Distance and dissimilarities

Contents


```
knitr::opts_chunk$set(echo = TRUE)
install.packages("dplyr")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
install.packages("stargazer")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```
install.packages("ade4")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

Definition of a distance

- A distance function or a metric on $\mathbb{R}^n$, $n \geq 1$, is a function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.
- A distance function must satisfy some required properties or axioms.
- There are three main axioms.
- A1. $d(\mathbf{x}, \mathbf{y}) = 0 \iff \mathbf{x} = \mathbf{y}$ (identity of indiscernibles);
- A2. $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ (symmetry);
- A3. $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ (triangle inequality), where $\mathbf{x} = (x_1, \cdots, x_n)$, $\mathbf{y} = (y_1, \cdots, y_n)$ and $\mathbf{z} = (z_1, \cdots, z_n)$ are all vectors of $\mathbb{R}^n$.
- We should use the term *dissimilarity* rather than *distance* when not all the three axioms A1-A3 are valid.
- Most of the time, we shall use, with some abuse of vocabulary, the term distance.

Exercice 1

- Prove that the three axioms A1-A3 imply the non-negativity condition:

$$d(\mathbf{x}, \mathbf{y}) \geq 0.$$

Euclidean distance

- It is defined by:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

* A1-A2 ae obvious. * The proof of A3 is provided below. # Exercice 2 * Is the squared Euclidian distance a true distance?

Manhattan distance

- The Manhattan distance also called taxi-cab metric or city-block metric is defined by:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} |x_i - y_i|.$$

- A1-A2 hold.
- A3 also holds using the fact that $|a + b| \leq |a| + |b|$ for any reals a, b.
- There exists also a weighted version of the Manhattan distance called the Canberra distance.

Manhattan distance vs Euclidean distance Graph

```
x = c(0, 0)
y = c(6,6)
dist(rbind(x, y), method = "euclidian")
```

```
##          x
## y 8.485281
```

```r
6*sqrt(2)
```

```
## [1] 8.485281
```

```r
dist(rbind(x, y), method = "manhattan")
```

```
##    x
## y 12
```

Canberra distance

- It is defined by:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} \frac{|x_i - y_i|}{|x_i| + |y_i|}.$$

- Note that the term $|x_i - y_i|/(|x_i| + |y_i|)$ is not properly defined when $x_i = y_i = 0$.
- By convention we set the ratio to be zero in that case.
- The Canberra distance is specially sensitive to small changes near zero.

```r
x = c(0, 0)
y = c(6,6)
dist(rbind(x, y), method = "canberra")
```

```
##   x
## y 2
```

```r
6/6+6/6
```

```
## [1] 2
```

Exercice 3

- Prove that the Canberra distance is a true distance.

Minkowski distance

- Both the Euclidian and the Manattan distances are special cases of the Minkowski distance which is defined, for $p \geq 1$, by:

$$d(\mathbf{x}, \mathbf{y}) = \left[\sum_{i=1} |x_i - y_i|^p \right]^{1/p}.$$

- For $p = 1$, we get the Manhattan distance.
- For $p = 2$, we get the Euclidian distance.
- Let us also define:

$$\|\mathbf{x}\|_p \equiv \left[\sum_{i=1}^{n} |x_i|^p \right]^{1/p},$$

  where $\|\cdot\|_p$ is known as the p-norm or Minkowski norm.
- Note that the Minkowski distance and norm are related by:

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_p.$$

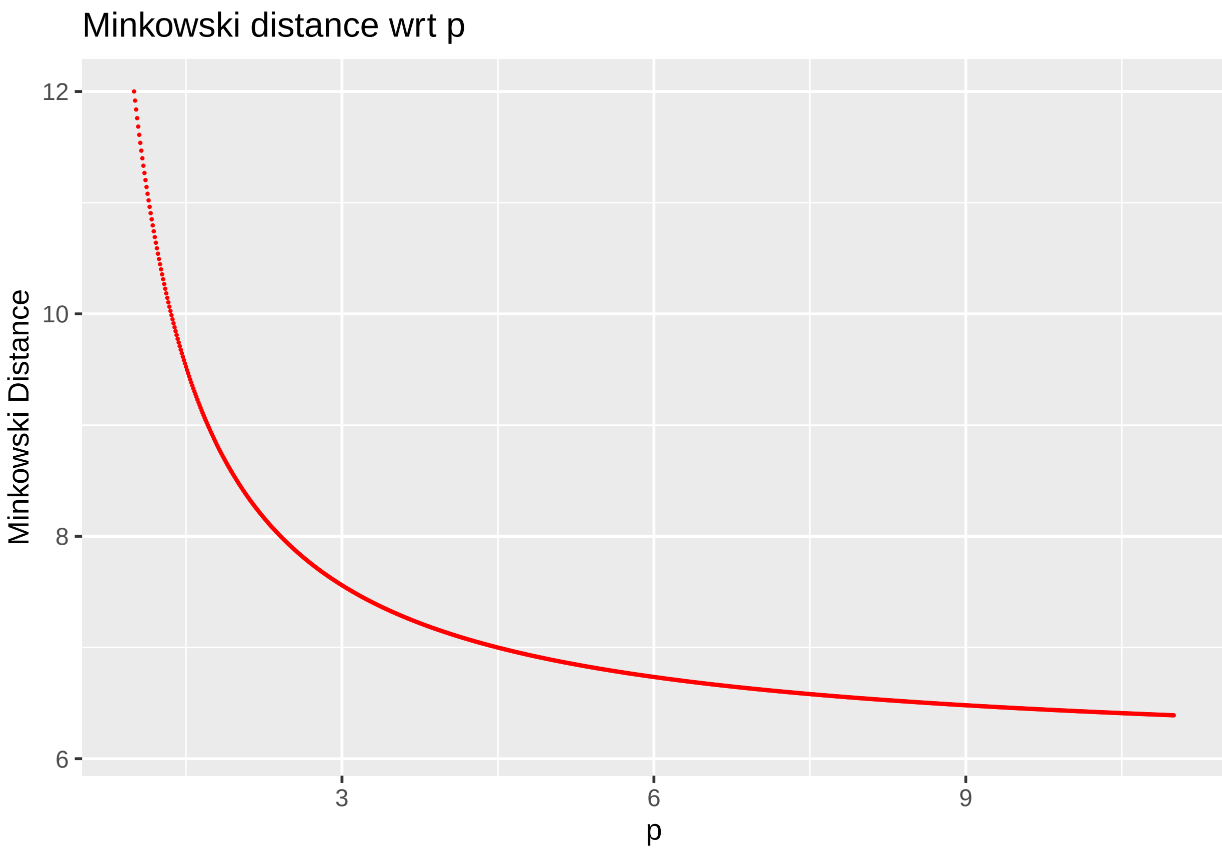
- Conversely, we have:

$$\|\mathbf{x}\|_p = d(\mathbf{x}, \mathbf{0}),$$

where $\mathbf{0}$ is the null-vetor of $\mathbb{R}^n$.

```
library("ggplot2")
x = c(0, 0)
y = c(6,6)
MinkowDist=c()
for (p in seq(1,30,.01))
{
MinkowDist=c(MinkowDist,dist(rbind(x, y), method = "minkowski", p = p))
}
ggplot(data =data.frame(x = seq(1,30,.01), y=MinkowDist ) , mapping = aes(x = x, y = y))+geom_point(siz
```

```
## Warning: Removed 1900 rows containing missing values (geom_point).
```



Chebyshev distance

- At the limit, we get the Chebyshev distance which is defined by:

$$d(\mathbf{x}, \mathbf{y}) = \max_{i=1,\cdots,n}(|x_i - y_i|) = \lim_{p\to\infty}\left[\sum_{i=1}|x_i - y_i|^p\right]^{1/p}.$$

- The corresponding norm is:

$$\|\mathbf{x}\|_\infty = \max_{i=1,\cdots,n}(|x_i|).$$

Minkowski inequality

- The proof of the triangular inequality A3 is based on the Minkowski inequality:
- For any nonnegative real numbers $a_1, \cdots, a_n$; $b_1, \cdots, b_n$, and for any $p \geq 1$, we have:

$$\left[\sum_{i=1}^{n}(a_i+b_i)^p\right]^{1/p} \leq \left[\sum_{i=1}^{n} a_i^p\right]^{1/p} + \left[\sum_{i=1}^{n} b_i^p\right]^{1/p}.$$

- To prove that the Minkowski distance satisfies A3, notice that

$$\sum_{i=1}^{n} |x_i - z_i|^p = \sum_{i=1}^{n} |(x_i - y_i) + (y_i - z_i)|^p.$$

- Since for any reals x, y, we have: $|x+y| \leq |x| + |y|$, and using the fact that x^p is increasing in $x \geq 0$, we obtain:

$$\sum_{i=1}^{n} |x_i - z_i|^p \leq \sum_{i=1}^{n} (|x_i - y_i| + |y_i - z_i|)^p.$$

- Applying the Minkowski inequality with $a_i = |x_i - y_i|$ and $b_i = |y_i - z_i|$, $i = 1, \cdots, n$, we get:

$$\sum_{i=1}^{n} |x_i - z_i|^p \leq \left(\sum_{i=1}^{n} |x_i - y_i|^p\right)^{1/p} + \left(\sum_{i=1}^{n} |y_i - z_i|^p\right)^{1/p}.$$

Hölder inequality

- The proof of the Minkowski inequality itself requires the Hölder inequality:
- For any nonnegative real numbers $a_1, \cdots, a_n$; $b_1, \cdots, b_n$, and any $p, q > 1$ with $1/p + 1/q = 1$, we have:

$$\sum_{i=1}^{n} a_i b_i \leq \left[\sum_{i=1}^{n} a_i^p\right]^{1/p} \left[\sum_{i=1}^{n} b_i^q\right]^{1/q}$$

- The proof of the Hölder inequality relies on the Young inequality:
- For any $a, b > 0$, we have

$$ab \leq \frac{a^p}{p} + \frac{b^q}{q},$$

  with equality occuring iff: $a^p = b^q$.
- To prove the Young inequality, one can use the (strict) convexity of the exponential function.
- For any reals x, y, we have:

$$e^{\frac{x}{p} + \frac{y}{q}} \leq \frac{e^x}{p} + \frac{e^y}{q}.$$

- We then set: $x = p \ln a$ and $y = q \ln b$ to get the Young inequality.
- A good reference on inequalities is: Z. Cvetkovski, Inequalities: theorems, techniques and selected problems, 2012, Springer Science & Business Media. # Cauchy-Schwartz inequality
- Note that the triangular inequality for the Minkowski distance implies:

$$\sum_{i=1}^{n} |x_i| \leq \left[\sum_{i=1}^{n} |x_i|^p\right]^{1/p}.$$

- Note that for $p = 2$, we have $q = 2$. The Hölder inequality implies for that special case

$$\sum_{i=1}^{n} |x_i y_i| \leq \sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}.$$

- Since the LHS od thes above inequality is greater then $|\sum_{i=1}^{n} x_i y_i|$, we get the Cauchy-Schwartz inequality

$$|\sum_{i=1}^{n} x_i y_i| \leq \sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2}.$$

* Using the dot product notation called also scalar product noation: $\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} x_i y_i$, and the norm notation $\|\cdot\|_2\|$, the Cauchy-Schwart inequality is:

$$|\mathbf{x} \cdot \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2.$$

Pearson correlation distance

- The Pearson correlation coefficient is a similarity measure on $\mathbb{R}^n$ defined by:

$$\rho(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} (x_i - \bar{\mathbf{x}})(y_i - \bar{\mathbf{y}})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{\mathbf{x}})^2 \sum_{i=1}^{n} (y_i - \bar{\mathbf{y}})^2}},$$

  where $\bar{\mathbf{x}}$ is the mean of the vector $\mathbf{x}$ defined by:

$$\bar{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} x_i,$$

- Note that the Pearson correlation coefficient satisfies P2 and is invariant to any positive linear transformation, i.e.:

$$\rho(\alpha \mathbf{x}, \mathbf{y}) = \rho(\mathbf{x}, \mathbf{y}),$$

  for any $\alpha > 0$.

- The Pearson distance (or correlation distance) is defined by:

$$d(\mathbf{x}, \mathbf{y}) = 1 - \rho(\mathbf{x}, \mathbf{y}).$$

- Note that the Pearson distance does not satisfy A1 since $d(\mathbf{x}, \mathbf{x}) = 0$ for any non-zero vector $\mathbf{x}$. It neither satisfies the triangle inequality. However, the symmetry property is fullfilled.

Cosine correlation distance

- The cosine of the angle θ between two vectors $\mathbf{x}$ and $\mathbf{y}$ is a measure of similarity given by:

$$\cos(\theta) = \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\|_2 \|\mathbf{y}\|_2} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}}.$$

- Note that the cosine of the angle between the two centred vectors $\mathbf{x} - \bar{\mathbf{x}}\mathbf{1}$ and $\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}$ coincides with the Pearson correlation coefficient of $\mathbf{x}$ and $\mathbf{y}$, where $\mathbf{1}$ is a vector of units of $\mathbb{R}^n$.
- The cosine correlation distance is defined by:

$$d(\mathbf{x}, \mathbf{y}) = 1 - \cos(\theta).$$

- It shares similar properties than the Pearson correlation distance. Likewise, Axioms A1 and A3 are not satisfied.

Spearman correlation distance

- To calculate the Spearman's rank-order correlation, we need to map seperately each of the vectors to ranked data values:
$$\mathbf{x} \rightarrow \text{rank}(\mathbf{x}) = (x_1^r, \cdots, x_n^r).$$
- Here, x_i^r is the rank of x_i among the set of values of $\mathbf{x}$.
- We illustrate this transformation with a simple example:
- If $\mathbf{x} = (3, 1, 4, 15, 92)$, then the rank-order vector is $\text{rank}(\mathbf{x}) = (2, 1, 3, 4, 5)$.

```
x=c(3, 1, 4, 15, 92)
rank(x)
```

```
## [1] 2 1 3 4 5
```

- The Spearman's rank correlation of two numerical variables $\mathbf{x}$ and $\mathbf{y}$ is simply the Pearson correlation of the two correspnding rank-order variables $\text{rank}(\mathbf{x})$ and $\text{rank}(\mathbf{y})$, i.e. $\rho(\text{rank}(\mathbf{x}), \text{rank}(\mathbf{y}))$. This measure is is useful because it is more robust against outliers than the Pearson correlation.
- If all the n ranks are distinct, it can be computed using the following formula:
$$\rho(\text{rank}(\mathbf{x}), \text{rank}(\mathbf{y})) = 1 - \frac{6\sum_{i=1}^n d_i^2}{n(n^2-1)},$$
where $d_i = x_i^r - y_i^r$, $i = 1, \cdots, n$.
- The spearman distance is then defined by:
$$d(\mathbf{x}, \mathbf{y}) = 1 - \rho(\text{rank}(\mathbf{x}), \text{rank}(\mathbf{y})).$$
- It can be shown that easaly that it is not a proper distance.
- If all the n ranks are distinct, we get:
$$d(\mathbf{x}, \mathbf{y}) = \frac{6\sum_{i=1}^n d_i^2}{n(n^2-1)}.$$

```
x=c(3, 1, 4, 15, 92)
rank(x)
```

```
## [1] 2 1 3 4 5
```

```
y=c(30,2 , 9, 20, 48)
rank(y)
```

```
## [1] 4 1 2 3 5
```

```
d=rank(x)-rank(y)
d
```

```
## [1] -2  0  1  1  0
```

```
cor(rank(x),rank(y))
```

```
## [1] 0.7
```

```
1-6*sum(d^2)/(5*(5^2-1))
```

```
## [1] 0.7
```

Kendall tau distance

- The Kendall rank correlation coefficient is calculated from the number of correspondances between the rankings of $\mathbf{x}$ and the rankings of $\mathbf{y}$.

- The number of pairs of observations among n observations or values is:

$$\binom{n}{2} = \frac{n(n-1)}{2}.$$

- The pairs of observations (x_i, x_j) and (y_i, y_j) are said to be *concordant* if:

$$\text{sign}(x_j - x_i) = \text{sign}(y_j - y_i),$$

  and to be *discordant* if:

$$\text{sign}(x_j - x_i) = -\text{sign}(y_j - y_i),$$

  where sign$(\cdot)$ returns 1 for positive numbers and -1 negative numbers and 0 otherwise.
- If $x_i = x_j$ or $y_i = y_j$ (or both), there is a tie.
- The Kendall τ coefficient is defined by (neglecting ties):

$$\tau = \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j=1}^{n} \text{sign}(x_j - x_i)\text{sign}(y_j - y_i).$$

- Let n_c (resp. n_d) be the number of concordant (resp. discordant) pairs, we have

$$\tau = \frac{2(n_c - n_d)}{n(n-1)}.$$

- The Kendall tau distance is then:

$$d(\mathbf{x}, \mathbf{y}) = 1 - \tau.$$

- Remark: the triangular inequality may fail in cases where there are ties.

```
x=c(3, 1, 4, 15, 92)
y=c(30,2 , 9, 20, 48)
tau=0
for (i in 1:5)
{
tau=tau+sign(x -x[i])%*%sign(y -y[i])
}
tau=tau/(5*4)
tau
```

```
##      [,1]
## [1,]  0.6
```

```
cor(x,y, method="kendall")
```

```
## [1] 0.6
```

Variables standardization

- Variables are often standardized before measuring dissimilarities.
- Standardization converts the original variables into uniteless variables.
- A well known method is the z-score transformation:

$$\mathbf{x} \rightarrow (\frac{x_1 - \bar{\mathbf{x}}}{s_{\mathbf{x}}}, \cdots, \frac{x_n - \bar{\mathbf{x}}}{s_{\mathbf{x}}}),$$

  where $s_{\mathbf{x}}$ is the sample standard deviation given by:

$$s_{\mathbf{x}} = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{\mathbf{x}})^2.$$

- The transformed variable will have a mean of 0 and a variance of 1.
- The result obtained with Pearson correlation measures and standardized Euclidean distances are comparable.
- For other methods, see: Milligan, G. W., & Cooper, M. C. (1988). A study of standardization of variables in cluster analysis. *Journal of classification*, *5*(2), 181-204.

```
x=c(3, 1, 4, 15, 92)
y=c(30,2 , 9, 20, 48)
(x-mean(x))/sd(x)
```

```
## [1] -0.5134116 -0.5647527 -0.4877410 -0.2053646  1.7712699
```

```
scale(x)
```

```
##            [,1]
## [1,] -0.5134116
## [2,] -0.5647527
## [3,] -0.4877410
## [4,] -0.2053646
## [5,]  1.7712699
## attr(,"scaled:center")
## [1] 23
## attr(,"scaled:scale")
## [1] 38.9551
```

```
(y-mean(y))/sd(y)
```

```
## [1]  0.45263128 -1.09293895 -0.70654639 -0.09935809  1.44621214
```

```
scale(y)
```

```
##             [,1]
## [1,]  0.45263128
## [2,] -1.09293895
## [3,] -0.70654639
## [4,] -0.09935809
## [5,]  1.44621214
## attr(,"scaled:center")
## [1] 21.8
## attr(,"scaled:scale")
## [1] 18.11629
```

Similarity measures for binary data

- A common simple situation occurs when all information is of the presence/absence of 2-level qualitative characters.
- We assume there are n characters.
- *The presence of the character is coded by 1 and the absence by 0.
- We have have at our disposal two vectors.
- $\mathbf{x}$ is observed for a first individual (or object).
- $\mathbf{y}$ is observed for a second individual.
- We can then calculate the following four statistics:

$a = \mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} x_i y_i$.

$b = \mathbf{x} \cdot (\mathbf{1} - \mathbf{y}) = \sum_{i=1}^{n} x_i (1 - y_i)$.

$c = (\mathbf{1} - \mathbf{x}) \cdot \mathbf{y} = \sum_{i=1}^{n} (1 - x_i) y_i$.

$d = (\mathbf{1} - \mathbf{x}) \cdot (\mathbf{1} - \mathbf{y}) = \sum_{i=1}^{n} (1 - x_i)(1 - y_i)$.

- The counts of matches are a for $(1, 1)$ and d for $(0, 0)$;
- The counts of mismatches are b for $(1, 0)$ and c for $(0, 1)$.
- Note that obviously: $a + b + c + d = n$.
- This gives a very useful 2×2 association table.

		Second individual		
		1	0	*Totals*
First individual	1	a	b	$a + b$
	0	c	d	$c + d$
Totals		$a + c$	$b + d$	n

- The data shows 8 people (individuals) and 10 binary variables:
- Sex, Married, Fair Hair, Blue Eyes, Wears Glasses, Round Face, Pessimist, Evening Type, Is an Only Child, Left-Handed.

```
data=c(
1,0,1,1,0,0,1,0,0,0,
0,1,0,0,1,0,0,0,0,0,
0,0,1,0,0,0,1,0,0,1,
0,1,0,0,0,0,0,1,1,0,
1,1,0,0,1,1,0,1,1,0,
1,1,0,0,1,0,1,1,0,0,
0,0,0,1,0,1,0,0,0,0,
0,0,0,1,0,1,0,0,0,0
)
data=data.frame(matrix(data, nrow=8,byrow=T))
row.names(data)=c("Ilan","Jacqueline","Kim","Lieve","Leon","Peter","Talia","Tina")
names(data)=c("Sex", "Married", "Fair Hair", "Blue Eyes", "Wears Glasses", "Round Face", "Pessimist", "
```

- We are comparing the records for Ilan with Talia.

```
x=data["Ilan",]
y=data["Talia",]
knitr::kable(table(x, y)[2:1,2:1],"pipe")
```

	1	0
1	1	3
0	1	5

- Therefore: $a = 1,\ b = 3,\ c = 1,\ d = 5$.
- Note that interchanging Ilan and Talia would permute b and c while leaving a and d unchanged.
- A good similarity or dissimilarity coefficient must treat b and c symmetrically.
- A similarity measure is denoted by: $s(\mathbf{x}, \mathbf{y})$.
- The corresponding distance is then defined as:

$$d(\mathbf{x}, \mathbf{y}) = 1 - s(\mathbf{x}, \mathbf{y}).$$

Table 9 Binary Variables for Eight People

Person	Sex (Male = 1, Female = 0)	Married (Yes = 1, No = 0)	Fair Hair = 1, Dark Hair = 0	Blue Eyes = 1, Brown Eyes = 0	Wears Glasses (Yes = 1, No = 0)	Round Face = 1, Oval Face = 0	Pessimist = 1, Optimist = 0	Evening Type = 1, Morning Type = 0	Is an Only Child (Yes = 1, No = 0)	Left-Handed = 1, Right-Handed = 0
Ilan	1	0	1	1	0	0	1	0	0	0
Jacqueline	0	1	0	0	1	0	0	0	0	0
Kim	0	0	1	0	0	0	1	0	0	1
Lieve	0	1	0	0	0	0	0	1	1	0
Leon	1	1	0	0	1	1	0	1	1	0
Peter	1	1	0	0	1	0	1	1	0	0
Talia	0	0	0	1	0	1	0	0	0	0
Tina	0	0	0	1	0	1	0	0	0	0

Figure 1: Table from Kaufman, L., & Rousseeuw, P. J. (2009). *Finding groups in data: an introduction to cluster analysis* (Vol. 344). John Wiley & Sons

- Alternatively, we have:
$$d(\mathbf{x},\mathbf{y}) = \sqrt{1 - s(\mathbf{x},\mathbf{y})}.$$
- A list of some of the similarity measures $s(\mathbf{x},\mathbf{y})$ that have been suggested for binary data is shown below.
- A more extensive list can be found in: Gower, J. C., & Legendre, P. (1986). Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification*, *3*(1), 5-48.

Coefficient	$s(\mathbf{x},\mathbf{y})$	$d(\mathbf{x},\mathbf{y}) = 1 - s(\mathbf{x},\mathbf{y})$
Simple matching	$\frac{a+d}{a+b+c+d}$	$\frac{b+c}{a+b+c+d}$
Jaccard	$\frac{a}{a+b+c}$	$\frac{b+c}{a+b+c}$
Rogers and Tanimoto (1960)	$\frac{a+d}{a+2(b+c)+d}$	$\frac{2(b+c)}{a+2(b+c)+d}$
Gower and Legendre (1986)	$\frac{2(a+d)}{2(a+d)+b+c}$	$\frac{b+c}{2(a+d)+b+c}]$
Gower and Legendre (1986)	$\frac{2a}{2a+b+c}$	$\frac{b+c}{2a+b+c}$

- To calculate these coefficients, we use the function: dist.binary().
- All the distances in this package are of type $d(\mathbf{x}.\mathbf{y}) = \sqrt{1 - s(\mathbf{x}.\mathbf{y})}$.

```r
library(ade4)
a=1
b=3
c=1
d=5
dist.binary(data[c("Ilan","Talia"),],method=2)^2
```

```
  Ilan
```

Talia 0.4

```r
1-(a+d )/(a+b+c+d)
```

[1] 0.4

```r
dist.binary(data[c("Ilan","Talia"),],method=1)^2
```

```
  Ilan
```

Talia 0.8

```r
1-a/(a+b+c)
```

[1] 0.8

```r
dist.binary(data[c("Ilan","Talia"),],method=4)^2
```

```
       Ilan
```

Talia 0.5714286

```r
1-(a+d )/(a+2*(b+c)+d)
```

[1] 0.5714286

```r
# One Gower coefficient is missing
dist.binary(data[c("Ilan","Talia"),],method=5)^2
```

```
       Ilan
```

Talia 0.6666667

```
1-2*a/(2*a+b+c)
```

[1] 0.6666667 * The reason for such a large number of possible measures has to do with the apparent uncertainty as to how to deal with the count of zero-zero matches d. * The measues embedding d are sometimes called symmetrical. * The other measues are called assymmetrical. * In some cases, of course, zero_zero matches are completely equivalent to one–one matches, and therefore should be included in the calculated similarity measure. * An example is gender, where there is no preference as to which of the two categories should be coded zero or one. * But in other cases the inclusion or otherwise of d is more problematic; for example, when the zero category corresponds to the genuine absence of some property, such as wings in a study of insects. # Exercice 4 * Prove that the distances based on the SimplemMatching coefficient and the Jaccard coefficient satisfy A3. * Prove that the distances proposed by Gower and Legendre (1986) do not satisfy A3. * Hint: Proofs and counterexamples have to be adapted from in the paper: * Gower, J. C., & Legendre, P. (1986). Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification*, *3*(1), 5-48. # Nominal variables * We studied above binary variables which can only take on two states coded as $0, 1$. * We generalize this approach to nominal variable which may take on more than two states. * Eye's color may have for example four states: blue/brown/green/grey . * Le M be the number of states and code the outcomes as $1, \cdots, M$. * We could choose $1 =$ blue, $2 =$ brown, $3 =$ green, and $4 =$ grey. * Note that these states are not ordered in any way * One strategy would be creating a new binary variable for each of the M nominal states. * Then to put it equal to 1 if the corresponding state occurs and to 0 otherwise. * After that, one could resort to one of the dissimilarity coeffi- cients of the previous subsection. * The most common way of measuring the similarity or dissimilarity between some objects i and j that are characterized through nominal variables is to use the simple matching approach: U P-U s(i, j) = - and d(i, j) = - P P (Sokal and Michener, 1958). Here, u is the number of matches, that is, the number of variables for which objects i and j happen to be in the same state. As before, p is the total number of variables (or, in a situation with missing values, the number of variables that are available for both i and j). Therefore, simple matching has exactly the same meaning as in the preced- ing section. For instance, it is invariant with respect to different codings of the variables because this does not affect the number of matches. Again it is possible to give a higher weight to u, the number of agreements, or to p - u, the number of disagreements. Such variants were considered by Rogers and Tanimoto (1960) and Sokal and Sneath (1963), corresponding to the formulas in Table 10. It must also be noted that different variables may have different values of M. Therefore, Hyvarinen (1962) assigns more weight to matches in variables with a large number of states. Lingoes (1967) extends this by counting, for all variables, the frequency with which each state actually occurs and by giving a higher weight to matches corresponding to rare states. (This is reminiscent of the treatment of asymmetric binary variables.) Some other variants can be found in Bock (1974, Section 5). We personally prefer the simple matching approach (19) because of its intuitive appeal and widespread acceptance. Simple matching dissimilarities can be computed by means of the program DAISY. It is not necessary to know the number of states for each variable because the program will itself produce an inventory with the number of states and the number of missing values. Also, the codes entered may be arbitrary real numbers, so the variables do not have to be coded in a discrete way. The main purpose of DAISY is to deliver a dissimilarity matrix which can be used by some of the clustering algorithms described in the following chapters.

Election data set of Table 15.5 (1 = Dem, 0 = Rep)**

State e1 e2 e3 e4 e5 e6 e7 e8 Alab 0 0 0 0 0 0 0 0
Alas 0 0 0 0 0 0 0 0
Ariz 0 0 0 0 1 0 0 0
Arka 0 0 0 1 1 0 0 0
Cali 0 0 0 1 1 1 1 1
Colo 0 0 0 1 0 0 0 1
Conn 0 0 0 1 1 1 1 1
Dela 0 0 0 1 1 1 1 1
DisC 1 1 1 1 1 1 1 1
Flor 0 0 0 0 1 0 0 1
Geor 1 0 0 1 0 0 0 0
Hawa 1 0 1 1 1 1 1 1

Idah 0 0 0 0 0 0 0 0
Illi 0 0 0 1 1 1 1 1
Indi 0 0 0 0 0 0 0 1
Iowa 0 0 1 1 1 1 0 1
Kans 0 0 0 0 0 0 0 0
Kent 0 0 0 1 1 0 0 0
Loui 0 0 0 1 1 0 0 0
Main 0 0 0 1 1 1 1 1
Mary 1 0 0 1 1 1 1 1
Mass 0 0 1 1 1 1 1 1
Mich 0 0 0 1 1 1 1 1
Minn 1 1 1 1 1 1 1 1
Miss 0 0 0 0 0 0 0 0
Miso 0 0 0 1 1 0 0 0
Mont 0 0 0 1 0 0 0 0
Nebr 0 0 0 0 0 0 0 0
Neva 0 0 0 1 1 0 0 1
NewH 0 0 0 1 1 0 1 1
NewJ 0 0 0 1 1 1 1 1
NewM 0 0 0 1 1 1 0 1
NewY 0 0 1 1 1 1 1 1
NorC 0 0 0 0 0 0 0 1
NorD 0 0 0 0 0 0 0 0
Ohio 0 0 0 1 1 0 0 1
Okla 0 0 0 0 0 0 0 0
Oreg 0 0 1 1 1 1 1 1
Penn 0 0 0 1 1 1 1 1
Rhod 1 0 1 1 1 1 1 1
SouC 0 0 0 0 0 0 0 0
SouD 0 0 0 0 0 0 0 0
Tenn 0 0 0 1 1 0 0 1
Texa 0 0 0 0 0 0 0 0
Utah 0 0 0 0 0 0 0 0
Verm 0 0 0 1 1 1 1 1
Virg 0 0 0 0 0 0 0 1
Wash 0 0 1 1 1 1 1 1
WesV 1 0 1 1 1 0 0 0
Wisc 0 0 1 1 1 1 1 1
Wyom 0 0 0 0 0 0 0 0

Distance matrix computation

- We'll use a subset of the data USArrests
- We'll use only a by taking 15 random rows among the 50 rows in the data set.
- Next, we standardize the data using the function scale():

```r
install.packages("FactoMineR")
```

```
## Installing package into '/home/rstudio-user/R/x86_64-pc-linux-gnu-library/4.0'
## (as 'lib' is unspecified)
```

```r
library("FactoMineR")
```

```
##
```

```
## Attaching package: 'FactoMineR'

## The following object is masked from 'package:ade4':
##
##     reconst
```

```r
data("USArrests") # Loading
head(USArrests, 3) # Print the first 3 rows
```

```
##         Murder Assault UrbanPop Rape
## Alabama   13.2     236       58 21.2
## Alaska    10.0     263       48 44.5
## Arizona    8.1     294       80 31.0
```

```r
set.seed(123)
ss <- sample(1:50, 15) # Take 15 random rows
df <- USArrests[ss, ] # Subset the 15 rows
df.scaled <- scale(df) # Standardize the variables
```