G. Rohwer, C. Dudel Wintersemester 2010/11

Methoden der Datenrepräsentation und Klassifikation

Kapitel 7: Unscharfe Klassifikation

7 Unscharfe Klassifikation

Die bisher besprochenen Ansätze zur Clusteranalyse orientierten sich an der Idee einer Partition, durch die jedes Objekt genau einem Cluster zugeordnet wird. In diesem Kapitel besprechen wir einige Ansätze der unscharfen Clusteranalyse, bei denen Objekte zu mehreren Clustern gehören können. Im ersten Abschnitt wird eine Verallgemeinerung der hierarchischen Klassifikation dargestellt. Im zweiten Abschnitt werden Methoden behandelt, die von vergröberten Abstandsmatrizen ausgehen, die als Adjazenzmatrizen von Graphen betrachtet werden. Hiervon ausgehend können dann sich überlappende Cluster definiert werden. Im dritten Abschnitt wird zunächst ein Ansatz besprochen, bei dem Objekte bestimmt werden, die als Prototypen für Cluster dienen sollen, um dann für alle Objekte „Grade der Zugehörigkeit“ zu diesen Clustern zu definieren. Daran anknüpfend wird ein Verfahren der Fuzzy-Clusteranalyse vorgestellt. Schließlich wird im vierten Abschnitt ein weiteres Verfahren zur Datenrepräsentation das auf einer Verwendung „unscharfer Archetypen“ beruht, erläutert.

7.1 Pyramidale Klassifikation

1. Pyramidale Klassifikationsschemas

Pyramidale Klassifikation wurde von E. Diday vorgeschlagen und dann von zahlreichen Autoren weiter verfolgt.[1] Es handelt sich um eine Verallgemeinerung der hierarchischen Klassifikation. Eine formale Definition kann folgendermaßen gegeben werden. Ausgangspunkt ist eine Objektmenge $\Omega = \{\omega_1, \ldots, \omega_n\}$. Eine Menge von Teilmengen von Ω, im Folgenden $\mathcal{H}$ genannt, ist ein *pyramidales Klassifikationsschema* (kurz: eine *Pyramide*) für Ω, wenn folgende Bedingungen erfüllt sind:

a) $\Omega \in \mathcal{H}$ und $\{\omega_i\} \in \mathcal{H}$ für $i = 1, \ldots, n$.

b) Für alle $h, h' \in \mathcal{H}$: $h \cap h' = \emptyset$ oder $h \cap h' \in \mathcal{H}$.

c) Es gibt eine vollständige lineare Ordnung für Ω, so dass alle Elemente von $\mathcal{H}$ bzgl. dieser Ordnung zusammenhängend sind.

Offenbar ist jedes hierarchische Klassifikationsschema auch ein pyramidales Klassifikationsschema. Ein Unterschied liegt darin, dass jedes Element einer Pyramide zwei unmittelbare Nachfolger haben kann.[2] Zum Beispiel: $\Omega = \{\omega_1, \omega_2, \omega_3\}$ und

$$\mathcal{H} = \{\{\omega_1\}, \{\omega_2\}, \{\omega_3\}, \{\omega_1, \omega_2\}, \{\omega_2, \omega_3\}, \{\omega_1, \omega_2, \omega_3\}\}$$

In diesem Beispiel hat $\{\omega_2\}$ zwei unmittelbare Nachfolger. Mehr als zwei unmittelbare Nachfolger sind jedoch nicht möglich.[3] Daraus folgt, dass ein pyramidales Klassifikationsschema für n Objekte maximal $n\,(n+1)/2$ Elemente hat.

2. Ein agglomerativer Algorithmus

Zur praktischen Berechnung von pyramidalen Klassifikationsschemas verwenden wir eine Variante eines agglomerativen Algorithmus, der von Diday (1986: 215) vorgeschlagen wurde. Ausgangspunkt ist eine Objektmenge $\Omega = \{\omega_1, \ldots, \omega_n\}$ und eine Distanzmatrix $\mathbf{D} = (d_{ij})$.

1) Setze $\mathcal{H} := \{\{\omega_1\}, \ldots, \{\omega_n\}\}$.
2) Bilde eine Menge $\mathcal{M}$, die aus allen Paaren (h_1, h_2) von Elementen von $\mathcal{H}$ besteht, die folgenden Bedingungen genügen:

[1] Vgl. Diday (1986); Gaul und Schader (1994); Lasch (1993; 1996); Gil, Capdevila und Arcas (????); Aude, Diaz-Lazcoz, Codani und Risler (1999).

[2] h' ist ein unmittelbarer Nachfolger von h, wenn $h \subset h'$ ist und es kein h'' mit der Eigenschaft $h \subset h'' \subset h'$ gibt.

[3] Vgl. Diday (1986: 207).

a) Wenn $h_1 = \{\omega\}$ ist, kann ω bzgl. der (während des Verfahrens zu bildenden) linearen Ordnung unmittelbar vor dem kleinsten Element oder unmittelbar nach dem größten Element von h_2 plaziert werden.

b) Wenn $h_2 = \{\omega\}$ ist, kann ω bzgl. der (während des Verfahrens zu bildenden) linearen Ordnung unmittelbar vor dem kleinsten Element oder unmittelbar nach dem größten Element von h_1 plaziert werden.

c) $h_1 \cup h_2$ ist bzgl. der bisher gebildeten linearen Ordnung zusammenhängend.

d) Es gibt kein $h \in \mathcal{H}$, so dass $h_1 \subseteq h$ und $h_2 \subseteq h$ ist.

e) Es gibt kein $h \in \mathcal{H}$, so dass $h \subset h_1 \cup h_2$ und $h \not\subseteq h_1$ und $h \not\subseteq h_2$.

Wenn kein Paar gefunden werden kann, dass diesen Bedingungen genügt, wird abgebrochen.

3) Wähle das Paar, bei dem h_1 und h_2 den kleinsten Abstand aufweisen und füge $h_1 \cup h_2$ als neues Element zur bisher gebildeten Menge $\mathcal{H}$ hinzu. Soweit erforderlich, ergänze die lineare Ordnung für die Elemente von Ω. Dann Fortsetzung bei (2).

Ergänzend ist festzulegen, wie Abstände zwischen den Elementen von $\mathcal{H}$ gebildet werden sollen. Das kann auf analoge Weise geschehen, wie in Abschnitt 6.2-2 für hierarchische Klassifikationsverfahren besprochen wurde. Es wird also durch den Algorithmus zugleich eine neue Abstandsfunktion $d_{\mathcal{H}}$ für $\mathcal{H}$ gebildet. Anfangs wird mit der Definition

$$d_{\mathcal{H}}(\{\omega_i\}, \{\omega_j\}) := d_{ij}$$

begonnen. Dann kann für die Fortsetzung beispielsweise die Complete-Link-Methode verwendet werden, also die Definition

$$d_{\mathcal{H}}(h, h_1 \cup h_2)) := \max\{d_{\mathcal{H}}(h, h_1), d_{\mathcal{H}}(h, h_2)\}$$

Diese Methode hat auch den Vorteil, dass eine Pyramide ohne Inversionen entsteht.[4]

3. Illustration der Berechnung

Um die Berechnung eines pyramidalen Klassifikationsschemas zu verdeutlichen, verwenden wir vier Objekte mit folgender Abstandsmatrix:

$$\mathbf{D} := \begin{pmatrix} 0 & 1 & 3 & 2 \\ 1 & 0 & 6 & 5 \\ 3 & 6 & 0 & 4 \\ 2 & 5 & 4 & 0 \end{pmatrix} \tag{7.1}$$

[4] Vgl. die Diskussion bei Lasch (1996).

Box 7.1-1 Pyramidales Klassifikationsschema für die Abstandsmatrix (7.1).

```
 1. cluster number
 2. first element
 3. second element
 4. number of objects
 5. minimal element
 6. maximal element
 7. next on left side
 8. next on right side
 9. index

 1    2    3    4    5    6    7    8      9
-------------------------------------------------
 1    0    0    1    1    1    2    4     0.0000
 2    0    0    1    2    2    0    1     0.0000
 3    0    0    1    3    3    4    0     0.0000
 4    0    0    1    4    4    1    3     0.0000
 5    2    1    2    2    1    0    0     1.0000
 6    4    1    2    1    4    0    0     2.0000
 7    4    3    2    4    3    0    0     4.0000
 8    7    6    3    1    3    0    0     4.0000
 9    6    5    3    2    4    0    0     5.0000
10    9    8    4    2    3    0    0     6.0000


Abstandsmatrix fuer die Cluster

 1 :  0  1  3  2  1  2  3  3  2  3
 2 :  1  0  6  5  1  5  6  6  5  6
 3 :  3  6  0  4  6  4  4  4  6  6
 4 :  2  5  4  0  5  2  4  4  5  5
 5 :  1  1  6  5  0  5  6  6  5  6
 6 :  2  5  4  2  5  0  4  4  5  5
 7 :  3  6  4  4  6  4  0  4  6  6
 8 :  3  6  4  4  6  4  4  0  6  6
 9 :  2  5  6  5  5  5  6  6  0  6
10 :  3  6  6  5  6  5  6  6  6  0
```

Box 7.1-1 zeigt das Ergebnis des in Abschnitt 7.1-2 beschriebenen Algorithmus mit der Complete-Link-Methode.[5] Das pyramidale Klassifikationsschema $\mathcal{H}$ hat in diesem Beispiel 10 Elemente $h_1, \ldots, h_{10}$, deren Nummern in der ersten Spalte angegeben sind. Im unteren Teil der Box wird die durch den Algorithmus gebildete Abstandsfunktion $d_{\mathcal{H}}$ dargestellt.

Zur bildlichen Veranschaulichung können, ähnlich zu Dendrogrammen, pyramidenartige Graphen vewendet werden. Abbildung 7.1-1 zeigt die Darstellung für das gegenwärtige Beispiel.

[5] Erzeugt mit dem Skript `clpyr1.cf`.

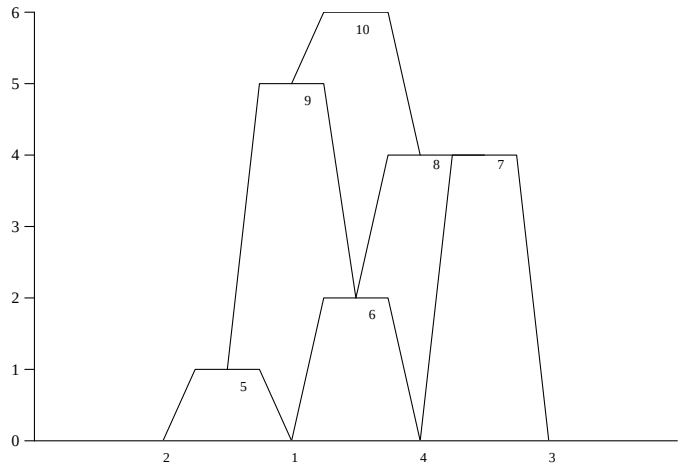

Abb. 7.1-1 Darstellung des pyramidalen Klassifikationsschemas mit den Daten aus Box 7.1-1 (erzeugt mit `clplot4.cf`).

4. Indexfunktionen und Abstände

Analog zur Vorgehensweise bei der hierarchischen Klassifikation können Indexfunktionen verwendet werden. Ist ein pyramidales Klassifikationsschema $\mathcal{H}$ gegeben, ist eine Indexfunktion eine Funktion $l : \mathcal{H} \longrightarrow \mathbf{R}$, die jedem Cluster $h \in \mathcal{H}$ eine nicht-negative Zahl $l(h)$ zuordnet und folgende Bedingung erfüllt: Wenn $h \subset h'$, dann $l(h) < l(h')$.

Um eine bestimmte Indexfunktion zu definieren, kann die für die Cluster gebildete Abstandsfunktion $d_{\mathcal{H}}$ verwendet werden.[6] Für den in Abschnitt 7.1-2 beschriebenen Algorithmus eignet sich folgende rekursive Definition:

$$l(\{\omega_i\}) := 0\,, \quad l(h_1 \cup h_2) := d_{\mathcal{H}}(h_1, h_2) \tag{7.2}$$

Die letzte Spalte in Box 7.1-1 zeigt die Werte dieser Indexfunktion für das Beispiel.

Eine Indexfunktion kann dann verwendet werden, um eine neue, das Klassifikationsschema charakterisierende Abstandsmatrix $\mathbf{D}^* = (d^*_{ij})$ für die Ausgangsobjekte zu bilden:

$$d^*_{ij} := \min\{l(h) \,|\, \omega_i, \omega_j \in h\} \tag{7.3}$$

[6] Vgl. Lasch (1996: 239).

Für das Beispiel sieht diese Abstandsmatrix folgendermaßen aus:

$$\mathbf{D}^{*} := \begin{pmatrix} 0 & 1 & 4 & 2 \\ 1 & 0 & 6 & 5 \\ 4 & 6 & 0 & 4 \\ 2 & 5 & 4 & 0 \end{pmatrix} \qquad \mathbf{D}^{**} := \begin{pmatrix} 0 & 1 & 6 & 6 \\ 1 & 0 & 6 & 6 \\ 6 & 6 & 0 & 4 \\ 6 & 6 & 4 & 0 \end{pmatrix} \tag{7.4}$$

Mit einer Ausnahme entspricht sie der Ausgangsmatrix in (7.1). $\mathbf{D}^{**}$ ist zum Vergleich die durch eine hierarchische Klassifikation mit der Complete-Link-Methode erzeugte Abstandsmatrix. Offenbar erlaubt die pyramdiale Klassifikation eine bessere Anpassung an die vorgegebenen Abstände.

5. Illustration mit Berufsstrukturdaten

Für eine weitere Illustration verwenden wir die Abstandsmatrix 2.3-3 für die Berufsstrukturdaten. Box 7.1-2 zeigt das pyramidale Klassifikationsschema, Abbildung 7.1-2 liefert eine graphische Veranschaulichung.[7]

Tabelle 7.1-1 zeigt die aus dem Klassifikationsschema gebildete Abstandsmatrix. Wiederum repräsentiert sie die ursprüngliche Abstandsmatrix (Tabelle 2.3-3) wesentlich besser als mit einem ultrametrischen Verfahren möglich ist (vgl. beispielsweise die Abstandsmatrix in Tabelle 6.3-1).

So wie man ausgehend von einem Dendrogramm Cluster bilden kann (vgl. Abschnitt 6.2-6), kann man auch von einem pyramidalen Klassifikationsschema ausgehen. Man gelangt dann zu sich überlappenden Clustern. Tabelle 7.1-2 zeigt die Cluster, die man mit dem Schema in Box 7.1-2 bis zu einem Niveau 0.16 erhält,

7.2 Abstandsmatrizen als Graphen

In diesem Abschnitt besprechen wir einige Möglichkeiten der Clusteranalyse, die davon ausgehen, eine Abstandsmatrix als Adjazenzmatrix eines bewerteten Graphen zu betrachten.[8]

1. Vergröberung von Abstandsmatrizen

Eine Abstandsmatrix $\mathbf{D} = (d_{ij})$ für n Objekte kann offenbar als eine Adjazenzmatrix eines bewerteten Graphen betrachtet werden, dessen Knoten den Objekten entsprechen. Es handelt sich dann um einen vollständigen Graphen: Jeder Knoten ist mit jedem anderen Knoten durch eine Kante verbunden, deren Bewertung durch den Abstand der entsprechenden Objekte gegeben ist. Um zu Clusteranalysen zu gelangen, kann man hier anknüpfen und einen vergröberten unbewerteten Graphen betrachten. Man

[7] Das TDA-Skript ist `clpyr2.cf`.

[8] Man vgl. etwa Augustson und Minker (1970), Hubert (1974a).

Box 7.1-2 Pyramidales Klassifikationsschema für die Abstandsmatrix der Berufsstrukturdaten.

```
1. cluster number
2. first element
3. second element
4. number of objects
5. minimal element
6. maximal element
7. next on left side
8. next on right side
9. index
```

1	2	3	4	5	6	7	8	9
1	0	0	1	1	1	2	0	0.0000
2	0	0	1	2	2	8	1	0.0000
3	0	0	1	3	3	5	8	0.0000
4	0	0	1	4	4	7	5	0.0000
5	0	0	1	5	5	4	3	0.0000
6	0	0	1	6	6	0	7	0.0000
7	0	0	1	7	7	6	4	0.0000
8	0	0	1	8	8	3	2	0.0000
9	5	3	2	5	3	0	0	0.0520
10	7	4	2	7	4	0	0	0.1002
11	5	4	2	4	5	0	0	0.1027
12	11	9	3	4	3	0	0	0.1127
13	11	10	3	7	5	0	0	0.1328
14	8	3	2	3	8	0	0	0.1457
15	14	9	3	5	8	0	0	0.1457
16	2	1	2	2	1	0	0	0.1551
17	8	2	2	8	2	0	0	0.1652
18	13	12	4	7	3	0	0	0.1696
19	7	6	2	6	7	0	0	0.1769
20	19	10	3	6	4	0	0	0.1769
21	17	14	3	3	2	0	0	0.1801
22	21	15	4	5	2	0	0	0.1801
23	20	13	4	6	5	0	0	0.1821
24	23	18	5	6	3	0	0	0.1821
25	15	12	4	4	8	0	0	0.2028
26	25	18	5	7	8	0	0	0.2135
27	25	22	5	4	2	0	0	0.2486
28	26	24	6	6	8	0	0	0.2624
29	27	26	6	7	2	0	0	0.2645
30	29	28	7	6	2	0	0	0.2950
31	17	16	3	8	1	0	0	0.3040
32	31	21	4	3	1	0	0	0.3235
33	32	22	5	5	1	0	0	0.3235
34	33	27	6	4	1	0	0	0.3761
35	34	29	7	7	1	0	0	0.3901
36	35	30	8	6	1	0	0	0.4230

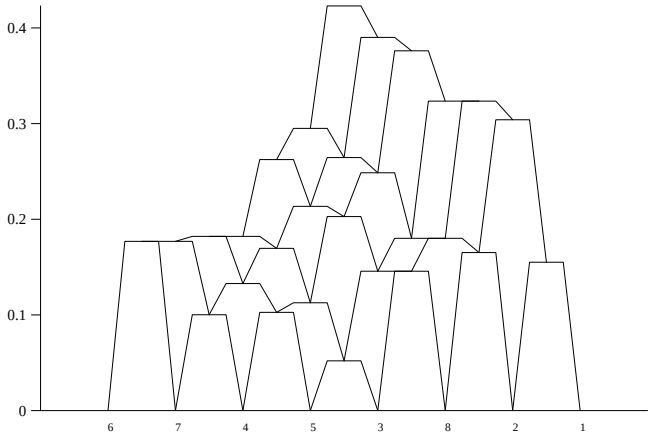

Abb. 7.1-1 Darstellung des pyramidalen Klassifikationsschemas mit den Berufsstrukturdaten (erzeugt mit `clplot5.cf`).

Tabelle 7.1-1 Aus der pyramidalen Klassifikation der Berufsstrukturdaten gebildete Abstandsmatrix.

0.0000	0.1551	0.3235	0.3761	0.3235	0.4230	0.3901	0.3040
0.1551	0.0000	0.1801	0.2486	0.1801	0.2950	0.2645	0.1652
0.3235	0.1801	0.0000	0.1127	0.0520	0.1821	0.1696	0.1457
0.3761	0.2486	0.1127	0.0000	0.1027	0.1769	0.1002	0.2028
0.3235	0.1801	0.0520	0.1027	0.0000	0.1821	0.1328	0.1457
0.4230	0.2950	0.1821	0.1769	0.1821	0.0000	0.1769	0.2624
0.3901	0.2645	0.1696	0.1002	0.1328	0.1769	0.0000	0.2135
0.3040	0.1652	0.1457	0.2028	0.1457	0.2624	0.2135	0.0000

Tabelle 7.1-2 Aus dem pyramidalen Klassifikationsschema in Box 7.1-2 bis zum Niveau 0.16 gebildete überlappende Cluster.

$\{3,5\}$	Schweiz, Deutschland
$\{4,7\}$	Grossbritannien, USA
$\{4,5\}$	Grossbritannien, Deutschland
$\{3,4,5\}$	Schweiz, Grossbritannien, Deutschland
$\{4,5,7\}$	Grossbritannien, Deutschland, USA
$\{3,8\}$	Schweiz, Japan
$\{3,5,8\}$	Schweiz, Deutschland, Japan
$\{1,2\}$	Türkei, Griechenland

bildet dann aus $\mathbf{D}$ eine Adjazenzmatrix $\mathbf{D}^\nu = (d_{ij}^\nu)$, wobei

$$d_{ij}^\nu := \begin{cases} 1 & \text{wenn } d_{ij} \le \nu \\ 0 & \text{andernfalls} \end{cases}$$

Tabelle 7.2-1 Eine aus der Abstandsmatrix in Tabelle 2.3-3 gebildete Adjazenzmatrix ($\nu = 0.16$).

0.0000	0.1551	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000
0.1551	0.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	-1.0000
-1.0000	-1.0000	0.0000	0.1127	0.0520	-1.0000	-1.0000	0.1457
-1.0000	-1.0000	0.1127	0.0000	0.1027	-1.0000	0.1002	-1.0000
-1.0000	-1.0000	0.0520	0.1027	0.0000	-1.0000	0.1328	0.1341
-1.0000	-1.0000	-1.0000	-1.0000	-1.0000	0.0000	-1.0000	-1.0000
-1.0000	-1.0000	-1.0000	0.1002	0.1328	-1.0000	0.0000	-1.0000
-1.0000	-1.0000	0.1457	-1.0000	0.1341	-1.0000	-1.0000	0.0000

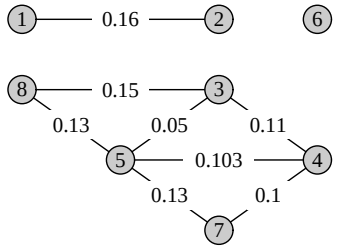

Abb. 7.2-1 Der Graph mit der Adjazenzmatrix in Tabelle 7.2-1.

Hierbei ist ν eine positive Zahl, die das Niveau angibt, bis zu dem Objekte als verbunden betrachtet werden sollen.

Zur Illustration verwenden wir die Abstandsmatrix in Tabelle 2.3-3 für die Berufsstrukturdaten. Mit $\nu = 0.16$ findet man die in Tabelle 7.2-1 gezeigte Adjazenzmatrix (wobei auch noch die ursprünglichen Abstände angegeben werden und -1 die nicht-vorhandenen Kanten anzeigt). Abbildung 7.2-1 zeigt den Graphen.

2. Clusterdefinitionen für Graphen

Ausgehend von der Repräsentation einer vergröberten Abstandsmatrix in der Form eines (unbewerteten oder bewerteten) Graphen kann man unterschiedliche Clusterdefinitionen verwenden. Wir verweisen kurz auf drei Möglichkeiten und illustrieren sie mit dem Graphen in Abbildung 7.2-1.

1. Die einfachste Möglichkeit besteht darin, als Cluster die Komponenten eines Graphen zu verwenden. In unserm Beispiel gibt es dann drei Cluster:

$$\{1,2\}, \{6\}, \{3,4,5,7,8\}$$

Dieser Ansatz liefert offenbar eine Parition.

2. Definiert man dagegen Cluster durch Cliquen, können sie sich überlappten. Eine Clique ist eine maximale Menge von Knoten, bei der jeder Knoten mit jedem anderen Knoten durch eine Kante verbunden ist. In unserem Beispiel findet man:

$$\{6\}, \{1,2\}, \{3,4,5\}, \{3,5,8\}, \{4,5,7\}$$

Zum Beispiel gehört der Knoten 5 (Deutschland) zu drei Cliquen.

3. Geht man von einem bewerteten Graphen aus, kann man Cluster auch durch sogenannte kompakte Mengen definieren, die den in Abschnitt 5.1 eingeführten S-Clustern entsprechen. Eine Menge C, die aus Knoten eines bewerteten Graphen mit der Knotenmenge $\mathcal{N}$ besteht, wird als eine *kompakte Menge* bezeichnet, wenn gilt:[9]

$$\max\{d^{\nu}_{ij} \mid i,j \in C\} < \min\{d^{\nu}_{ij} \mid i \in C, j \in \mathcal{N} - C\}$$

In unserm Beispiel gibt es folgende kompakte Mengen:

$$\{6\}, \{1,2\}, \{3,5\}, \{4,7\}, \{3,4,5,7\}$$

Für die Berechnung wurde das Programm TDA verwendet.[10]

3. Berechnungen mit R

In Box 7.2-1 findet man die R-Syntax für eine Vergröberung der Abstandsmatrix der Berufsstrukturdaten sowie die Bildung von Clustern für den resultierenden Graphen. Als erstes wird das Paket `sna` geladen, welches die weiter unten verwendeten Befehle zum Auffinden von Komponenten und Cliquen eines Graphen enthält. Anschließend wird mit den bereits bekannten Befehlen die Abstandsmatrix aus Tabelle 2.3-3 erzeugt. Für die Vergröberung wird eine neue Matrix `d2` erstellt, bei der zunächst alle Einträge gleich 0 sind. Die Einträge in dieser Matrix, für die die korrespondierenden Einträge in der Abstandsmatrix kleiner oder gleich 0.16 sind, werden anschließend auf den Wert 1 gesetzt. Das Ergebnis ist die in der Box dokumentierte Adjazenzmatrix.

Um die Komponenten des durch diese Adjazenzmatrix gegebenen Graphen anzuzeigen, kann der Befehl `component.dist` benutzt werden. Zunächst wird für jedes der acht Länder angegeben, zu welcher Komponente es gehört. Die Türkei und Griechenland bilden beispielsweise eine Komponente, ebenso wie Schweden. Dabei ist die Nummerierung der

[9] $\mathcal{N} - C$ bezeichnet die Menge aller Knoten von $\mathcal{N}$, die nicht zu C gehören.

[10] Die vergröberte Abstandsmatrix wurde mit dem Befehl `gdp` erzeugt (Skript:`ucl5.cf`). Komponenten können mit dem Befehl `gcon`, Cliquen mit `gcliq` und kompakte Mengen mit `gcset` berechnet werden (Skript `ucl6.cf`).

Box 7.2-1 R-Befehle für eine Illustration mit den Berufsstrukturdaten.

```
> library(sna)

# Abstandsmatrix
> dat <- read.table("bs1.dat")
> names(dat) <- c("X","Y","Z","h")
> tab <- xtabs(h~.,data=dat)
> tab <- ftable(tab,row.vars="X",col.vars=c("Y","Z"))
> tab <- prop.table(tab,1)
> d <- dist(tab,method="manhattan")*0.5

# Vergr"oberung
> d2 <- matrix(0,nrow=8,ncol=8)
> d2[as.matrix(d)<=0.16] <- 1
> d2
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]
[1,]    1    1    0    0    0    0    0    0
[2,]    1    1    0    0    0    0    0    0
[3,]    0    0    1    1    1    0    0    1
[4,]    0    0    1    1    1    0    1    0
[5,]    0    0    1    1    1    0    1    1
[6,]    0    0    0    0    0    1    0    0
[7,]    0    0    0    1    1    0    1    0
[8,]    0    0    1    0    1    0    0    1

# Komponenten & Cliquen
> component.dist(d2)
$membership
[1] 1 1 2 2 2 3 2 2

$csize
[1] 2 5 1

$cdist
[1] 1 1 0 0 1 0 0 0

> clique.census(d2)
```

Komponenten (1 bis 3) willkürlich und hat keine inhaltliche Bedeutung. Die Anzahl der Elemente der einzelnen Komponenten wird unter `$csize` wiedergegeben. Die Verteilung der Größen der Komponenten kann aus `$cdist` abgelesen werden. Es gibt eine Komponente mit einem Element und eine mit 2 Elementen. Komponenten mit 3 oder 4 Elementen kommen nicht vor, während es eine Komponente mit 5 Knoten gibt.

Der Befehl `clique.census` kann zur Berechnung der Cliquen eines Graphen verwendet werden.

7.3 Graduelle Clusterzugehörigkeit

In diesem Abschnitt besprechen wir Ansätzea der unscharfen Clusteranalyse, die auf der Idee beruhen, dass es sinnvoll ist, „Grade der Clusterzugehörigkeit“ zu unterscheiden; in der Literatur wird of von „Fuzzy-Clusteranalyse“ gesprochen. Zur Vorbereitung der Überlegungen beginnen wir im ersten Unterabschnitt mit einem einfachen Verfahren der partitionierenen Clusteranalyse.

1. Clusteranalyse mit Prototypen

Ein einfacher Ansatz, um p Cluster zu ermitteln, besteht darin, zunächst p Objekte zu bestimmen, die als Prototypen dienen sollen, und dann alle übrigen Objekte jeweils demjenigen Prototyp zuzurechnen, zu dem sie den geringsten Abstand haben. Sie P die Menge der Indizes für die Prototypen, erhält man die Cluster[11]

$$C_t := \{\omega_i \,|\, d_{it} < d_{it'} \text{ für } t' \in P - \{t\}\} \tag{7.5}$$

Um die Prototypen ω_t ($t \in P$) zu finden, kann man sich daran orientieren, dass folgendes Kriterium minimal sein sollte:

$$\sum_{i=1}^{n} d_{i,t(i)} \tag{7.6}$$

Dabei soll $t(i)$ der Index desjenigen Prototyps sein, zu dem ω_i den kleinsten Abstand aufweist.[12]

Zur Illustration verwenden wir die Abstandsmatrix in Tabelle 2.3-3 für die Berufsstrukturdaten. Sucht man drei Prototypen, die das Kriterium (7.6) minimieren, findet man die Objekte ω_1 (Türkei), ω_5 (Deutschland) und ω_6 (Schweden).

Bei dieser Ansatz wird jedes Objekt demjenigen Prototypen zugeordnet, zu dem es den kleinsten Abstand aufweist. Um zu ermitteln, wie gut diese Zuordnung ist, sollten allerdings auch die Abstände zu den übrigen Prototypen in Betracht gezogen werden. Die obere Hälfte von Box 7.3-1 zeigt diese Abstände für unser Beispiel. Man erkennt zum Beispiel, dass die Zuordnung von ω_2 (Griechenland) zum Prototyp ω_1 (Türkei) nicht sehr trennscharf ist.

Um das explizit zu erfassen, kann anstelle von (7.5) folgende allgemeinere Clusterdefinition verwendet werden:

$$C_t(\alpha) := \{\omega_i \,|\, d_{it} \le \alpha\} \tag{7.7}$$

[11] In dieser Formulierungen wird angenommen, dass keine Bindungen auftreten.

[12] Dies ist eine Variante des sog. p-Median-Problems, zu dem es eine umfangreiche Literatur gibt.

Box 7.3-1 Zurordnung der Länder aus den Berufsstrukturdaten zu drei Prototypen.

i	t(i)	d_i,1	d_i,5	d_i,6
1	1	0.0000	0.3142	0.4230
2	1	0.1551	0.1663	0.2950
3	5	0.3235	0.0520	0.1746
4	5	0.3761	0.1027	0.1664
5	5	0.3142	0.0000	0.1821
6	6	0.4230	0.1821	0.0000
7	5	0.3901	0.1328	0.1769
8	5	0.3040	0.1341	0.2624

i	t(i)	u_i,1	u_i,5	u_i,6
1	1	0.7954	0.2046	0.0000
2	1	0.4105	0.3933	0.1961
3	5	0.1384	0.5161	0.3455
4	5	0.0752	0.5135	0.4113
5	5	0.1408	0.5474	0.3118
6	6	0.0000	0.3629	0.6371
7	5	0.0578	0.5098	0.4324
8	5	0.2093	0.5082	0.2825

Das durch den Prototyp ω_t konstituierte Cluster besteht dann aus allen Objekten, deren Abstand zu dem Prototyp nicht größer als ein vorgegebener Wert α ist. Dies führt dann meistens zu überlappenden Clustern. Wählt man etwa $\alpha = 0.17$, erhält man in unserem Beispiel die Cluster

$$C_1(0.17) = \{1, 2\},\ \ C_5(0.17) = \{2, 3, 4, 5, 7, 8\},\ \ C_6(0.17) = \{4, 6\}$$

2. Grade der Clusterzugehörigkeit

Eine weiterführende Überlegung beruht auf der Idee, dass man die Zugehörigkeit von Objekten zu Clustern graduell abstufen kann. Für jedes Objekt ω_i und jedes Cluster C_t wird dann eine (zunächst unbestimmte) Größe u_{it} angenommen, die den Grad der Zugehörigkeit von ω_i zum Cluster C_t quantifiziert. Dabei werden folgende Bedingungen vorausgesetzt:

$$0 \le u_{it} \le 1 \quad \text{und} \quad \sum_{t=1,p} u_{it} = 1 \tag{7.8}$$

Es gibt zahlreiche Methoden, um diese Größen zu konstruieren. Oft wird in diesem Zusammenhang von „Fuzzy-Clusteranalyse“ gesprochen. Bei einer Clusteranalyse, die von Prototypen ausgeht, kann man indessen Grade der Clusterzugehörigkeit auch sehr einfach definieren und berechnen, indem man die vorgegebenen Abstände zunächst in Ähnlichkeiten transformiert;

zum Beispiel mit folgender Definition:

$$s_{ij} := 1 - d_{ij}/d_{\max} \tag{7.9}$$

wobei $d_{\max}$ der größte Wert in der Abstandsmatrix $\mathbf{D} = (d_{ij})$ ist. Dann kann man annehmen, dass die Größen u_{it} proportional zu den Ähnlichkeiten s_{it} sein sollen; d.h. je ähnlicher ein Objekt zu einem Prototyp ist, desto größer soll der Grad seiner Zugehörigkeit zu dem entsprechenden Cluster sein. Um der Normierungsbedingung (7.8) zu genügen, kann man also definieren:

$$u_{it} := \frac{s_{it}}{\sum_{k \in P} s_{ik}} \tag{7.10}$$

In unserem Beispiel erhält man die in der unteren Hälfte von Box 7.3-1 angegebenen Werte. Sie können offenbar als Anhaltspunkte verwendet werden, um zu überlegen, wie gut sich die Objekte den Clustern zurechnen lassen.

3. Fuzzy-Clusteranalyse mit R

Ein Vorschlag, die Idee gradueller Clusterzugehörigkeit im Rahmen partitionierender Verfahren umzusetzen, stammt von Kaufman und Rousseeuw (1990: 164ff.). Wie bei den in Kapitel 5 beschriebenen Ansätzen wird die Anzahl der Cluster k vorgegeben. Dann sollen die Cluster so bestimmt werden, dass folgendes Kriterium minimiert wird:

$$\sum_{t=1}^{k} \frac{\sum_{i<j} u_{it}^r u_{jt}^r d_{ij}}{\sum_{i=1}^{n} u_{it}^r} \to \min \tag{7.11}$$

Hierbei gelten für u_{it} die in (7.8) gegebenen Bedingungen. Wenn dann ein Abstand d_{ij} groß ist, wird eine Minimierung des Kriteriums erreicht, wenn für ein Cluster C_t entweder einer der Werte u_{it} und u_{jt} oder beide klein sind – also höchstens eines der beiden Objekte relativ stark zu C_t gehört. r muss größer als 1 sein und wird üblicherweise auf den Wert 2 gesetzt. Je größer der Wert, desto unschärfer wird die Klassifikation. Der Algorithmus zur Minimierung des Kriteriums ist kompliziert; genaue Erläuterungen findet man bei Kaufman und Rousseeuw (1990: 182ff.).

Nachdem Werte für u_{it} berechnet wurden, stellt sich die Frage, wie stark die Lösung von einer scharfen Klassifikation abweicht. Ein mögliches Kriterium für die Beantwortung dieser Frage wurde von Dunn (1976) vorgeschlagen[13]:

$$\sum_{t=1}^{k} \sum_{i=1}^{n} \frac{u_{it}^2}{n} \tag{7.12}$$

[13] Vgl. Kaufman und Rousseeuw (1990: 171)

Box 7.3-2 Fuzzy-Clusteranalyse mit den Berufsstrukturdaten.

```
> library(cluster)

> dat <- read.table("bs1.dat")
> names(dat) <- c("X","Y","Z","h")
> tab <- xtabs(h~.,data=dat)
> tab <- ftable(tab,row.vars="X",col.vars=c("Y","Z"))
> tab <- prop.table(tab,1)
> d <- dist(tab,method="manhattan")*0.5

> cl <- fanny(d,k=2)
> cl
Fuzzy Clustering object of class 'fanny' :
m.ship.expon.        2
objective     0.3533829
tolerance        1e-15
iterations          31
converged            1
maxit              500
n                    8
Membership coefficients (in , rounded):
     [,1] [,2]
[1,]   79   21
[2,]   84   16
[3,]   25   75
[4,]   19   81
[5,]   22   78
[6,]   30   70
[7,]   24   76
[8,]   50   50
Fuzzyness coefficients:
dunn_coeff normalized
 0.6348868  0.2697735
Closest hard clustering:
[1] 1 1 2 2 2 2 2 2

Available components:
 [1] "membership"  "coeff"       "memb.exp"    "clustering"  "k.crisp"
 [6] "objective"   "convergence" "diss"        "call"        "silinfo"
```

Diese Größe kann Werte zwischen $1/k$ bis 1 annehmen. Ein Wert von 1 ergibt sich, wenn für jede Beobachtung ein $u_{it} = 1$ ist, also jedes Objekt zu genau einem Cluster gehört. Den Wert $1/k$ erhält man, wenn alle $u_{it} = 1/k$ sind, also alle Objekte gleichermaßen zu allen Clustern gehören.

In R ist das Verfahren im Paket `cluster` enthalten und kann mit dem Befehl `fanny` aufgerufen werden. Ein Beispiel mit den Berufsstrukturdaten findet man in Box 7.3-2. Nachdem das Paket geladen und eine Abstandsmatrix `d` erstellt wurde, wird diese an den Befehl `fanny` übergeben. Das Argument `k=2` gibt an, dass eine Partition mit zwei Clustern gefunden werden soll. Die maximale Anzahl an Clustern, die über dieses Argument

angegeben werden kann, beträgt $n/2-1$, was bei den Berufsstrukturdaten einem maximalen Wert von 3 entspricht.

Beim Aufruf der Ergebnisse wird unter `m.ship.expon.` der Wert von r angezeigt. Als Standard wird $r = 2$ verwendet, einen anderen Wert kann man bei Bedarf über das Argument `memb.exp` bestimmen. Der Wert des Zielkriteriums wird bei `objective` ausgegeben. Hierauf folgen einige weitere technische Details.

Die Werte für einzelne u_{it} sind unter `Membership coefficients` abzulesen. Beispielsweise ist $u_{11} = 0.79$ und $u_{12} = 0.21$. Die Zugehörigkeit der Türkei zu Cluster 1 ist also größer als die zu Cluster 2. Gleiches gilt für Griechenland. Die übrigen Objekte gehören „eher“ zu Cluster 2, mit Ausnahme von Objekt 8 (Japan), für das u_{81} und u_{82} in etwa gleich sind.

Das von Dunn vorgeschlagene Kriterium zur Quantifizierung der Unschärfe der Klassifikation ist unter `dunn_coeff` ausgewiesen und beträgt 0.635. Eine standardisierte Variante, die Werte zwischen 0 und 1 annehmen kann, wird bei `normalized` gezeigt. Auch für diese gilt, dass niedrige Werte auf eine relativ unscharfe und hohe Werte auf eine relativ eindeutige Klassifikation hinweisen. In diesem Beispiel beträgt der Wert etwa 0.270, was auf eine vergleichsweise unscharfe Klassifikation hinweist.

Schließlich ist bei `Closest hard clustering` eine scharfe Klassifikation angegeben. Bei dieser wird jede Beobachtung i dem Cluster t zugeordnet, für das es den höchsten Wert u_{it} aufweist. Bei diesem Beispiel entspricht diese Partition den Partitionen mit zwei Clustern, die über das *k-means* und das *k-medoids*-Verfahren gefunden wurden.

7.4 Unscharfe Archetypen

1. Der theoretische Ansatz

Ein der Verwendung von Prototypen nicht unähnliches Verfahren wurde von Cutler und Breiman (1994) vorgeschlagen und ist unter der Bezeichnung *archetypische Analyse* (engl.: *archetypal analysis*) bekannt[14]. Die Idee ist, eine Datenmatrix $\mathbf{X}$ mit n Objekten durch eine neue Datenmatrix $\mathbf{Z}$ mit k „Archetypen“ optimal abzubilden. Hierfür wird versucht, folgendes Kriterium zu minimieren:

$$\sum_{i=1}^{n} ||\mathbf{x}_i - \sum_{t=1}^{k} u_{it}\mathbf{z}_t||^2 \rightarrow \min \tag{7.13}$$

Dabei gelten wieder folgende Bedingungen:

$$u_{it} \geq 0 \quad \text{und} \quad \sum_{t=1}^{k} u_{it} = 1 \quad (i = 1, 2, \ldots, n)$$

[14] Anwendungen und Erweiterungen werden von Corsaro und Marino (2010), D'Esposito et al. (2010), Eugster und Leisch (2010) und Porzio et al. (2008) diskutiert.

$\mathbf{x}_i$ steht für Datenzeile i von $\mathbf{X}$ und $\mathbf{z}_t$ für Zeile t von $\mathbf{Z}$. Durch die einzelnen u_{it} werden die Objekte als „Mischungen“ von Archetypen repräsentiert. Ziel ist dann die Minimierung des euklidischen Abstands zwischen den ursprünglichen Beobachtungen und ihren Repräsentationen durch die zu findenden Typen.

Ferner wird gefordert, dass für einzelne $\mathbf{z}_t$

$$\mathbf{z}_t = \sum_{i=1}^{n} v_{ti}\mathbf{x}_i$$

mit den Bedingungen

$$v_{ti} \geq 0 \quad \text{und} \quad \sum_{i=1}^{n} v_{ti} = 1 \quad (t = 1, 2, \ldots, k)$$

gilt. Insofern können die zu konstruierenden Typen als „Mischungen“ der ursprünglichen Objekte aufgefasst werden, wobei die Koeffizienten v_{ti} erfassen, zu welchem Anteil sich Typus t aus Objekt i ergibt.

Dieses Vorgehen führt dazu, dass für $k > 1$ die aus diesem Verfahren resultierenden Typen „Extremtypen“ sind[15]. Bei $k = 1$ ergibt sich die Lösung als Mittelwert $1/n \sum_i \mathbf{x}_i$.

Bei der Berechnung einer Lösung für diese Problemstellung wird von irgendwie gegebenen Startwerten für v_{ti} ausgegangen. Die ausgehend von diesen Startwerten gefundene Lösung muss nicht unbedingt einem globales Optimum entsprechen, weshalb das Verfahren mehrmals mit unterschiedlichen Startwerten durchgeführt werden sollte.

2. Berechnungen mit `R`

In `R` ist das beschriebene Verfahren im Paket `archetypes` implementiert[16] Ein Beispiel mit den Klausurdaten findet man in Box 7.4-1.

Zur Durchführung des Verfahrens wird der Befehl `stepArchetypes` benutzt. Als Datengrundlage werden die Spalten 2 bis 6 der ursprünglichen Datenmatrix verwendet, womit die erste Spalte, die die durchlaufende Nummerierung enthält, nicht berücksichtigt wird. Über das Argument `k=3` wird die Zahl der zu suchenden Typen auf 3 gesetzt und über `nrep=100` werden 100 Wiederholungen des Verfahrens mit unterschiedlichen Startwerten durchgeführt.

Das Objekt `as` umfasst die Resultate aller 100 Wiederholungen. Um die beste Lösung auszuwählen, wird der Befehl `bestModel` benutzt. Über `as$archetypes` lassen sich die gefundenen Typen ausgeben. Angezeigt

[15] Wenn C gleich der Menge der Datenpunkte $\mathbf{x}_i$ ist, die auf der konvexen Hülle der Datenmatrix liegen, ist für den Fall, dass $k = |C|$ Typen gesucht sind, die Lösungsmenge gleich C.

[16] Für Details hierzu siehe Eugster und Leisch (2009,2010).

Box 7.4-1 Berechnung unscharfer Archetypen für die Klausurdaten.

```
> library(archetypes)
> dat <- read.table("klaus1.dat")

> as <- stepArchetypes(dat[,2:6],k=3,nrep=100)

> as <- bestModel(as)
> as$archetypes
            V2        V3            V4         V5        V6
[1,]  6.002062 10.000020 -8.389044e-05  3.5928485  4.944875
[2,]  9.999977  1.338121  4.999930e+00  0.7326545 11.521605
[3,] 10.000024 10.000026  4.337718e+00 10.0001031 14.561884

> as$alphas[1:5,]
          [,1]       [,2]      [,3]
[1,] 0.8252791 0.00000000 0.1747183
[2,] 0.7622686 0.05154248 0.1861818
[3,] 1.0000433 0.00000000 0.0000000
[4,] 0.6201260 0.37991219 0.0000000
[5,] 0.4986161 0.12234762 0.3790137

> as$betas[,1:5]
             [,1] [,2]      [,3] [,4]          [,5]
[1,] 1.265680e-02    0 0.6494504    0 0.000000e+00
[2,] 0.000000e+00    0 0.0000000    0 0.000000e+00
[3,] 2.381032e-15    0 0.0000000    0 1.096415e-16

> rowSums(as$alphas)[1:5]
 [1] 0.9999974 0.9999929 1.0000433 1.0000382 0.9999774

> rowSums(as$betas)
[1] 1.0000282 0.9999652 1.0000362
```

wird eine Matrix, bei der die Zeilen zu den Typen und die Spalten zu den Merkmalen korrespondieren. Archetypus 3 weist beispielsweise bei allen Klausuraufgaben sehr hohe Punktzahlen auf. Allerdings sieht man, dass bei den Klausuraufgaben 1, 2 und 3 mehr Punkte erreicht werden, als maximal möglich sind. Bei Typus 1 und Aufgabe 3 wiederum erscheint ein negativer Wert, der ebenfalls nicht plausibel erscheint und nicht durch eine Kombination von Beobachtungen aus den ursprünglichen Daten zu diesem Typus zu erklären ist. Ursache hierfür ist, dass nur eine näherungsweise Lösung für $\mathbf{Z}$ gefunden werden kann.

Die Werte von u_{it} können mittels `as$alphas` angezeigt werden. In Box 7.4-1 werden die Werte nur für die ersten fünf Beobachtungen des Datensatzes angefordert. Bei der ausgegebenen Matrix entsprechen die Zeilen den Beobachtungen und die Spalten den unterschiedlichen Typen. Beobachtung 1 kann beispielsweise durch eine Mischung der Typen 1 und 3 abgebildet werden. Werte für v_{it} erhält man mittels `as$betas`. Hier

beziehen sich die Zeilen auf Typen und die Spalten auf Beobachtungen. Es werden wieder nur die Werte der ersten fünf Beobachtungen angezeigt. Dann ist zum Beispiel zu sehen, dass v_{13} etwa 0.65 beträgt.

Auch für u_{it} und v_{ti} werden nur näherungsweise Lösungen gefunden. Dies sieht man unter anderem, wenn man die Bedingungen $\sum_{t=1}^{k} u_{it} = 1$ und $\sum_{i=1}^{n} v_{ti} = 1$ überprüft. Hierfür wird der Befehl `rowSums` benutzt. Für die ersten fünf Beobachtungen sieht man dann, dass $\sum_{t=1}^{k} u_{it} \neq 1$ gilt. Gleiches lässt sich für die Summen der v_{ti} feststellen.