Data Analysis

Clustering algorithms

National Research University Higher School of Economics
Master's Program "Big Data Systems"

Fall 2019

Clustering

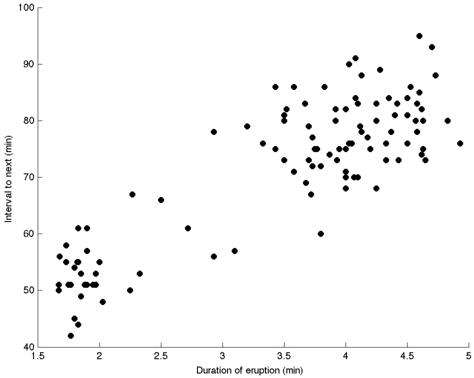
- Clustering refers to a very broad set of techniques for finding subgroups, or clusters, in a data set, i.e., divide the data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ into groups
- We seek for a partition of the data into distinct groups so that the observations within each group are quite similar to each other.
- Contrast to classification (discrimination)
    - Classification means predefined classes (supervised)
    - Clustering is about determination of unknown classes (unsupervised)
- "Unsupervised learning": data segmentation, class discovery, examples include:
    - Marketers use demographics and consumer profiles to segment the marketplace into small, homogeneous groups
    - Physicians use medical records to cluster patients for personalized treatment

Clustering for Market Segmentation

- Suppose we have access to a large number of measurements (e.g. median household income, occupation, distance from nearest urban area, and so forth) for a large number of people.
- Our goal is to perform market segmentation by identifying subgroups of people who might be more receptive to a particular form of advertising, or more likely to purchase a particular product.
- The task of performing market segmentation is to uncover clusters the people in the data set.

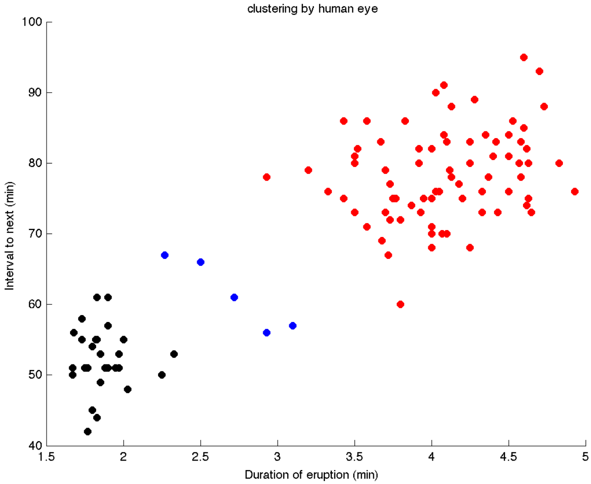
Example: Old Faithful Geyser

- 107 bivariate observation for duration of eruption,X_1), and the waiting time until the next eruption, X_2.
- Can this dataset be divided into two or three sub-groups?



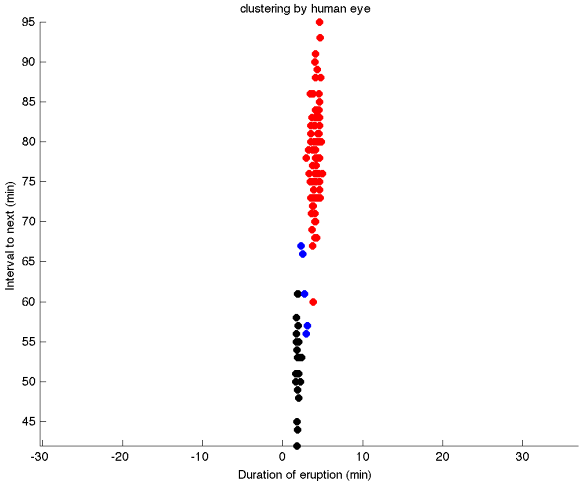
Example: Old Faithful Geyser (2)

- Human perception is excellent??
- Depends on many things, e.g., on scaling of axes...



Example: Old Faithful Geyser (3)

- Example of inappropriate scaling and, hence, bad visibility of the clusters.

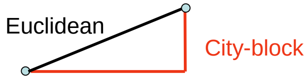


Ingredients for clustering

- Input data: table individuals $\times$ variables (or a distance matrix)
- Need for a distance to measure similarity or dissimilarity between different observations
- Quality of clusters, number of clusters?

Distances and dissimilarities

- Given data $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^d$, dissimilarity $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$, e.g.,



- the usual L_2-norm (Euclidean):

$$d(\boldsymbol{x}, \boldsymbol{y}) = \|x - y\|_2 = \sqrt{\sum_{j=1}^{d} (x_j - y_j)^2}$$

- L_1-norm (city block, or taxi driver's distance):

$$d(\boldsymbol{x}, \boldsymbol{y}) = \|x - y\|_2 = \sum_{j=1}^{d} |x_j - y_j|$$

- L_p-norm:

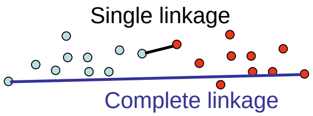
$$d(\boldsymbol{x}, \boldsymbol{y}) = \|x - y\|_p = \sqrt[p]{\sum_{j=1}^{d} (x_j - y_j)^p}$$

Agglomerative Hierarchical Clustering (AHC)

- A wide-spread approach
- Principle: sequentially agglomerate clusters of individuals using
  - a distance between individuals: city block, Euclidean, etc.
  - an agglomerative criterion: single linkage, complete linkage, average linkage, Ward's criterion
- Builds a hierarchy in a “bottom-up” fashion

AHC: distance between clusters

- Dissimilarities d_{ij} between any pair of observations i and j.
- Clusters G_1 and G_2 (an example)



- Linkage: function $d(G_1, G_2)$ that takes two groups G_1,G_2 and returns a dissimilarity score between them:
  - Single linkage (nearest-neighbor linkage):

$$d(G_1, G_2) = \min_{i\in G_1, j\in G_2} d_{ij}$$

  - Complete linkage (furthest-neighbor linkage):

$$d(G_1, G_2) = \max_{i\in G_1, j\in G_2} d_{ij}$$

  - Average linkage:

$$d(G_1, G_2) = \text{Average}_{i\in G_1, j\in G_2} d_{ij}$$

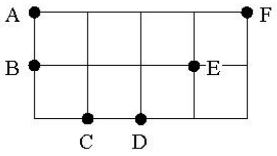
AHC: algorithm

Input $D = \{d_{ij}\}$, the $n \times n$ (symmetric) matrix of dissimilarities $d_{ij} = d(\boldsymbol{x}_i, \boldsymbol{x}_j)$ between the n clusters, given a linkage $d(G, H)$

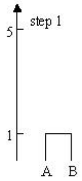
1. Merge two clusters G and H whose $d(G, H)$ is the smallest.
2. With the new cluster GH and remaining clusters, repeat Step 1 until there is only one cluster

AHC: example

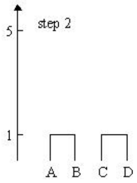
- For ease of presentation, the city block distance and the complete linkage are used.



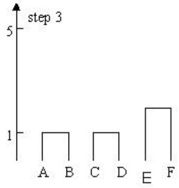
	A	B	C	D	E	F
A	0					
B	1	0				
C	3	2	0			
D	4	3	1	0		
E	4	3	3	2	0	
F	4	5	5	4	2	0



	AB	C	D	E	F
AB	0				
C	3	0			
D	4	1	0		
E	4	3	2	0	
F	5	5	4	2	0

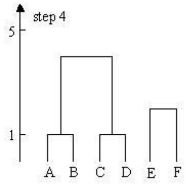


	AB	CD	E	F
AB	0			
CD	4	0		
E	4	3	0	
F	5	5	2	0

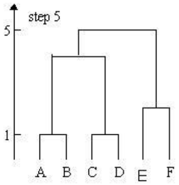


AHC: example continued

	AB	CD	EF
AB	0		
CD	4	0	
EF	5	5	0



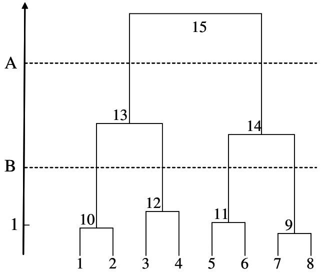
	ABCD	EF
ABCD	0	
EF	5	0



- Vertical axis: distance between clusters
- Horizontal axis: observations
- Dendrogram is a binary tree where:
  - Each node represents a cluster
  - Each leaf node is an observation
  - Root node is the whole data with all observations.

AHC: hierarchy and partition

- With I individuals, there are $I - 1$ nodes, often numbered from $I + 1$ to $2I - 1$ according to the order of construction.
- The first I numbers are reserved for the leaf nodes.



AHC: Airline distances

- Consider using single linkage.

```
D(7) =
            Fr       HK      Lnd      Mnt      Mos       NY       Tk
    Fr       0     8277      400     3640     1253     3851     9776
    HK    8277        0     8252    10345     6063    10279     1788
   Lnd     400     8252        0     3251     1557     3456     9536
   Mnt    3640    10345     3251        0     5259      330     8199
   Mos    1253     6063     1557     5259        0     5620     4667
    NY    3851    10279     3456      330     5620        0     8133
    Tk    9776     1788     9536     8199     4667     8133        0
```

- Merge two clusters `Mnt` and `NY` as `d(Mnt, NY)` is smallest.
- Compute new $(n-1) \times (n-1)$ dissimilarity matrix

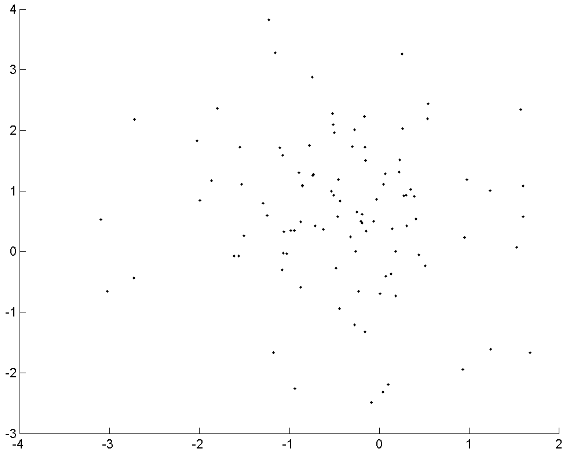
AHC: Airline distances (2)

```
D(6) =
          MntNY      Fr      HK     Lnd     Mos      Tk
  MntNY       0    3640   10279    3251    5259    8133
     Fr    3640       0    8277     400    1253    9776
     HK   10279    8277       0    8252    6063    1788
    Lnd    3251     400    8252       0    1557    9536
    Mos    5259    1253    6063    1557       0    4667
     Tk    8133    9776    1788    9536    4667       0
```

- Merge two clusters `Fr` and `Lnd` as `d(Fr, Lnd)` is smallest
- Repeat until there is only one cluster.

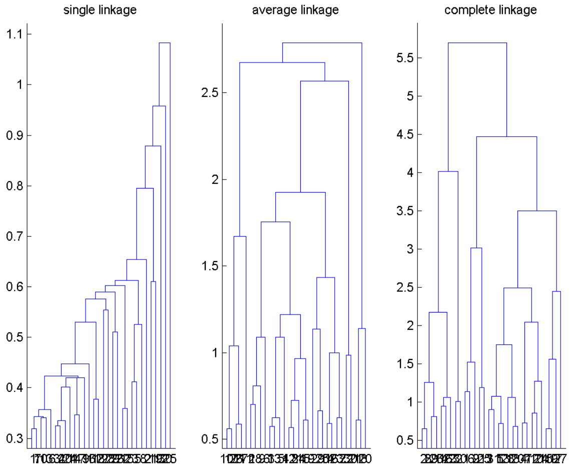
AHC: type of linkage

- Randomly generated data:



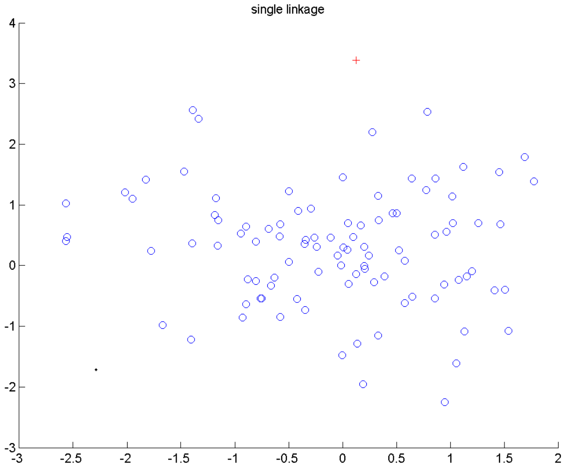
AHC: type of linkage (2)

- Three different linkages (single, average and complete)



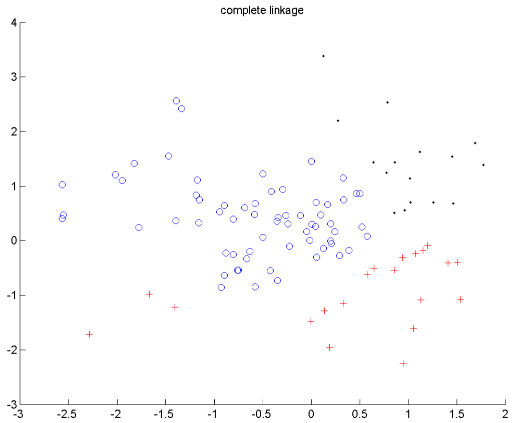
AHC: single linkage

- Tends to leave single points as clusters
- Suffers from chaining (clusters spread out, not compact)



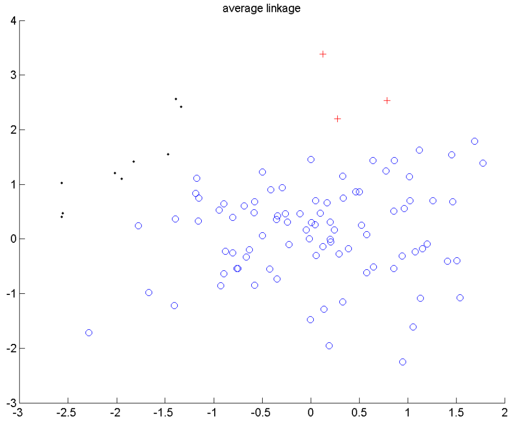
AHC: complete linkage

- Can have a disjoint cluster
- Suffers from crowding (a point can be closer to points in other clusters than to points in its own cluster)



AHC: average linkage

- A good balance – relatively compact, relatively far apart other clusters than to points in its own cluster)



Quality of partition

The partition is of high quality when:

- The individuals within a cluster are homogeneous (small within-cluster variability)
- The individuals differ from one cluster to another (high between-cluster variability)

The total variance can be decomposed into two parts:

- within-cluster variance w.r.t. the center of mass of the cluster and
- between-cluster variance accounting for deviation between each center of mass for a given cluster and the overall center of mass.

$$\underbrace{\sum_{k=1}^{K}\sum_{q=1}^{Q}\sum_{i=1}^{I_q}(x_{iqk}-\bar{x}_k)^2}_{\text{total variance}} = \underbrace{\sum_{k=1}^{K}\sum_{q=1}^{Q} I_q(\bar{x}_{qk}-\bar{x}_k)^2}_{\text{between - cluster variance}} + \underbrace{\sum_{k=1}^{K}\sum_{q=1}^{Q}\sum_{i=1}^{I_q}(x_{iqk}-\bar{x}_{qk})^2}_{\text{within - cluster variance}}$$

Quality of partition (2)

- As a result, quality of partition can be measured by:

$$\frac{\text{Between-cluster variance}}{\text{Total variance}} \tag{1}$$

- The closer ratio (1) to 1, the better the quality of clustering

Ward's algorithm

- At step n of the agglomerative algorithm, the individuals are distributed over $Q = I - n + 1$ clusters obtained from previous steps.
- The issue is in choosing the two clusters (among Q) to be agglomerated (merged).
- When grouping two clusters together, we move from a partition in Q clusters to a partition in $Q - 1$ clusters; the within-cluster inertia can only increase.
- Grouping according to inertia (=Ward's algorithm) means choosing the two clusters to be agglomerated so as to minimize the increase of within-cluster inertia.

Ward's algorithm (2)

- Consider clusters p (with the center of mass g_p and sample size I_p) and q (with the center of mass g_q and sample size I_q).
- The increase $\Delta(p,q)$ in within-cluster variance caused by grouping together clusters p and q reads:

$$\Delta(p,q) = \frac{I_p I_q}{I_p + I_q} d^2\left(g_p, g_q\right) \tag{2}$$

Choosing clusters p and q so as to minimize (2) means choosing:

- Clusters whose centers of mass are close (small $d^2(g_p, g_q)$).
- Clusters with small sample sizes (small $\frac{I_q I_p}{I_q + I_p}$)

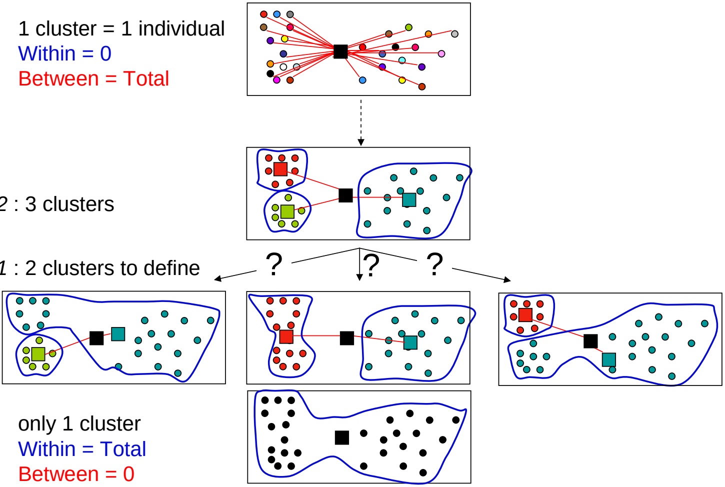
Ward's algorithm (3)

Step 1: 1 cluster = 1 individual
Within = 0
Between = Total

Step *I-2* : 3 clusters

Step *I-1* : 2 clusters to define

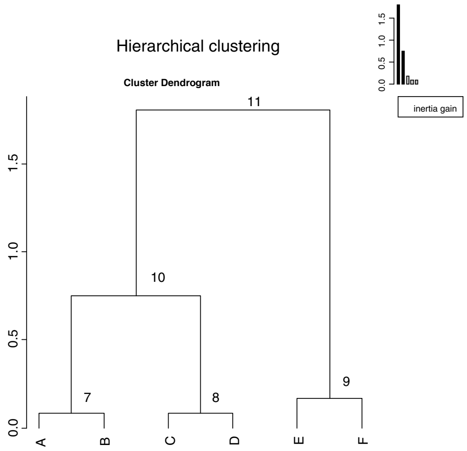
Step *I* : only 1 cluster
Within = Total
Between = 0



- Ward's rule minimizes the increasing within-cluster inertia.

Ward's algorithm: example

- Tree obtained by applying Ward's algorithm to the data in Slide 12 and by using the usual Euclidean metric.



- The levels of the nodes, top right corner.
- Each irregularity in the decrease of nodes claims further division.

Ward's algorithm: analysis

Number of the node	p	q	$\frac{I_p I_q}{I_p+I_q}$	$d^2(g_p, g_q)$	Index	in %	Cumulative %	Within inertia	Within variance
7	2	1	0.5	0.167	0.083	2.88	100	0.083	0.250
8	4	3	0.5	0.167	0.083	2.88	97.12	0.083	0.250
9	6	5	0.5	0.333	0.167	5.77	94.23	0.167	0.500
10	8	7	1	0.750	0.750	25.96	88.46	0.917	1.375
11	9	10	1.33	1.354	1.806	62.50	62.50	2.889	2.889
Sum					2.889	100			

Note: The individuals are considered as nodes numbered in the order in which they appear in the file (here in alphabetical order).

- The quantity $\Delta(p,q)$ serves as index.
- This index is ever-increasing, i.e., $\Delta_n \geq \Delta_{n-1}$ where $\Delta_n = \Delta(p,q)$ at the n-th step of clustering.
- The sum of all Δ_n (from the same hierarchy) is equal to the total inertia, i.e.,

$$\sum_{n}^{I-1} \Delta_n = \text{total inertia}$$

K-means clustering

- Another popular approach.
- Unlike the hierarchical clustering, K-means algorithm seeks for a partition of the observations into a **pre-specfied** number of clusters.
- Denote this number by Q.

K-means clustering: algorithm

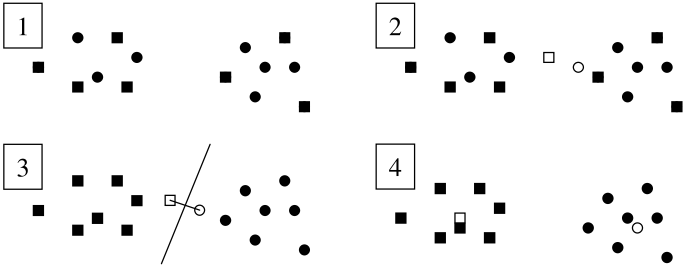
Let P_n be the partition of the individuals at step n of the algorithm and ρ_n is the ratio (1) of this partition P_n.

0. Consider a given initial partition P_0 (e.g., random), calculate ρ_0.

At step n of the algorithm:

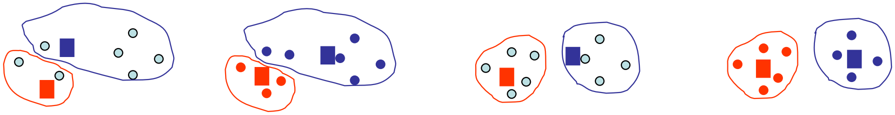
1. Compute the center of mass $g_n(q)$ for each cluster q of P_n.
2. Reassign each individual to the cluster q to which it is closest (in terms of distance to the centers of gravity $g_n(q)$!).
3. Thus, obtain a new partition P_{n+1}. Compute ρ_{n+1}.
4. As long as
$$\rho_{n+1} - \rho_n > \text{threshold}$$
i.e., partition P_{n+1} is better than P_n, return to step 1.
Otherwise, P_{n+1} is the optimal partition.

K-means clustering: algorithm



- The convergence is ensured by the fact that, at each step, ρ_n decreases.
- In the above demonstration, the algorithm has converged at stage 4.

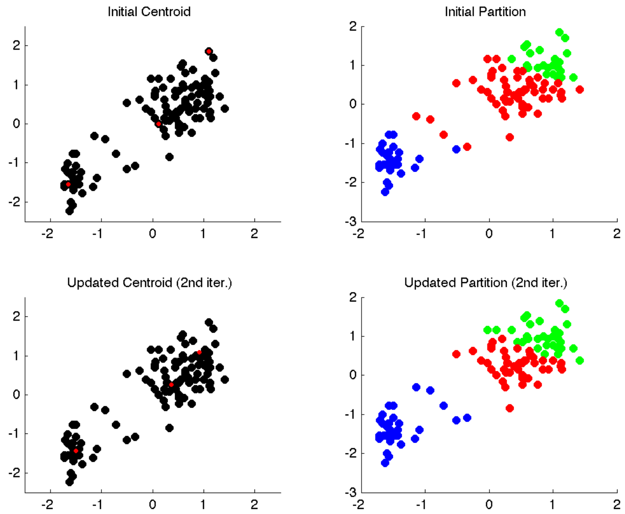
K-means clustering: another illustration



- Here, the algorithm has also converged at the last stage.

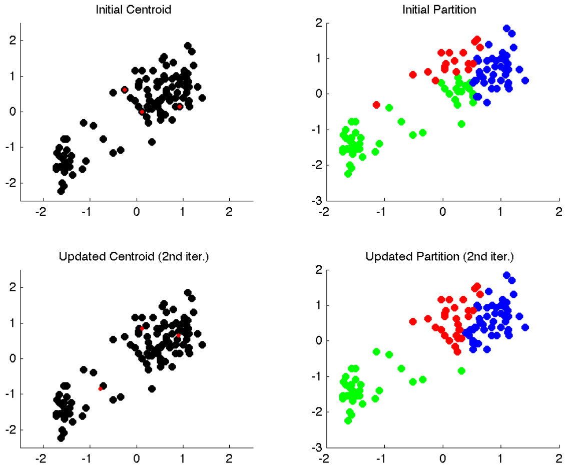
K-means example 1

- First two iterations of the algorithm



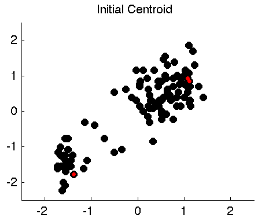
K-means example 1 (2)

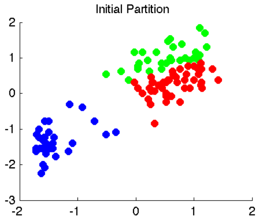
- First two iterations of the algorithm (*another initial partition*)

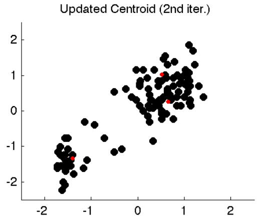


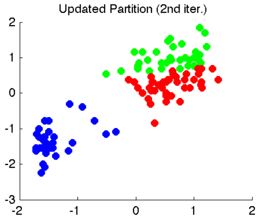
K-means example 1 (3)

- First two iterations of the algorithm (*yet another initial partition*)









K-means algorithm: properties

- Iterations always converge.
- Different initial values may lead to different solutions.
- K-means is typically run multiple times, with random initial values for each run. Final solution is chosen among from the collection of centers based on which one gives the smallest within-cluster variation.
- K-means can deal with much greater numbers of individuals than the hierarchical clustering.

When K-means may not be preferred

- In K-means, cluster p is represented by the center of mass $g_n(p)$ = the average of all points in that cluster.
- In some applications
  - we may want each cluster to be represented by one of the points in the data (instead of some averaged point which may be meaningless).
  - we may have only pairwise dissimilarities d_{ij} rather than actual points (thus no averaging)
- This is where K-medoids comes in

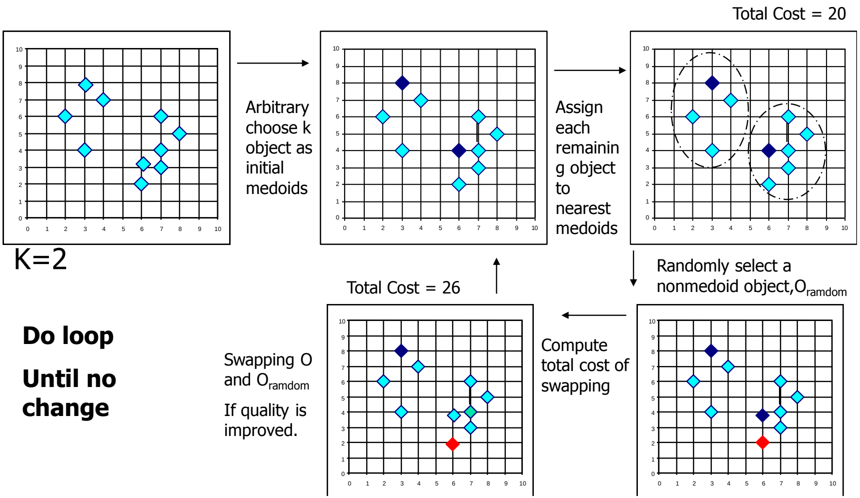
K-medoids: algorithm

- K-medoids is similar to K-means, but searches for K representative objects (medoids).
- Starts from an initial set $m_i \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$, $i = 1, \ldots, K$ of K medoids and iteratively replaces one of the medoids by one of the non-medoids if it improves the within-cluster inertia:

```
do
    ‣ Creating clusters: distribute $x_i$ over the clusters according to
      proximity to medoids $m_k$
    ‣ Choosing new medoids: within the clusters, choose the new
      medoid so that the within-cluster inertia reduces.
end do
```

- K-medoids is more robust than K-means in the presence of noise and outliers because a medoid is less influenced by outliers or other extreme values than the mean.
- Works effectively for small data sets but does not scale well for large data sets.

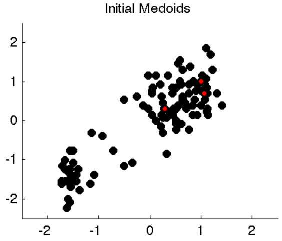
K-medoids: schematic

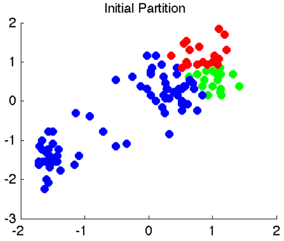

Total Cost = 20
Arbitrary choose k object as initial medoids
Assign each remaining object to nearest medoids
K=2
Randomly select a nonmedoid object,O_{ramdom}
Total Cost = 26
Compute total cost of swapping
Swapping O and O_{ramdom}
If quality is improved.

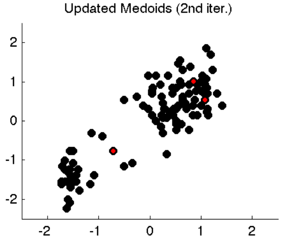
Do loop

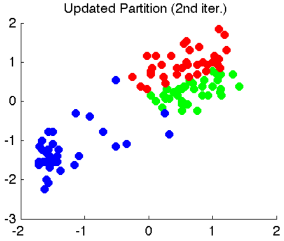
Until no change

K-medoids: example





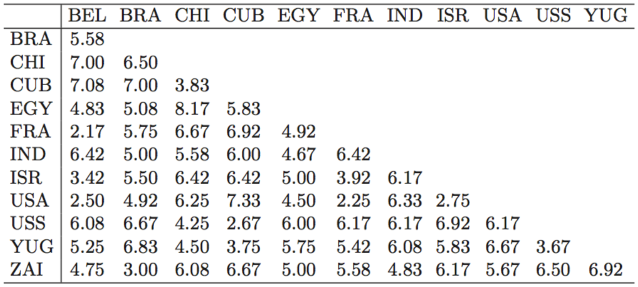




K-medoids: example

- The average dissimilarity scores are given.
- K-means clustering could not be applied because we have only distances rather than raw observations.

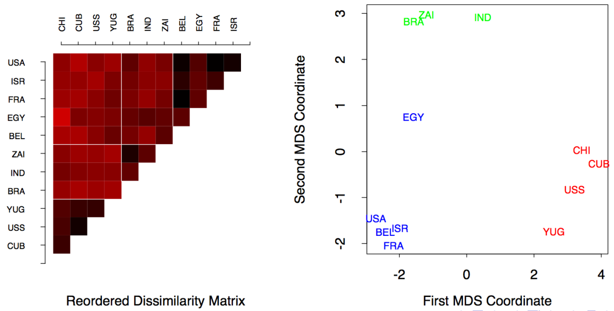
TABLE 14.3. *Data from a political science survey: values are average pairwise dissimilarities of countries from a questionnaire given to political science students.*



	BEL	BRA	CHI	CUB	EGY	FRA	IND	ISR	USA	USS	YUG
BRA	5.58										
CHI	7.00	6.50									
CUB	7.08	7.00	3.83								
EGY	4.83	5.08	8.17	5.83							
FRA	2.17	5.75	6.67	6.92	4.92						
IND	6.42	5.00	5.58	6.00	4.67	6.42					
ISR	3.42	5.50	6.42	6.42	5.00	3.92	6.17				
USA	2.50	4.92	6.25	7.33	4.50	2.25	6.33	2.75			
USS	6.08	6.67	4.25	2.67	6.00	6.17	6.17	6.92	6.17		
YUG	5.25	6.83	4.50	3.75	5.75	5.42	6.08	5.83	6.67	3.67	
ZAI	4.75	3.00	6.08	6.67	5.00	5.58	4.83	6.17	5.67	6.50	6.92

K-medoids: example (2)

- Dissimilarities reordered and blocked according to 3-medoid clustering.
- Left panel: heat map is coded from most similar (dark red) to least similar (bright red).
- Right panel: two-dimensional multidimensional scaling plot, with 3-medoid clusters indicated by different colors.

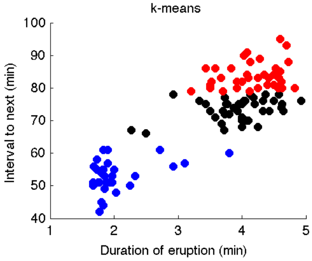


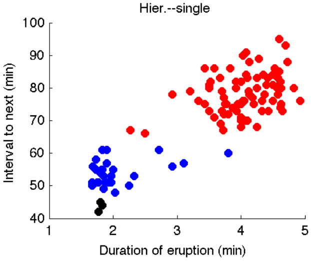
More on dissimilarities

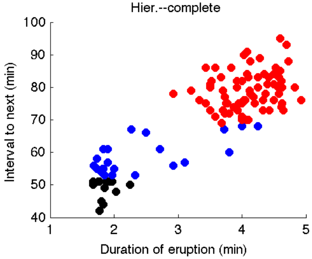
- For $\boldsymbol{x}, \boldsymbol{y} \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^d$:
- p-norm: $\|x - y\|_p = \sqrt[p]{\sum_{i=1}^{d} (x_i - y_i)^p}$
- Standardized distance: $d^2(x, y) = \sum_{i=1}^{d} \frac{(x_i - y_i)^2}{s_i^2} = (x - y)^T D (x - y)$ where $\mathbf{D} = \text{diag}\{s_1, \ldots, s_d\}$ and s_i is the standard deviation of the i-th measurement.
- Mahalanobis distance: $d^2(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{x} - \boldsymbol{y})^T \mathbf{S}^{-1} (\boldsymbol{x} - \boldsymbol{y})$ where $\mathbf{S}$ is the covariance matrix.
- Many others...

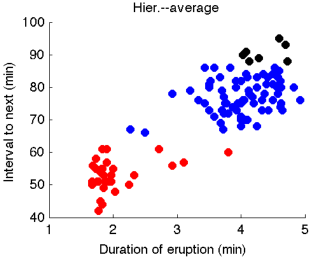
More on dissimilarities (2)

Using squared 2-norm:



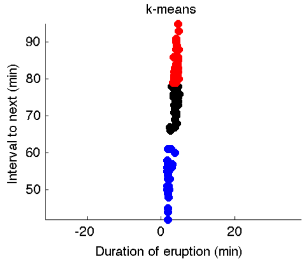


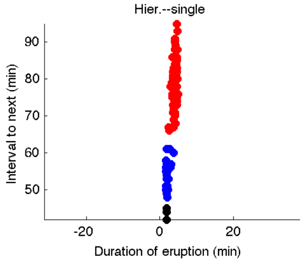


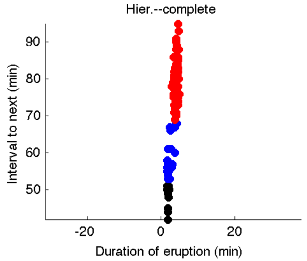


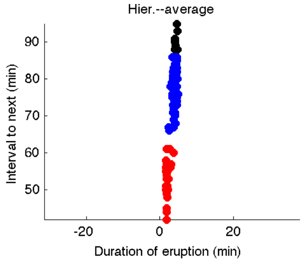
More on dissimilarities (3)

Using squared 2-norm (squared Euclidean distance):



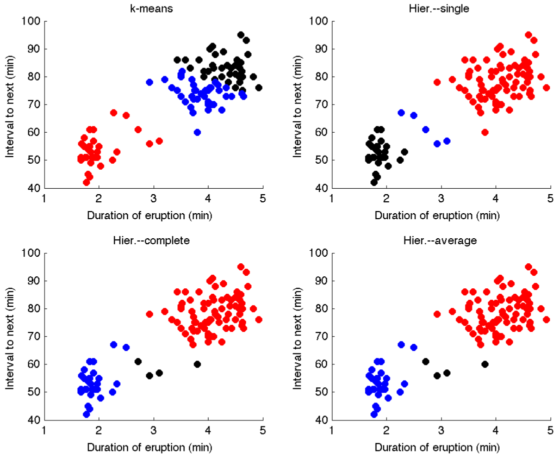






More on dissimilarities (4)

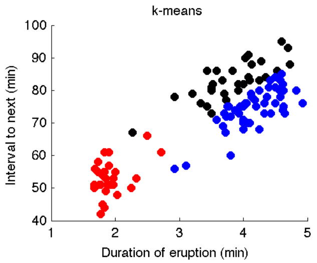
Using Standardized distance:

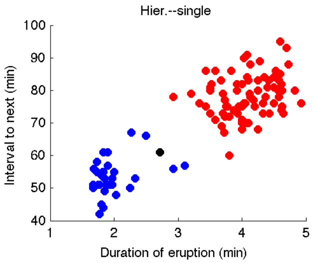


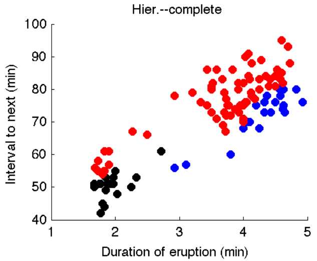
Using standardized distance = using squared Euclidean distance for standardized variables

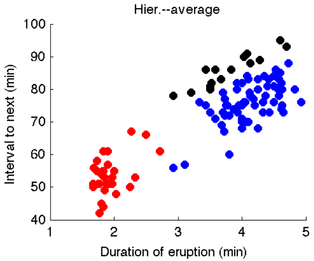
More on dissimilarities (5)

Using Mahalanobis distance:









Probabilistic clustering: Gaussian mixture model (GMM)

Assumptions:

- There are K components (multivariate normal distributions).
- Y is a discrete random variable labeling the components.
- The distribution $\Pr(Y = k)$, $k = 1, \dots, K$ is not known in general.
- Each observed datum $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \in \mathbb{R}^d$ comes from a mixture of multivariate normal distributions:

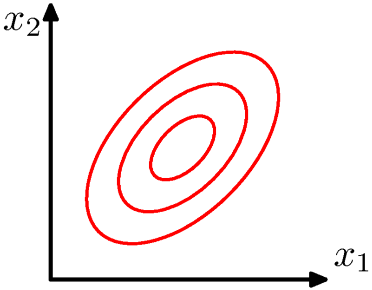
$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \Pr(Y = k)\mathcal{N}(\boldsymbol{x}|\mu_k, \Sigma_k), \quad \sum_{k=1}^{K} \Pr(Y = k) = 1.$$

where μ_k, Σ_k are the mean and covariance matrix of the k-th multivariate normal.

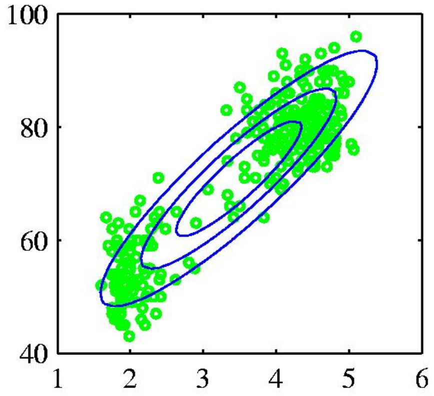
Multivariate Gaussians

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]$$

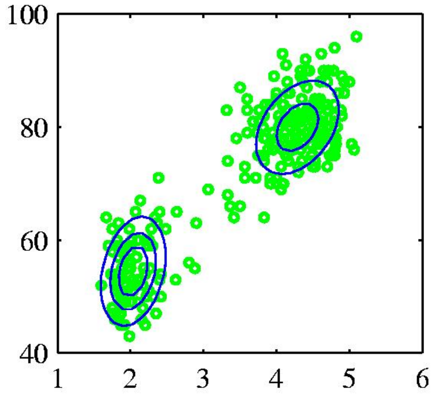
- $\boldsymbol{x} \in \mathbb{R}^d$
- $\boldsymbol{\mu}$ is a vector with means, Σ is a covariance matrix.



GMM: illustration

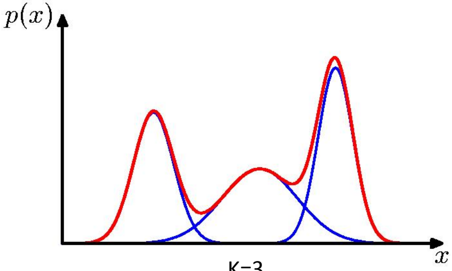


Single Gaussian

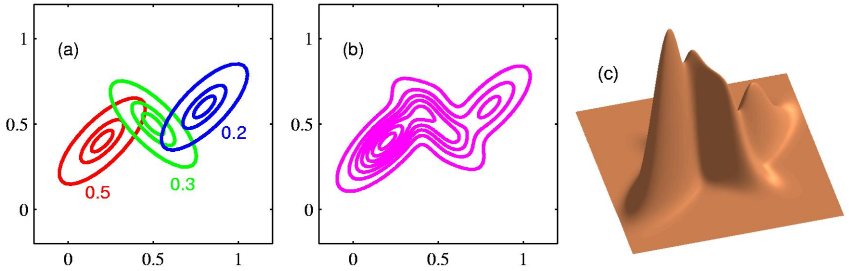


Mixture of two Gaussians

GMM: illustration (2)



K=3



GMM: maximum likelihood estimation

- MLE:

$$\arg\max_{\theta} \prod_{j=1}^{m} \Pr(y_j, x_j)$$

- $y_j = 1, \dots, K$ is a number of cluster where jth observation comes from
- θ = all the parameters of the model:

$$\theta = \{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K, \Sigma_1, \dots, \Sigma_K; \Pr(Y = 1), \dots, \Pr(Y = K)\}$$

- We do not know y_j!
- Remedy: maximize the marginal likelihood

$$\arg\max_{\theta} \prod_{j=1}^{m} \Pr(x_j) = \arg\max_{\theta} \prod_{j=1}^{m} \sum_{k=1}^{K} \Pr(Y = k, x_j) \tag{3}$$

- (3) is a tough problem: no closed form solutions, numerically intensive, non-convex function => many local optima

Expectation maximization (EM)

Iterate:
On the n-th iteration, let the estimates be

$$\theta^{(n)} = \left\{\mu_1^{(n)}, \ldots, \mu_K^{(n)}, \Sigma_1^{(n)}, \ldots, \Sigma_K^{(n)}, \Pr^{(n)}(Y = 1), \ldots, \Pr^{(n)}(Y = K)\right\}$$

- E-step: Compute "expected" classes of all datapoints for each class

  $$\Pr\left(Y_j = k | x_j, \theta^{(n)}\right) \propto \mathsf{Pr}^{(n)}(Y = k) p(x_j | \mu_k^{(n)}, \Sigma_k^{(n)})$$

  i.e., just evaluate a Gaussian at $\boldsymbol{x}_j$
- M-step: Compute weighted MLE for $\boldsymbol{\mu}_k$, Σ_k and $\Pr(Y = k)$ given expected classes above:

  $$\mu_k^{(n+1)} = \frac{\sum_{j=1}^{m} \Pr\left(Y_j = k | x_j, \theta^{(n)}\right) \boldsymbol{x}_j}{\sum_{j=1}^{m} \Pr\left(Y_j = k | \boldsymbol{x}_j, \theta^{(n)}\right)}$$

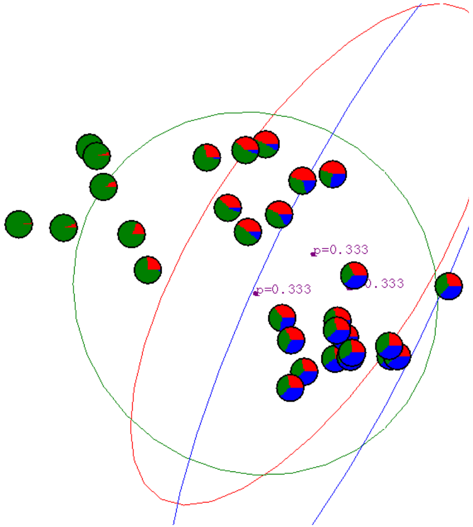
EM (2)

- M-step: MLE for Σ and $\mathrm{Pr}(Y = k)$:

$$\Sigma_k^{(n+1)} = \frac{\sum_{j=1}^{m} \mathrm{Pr}\left(Y_j = k | \boldsymbol{x}_j, \theta^{(n)}\right) \left[\boldsymbol{x}_j - \boldsymbol{\mu}_k^{(n+1)}\right] \left[\boldsymbol{x}_j - \boldsymbol{\mu}_k^{(n+1)}\right]^T}{\sum_{j=1}^{m} \mathrm{Pr}\left(Y_j = k | \boldsymbol{x}_j, \theta^{(n)}\right)}$$

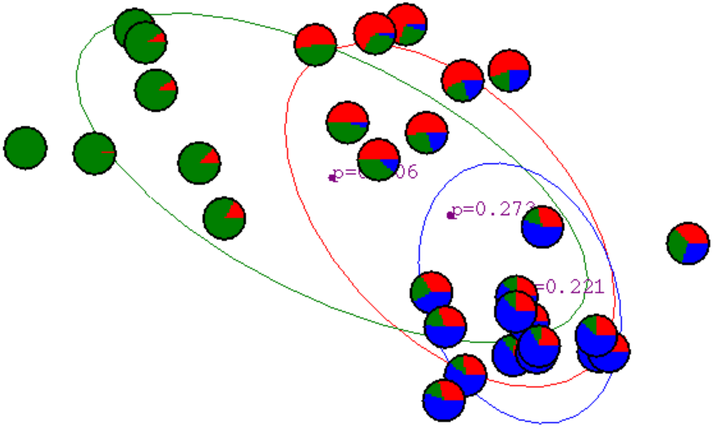
$$\mathsf{Pr}^{(n+1)}(Y = k) = \frac{\sum_{j=1}^{m} \mathrm{Pr}\left(Y_j = k | x_j, \theta^{(n)}\right)}{m}$$

GMM: example (1)



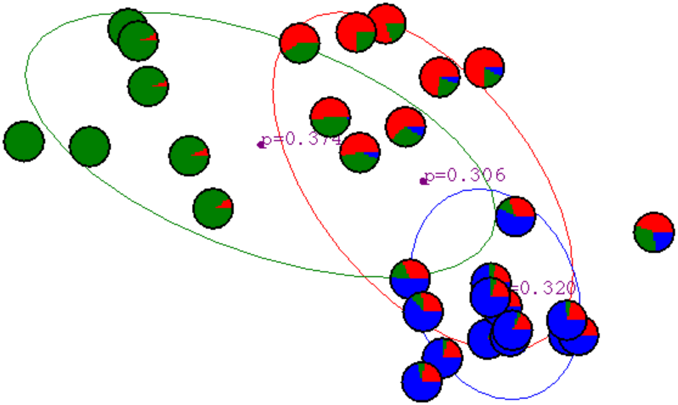
Initial model parameters.

GMM: example (2)



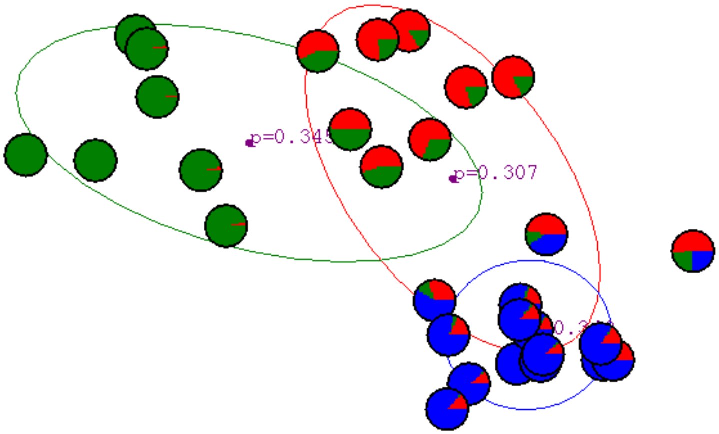
After first iteration

GMM: example (3)



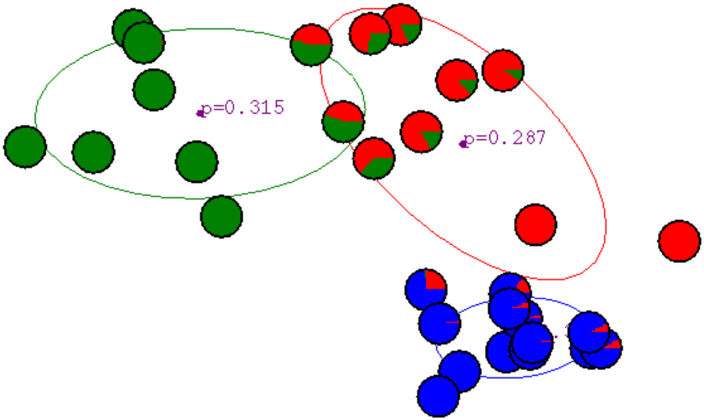
After second iteration

GMM: example (4)



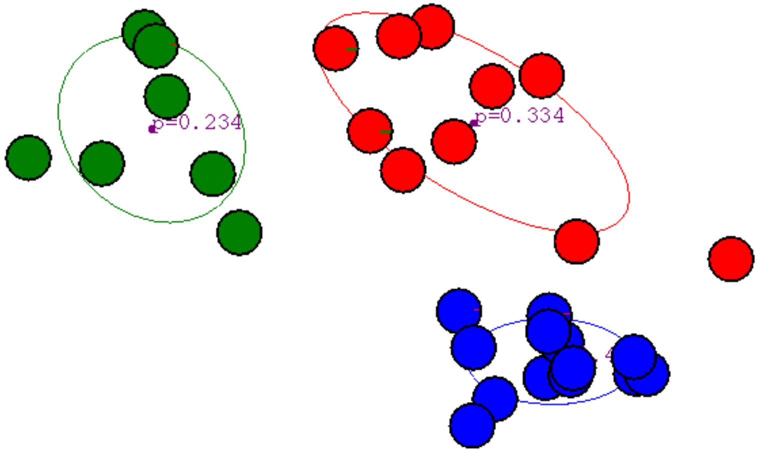
After third iteration

GMM: example (5)



After sixth iteration

GMM: example (6)



After convergence

References

See Chapter 4 of [1] for more.

F. Husson, S. Le, J. Pagès, Exploratory Multivariate Analysis by Example Using R, Second Edition, Chapman & Hall/CRC Computer Science & Data Analysis, CRC Press, 2017.
URL
https://books.google.com/books?id=nLrODgAAQBAJ