

Clustering

Clustering

- What is Clustering?
- Types of Data in Cluster Analysis
- A Categorization of Major Clustering Methods
- Partitioning Methods
- Hierarchical Methods

What is Clustering?

- **Clustering** of data is a method by which large sets of data are grouped into clusters of smaller sets of similar data.



- **Cluster**: a collection of data objects
 - Similar to one another within the same cluster
 - Dissimilar to the objects in other clusters



- Clustering is unsupervised classification: **no predefined classes**

What is Clustering?

- Typical applications
 - As a stand-alone tool to get insight into data distribution
 - As a preprocessing step for other algorithms
- Use cluster detection when you **suspect** that there are **natural groupings** that may represent groups of customers or products that have lot in common.
- When there are **many competing patterns** in the data, making it hard to spot a single pattern, creating clusters of similar records **reduces the complexity** within clusters so that other data mining techniques are more likely to succeed.

Examples of Clustering Applications

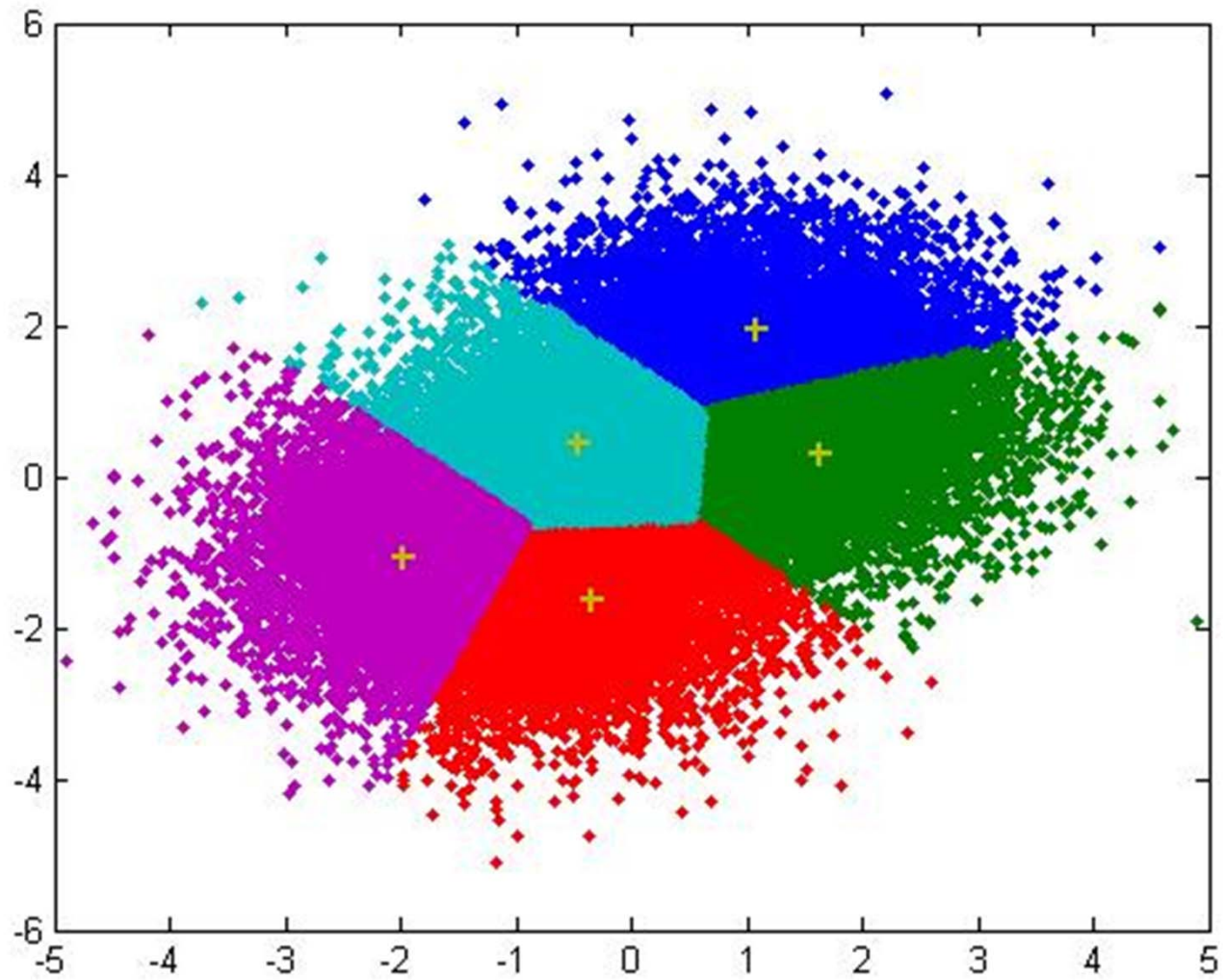
- **Marketing:** Help marketers discover distinct groups in their customer bases, and then use this knowledge to develop targeted marketing programs
- **Land use:** Identification of areas of similar land use in an earth observation database
- **Insurance:** Identifying groups of motor insurance policy holders with a high average claim cost
- **City-planning:** Identifying groups of houses according to their house type, value, and geographical location
- **Earth-quake studies:** Observed earth quake epicenters should be clustered along continent faults

Clustering definition

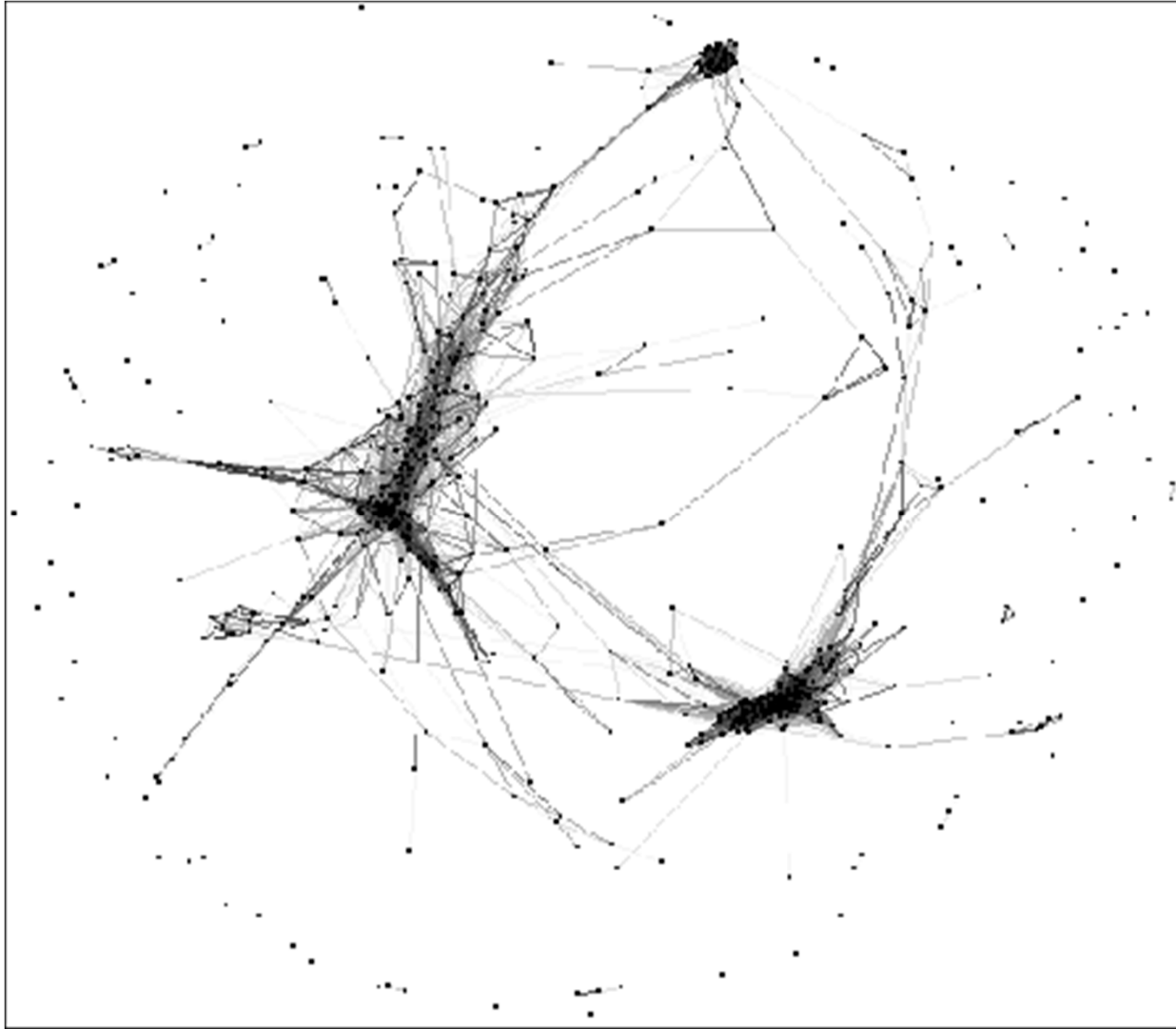
- Given a set of data points, each having a set of attributes, and a similarity measure among them, find clusters such that:
 - data points in one cluster are more similar to one another (**high intra-class similarity**)
 - data points in separate clusters are less similar to one another (**low inter-class similarity**)
- Similarity measures: e.g. Euclidean distance if attributes are continuous.

Requirements of Clustering in Data Mining

- Scalability
- Ability to deal with different types of attributes
- Discovery of clusters with arbitrary shape
- Minimal requirements for domain knowledge to determine input parameters
- Able to deal with noise and outliers
- Insensitive to order of input records
- High dimensionality
- Incorporation of user-specified constraints
- Interpretability and usability



http://webscripts.softpedia.com/screenshots/Efficient-K-Means-Clustering-using-JIT_1.png

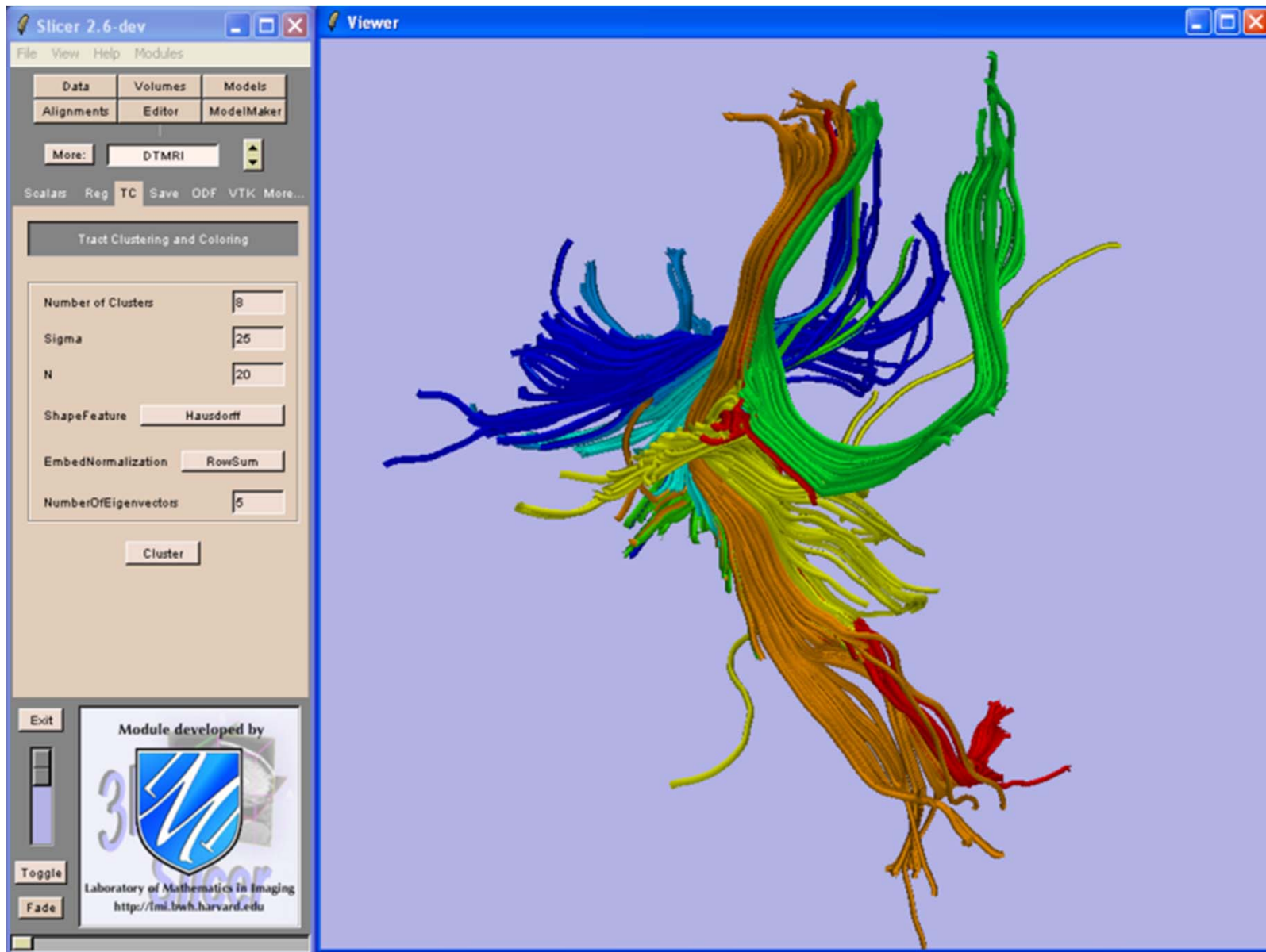


http://api.ning.com/files/ul4*osegkS5tF-JjFYZai3mGuslDu*-BQ1rFsozaAaDw9lBdc99OjNas3FPKlrdgPXAz34DU0KsbZwl7G8tM5-n4DXTk6Fab/clustering.gif

http://wiki.na-mic.org/Wiki/index.php/Progress_Report:DTI_Clustering

Project aiming at developing tools in the 3D Slicer for automatic clustering of tractographic paths through diffusion tensor MRI (DTI) data.

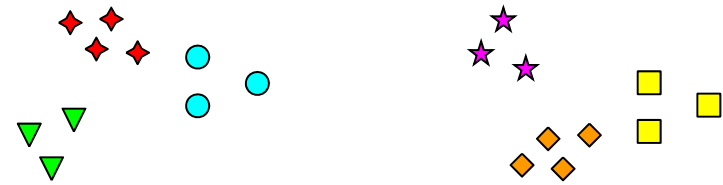
'characterize the strength of connectivity between selected regions in the brain'



Notion of a Cluster is Ambiguous



Initial points.



Six Clusters

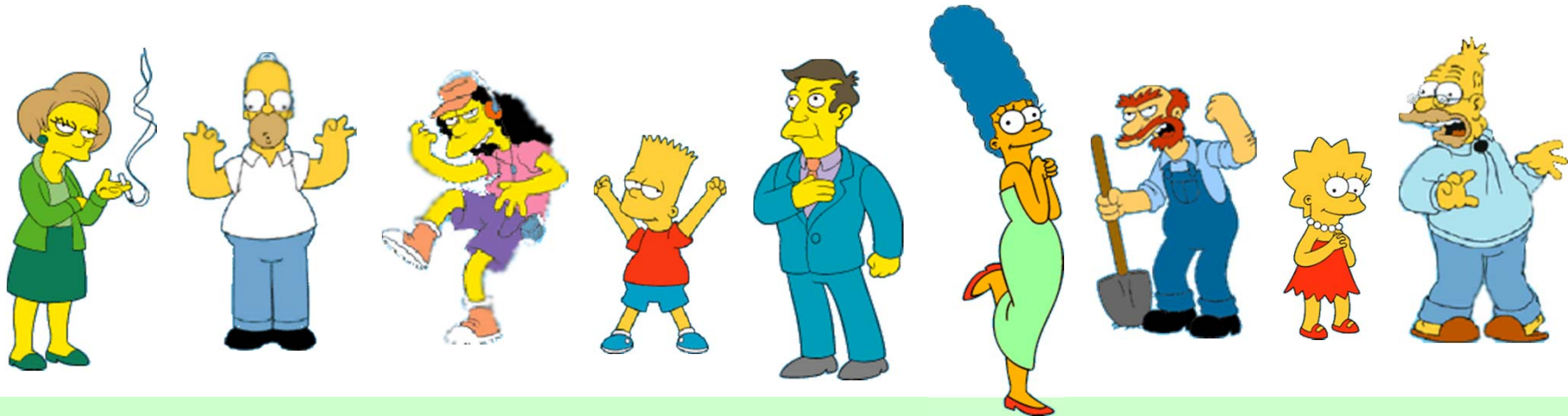


Two Clusters

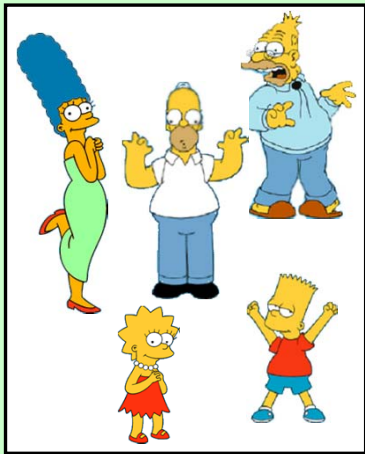


Four Clusters

What is a natural grouping among these objects?



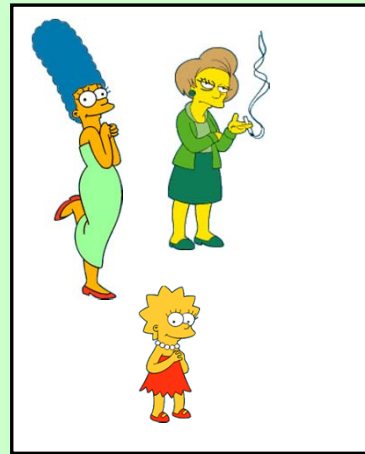
Clustering is subjective



Simpson's Family



School Employees



Females



Males

Clustering

- What is Cluster Analysis?
- Types of Data in Cluster Analysis
- A Categorization of Major Clustering Methods
- Partitioning Methods
- Hierarchical Methods

Data Matrix

- Represents n objects with p variables (attributes, measures)
 - A relational table

$$\begin{bmatrix} x_{11} & \cdots & x_{1f} & \cdots & x_{1p} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{i1} & \cdots & x_{if} & \cdots & x_{ip} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{n1} & \cdots & x_{nf} & \cdots & x_{np} \end{bmatrix}$$

| Name | Energy | Protein | Fat | Calcium | Iron |
|---------------------|--------|---------|-----|---------|------|
| Braised beef | 340 | 20 | 28 | 9 | 2.6 |
| Hamburger | 245 | 21 | 17 | 9 | 2.7 |
| Roast beef | 420 | 15 | 39 | 7 | 2 |
| Beefsteak | 375 | 19 | 32 | 9 | 2.6 |
| Canned beef | 180 | 22 | 10 | 17 | 3.7 |
| Broiled chicken | 115 | 20 | 3 | 8 | 1.4 |
| Canned chicken | 170 | 25 | 7 | 12 | 1.5 |
| Beef heart | 160 | 26 | 5 | 14 | 5.9 |
| Roast lamb leg | 265 | 20 | 20 | 9 | 2.6 |
| Roast lamb shoulder | 300 | 18 | 25 | 9 | 2.3 |
| Smoked ham | 340 | 20 | 28 | 9 | 2.5 |
| Pork roast | 340 | 19 | 29 | 9 | 2.5 |
| Pork simmered | 355 | 19 | 30 | 9 | 2.4 |
| Beef tongue | 205 | 18 | 14 | 7 | 2.5 |
| Veal cutlet | 185 | 23 | 9 | 9 | 2.7 |
| Baked bluefish | 135 | 22 | 4 | 25 | 0.6 |
| Raw clams | 70 | 11 | 1 | 82 | 6 |
| Canned clams | 45 | 7 | 1 | 74 | 5.4 |
| Canned crabmeat | 90 | 14 | 2 | 38 | 0.8 |
| Fried haddock | 135 | 16 | 5 | 15 | 0.5 |
| Broiled mackerel | 200 | 19 | 13 | 5 | 1 |
| Canned mackerel | 155 | 16 | 9 | 157 | 1.8 |
| Fried perch | 195 | 16 | 11 | 14 | 1.3 |
| Canned salmon | 120 | 17 | 5 | 159 | 0.7 |
| Canned sardines | 180 | 22 | 9 | 367 | 2.5 |
| Canned tuna | 170 | 25 | 7 | 7 | 1.2 |
| Canned shrimp | 110 | 23 | 1 | 98 | 2.6 |

Dissimilarity Matrix

- Proximities of pairs of objects
- $d(i,j)$: dissimilarity between objects i and j
- Nonnegative
- Close to 0: similar

$$\begin{bmatrix} 0 & & & & & \\ d(2,1) & 0 & & & & \\ d(3,1) & d(3,2) & 0 & & & \\ \vdots & \vdots & \vdots & & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 & \end{bmatrix}$$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 |
|-----------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|----|
| 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 0.27 | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 0.31 | 0.42 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 0.21 | 0.31 | 0.27 | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 0.37 | 0.27 | 0.52 | 0.41 | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 0.41 | 0.33 | 0.52 | 0.46 | 0.32 | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 0.41 | 0.32 | 0.52 | 0.45 | 0.28 | 0.26 | | | | | | | | | | | | | | | | | | | | | |
| 8 | 0.50 | 0.40 | 0.65 | 0.54 | 0.30 | 0.39 | 0.32 | | | | | | | | | | | | | | | | | | | | |
| 9 | 0.24 | 0.20 | 0.38 | 0.28 | 0.30 | 0.34 | 0.34 | 0.43 | | | | | | | | | | | | | | | | | | | |
| 10 | 0.22 | 0.26 | 0.32 | 0.25 | 0.37 | 0.39 | 0.39 | 0.50 | 0.23 | | | | | | | | | | | | | | | | | | |
| 11 | 0.17 | 0.27 | 0.31 | 0.21 | 0.37 | 0.41 | 0.41 | 0.51 | 0.24 | 0.22 | | | | | | | | | | | | | | | | | |
| 12 | 0.18 | 0.29 | 0.30 | 0.20 | 0.39 | 0.42 | 0.42 | 0.52 | 0.25 | 0.22 | 0.18 | | | | | | | | | | | | | | | | |
| 13 | 0.20 | 0.30 | 0.28 | 0.19 | 0.40 | 0.43 | 0.43 | 0.53 | 0.27 | 0.22 | 0.19 | 0.18 | | | | | | | | | | | | | | | |
| 14 | 0.31 | 0.23 | 0.41 | 0.33 | 0.27 | 0.31 | 0.31 | 0.40 | 0.24 | 0.26 | 0.31 | 0.30 | 0.32 | | | | | | | | | | | | | | |
| 15 | 0.35 | 0.25 | 0.49 | 0.39 | 0.22 | 0.29 | 0.24 | 0.32 | 0.28 | 0.34 | 0.35 | 0.36 | 0.38 | 0.25 | | | | | | | | | | | | | |
| 16 | 0.45 | 0.35 | 0.56 | 0.49 | 0.31 | 0.23 | 0.25 | 0.38 | 0.38 | 0.43 | 0.45 | 0.46 | 0.47 | 0.34 | 0.29 | | | | | | | | | | | | |
| 17 | 0.62 | 0.54 | 0.68 | 0.65 | 0.45 | 0.45 | 0.53 | 0.39 | 0.55 | 0.58 | 0.62 | 0.62 | 0.63 | 0.49 | 0.49 | 0.50 | | | | | | | | | | | |
| 18 | 0.65 | 0.56 | 0.70 | 0.67 | 0.48 | 0.47 | 0.55 | 0.44 | 0.58 | 0.61 | 0.65 | 0.64 | 0.66 | 0.51 | 0.52 | 0.52 | 0.23 | | | | | | | | | | |
| 19 | 0.51 | 0.43 | 0.54 | 0.54 | 0.41 | 0.27 | 0.35 | 0.48 | 0.44 | 0.45 | 0.51 | 0.50 | 0.51 | 0.37 | 0.39 | 0.28 | 0.38 | 0.41 | | | | | | | | | |
| 20 | 0.46 | 0.38 | 0.50 | 0.48 | 0.36 | 0.25 | 0.30 | 0.43 | 0.39 | 0.40 | 0.46 | 0.45 | 0.46 | 0.32 | 0.34 | 0.23 | 0.45 | 0.48 | 0.24 | | | | | | | | |
| 21 | 0.35 | 0.28 | 0.44 | 0.38 | 0.30 | 0.27 | 0.28 | 0.43 | 0.29 | 0.31 | 0.35 | 0.35 | 0.35 | 0.23 | 0.28 | 0.28 | 0.53 | 0.56 | 0.33 | 0.28 | | | | | | | |
| 22 | 0.46 | 0.38 | 0.50 | 0.48 | 0.36 | 0.33 | 0.34 | 0.46 | 0.39 | 0.40 | 0.46 | 0.45 | 0.46 | 0.32 | 0.34 | 0.35 | 0.45 | 0.48 | 0.33 | 0.30 | 0.32 | | | | | | |
| 23 | 0.38 | 0.30 | 0.42 | 0.41 | 0.30 | 0.28 | 0.28 | 0.44 | 0.31 | 0.32 | 0.38 | 0.38 | 0.38 | 0.24 | 0.29 | 0.30 | 0.48 | 0.51 | 0.30 | 0.24 | 0.22 | 0.27 | | | | | |
| 24 | 0.52 | 0.44 | 0.58 | 0.54 | 0.42 | 0.29 | 0.36 | 0.49 | 0.45 | 0.46 | 0.52 | 0.51 | 0.52 | 0.38 | 0.40 | 0.29 | 0.46 | 0.49 | 0.28 | 0.25 | 0.33 | 0.24 | 0.32 | | | | |
| 25 | 0.51 | 0.41 | 0.65 | 0.55 | 0.37 | 0.44 | 0.40 | 0.49 | 0.44 | 0.50 | 0.50 | 0.52 | 0.53 | 0.40 | 0.35 | 0.42 | 0.58 | 0.62 | 0.51 | 0.48 | 0.43 | 0.35 | 0.43 | 0.41 | | | |
| 26 | 0.42 | 0.33 | 0.53 | 0.46 | 0.29 | 0.26 | 0.18 | 0.33 | 0.35 | 0.40 | 0.42 | 0.43 | 0.44 | 0.31 | 0.25 | 0.25 | 0.54 | 0.56 | 0.35 | 0.29 | 0.27 | 0.35 | 0.28 | 0.35 | 0.41 | | |
| 27 | 0.45 | 0.36 | 0.60 | 0.50 | 0.32 | 0.28 | 0.31 | 0.37 | 0.39 | 0.45 | 0.46 | 0.47 | 0.48 | 0.35 | 0.28 | 0.29 | 0.40 | 0.43 | 0.34 | 0.36 | 0.39 | 0.33 | 0.39 | 0.33 | 0.37 | 0.32 | |

Type of data in clustering analysis

- Continuous variables
- Binary variables
- Nominal and ordinal variables
- Variables of mixed types

Continuous variables

- To avoid dependence on the choice of measurement units the data should be standardized.
- **Standardize data**

- Calculate the mean absolute deviation:

$$s_f = \frac{1}{n} (|x_{1f} - m_f| + |x_{2f} - m_f| + \dots + |x_{nf} - m_f|)$$

- where $m_f = \frac{1}{n} (x_{1f} + x_{2f} + \dots + x_{nf})$

- Calculate the standardized measurement (z-score)

$$z_{if} = \frac{x_{if} - m_f}{s_f}$$

- Using mean absolute deviation is **more robust than using standard deviation**. Since the deviations are not squared the effect of outliers is somewhat reduced but their z-scores do not become too small; therefore, the outliers remain detectable.

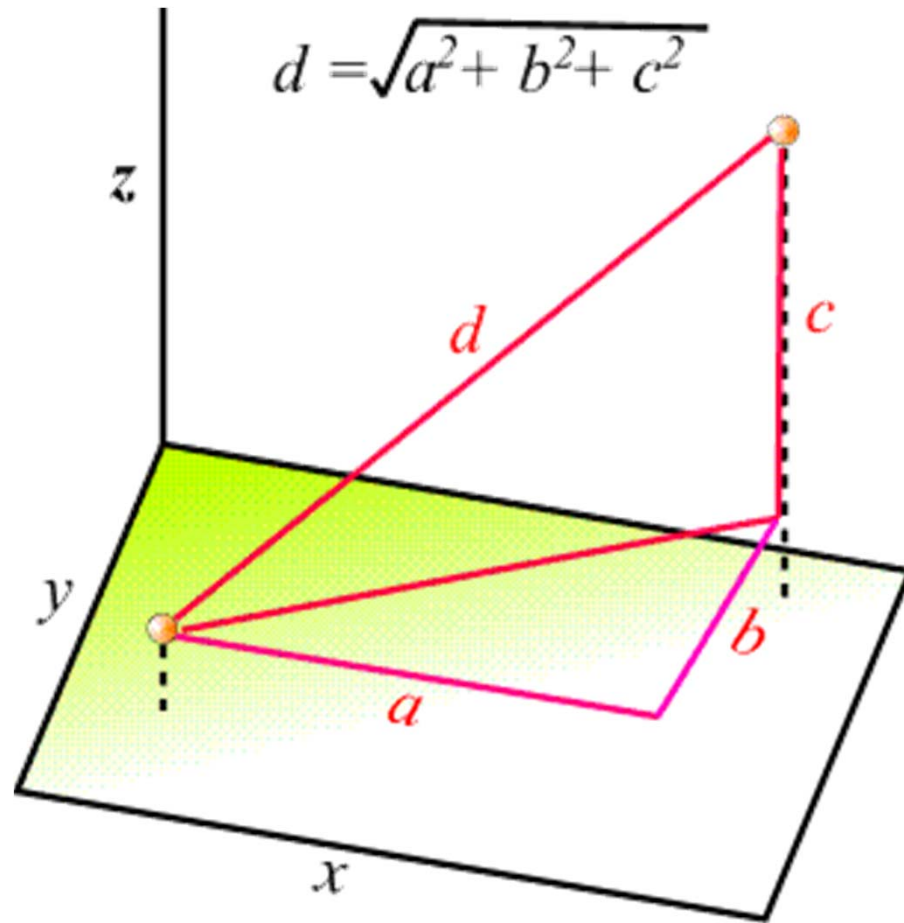
Similarity/Dissimilarity Between Objects

- **Distances** are normally used to measure the similarity or dissimilarity between two data objects
- Euclidean distance is probably the most commonly chosen type of distance. It is the geometric distance in the multidimensional space:

- Required properties for a distance function

- $d(i,j) \geq 0$
- $d(i,i) = 0$
- $d(i,j) = d(j,i)$
- $d(i,j) \leq d(i,k) + d(k,j)$

$$d(i,j) = \sqrt{\sum_{k=1}^p (x_{ki} - x_{kj})^2}$$



http://uk.geocities.com/ahf_alternate/dist.htm#S2

Similarity/Dissimilarity Between Objects

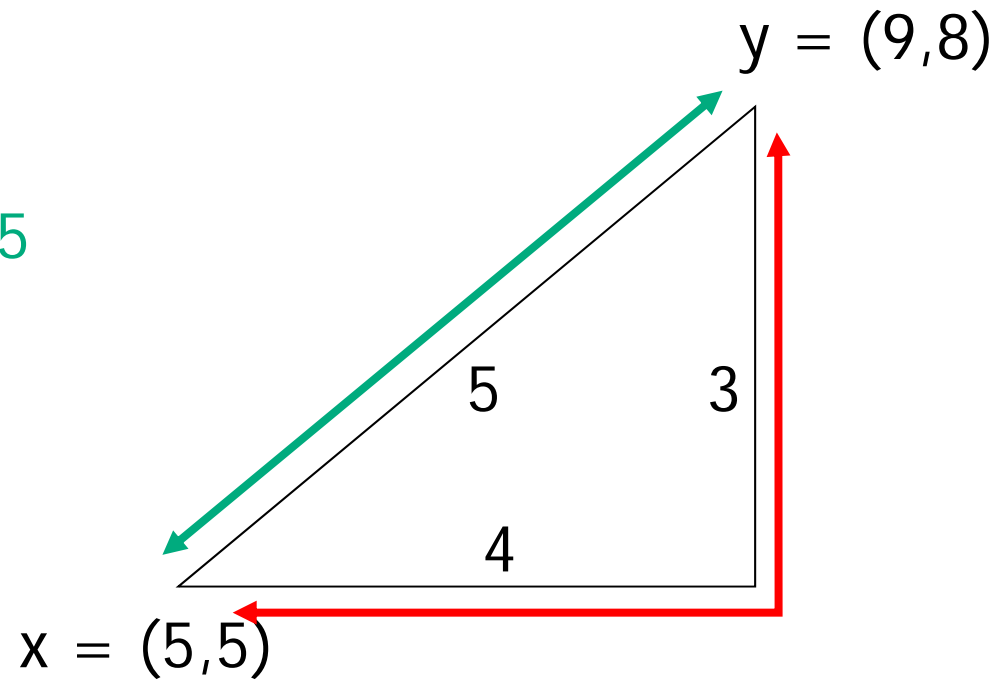
- City-block (Manhattan) distance. This distance is simply the sum of differences across dimensions. In most cases, this distance measure yields results similar to the Euclidean distance. However, note that in this measure, the effect of single large differences (outliers) is dampened (since they are not squared).

$$d(i, j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ip} - x_{jp}|$$

- The properties stated for the Euclidean distance also hold for this measure.

- *Manhattan distance* = distance if you had to travel along coordinates only.

euc.:
 $\text{dist}(x,y) = \sqrt{4^2 + 3^2} = 5$



manh.:
 $\text{dist}(x,y) = 4 + 3 = 7$

Similarity/Dissimilarity Between Objects

- Minkowski distance. Sometimes one may want to increase or decrease the progressive weight that is placed on dimensions on which the respective objects are very different. This measure enables to accomplish that and is computed as:

$$d(i, j) = \left(|x_{i1} - x_{j1}|^q + |x_{i2} - x_{j2}|^q + \dots + |x_{ip} - x_{jp}|^q \right)^{1/q}$$

Similarity/Dissimilarity Between Objects

- **Weighted distances**
- If we have some idea of the relative importance that should be assigned to each variable, then we can weight them and obtain a weighted distance measure.

$$d(i, j) = \sqrt{w_1 (x_{i1} - x_{j1})^2 + \dots + w_p (x_{ip} - x_{jp})^2}$$

Binary Variables

- Binary variable has only two states: 0 or 1
- A binary variable is **symmetric** if both of its states are equally valuable, that is, there is no preference on which outcome should be coded as 1.
- A binary variable is **asymmetric** if the outcome of the states are not equally important, such as positive or negative outcomes of a disease test.
- Similarity that is based on symmetric binary variables is called **invariant similarity**.

Binary Variables

- A contingency table for binary data

| | | Object j | | <i>sum</i> |
|------------|---|------------|-------|------------|
| | | 1 | 0 | |
| Object i | 1 | a | b | $a+b$ |
| | 0 | c | d | $c+d$ |
| <i>sum</i> | | $a+c$ | $b+d$ | p |

| | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|------------|
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | object i |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | object j |

Binary Variables

| | | Object j | | |
|------------|-------|------------|-------|-------|
| | | 1 | 0 | sum |
| Object i | 1 | a | b | $a+b$ |
| | 0 | c | d | $c+d$ |
| | sum | $a+c$ | $b+d$ | p |

- Symmetric binary dissimilarity:

$$d(i, j) = \frac{b + c}{a + b + c + d}$$

- Jaccard coefficient (asymmetric binary dissimilarity):

$$d(i, j) = \frac{b + c}{a + b + c}$$

Dissimilarity between Binary Variables

■ Example

| Name | Gender | Fever | Cough | Test-1 | Test-2 | Test-3 | Test-4 |
|------|--------|-------|-------|--------|--------|--------|--------|
| Jack | M | Y | N | P | N | N | N |
| Mary | F | Y | N | P | N | P | N |
| Jim | M | Y | P | N | N | N | N |

- gender is a symmetric attribute
- the remaining attributes are asymmetric binary
- let the values Y and P be set to 1, and the value N be set to 0

Jaccard coefficient
(for the asymmetric
variables)

$$d(jack, mary) = \frac{0 + 1}{2 + 0 + 1} = 0.33$$

$$d(jack, jim) = \frac{1 + 1}{1 + 1 + 1} = 0.67$$

$$d(jim, mary) = \frac{1 + 2}{1 + 1 + 2} = 0.75$$

Nominal Variables

- A generalization of the binary variable in that it can take more than 2 states, e.g., red, yellow, blue, green
- **Method 1:** simple matching
 - **m:** # of matches, **p:** total # of variables

$$d(i, j) = \frac{p - m}{p}$$

- **Method 2:** use a large number of binary variables
 - creating a new binary variable for each of the M nominal states

Ordinal Variables

- On ordinal variables order is important
 - e.g. Gold, Silver, Bronze
- Can be treated like continuous
 - the ordered states define the ranking $1, \dots, M_f$
 - replacing x_{if} by their rank $r_{if} \in \{1, \dots, M_f\}$
 - map the range of each variable onto $[0, 1]$ by replacing i-th object in the f-th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

- compute the dissimilarity using methods for continuous variables

Variables of Mixed Types

- A database may contain several/all types of variables
 - continuous, symmetric binary, asymmetric binary, nominal and ordinal.
- One may use a **weighted formula** to combine their effects.

$$d(i, j) = \frac{\sum_{f=1}^p \delta_{ij}^{(f)} d_{ij}^{(f)}}{\sum_{f=1}^p \delta_{ij}^{(f)}}$$

- $\delta_{ij}=0$ if x_{if} is missing or $x_{if}=x_{jf}=0$ and the variable f is asymmetric binary
- $\delta_{ij}=1$ otherwise
- continuous and ordinal variables d_{ij} : normalized absolute distance
- binary and nominal variables $d_{ij}=0$ if $x_{if}=x_{jf}$; otherwise $d_{ij}=1$

| ID | Gender | Age | Priority | WaitingTime | SpecialtyId | NumInterventions | ICD1 | NumSurg | NumInterv | OtherSpecialties | SurgeonId | SurgeonSex | ExperienceDisease | MainProcedureExpe | OtherDiagnosis | circulatorysystem | DiabetesAndAlike | | | |
|--------|--------|-----|----------|-------------|-------------|------------------|------|---------|-----------|------------------|-----------|------------|-------------------|-------------------|----------------|-------------------|------------------|--|--|--------------------|
| Col id | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | | | |
| 1 | 1 | 54 | 0 | 25 | 5 | 1 | 3317 | 0 | 0 | 0 | 881 | 0 | 80 | 737 | 1 | 0 | 0 | | | |
| 2 | 1 | 84 | 1 | 26 | 36 | 1 | 2034 | 0 | 0 | 0 | 1160 | 0 | 3 | 3 | 1 | 0 | 0 | | | |
| 3 | 1 | 88 | 1 | 97 | 36 | 1 | 2035 | 1 | 1 | 0 | 1160 | 0 | 20 | 8 | 1 | 0 | 0 | | | Binary symeric |
| 4 | 1 | 73 | 0 | 236 | 2 | 2 | 1548 | 0 | 0 | 0 | 1103 | 1 | 1 | 1 | 0 | 0 | 0 | | | Binary assymmetric |
| 5 | 1 | 36 | 0 | 38 | 20 | 3 | 494 | 0 | 0 | 0 | 894 | 1 | 515 | 259 | 0 | 0 | 0 | | | Numeric |
| 6 | 0 | 89 | 0 | 12 | 5 | 1 | 3317 | 0 | 0 | 0 | 233 | 1 | 8 | 101 | 1 | 0 | 0 | | | Ordinal |
| 7 | 1 | 61 | 0 | 140 | 5 | 1 | 1873 | 0 | 0 | 0 | 1182 | 0 | 32 | 25 | 0 | 0 | 0 | | | Nominal |
| 8 | 0 | 56 | 0 | 78 | 12 | 1 | 2339 | 0 | 0 | 0 | 592 | 1 | 0 | 0 | 0 | 1 | 0 | | | |
| 9 | 1 | 77 | 1 | 29 | 20 | 3 | 440 | 1 | 3 | 0 | 944 | 1 | 14 | 43 | 1 | 0 | 0 | | | |
| 10 | 1 | 76 | 0 | 129 | 20 | 3 | 440 | 0 | 0 | 0 | 1036 | 0 | 228 | 18 | 0 | 0 | 0 | | | |
| 11 | 1 | 60 | 0 | 41 | 5 | 1 | 3317 | 1 | 2 | 0 | 2109 | 0 | 20 | 156 | 1 | 0 | 0 | | | |
| 12 | 1 | 57 | 0 | 13 | 5 | 2 | 3317 | 0 | 0 | 0 | 233 | 1 | 6 | 79 | 0 | 0 | 0 | | | |
| 13 | 0 | 59 | 0 | 35 | 13 | 1 | 3249 | 0 | 0 | 0 | 1156 | 0 | 25 | 40 | 0 | 0 | 0 | | | |
| 14 | 0 | 60 | 0 | 24 | 13 | 2 | 1342 | 1 | 1 | 0 | 1156 | 0 | 54 | 13 | 1 | 0 | 0 | | | |
| 15 | 0 | 60 | 0 | 34 | 24 | 2 | 2919 | 2 | 3 | 1 | 826 | 0 | 136 | 253 | 1 | 0 | 0 | | | |
| 16 | 0 | 39 | 0 | 31 | 26 | 1 | 2324 | 2 | 2 | 1 | 2180 | 1 | 19 | 24 | 1 | 0 | 1 | | | |
| 17 | 0 | 36 | 0 | 84 | 5 | 1 | 3309 | 0 | 0 | 0 | 881 | 0 | 3 | 13 | 0 | 0 | 0 | | | |
| 18 | 0 | 37 | 0 | 12 | 5 | 1 | 3309 | 1 | 1 | 0 | 233 | 1 | 7 | 32 | 0 | 0 | 0 | | | |
| 19 | 0 | 81 | 0 | 14 | 20 | 2 | 478 | 1 | 2 | 0 | 996 | 1 | 309 | 323 | 1 | 0 | 0 | | | |
| 20 | 0 | 81 | 0 | 38 | 20 | 2 | 478 | 0 | 0 | 0 | 1860 | 0 | 21 | 57 | 0 | 0 | 0 | | | |

compute the dissimilarity matrix R

```
> data <- read.table("dataset.csv",header=TRUE,sep=",")  
> data$Priority <- as.factor(data$Priority)  
> data$SpecialtyId <- as.factor(data$SpecialtyId)  
> data$ICD1 <- as.factor(data$ICD1)  
> data$SurgeonId <- as.factor(data$SurgeonId)  
> library(cluster)  
> diss_matrix <- daisy(data, type = list(asymm=c(11,16,17,18),  
symm=c(2,13), ordratio = 4), metric = "gower", stand = TRUE)
```

dissimilarity matrix

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|----|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|------|----|
| 1 | | | | | | | | | | | | | | | | | | | | |
| 2 | 0.38 | | | | | | | | | | | | | | | | | | | |
| 3 | 0.47 | 0.15 | | | | | | | | | | | | | | | | | | |
| 4 | 0.54 | 0.52 | 0.56 | | | | | | | | | | | | | | | | | |
| 5 | 0.54 | 0.63 | 0.70 | 0.46 | | | | | | | | | | | | | | | | |
| 6 | 0.33 | 0.43 | 0.50 | 0.47 | 0.56 | | | | | | | | | | | | | | | |
| 7 | 0.33 | 0.42 | 0.45 | 0.39 | 0.52 | 0.42 | | | | | | | | | | | | | | |
| 8 | 0.55 | 0.57 | 0.61 | 0.45 | 0.54 | 0.39 | 0.43 | | | | | | | | | | | | | |
| 9 | 0.63 | 0.47 | 0.44 | 0.56 | 0.52 | 0.54 | 0.63 | 0.64 | | | | | | | | | | | | |
| 10 | 0.51 | 0.50 | 0.53 | 0.42 | 0.38 | 0.57 | 0.35 | 0.55 | 0.43 | | | | | | | | | | | |
| 11 | 0.25 | 0.43 | 0.38 | 0.56 | 0.60 | 0.35 | 0.34 | 0.55 | 0.47 | 0.50 | | | | | | | | | | |
| 12 | 0.35 | 0.51 | 0.59 | 0.35 | 0.40 | 0.23 | 0.32 | 0.41 | 0.51 | 0.43 | 0.34 | | | | | | | | | |
| 13 | 0.45 | 0.48 | 0.55 | 0.52 | 0.58 | 0.41 | 0.35 | 0.37 | 0.67 | 0.45 | 0.43 | 0.41 | | | | | | | | |
| 14 | 0.48 | 0.50 | 0.47 | 0.58 | 0.64 | 0.44 | 0.49 | 0.51 | 0.53 | 0.51 | 0.36 | 0.48 | 0.24 | | | | | | | |
| 15 | 0.57 | 0.64 | 0.60 | 0.71 | 0.70 | 0.57 | 0.62 | 0.63 | 0.57 | 0.62 | 0.43 | 0.61 | 0.51 | 0.36 | | | | | | |
| 16 | 0.64 | 0.67 | 0.63 | 0.67 | 0.68 | 0.49 | 0.65 | 0.54 | 0.59 | 0.74 | 0.49 | 0.58 | 0.54 | 0.47 | 0.40 | | | | | |
| 17 | 0.37 | 0.53 | 0.58 | 0.54 | 0.58 | 0.40 | 0.31 | 0.39 | 0.73 | 0.48 | 0.42 | 0.40 | 0.28 | 0.42 | 0.55 | 0.53 | | | | |
| 18 | 0.55 | 0.64 | 0.61 | 0.56 | 0.57 | 0.30 | 0.46 | 0.40 | 0.60 | 0.64 | 0.43 | 0.29 | 0.40 | 0.42 | 0.56 | 0.42 | 0.23 | | | |
| 19 | 0.61 | 0.65 | 0.61 | 0.61 | 0.52 | 0.43 | 0.68 | 0.57 | 0.43 | 0.55 | 0.48 | 0.53 | 0.56 | 0.40 | 0.44 | 0.47 | 0.60 | 0.45 | | |
| 20 | 0.54 | 0.51 | 0.58 | 0.50 | 0.53 | 0.44 | 0.43 | 0.46 | 0.58 | 0.36 | 0.51 | 0.43 | 0.31 | 0.38 | 0.51 | 0.60 | 0.34 | 0.46 | 0.35 | |

Clustering

- What is Cluster Analysis?
- Types of Data in Cluster Analysis
- **A Categorization of Major Clustering Methods**
- Partitioning Methods
- Hierarchical Methods

Major Clustering Approaches

- Partitioning algorithms: Construct various partitions and then evaluate them by some criterion
- Hierarchy algorithms: Create a hierarchical decomposition of the set of data (or objects) using some criterion
- Density-based: Based on connectivity and density functions. Able to find clusters of arbitrary shape. Continues growing a cluster as long as the density of points in the neighborhood exceeds a specified limit.
- Model-based: A model is hypothesized for each of the clusters and the idea is to find the best fit of that model to each other

Clustering

- What is Cluster Analysis?
- Types of Data in Cluster Analysis
- A Categorization of Major Clustering Methods
- Partitioning Methods
- Hierarchical Methods

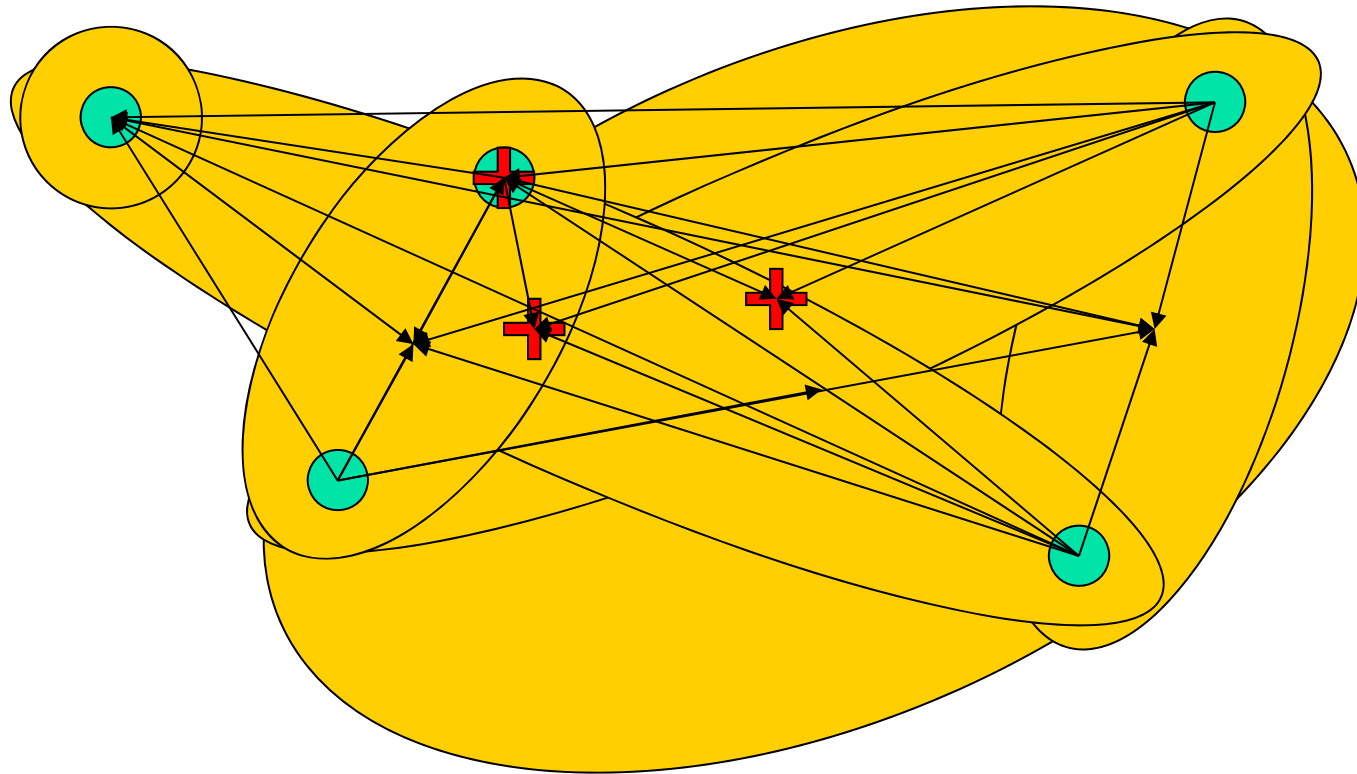
Partitioning Algorithms: Basic Concept

- Partitioning method: Construct a partition of a database D of n objects into a set of k clusters
- Given a k , find a partition of k clusters that optimizes the chosen partitioning criterion
 - **Global optimal**: exhaustively enumerate all partitions
 - **Heuristic methods**: k-means and k-medoids algorithms
 - **k-means**: Each cluster is represented by the center of the cluster
 - **k-medoids or PAM** (Partition around medoids): Each cluster is represented by one of the objects in the cluster

The K-Means Clustering Method

- Given k , the k-means algorithm is implemented in 4 steps:
 1. Partition objects into k nonempty subsets
 2. Compute **centroids** of the clusters of the current partition. The centroid is the center (mean point) of the cluster.
 3. **Assign** each object to the cluster with the **nearest centroid**.
 4. Go back to Step 2; **stop when no more new assignment**.

K-means clustering (k=3)

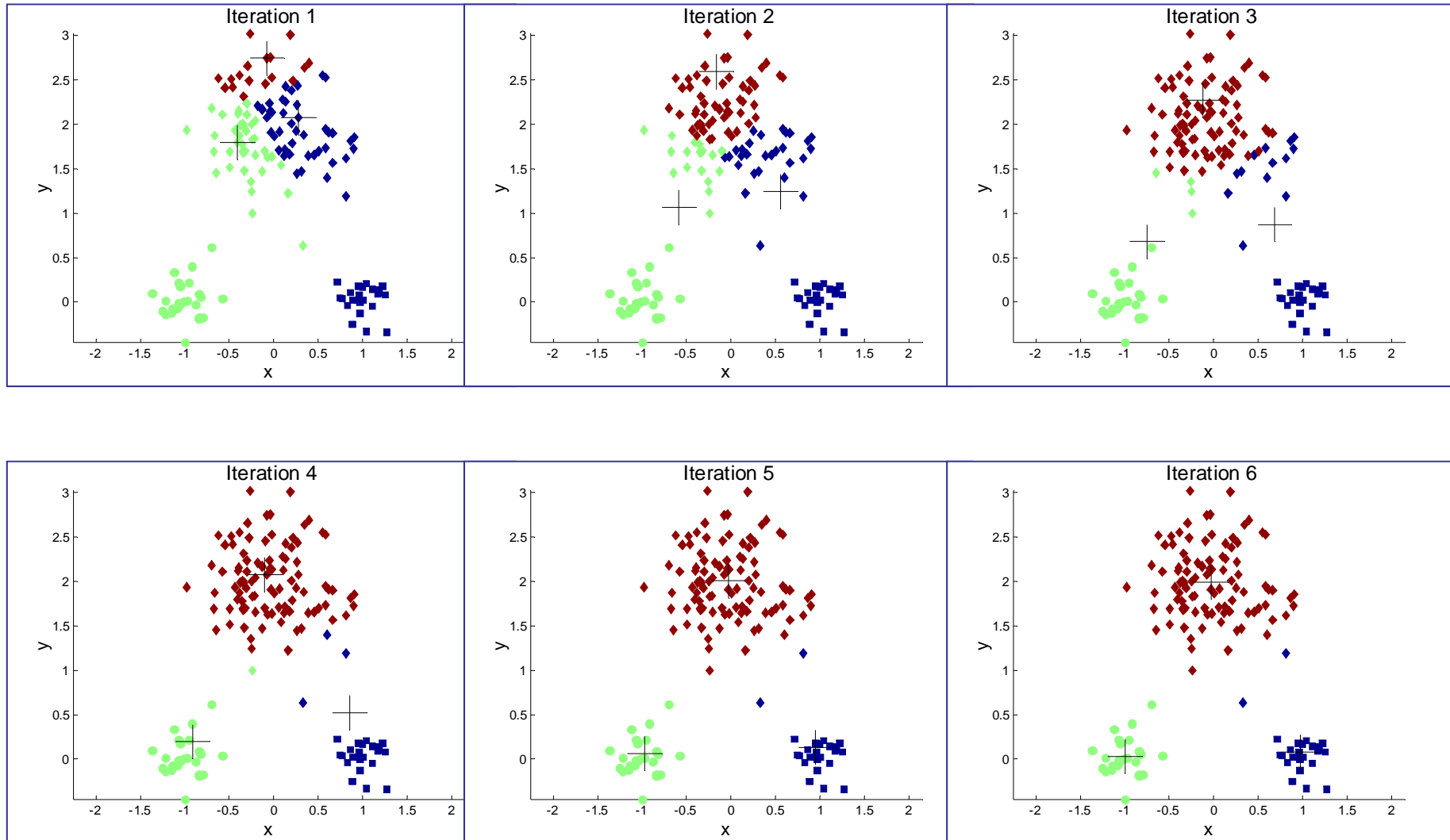


Comments on the K-Means Method

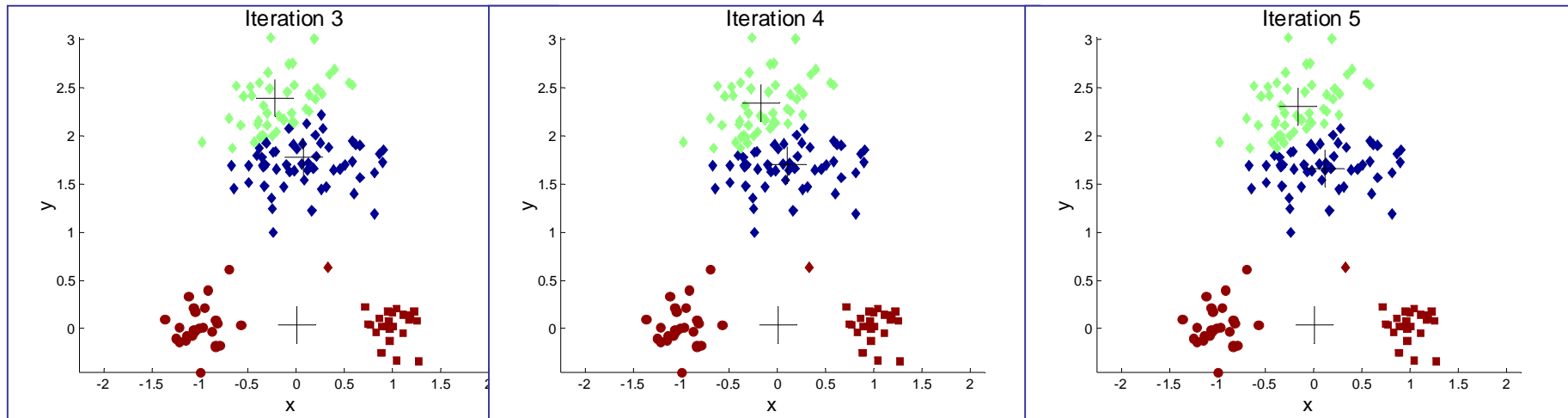
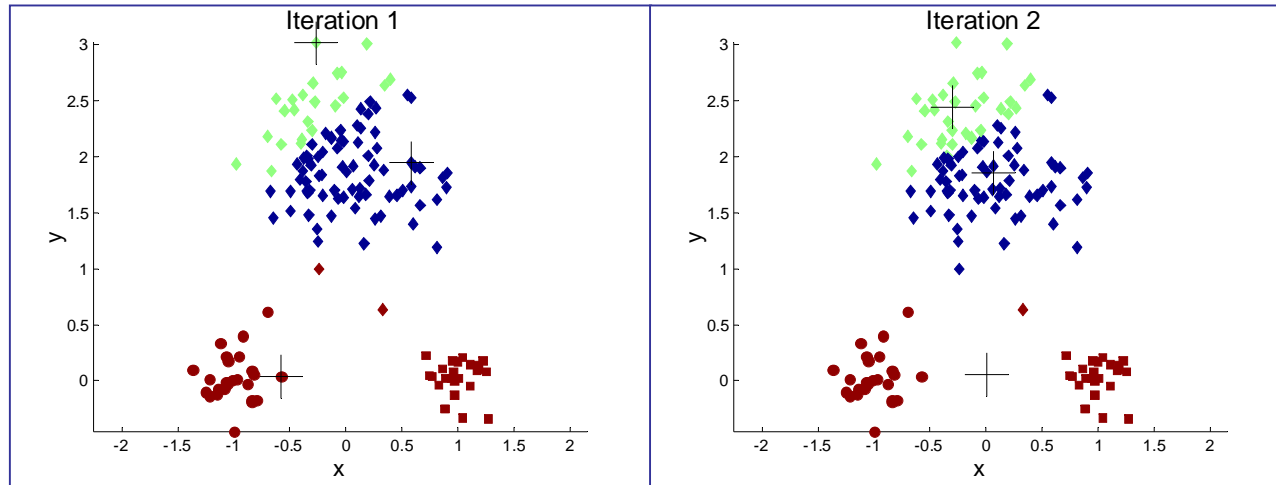
■ Strengths & Weaknesses

- Relatively efficient: $O(tkn)$, where n is # objects, k is # clusters, and t is # iterations. Normally, $k, t \ll n$.
- Often terminates at a local optimum
- Applicable only when mean is defined
- Need to specify k , the number of clusters, in advance
- Sensitive to noise and outliers as a small number of such points can influence the mean value
- Not suitable to discover clusters with non-convex shapes

Importance of Choosing Initial Centroids

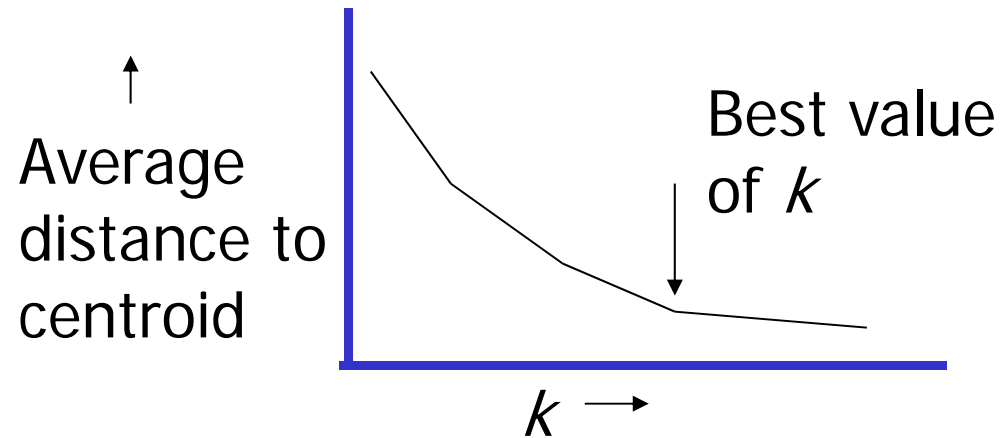


Importance of Choosing Initial Centroids



Getting k Right

- Try different k , looking at the change in the average distance to centroid, as k increases.
- Average falls rapidly until right k , then changes little.



Example

file:

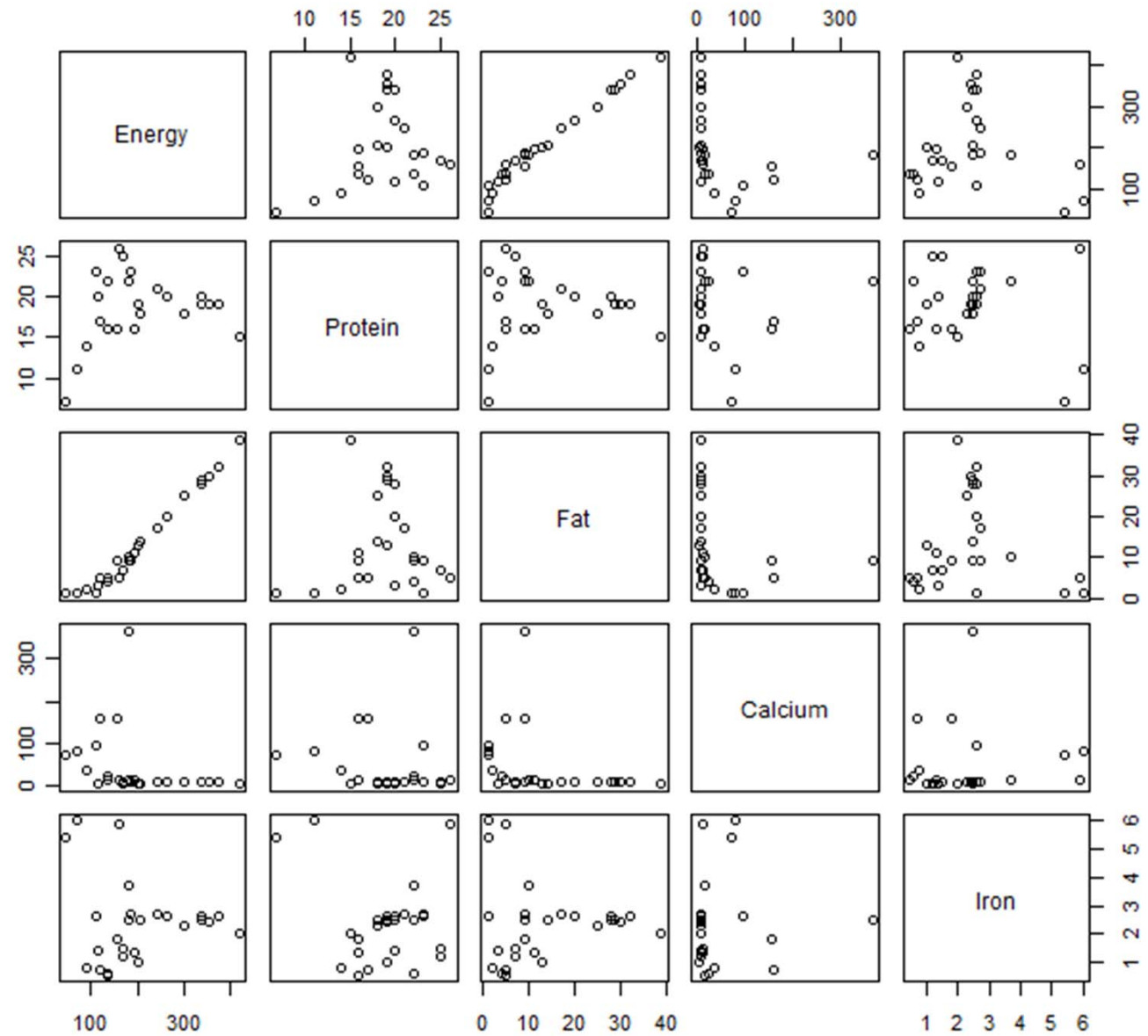
<http://orion.math.iastate.edu/burkardt/data/hartigan/file06.txt>

R instructions

```
> data <- read.table  
("nutrients_levels.txt",  
header = TRUE, sep = "\t")
```

| Name | Energy | Protein | Fat | Calcium | Iron |
|---------------------|--------|---------|-----|---------|------|
| Braised beef | 340 | 20 | 28 | 9 | 2.6 |
| Hamburger | 245 | 21 | 17 | 9 | 2.7 |
| Roast beef | 420 | 15 | 39 | 7 | 2 |
| Beefsteak | 375 | 19 | 32 | 9 | 2.6 |
| Canned beef | 180 | 22 | 10 | 17 | 3.7 |
| Broiled chicken | 115 | 20 | 3 | 8 | 1.4 |
| Canned chicken | 170 | 25 | 7 | 12 | 1.5 |
| Beef heart | 160 | 26 | 5 | 14 | 5.9 |
| Roast lamb leg | 265 | 20 | 20 | 9 | 2.6 |
| Roast lamb shoulder | 300 | 18 | 25 | 9 | 2.3 |
| Smoked ham | 340 | 20 | 28 | 9 | 2.5 |
| Pork roast | 340 | 19 | 29 | 9 | 2.5 |
| Pork simmered | 355 | 19 | 30 | 9 | 2.4 |
| Beef tongue | 205 | 18 | 14 | 7 | 2.5 |
| Veal cutlet | 185 | 23 | 9 | 9 | 2.7 |
| Baked bluefish | 135 | 22 | 4 | 25 | 0.6 |
| Raw clams | 70 | 11 | 1 | 82 | 6 |
| Canned clams | 45 | 7 | 1 | 74 | 5.4 |
| Canned crabmeat | 90 | 14 | 2 | 38 | 0.8 |
| Fried haddock | 135 | 16 | 5 | 15 | 0.5 |
| Broiled mackerel | 200 | 19 | 13 | 5 | 1 |
| Canned mackerel | 155 | 16 | 9 | 157 | 1.8 |
| Fried perch | 195 | 16 | 11 | 14 | 1.3 |
| Canned salmon | 120 | 17 | 5 | 159 | 0.7 |
| Canned sardines | 180 | 22 | 9 | 367 | 2.5 |
| Canned tuna | 170 | 25 | 7 | 7 | 1.2 |
| Canned shrimp | 110 | 23 | 1 | 98 | 2.6 |

Example (cont.)



> pairs(data[,2:6])

Example (cont.)

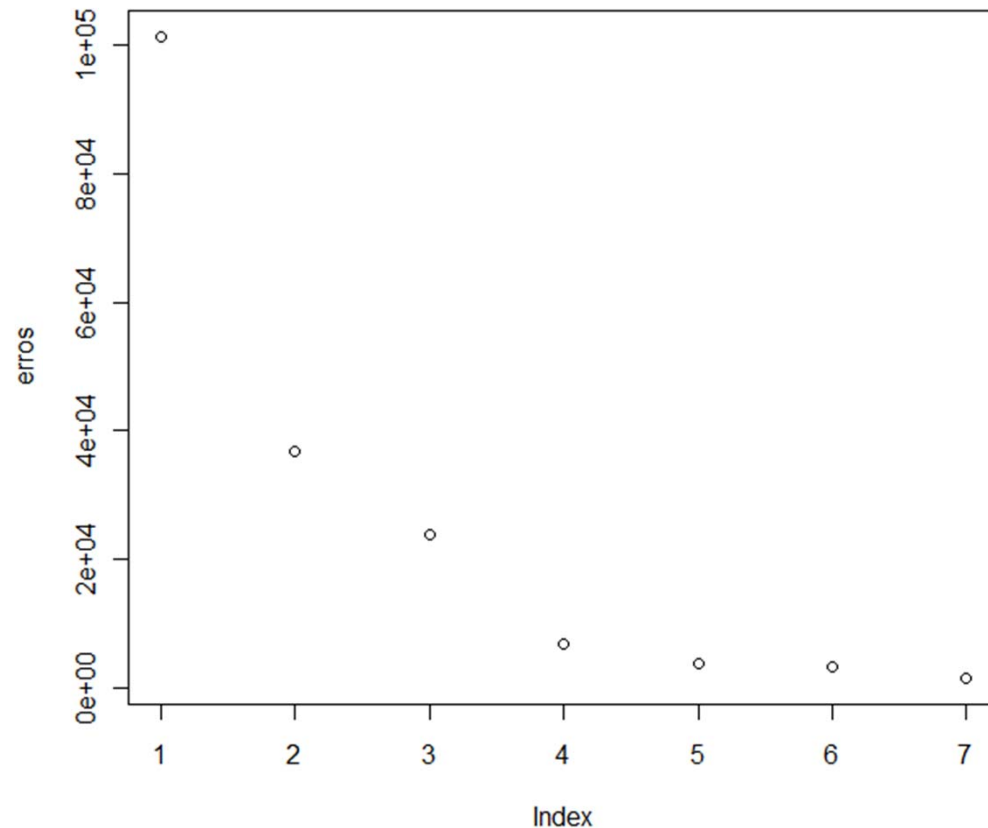
```
> cl3 <- kmeans(data[,2:6],centers=3)
```

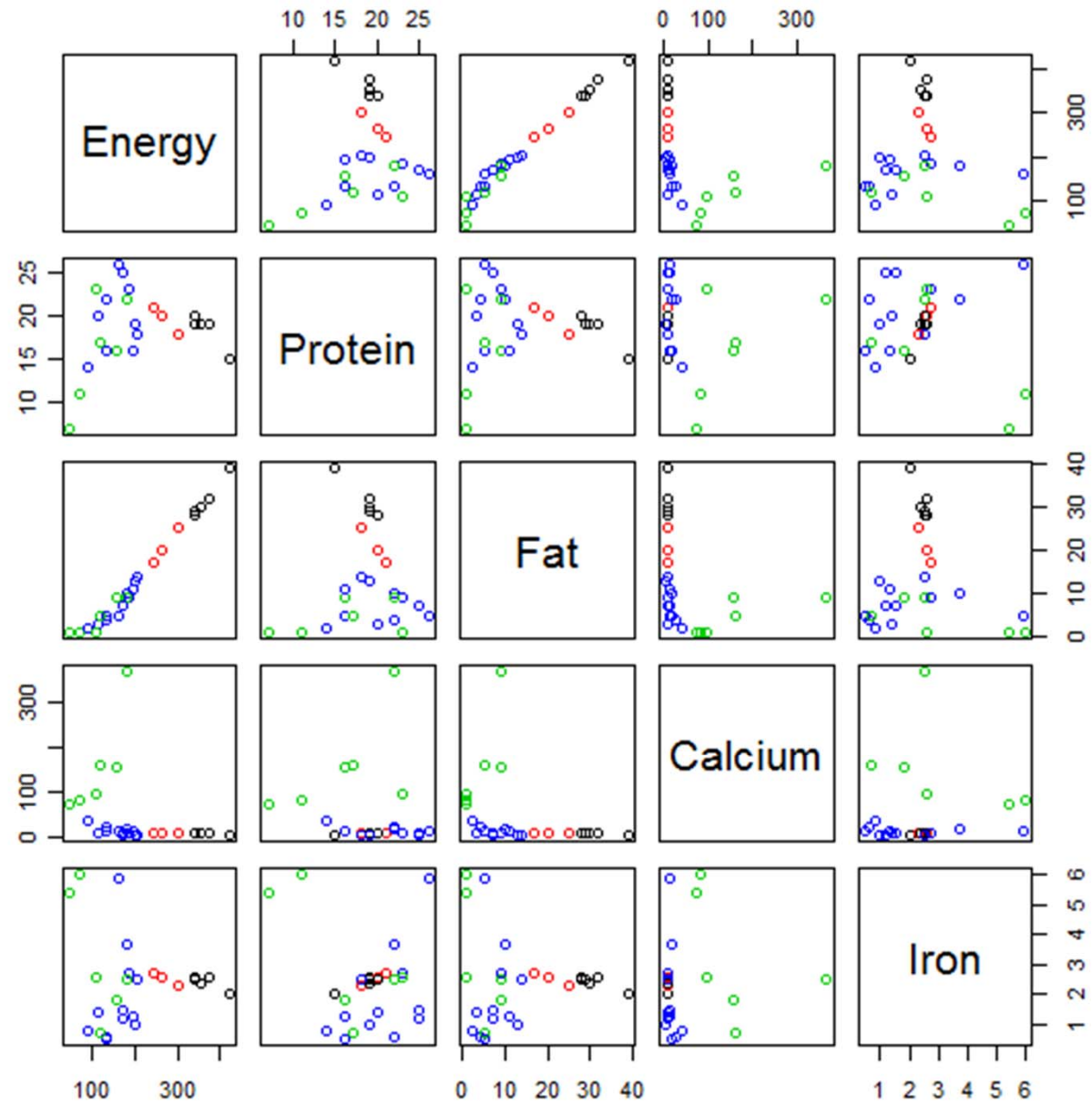
```
...
```

```
> cl8 <- kmeans(data[,2:6],centers=8)
```

```
> erros <-  
c(mean(cl2$withinss),  
  mean(cl3$withinss),  
  mean(cl4$withinss),  
  mean(cl5$withinss),  
  mean(cl6$withinss),  
  mean(cl7$withinss),  
  mean(cl8$withinss))
```

```
> plot(erros)
```





> plot(data[,2:6], col = cl4\$cluster)

| Name | Energy | Protein | Fat | Calcium | Iron | cl4\$cluster |
|---------------------|--------|---------|-----|---------|------|--------------|
| Braised beef | 340 | 20 | 28 | 9 | 2.6 | 1 |
| Roast beef | 420 | 15 | 39 | 7 | 2 | 1 |
| Beefsteak | 375 | 19 | 32 | 9 | 2.6 | 1 |
| Smoked ham | 340 | 20 | 28 | 9 | 2.5 | 1 |
| Pork roast | 340 | 19 | 29 | 9 | 2.5 | 1 |
| Pork simmered | 355 | 19 | 30 | 9 | 2.4 | 1 |
| Hamburger | 245 | 21 | 17 | 9 | 2.7 | 2 |
| Roast lamb leg | 265 | 20 | 20 | 9 | 2.6 | 2 |
| Roast lamb shoulder | 300 | 18 | 25 | 9 | 2.3 | 2 |
| Raw clams | 70 | 11 | 1 | 82 | 6 | 3 |
| Canned clams | 45 | 7 | 1 | 74 | 5.4 | 3 |
| Canned mackerel | 155 | 16 | 9 | 157 | 1.8 | 3 |
| Canned salmon | 120 | 17 | 5 | 159 | 0.7 | 3 |
| Canned sardines | 180 | 22 | 9 | 367 | 2.5 | 3 |
| Canned shrimp | 110 | 23 | 1 | 98 | 2.6 | 3 |
| Canned beef | 180 | 22 | 10 | 17 | 3.7 | 4 |
| Broiled chicken | 115 | 20 | 3 | 8 | 1.4 | 4 |
| Canned chicken | 170 | 25 | 7 | 12 | 1.5 | 4 |
| Beef heart | 160 | 26 | 5 | 14 | 5.9 | 4 |
| Beef tongue | 205 | 18 | 14 | 7 | 2.5 | 4 |
| Veal cutlet | 185 | 23 | 9 | 9 | 2.7 | 4 |
| Baked bluefish | 135 | 22 | 4 | 25 | 0.6 | 4 |
| Canned crabmeat | 90 | 14 | 2 | 38 | 0.8 | 4 |
| Fried haddock | 135 | 16 | 5 | 15 | 0.5 | 4 |
| Broiled mackerel | 200 | 19 | 13 | 5 | 1 | 4 |
| Fried perch | 195 | 16 | 11 | 14 | 1.3 | 4 |
| Canned tuna | 170 | 25 | 7 | 7 | 1.2 | 4 |

```

> data_w_cl4<-
cbind(data,cl4$cluster)
> data_w_cl4

```

| Name | Energy | Protein | Fat | Calcium | Iron | cl3\$cluster |
|---------------------|--------|---------|-----|---------|------|--------------|
| Braised beef | 340 | 20 | 28 | 9 | 2.6 | 1 |
| Roast beef | 420 | 15 | 39 | 7 | 2 | 1 |
| Beefsteak | 375 | 19 | 32 | 9 | 2.6 | 1 |
| Roast lamb leg | 265 | 20 | 20 | 9 | 2.6 | 1 |
| Roast lamb shoulder | 300 | 18 | 25 | 9 | 2.3 | 1 |
| Smoked ham | 340 | 20 | 28 | 9 | 2.5 | 1 |
| Pork roast | 340 | 19 | 29 | 9 | 2.5 | 1 |
| Pork simmered | 355 | 19 | 30 | 9 | 2.4 | 1 |
| Hamburger | 245 | 21 | 17 | 9 | 2.7 | 2 |
| Canned beef | 180 | 22 | 10 | 17 | 3.7 | 2 |
| Broiled chicken | 115 | 20 | 3 | 8 | 1.4 | 2 |
| Canned chicken | 170 | 25 | 7 | 12 | 1.5 | 2 |
| Beef heart | 160 | 26 | 5 | 14 | 5.9 | 2 |
| Beef tongue | 205 | 18 | 14 | 7 | 2.5 | 2 |
| Veal cutlet | 185 | 23 | 9 | 9 | 2.7 | 2 |
| Baked bluefish | 135 | 22 | 4 | 25 | 0.6 | 2 |
| Canned crabmeat | 90 | 14 | 2 | 38 | 0.8 | 2 |
| Fried haddock | 135 | 16 | 5 | 15 | 0.5 | 2 |
| Broiled mackerel | 200 | 19 | 13 | 5 | 1 | 2 |
| Fried perch | 195 | 16 | 11 | 14 | 1.3 | 2 |
| Canned tuna | 170 | 25 | 7 | 7 | 1.2 | 2 |
| Raw clams | 70 | 11 | 1 | 82 | 6 | 3 |
| Canned clams | 45 | 7 | 1 | 74 | 5.4 | 3 |
| Canned mackerel | 155 | 16 | 9 | 157 | 1.8 | 3 |
| Canned salmon | 120 | 17 | 5 | 159 | 0.7 | 3 |
| Canned sardines | 180 | 22 | 9 | 367 | 2.5 | 3 |
| Canned shrimp | 110 | 23 | 1 | 98 | 2.6 | 3 |

```

> data_w_cl3<-
cbind(data,cl3$cluster)
> data_w_cl3

```

Davies-Bouldin Validity Index

R Package 'clusterSim' – index.DB

Let C_i be a cluster of vector. Let X_i be a vector on C_i and S_i be a measure of scatter within the cluster

$$S_i = \sqrt[q]{\frac{1}{T_i} \sum_{j=1}^{T_i} |X_j - A_i|^q}$$

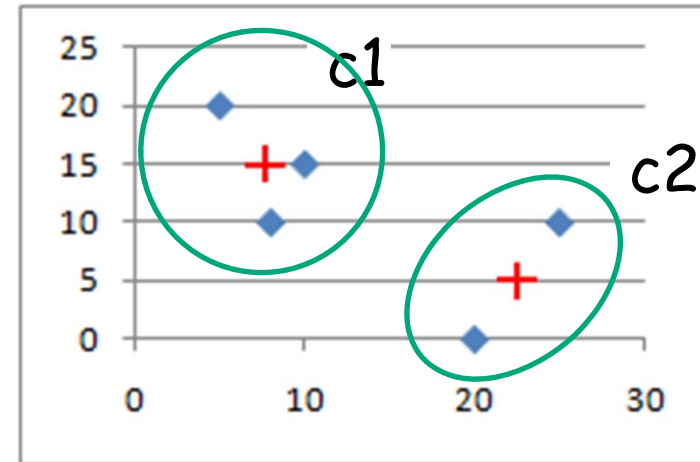
Let A_i be the centroid of C_i and $M_{i,j}$ a measure of separation between cluster C_i and C_j

$$M_{i,j} = \sqrt[p]{\sum_{k=1}^N |a_{k,i} - a_{k,j}|^p} \quad R_{i,j} = \frac{S_i + S_j}{M_{i,j}} \quad R_i = \max_{j:i \neq j} R_{i,j}$$
$$DB = \frac{1}{N} \sum_{i=1}^N R_i$$

DB should be small for good clustering

Example

| X | Y | Cluster |
|----|----|---------|
| 8 | 10 | C1 |
| 5 | 20 | C1 |
| 20 | 0 | C2 |
| 10 | 15 | C1 |
| 25 | 10 | C2 |



$$C1 \begin{cases} a_{1x} = \frac{8 + 5 + 10}{3} = 7.67 \\ a_{1y} = \frac{10 + 20 + 15}{3} = 15 \end{cases}$$

$$C2 \begin{cases} a_{2x} = 22.5 \\ a_{2y} = 5 \end{cases}$$

$$S_1 = \sqrt{\frac{1}{3} \left((8 - 7.67)^2 + (10 - 15)^2 + \dots + (15 - 15)^2 \right)} = 4.57$$

$$S_2 = \sqrt{\frac{1}{2} \left((20 - 22.5)^2 + (0 - 5)^2 + \dots \right)} = 5.59$$

$$M_{1,2} = \sqrt{(7.67 - 22.5)^2 + (15 - 5)^2} = 17.89$$

$$R_{1,2} = R_{2,1} = \frac{4.57 + 5.59}{17.89} = 0.568$$

$$DB = \frac{1}{2} (0.568 + 0.568) = 0.568$$

Silhouette Validation Method

R Package 'clusterSim' – index.DB

For each point i , let $a(i)$ be the average dissimilarity of i with all other points within the same cluster.

Any measure of similarity can be used (e.g. Euclidian distance)

Then, find the average dissimilarity of i with the data on another single cluster. Repeat this for every other cluster.

$b(i)$ is the lowest average similarity of i to any other cluster.

We can now define for each point i

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

$0 \leq s(i) \leq 1$ and an $s(i)$ close to one means that the datum is appropriately clustered. The average $s(i)$ of a cluster is a measure of how tightly grouped the data is in the cluster.

Therefore, the average $s(i)$ of the entire dataset is a measure of how appropriately the data has been clustered.

Hartigan index

R Package 'clusterSim' – index.DB

Assume a dataset with N samples X_1, \dots, X_N . Each sample with M dimensions

For k clusters the overall fitness can be expressed as the square of error for all samples, where d is the distance between sample X_j and the centroid X_{ci} of its cluster.

$$err(k) = \sum_{i=1}^k \sum_{j=1, j \in C_i}^N (d(X_j, X_{ci}))^2$$

The Hartigan index $H(k)$ is defined as follows

$$H(k) = (n - k - 1) \frac{err(k) - err(k + 1)}{err(k + 1)}$$

The multiplier correction term of $(N-k-1)$ is a penalty factor for large number of cluster partitioning. The optimal k number is the one that maximizes the $H(k)$.

Some links

HARTIGAN - Data for Clustering Algorithms

<http://orion.math.iastate.edu/burkardt/data/hartigan/hartigan.html>

R packages

cluster - <http://cran.r-project.org/web/packages/cluster/index.html>

clusterSim - <http://cran.r-project.org/web/packages/clusterSim/index.html>

References

- **Jiawei Han and Micheline Kamber**, “Data Mining: Concepts and Techniques”, 2 edition (4 Jun 2006),
- **Gordon S. Linoff and Michael J. Berry**, “Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management”, 3rd Edition edition (1 April 2011)