Hierarchisches Clustern in R

Achim Zeileis

2009-02-20

Um die Ergebnisse aus der Vorlesung zu reproduzieren, wird zunächst wieder der `GSA` Datensatz geladen

```
R> load("GSA.rda")
```

und wie schon in *PCA.pdf* aggregiert. Die nach `country` aggregierten "Erfolgsanteile" für 8 verschiedene Sommeraktivitäten werden berechnet:

```
R> gsa <- GSA[, c(5, 10:12, 14, 25:27, 29)]
R> gsa <- na.omit(gsa)
R> sucrate <- function(x) prop.table(table(x))[2]
R> gsa <- aggregate(gsa, list(gsa$country), sucrate)
R> rownames(gsa) <- gsa[, 1]
R> gsa <- gsa[, -(1:2)]
```

Die Daten können mit Hilfe der Funktion `hclust` hierarchisch geclustert werden. Diese Funktion nimmt zwei Argumente `hclust(d, method)`, wobei `d` eine Distanzmatrix (der Objekte) sein muß und `method` ein Spezifikation einer Distanzmethode (der Cluster). Die Voreinstellung für `method` ist `"complete"`, weiterhin stehen u.a. `"single"`, `"average"` und `"ward"` zur Verfügung.

Um also `hclust` anwenden zu können, müssen wir erst aus der Datenmatrix `gsa` die zugehörige Distanzmatrix zwischen allen Beobachtungen berechnen. Dies kann in R mit der Funktion `dist` getan werden. Auch diese nimmt zwei Argumente `dist(x, method)`, wobei `x` die Datenmatrix X (oder $\tilde{X}$ oder $\hat{X}$) ist und `method` wieder die Spezifikation einer Distanzmethode. Die Voreinstellung ist `"euclidean"`, weiterhin stehen u.a. `"manhattan"`, `"maximum"`, `"canberra"` oder `"binary"` zur Verfügung.

Da die aggregierten `gsa` Daten numerisch sind, verwenden wir die euklidische Distanz, jedoch skalieren wir die Daten vorher, indem wir jede Spalte um ihren Mittelwert bereinigen und durch ihre Standardabweichung skalieren. Das heißt also, daß wir mit $\hat{X}$ statt direkt mit X rechnen. Dies ist sinnvoll, da die verschiedenen Aktivitäten sehr unterschiedliche Popularität genießen. Die von uns verwendete Distanzmatrix ist also

```
R> gsa.dist <- dist(scale(gsa))
R> gsa.dist
```

```
                Austria (Vienna) Austria (other)  Belgium  Denmark   France
Austria (other)         1.715283
Belgium                 3.824754        3.319864
Denmark                 4.886288        3.994601 2.105802
France                  5.294393        4.485117 2.434923 3.391148
Germany                 2.392534        2.335256 1.756691 3.296428 3.596533
Hungary                 2.389625        2.814275 3.101408 3.955034 4.697927
```

```
Italy                   4.342296        3.708935 2.396942 3.166910 2.339537
Netherlands             3.689156        2.597313 1.681765 1.672746 3.220145
Spain                   8.334688        7.621627 5.317654 5.489192 4.028343
Sweden                  4.203597        3.591642 1.357107 1.640979 2.973355
Switzerland             1.411926        1.666905 3.254477 4.199413 4.543771
UK                      5.587735        4.878425 2.028275 1.958069 2.659113
USA                     7.724369        6.996523 4.338647 4.532401 3.136743
other                   5.744040        4.856727 2.513461 2.360332 2.138327
                 Germany  Hungary    Italy Netherlands    Spain   Sweden
Austria (other)
Belgium
Denmark
France
Germany
Hungary         2.032896
Italy           2.840806 3.407679
Netherlands     2.160484 3.240214 2.789935
Spain           6.350156 7.004168 4.275287    5.911184
Sweden          2.409621 3.288448 3.147176    1.755141 5.534666
Switzerland     2.267373 2.551538 3.496683    3.159937 7.635847 3.817893
UK              3.530582 4.681588 2.828258    2.623299 4.236130 2.299784
USA             5.685142 6.539240 3.689147    5.086255 2.133737 4.793200
other           3.798737 4.451054 2.271095    2.835356 3.356337 2.569099
                Switzerland       UK      USA
Austria (other)
Belgium
Denmark
France
Germany
Hungary
Italy
Netherlands
Spain
Sweden
Switzerland
UK                 4.778145
USA                6.839301 3.066656
other              5.045061 1.947578 2.671496
```

Bei der berechneten Distanzmatrix wird nur die untere Dreiecksmatrix angezeigt, da die Matrix symmetrisch ist und alle Einträge auf der Hauptdiagonalen 0 sein müssen.

Um nun zu illustrieren, wie die Daten auf Basis ihrer Distanzen geclustert werden, verwenden wir die `"average"` Methode. Die anderen Methoden können (und sollten) ganz analog ausprobiert werden.

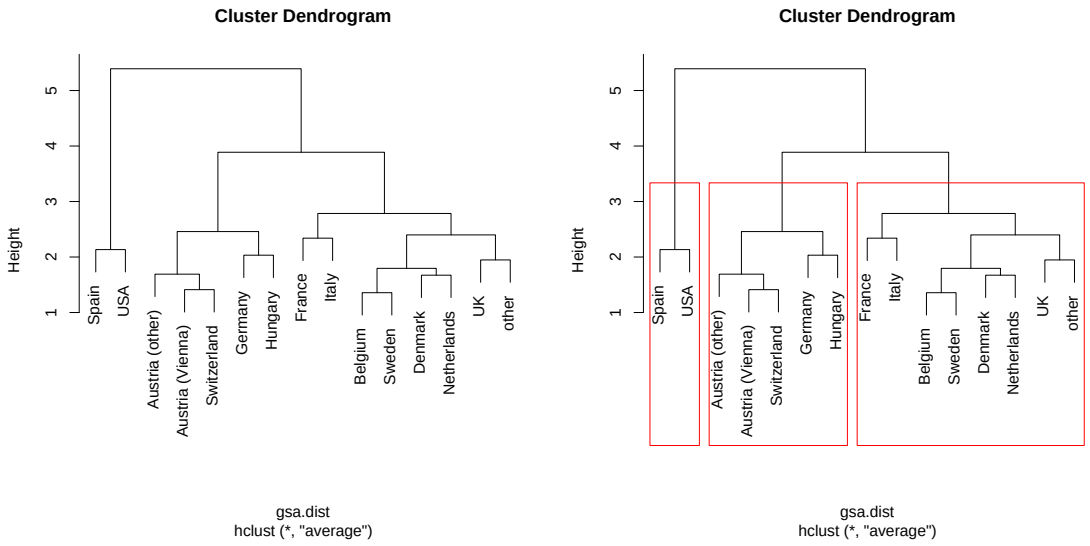
```
R> gsa.hclust <- hclust(gsa.dist, method = "average")
R> gsa.hclust

Call:
hclust(d = gsa.dist, method = "average")

Cluster method   : average
Distance         : euclidean
Number of objects: 15
```

Die `print`-Methode liefert eine sehr knappe Zusammenfassung. Informativer ist das zur Hierarchie gehörige Dendogramm, das durch die `plot`-Methode erzeugt wird.

```
R> plot(gsa.hclust)
```

Dies zeigt deutlich, daß man zumindest 3 Cluster vermuten würde, nämlich (Spanien, USA), (Österreich, Schweiz, Ungarn, Deutschland), und die übrigen Länder. Um diese drei Cluster noch hervorzuheben, kann der Befehl `rect.hclust` verwendet werden, dem man neben dem `"hclust"`-Objekt auch noch die gewünschte Anzahl Cluster übergibt.

```
R> rect.hclust(gsa.hclust, k = 3)
```



Eventuell könnte man auch noch eine Lösung mit 4 Clustern betrachten, die würde also (Frankreich, Italien) nochmal von den übrigen Ländern abtrennen. Eine noch höhere Zahl von Clustern scheint nicht so plausibel, da die Distanzen zwischen den dort zusammengefügten Clustern dann immer sehr klein ist.

Wenn man sich nun den Vektor der Clusterzugehörigkeiten berechnen möchte, kann man den Befehl `cutree` verwenden, der den Baum gleichsam zerschneidet und somit eine Partition returniert. Daher liefert

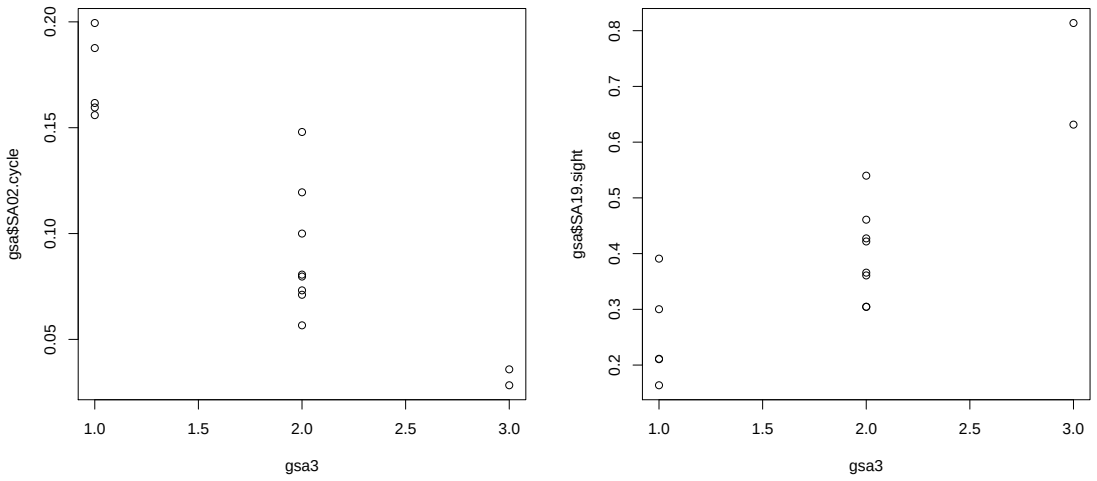
```
R> gsa3 <- cutree(gsa.hclust, k = 3)
R> gsa3
```

```
Austria (Vienna)  Austria (other)          Belgium          Denmark
               1                1                2                2
          France          Germany          Hungary            Italy
               2                1                1                2
     Netherlands            Spain           Sweden      Switzerland
               2                3                2                1
              UK              USA            other
               2                3                2
```

also die Zugehörigkeiten der Beobachtungen zu 3 Clustern wie bereits oben visualisiert. Dabei ist zu beachten, daß die Reihenfolge völlig willkürlich ist und keinerlei Information trägt. Man könnte die Labels `1`, `2` und `3` also beliebig vertauschen.

Um uns anzuschauen, welche Unterschiede zu genau dieser Klassifikation geführt haben, kann man sich ein Punktdiagramm der partitionierten Variablen für jeden Cluster anschauen.

```
R> plot(gsa$SA02.cycle ~ gsa3)
R> plot(gsa$SA19.sight ~ gsa3)
```



Stünden mehr Beobachtungen zur Verfügung, wäre ein Boxplot angemessener, aber da der kleinste Cluster sowieso nur 2 Beobachtungen hat, macht ein Boxplot hier keinen Sinn. Im kleinsten Cluster würden dann ja 2 Beobachtungen durch 5 Werte (Minimum, unteres Quartil, Median, oberes Quartil, Maximum) visualisiert, was sicher nicht sinnvoll ist.

Die Interpretation der Punktdiagramme ist recht klar und entspricht dem, was wir bereits in den vorherigen Tutorien herausgearbeitet haben: die Länder in Cluster 1 wollen eher sportlichen Aktivitäten nachgehen, die in Cluster 3 wollen kulturelle Aktivitäten verfolgen, und Cluster 2 liegt irgendwo dazwischen.