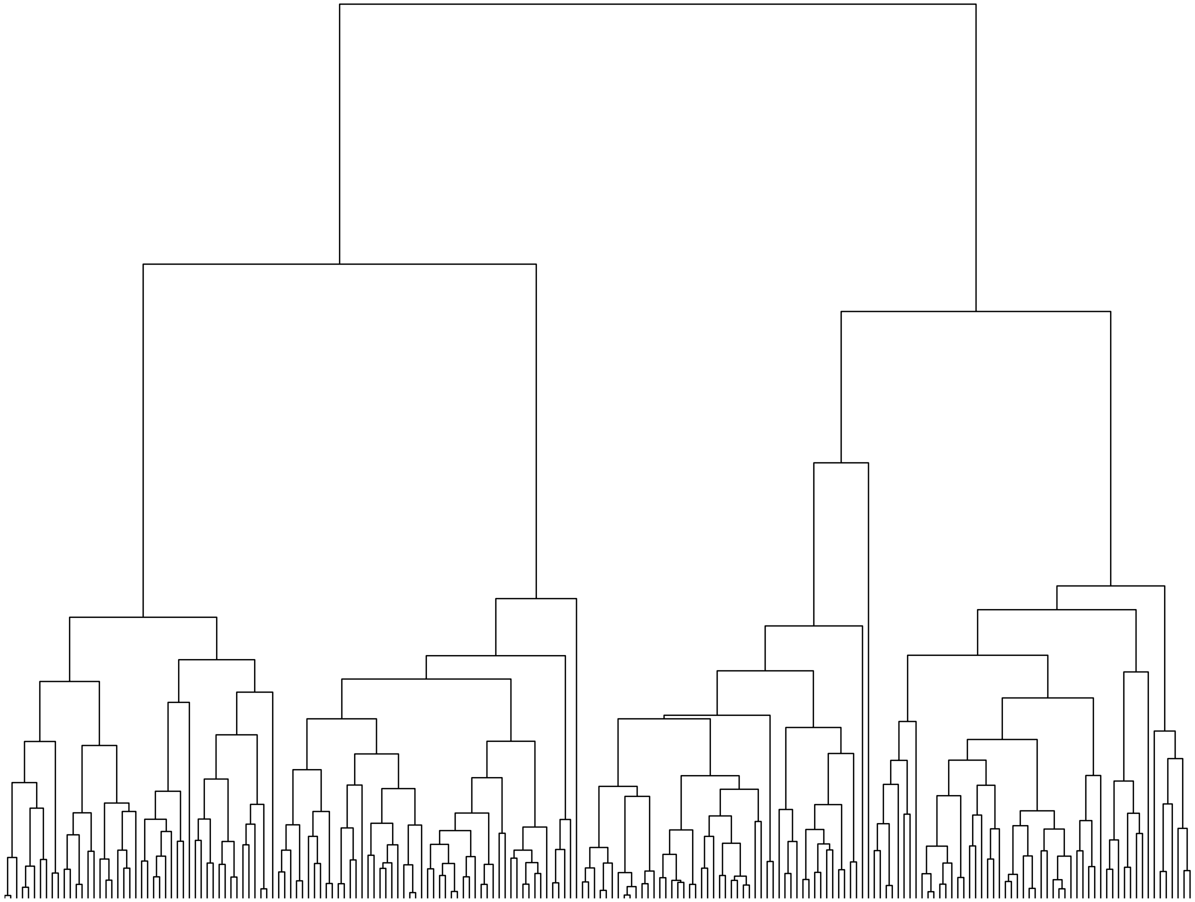
- A. Ben-Hur, A. Elisseeff and I. Guyon. A stability based method for discovering structure in clustered data. Pacific Symposium on Biocomputing, 2002.
- A. Ben-Hur and I. Guyon. Detecting stable clusters using principal component analysis. In Methods in Molecular Biology, M.J. Brownstein and A. Khodursky (eds.) Humana press, 2003 pp. 159-182.

Code:

https://bioconductor.org/packages/release/bioc/html/clusterStab.html

Dendrograms

Definition: Given a dataset D, a dendrogram is a binary tree with the elements of D at its leaves. An internal node of the tree represents the subset of elements in the leaves of the subtree rooted at that node.



Hierarchical/agglomerative clustering

Algorithm outline:

- Start with each data point in a separate cluster
- At each step merge the closest pair of clusters

Hierarchical clustering

Algorithm outline:

- Start with each data point in a separate cluster
- At each step merge the closest pair of clusters

Need to define a measure of distance between clusters:

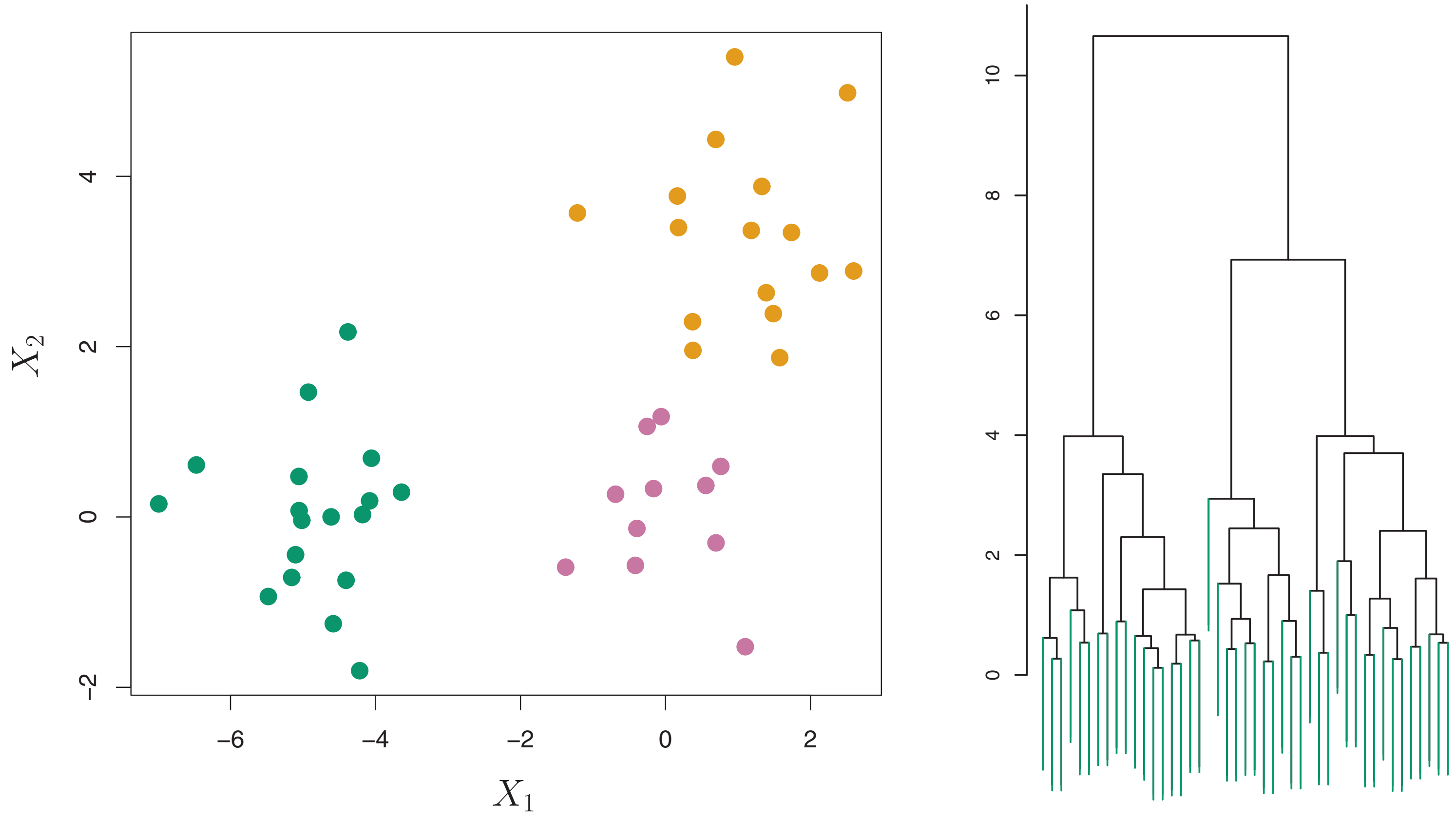
A *linkage function* $L : 2^{\mathscr{X}} \times 2^{\mathscr{X}} \to \mathbb{R}$ calculates the distance between arbitrary subsets of the instance space, given a distance metric $\mathrm{Dis} : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$.

Linkage functions

- Single linkage
  - Smallest pairwise distance between elements from each cluster
- Complete linkage
  - Largest distance between elements from each cluster
- Average linkage
  - The average distance between elements from each cluster
- Centroid linkage
  - Distance between cluster means
- Ward's method
  - Find the pair of clusters that leads to minimum increase in total within-cluster variance after merging.

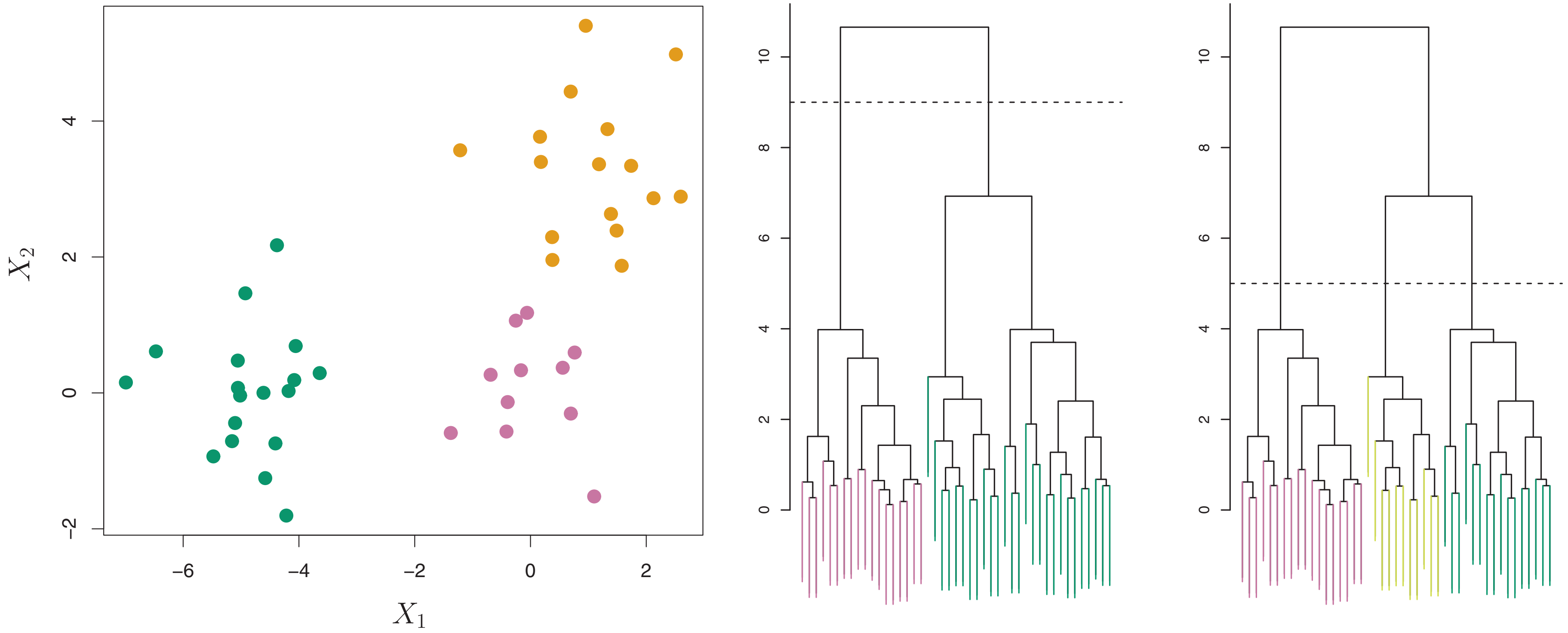
Dendrograms revisited

Interpretation of the vertical dimension: The distance between the clusters when they were merged (the level associated with the cluster). The leaves have level 0.



Dendrograms revisited

Interpretation of the vertical dimension: The distance between the clusters when they were merged (the level associated with the cluster). The leaves have level 0.



Hierarchical clustering

Algorithm 10.2 *Hierarchical Clustering*

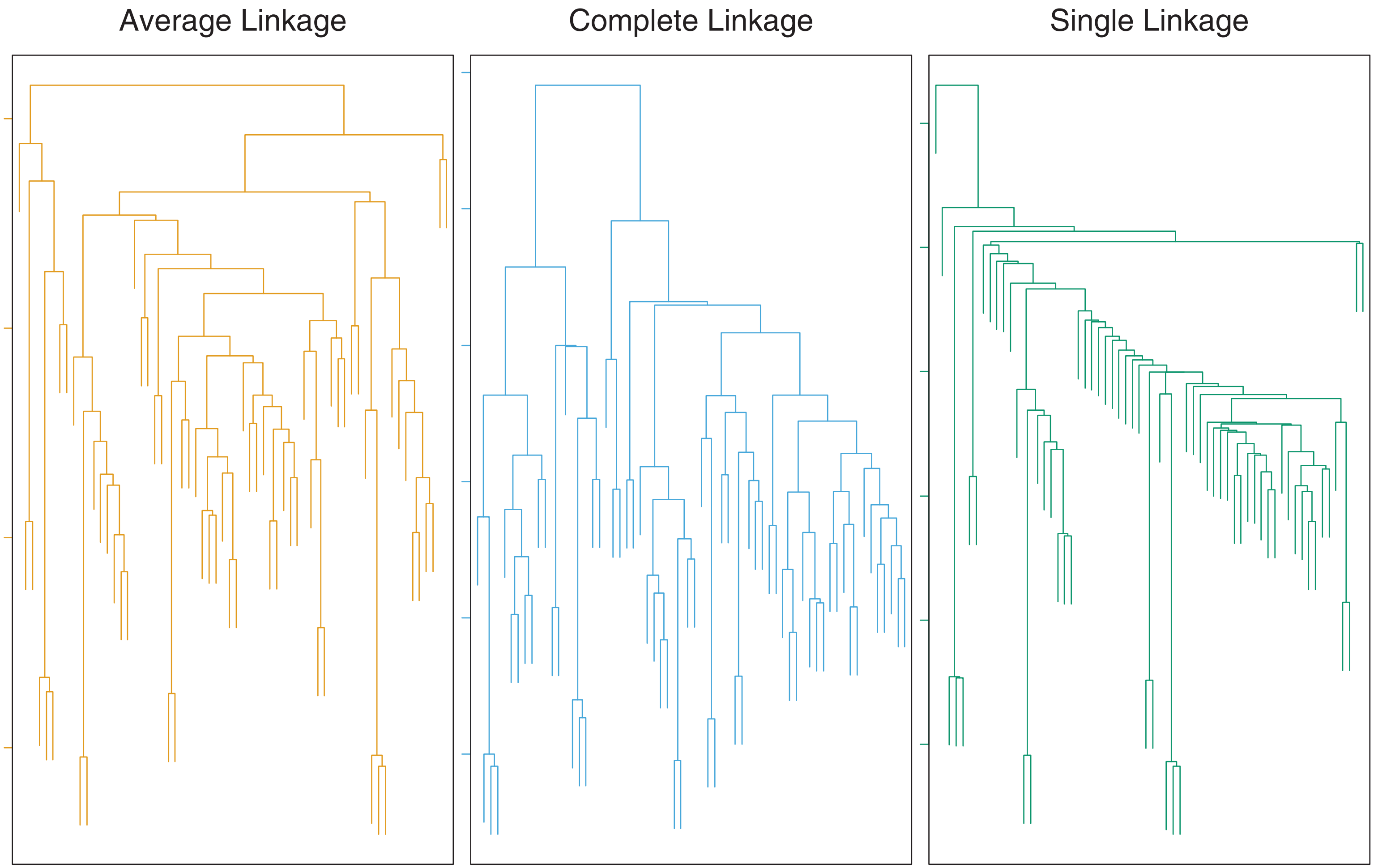
1. Begin with n observations and a measure (such as Euclidean distance) of all the $\binom{n}{2} = n(n-1)/2$ pairwise dissimilarities. Treat each observation as its own cluster.

2. For $i = n, n-1, \ldots, 2$:

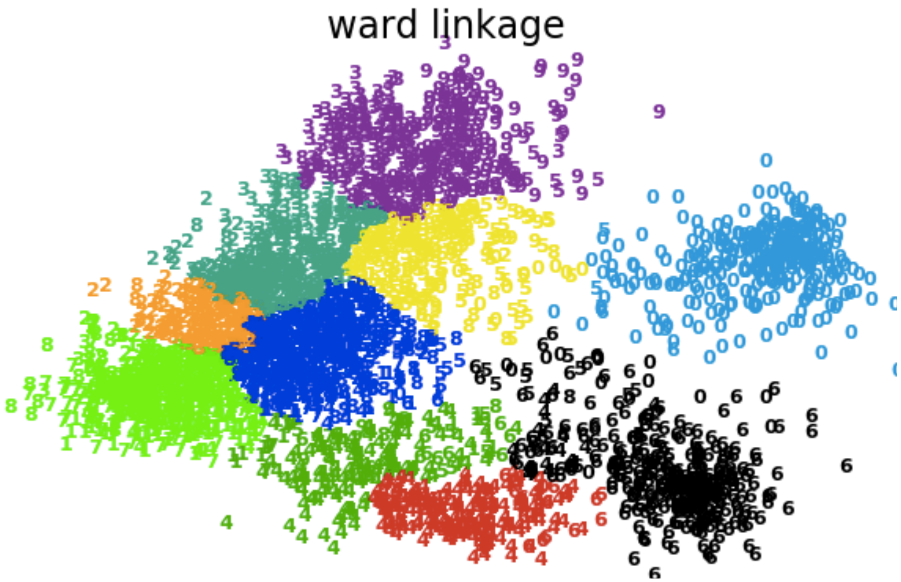
   (a) Examine all pairwise inter-cluster dissimilarities among the i clusters and identify the pair of clusters that are least dissimilar (that is, most similar). Fuse these two clusters. The dissimilarity between these two clusters indicates the height in the dendrogram at which the fusion should be placed.

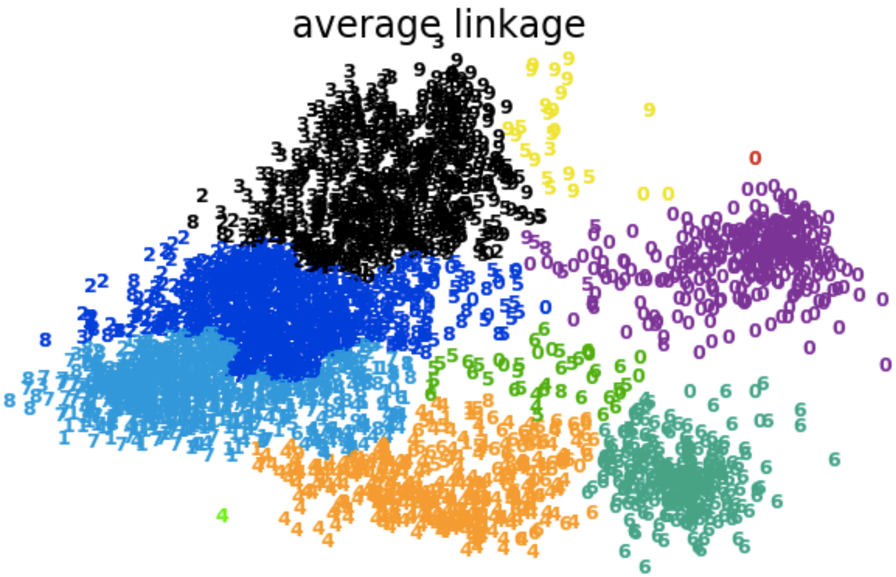
   (b) Compute the new pairwise inter-cluster dissimilarities among the $i-1$ remaining clusters.

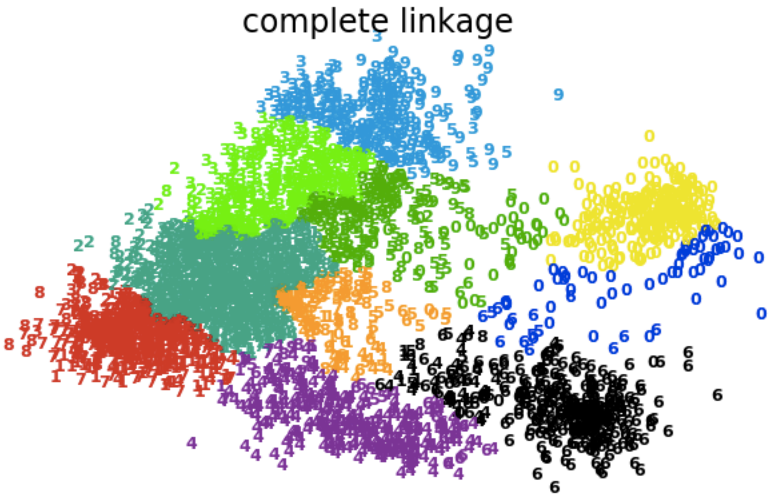
Linkage matters



Linkage matters


ward linkage


average linkage

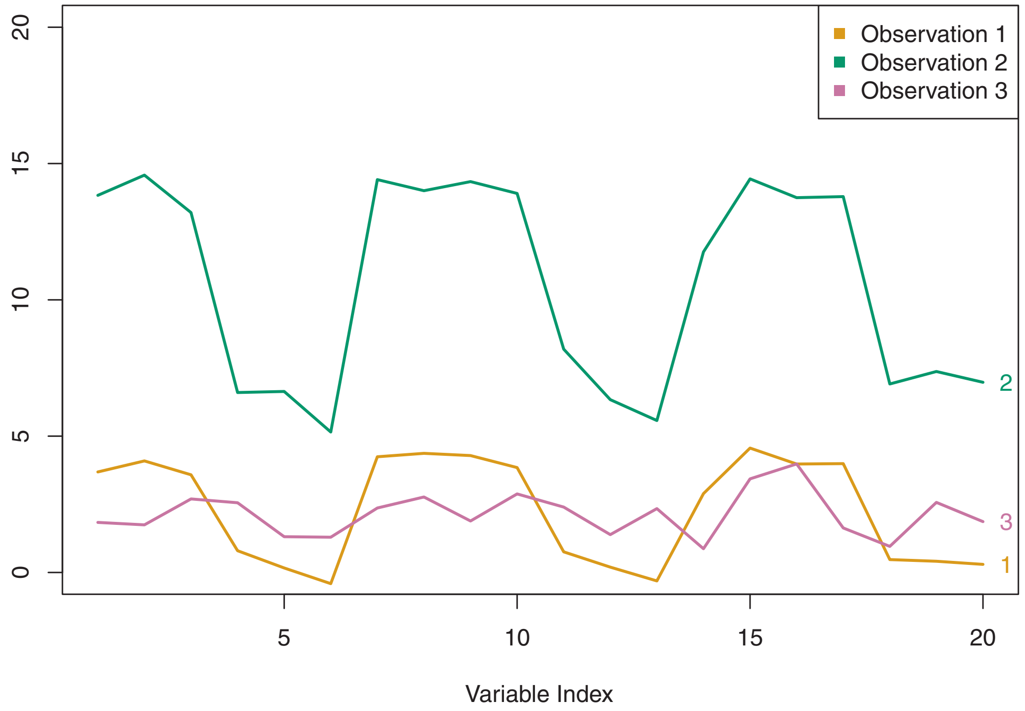

complete linkage

http://scikit-learn.org/stable/auto_examples/cluster/plot_digits_linkage.html#sphx-glr-auto-examples-cluster-plot-digits-linkage-py

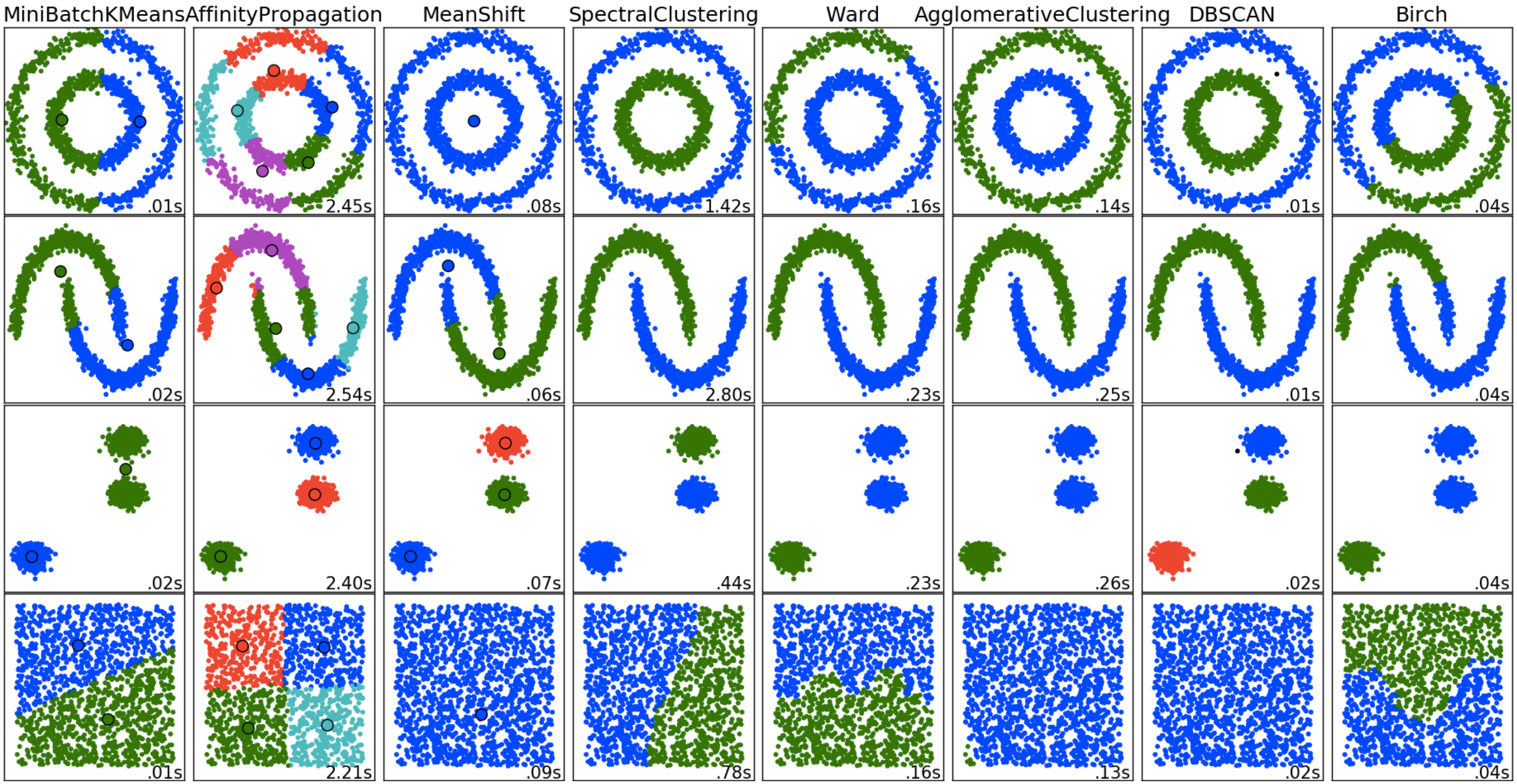
distance vs similarity

Hierarchical clustering can be performed with respect to a distance measure or a similarity measure.

Correlation is often a better choice:



Lots of clustering algorithms out there!



http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#comparing-different-clustering-algorithms-on-toy-datasets

Summary

Clustering depends on the choice of similarity/distance and preprocessing.

Different methods will give different results.

Clustering algorithms will find as many clusters as you ask for: need methods for deciding the number of clusters.

Clustering is sensitive to noise.

Hard choices to make – there is no teaching signal as we had in supervised learning.