Grundlagen clusteranalytischer Verfahren

Institut für Soziologie - Universität Duisburg-Essen

Prof. Petra Stein - Sven Vollnhals

1. April 2011

Inhaltsverzeichnis

KAPITEL 1

Einleitung

Clusteranalytische Verfahren sind immer dann relevant und von Interesse, wenn das Ziel ist, eine Gruppe von Objekten zu homogenen Gruppen zusammenzufassen, welche zum Erhebungszeitpunkt noch nicht vorlagen. Die Objekte können dabei sowohl Individuen (wie befragte Personen), Gegenstände (zum Beispiel verschiedene Produkte) als auch Aggregate (wie Länder oder Organisationen) sein. Durch die Anwendung clusteranalytischer Verfahren, kann man diese Objekte an Hand erhobener Merkmalsausprägungen (z.B. Einkommen, Berufsstatus oder aber auch Bruttonationaleinkommen und Entwicklungsstand bei Ländern) zu homogenen Gruppen zuordnen, um seine vorliegenden Daten zu strukturieren.

Die Clusteranalyse nimmt damit vor allem eine Vorbereitungsfunktion für weitergehende Analysen ein, welche eine exakte Gruppenzuordnung als Prämisse haben (wie zum Beispiel die Diskriminanzanalyse). Im Gegensatz zur Faktorenanalyse ist damit nicht die Gruppierung von Variablen von Interesse, sondern die Gruppierung der erhobenen Fälle (z.B. Personen).

Das Ziel einer Clusteranalyse sollte damit aber auch eine theoriegeleitete Strukturierung der vorliegenden Untersuchungseinheiten sein, um Typologisierungen wie sozialer Status von Personen oder ökonomische Entwicklungsniveaus von Ländern, auf der Grundlage theoretischer Überlegungen vorzunehmen. Dies impliziert eine Datenreduktion als Strukturierungsmaßnahme, wir wollen meist bestimmte Typologien von Objekten charakterisieren. Hieraus resultiert wiederum auch eine Vereinfachung der Datenstruktur. Wissen wir, dass eine bestimmte Gruppe von Personen gleichrangige Charakteristika aufweisen (d.h. sie bilden Typologien), ist es kostengünstiger nur einige wenige Einheiten dieser Gruppe zu erheben, anstatt eine mengemäßig große Anzahl.

Damit ist die Clusteranalyse kein direktes eigenständiges Verfahren, sondern ein Sammelbegriff für statistische Verfahren, welche es ermöglichen, einen Datensatz durch Gruppenzuordnungen zu strukturieren. Dabei handelt es sich bei diesen Verfahren um einen Subtyp

strukturentdeckender Verfahren als Teilmenge der multivariaten Verfahren. Regressionsanalytische Verfahren wiederum werden der zweiten Kategorie der strukturprüfenden Verfahren zugeordnet.[1]

Anwendung kann die Clusteranalyse damit interdisziplinär in vielen Bereichen finden. Biologische Populationen, archäologische Kategorisierungen von Funden, psychologische und medizinische Typenbildungen von Krankheiten, Clusterbildungsprozesse in der Informatik, Materialtypologisierungen in der Chemie und den Ingenieurwissenschaften sowie soziologische Untersuchungen sozioökonomischer Schichten oder betriebswirtschaftliche Kategorien des Käuferverhaltens sind dabei nur ein kleiner Ausschnitt des Möglichen.

Nicht zuletzt erlaubt die Klassifikation der Untersuchungseinheiten durch custeranalytische Verfahren eine Reduzierung der Komplexität von Datensätzen. Kategorisierung durch Clusterbildung erlaubt es uns z.B. Typen von Durchschnittswählern oder sozioökonomisch spezifischer Haushalte zu generieren.

Ziel dieses Skripts ist daher eine schrittweise Heranführung an die technischen Hintergründe der **clusteranalytischen Verfahren**, ihre Anforderungen an die Datenlage und ihre potentiellen Anwendungsbereiche. Mathematische Formalismen sind dabei unumgänglich, aber sollten nicht abschreckend wirken; sie stellen nur eine Anfangs gewöhnungsbedürftigere Art der Beschreibung dar. Lässt man sich aber auf diese ein, wird man mit einem hohen Maße an Prägnanz und Genauigkeit belohnt. Es wird aber aus diesem Grund versucht, jede mathematisch-formale Beschreibung kurz in ihrer Sinnaussage widerzuspiegeln.

1 Hierunter fallen Verfahren, welche Beziehungen kausalanalytisch untersuchen wie regressionsanalytische Verfahren.

KAPITEL 2

Grundlagen der Clusteranalyse

2.1 Einordnung der Clusteranalyse in den Bereich statistischer Verfahren

Als prominenteste Einführung in die statistischen Verfahren trifft man meist auf die Regressionsanalyse. An diesem Beispiel lässt sich sehr gut verdeutlichen, wie anders clusteranalytische Verfahren als Vertreter der strukturprüfenden Verfahren vorgehen. Die Regressionsanalyse untersucht immer einen (vermuteten) Zusammenhang zwischen verschiedenen (Zufalls-)Variablen. Damit steht immer ein Ursache-Wirkungszusammenhang im Mittelpunkt.

Ein klassisches Beispiel einer (multiplen) Regression wäre: Wir wollen untersuchen ob das Alter einer Person (in Jahren) und ihre Dauer einer Betriebszugehörigkeit (in Jahren) einen Einfluss auf das Einkommen (in Euro) ausüben. Im Bereich der Clusteranalyse interessiert uns die Analyse von Zusammenhängen zwischen verschiedenen *nicht*. Es ist dabei vielmehr von Interesse, in was für unbekannte (homogene) Subgruppen sich die verschiedenen Personen klassifizieren lassen. Dies unterscheidet clusteranalytische Verfahren auch von Strukturgleichungsmodellen. Letztere haben auch das Ziel der empirischen Überprüfung vermuteter Ursache-Wirkungsbeziehungen.[1] Clusteranalytische Verfahren bedienen sich damit keiner inferenzstatistischer Methoden, sondern klassifizieren die Untersuchungseinheiten an Hand ihrer konkreten Ausprägungen. Damit handelt es sich vor allem auch um explorative Verfahren.

1 Im Gegensatz zur klassischen Regressionsanalyse werden neben direkt beobachtbaren (manifesten) Variablen zusätzlich nicht direkt messbare, latente Variablen involviert.

2.2 Grundprinzip der Clusteranalyse

Wie lässt sich dieses Vorgehen adäquat beschreiben? Unser Ziel ist, eine Zuordnung unserer erhobenen Objekte zu möglichst **ähnlichen/homogenen** Gruppen zusammenzufassen. Homogenität als Ziel ist dabei zu verstehen als:

- hohe Intracluster-Homogenität - die Elemente *eines* Clusters sollen also möglichst ähnlich sein
- geringe Intercluster-Homogenität - die Unterschiede zwischen den Elementen *verschiedener* Cluster sollen möglichst groß sein

Hohe Intracluster-Homogenität und geringe Intercluster-Homogenität finden ihre Äquivalenz in einer niedrigen Varianz innerhalb der Cluster und einer hohen Varianz zwischen den Clustern.

Ist zu erwarten, dass diese Annahmen nicht erfülllt werden können, macht eine Clusteranalyse wenig Sinn. Die Inspektion der Verteilungen der relevanten Zufallsvariablen und ihrer elementaren Kennzahlen als Vorabtest kann dabei eine erste Orientierung geben, ob sich die Untersuchungsobjekte relevant zu unterscheiden scheinen.[1] Theoretische Fundierungen zur Typologisierung bieten ebenso eine Hilfe.

Von weiterem Interesse im Rahmen dieser Einführung sind vor allem die deterministischen Clusterverfahren. Deterministisch bedeutet hierbei, das wir eine exakte Zuordnung jeder Person zu einem *und nur einem* Cluster vornehmen. Ob eine Person $i \in I$ (Gesamtanzahl Objekte) damit Teil des Clusters $k \in K$ ist, lässt sich mit den beiden Komplementärwahrscheinlichkeiten 0 und 1 angeben.

Bezeichnen wir die Menge aller Cluster als $I_k = C_1,...,C_n$, so ergibt sich als Bedingung:

$$\bigcup_{k=1}^{n} C_k = I,\ C_k \cap C_j\ = \emptyset,\ k \neq j \tag{2.1}$$

Mathematisch ausgedrückt bedeutet dies nichts anderes als: Es soll keine Überlappung der Cluster geben. Jedes Element/Objekt wird exakt nur einem Cluster zugeordnet. Gehöre

1 Wäre die gemessene Varianz einer Zufallsvariable nahe Null, würde eine Clusteranalyse wegen mangelnder Variation der Daten keine Ergebnisse liefern können. Wir haben es dann nur mit einer homogenen Gruppe - unserem Gesamtdatensatz zu tun.

ich zum Cluster C_k, kann ich nicht auch zum Cluster C_j gehören ($\emptyset$ = Leere Menge), unter der Bedingung, dass beide Cluster nicht identisch sind ($C_k \neq C_j$).[1]

2.3 Modellgüte clusteranalytischer Verfahren

Die grundlegende Prüfung einer adäquaten Modellgüte lässt sich strukturieren als:

- *Prüfung der Modellanpassung* - Das Ziel ist dabei zu prüfen, wie gut die berechneten Modellergebnisse mit den empirischen Daten übereinstimmen.
- *Prüfung der inhaltlichen Interpretierbarkeit* - Trifft die datenanalytische Interpretierbarkeit auf keinerlei mögliche theoretische Interpretierbarkeit, ist die beste statistische Zuordnung obsolet.
- *Validitätsprüfung* - Validieren die Zuordnungen an Clustern die theoretischen Vorüberlegungen?

Treffen die Kriterien der Modellgüte nicht zu, so ist eine Analyse der Fehler unumgänglich. In der Regel trifft man dabei auf folgende Kategorien von Fehlern, wo Verbesserungen passieren müssen:

- *Theoriefehler* - Die postulierten theoretischen Vermutungen sind schlichtweg nicht adäquat und falsch formuliert.
- *Datenfehler* - Inadäquate Zusammenführungen von Variablen (Informationsverluste) oder stark fehlerhafte Erhebungen der Datengrundlagen verhindern sinnvoll interpretierbare Zuordnungen der Objekte zu Clustern.
- *Methodenfehler* - Clusteranalytische Verfahren sind schlicht nicht das richtige methodische Mittel zur Analyse des vorliegenden Datensatzes.

Durch die Fokussierung auf diese Kategorien lässt sich in der Regel jeder Fehler bestimmen.

1 Alternativ behandeln dies probabilistische Clusterverfahren (z.B. Fuzzy Clustering), wo jedes Objekt Wahrscheinlichkeiten der Zuordnungen für jeden Cluster erhält. Damit erfolgt dabei keine exakte Zuordnung. Probabilistische Clusterverfahren werden im Rahmen dieses Skripts aber nicht behandelt.

KAPITEL 3

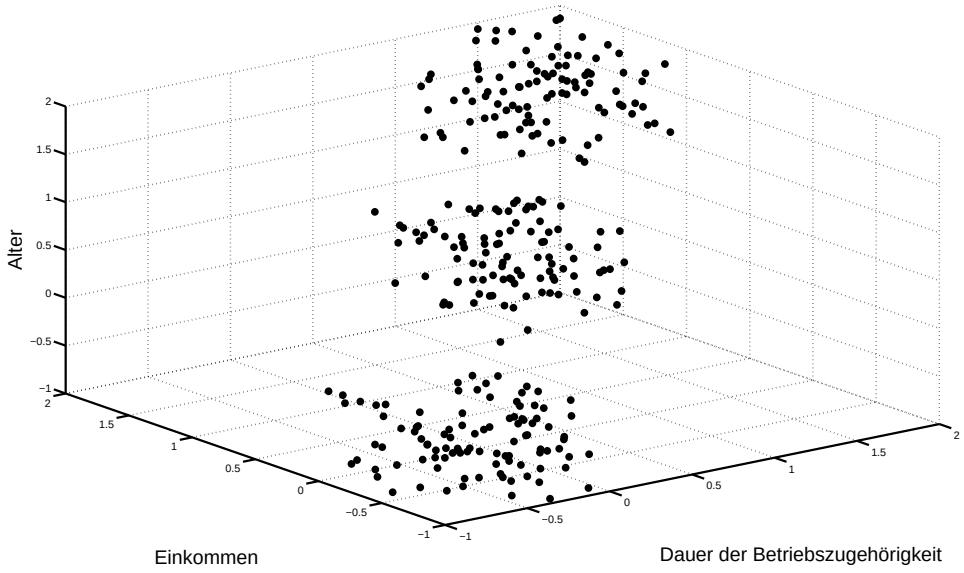
Das Konzept der Distanz- und Ähnlichkeitsmaße als Grundlagen für clusteranalytische Untersuchungen

Clusteranalytische Untersuchungen beruhen elementar auf der Betrachtung der Ähnlichkeit (oder als äquivalente Messung der Distanz) der Untersuchungsobjekte zueinander. Deshalb ist es unumgänglich, vor der Vorstellung der konkreten Verfahren zur Clusterbestimmung, zu klären, was man algebraisch und statistisch unter dieser Ähnlichkeit (und analog Distanz) von Objekten versteht. Dazu stellen wir uns unsere Untersuchungseinheiten in einem n-dimensionalen Raum vor (dem R^n). Für Objekte mit drei Ausprägungen erhalten wir zum Beispiel den dreidimensionalen Raum R^3, der gerade noch in dem Bereich des menschlich Vorstellbaren liegt.

Abbildung 3.1 veranschaulicht eine simple dreidimensionale Darstellung einer Gruppe von Personen, welche nach den Merkmalen der jeweiligen Achsen geplottet sind: Dabei sind die drei Variablen standardisiert, um sie einerseits direkt vergleichbar zu machen und andererseits unverzerrt plotten zu können.[1] Wie bereits klar erkennbar ist, scheinen wir drei Gruppen von Personen getroffen zu haben, welche sich von ihren Ausprägungen auf den drei Variablen relevant unterscheiden; es sind klar drei Punktwolken erkennbar. Unterteilt man diese Personen in drei Cluster, so ergibt sich Abbildung 3.2. Hier sind nun die Einteilungen der Beobachtungen in drei verschiedene Gruppierung grafisch erkennbar. Es gibt also Möglichkeiten, einen Datensatz, welcher Informationen über die Untersuchungseinheiten widerspiegelt, zu benutzen, um ebendiese in Cluster zu kategorisieren. Innerhalb der eingefärbten Cluster scheinen die Personen also ähnlicher zueinander zu sein als zwischen den Clustern. Hier sollten wir uns bereits klar machen: Wir betrachten zwar inhaltlich und von den theoretischen Vorüberlegungen her immer die Ähnlichkeit von Objekten;

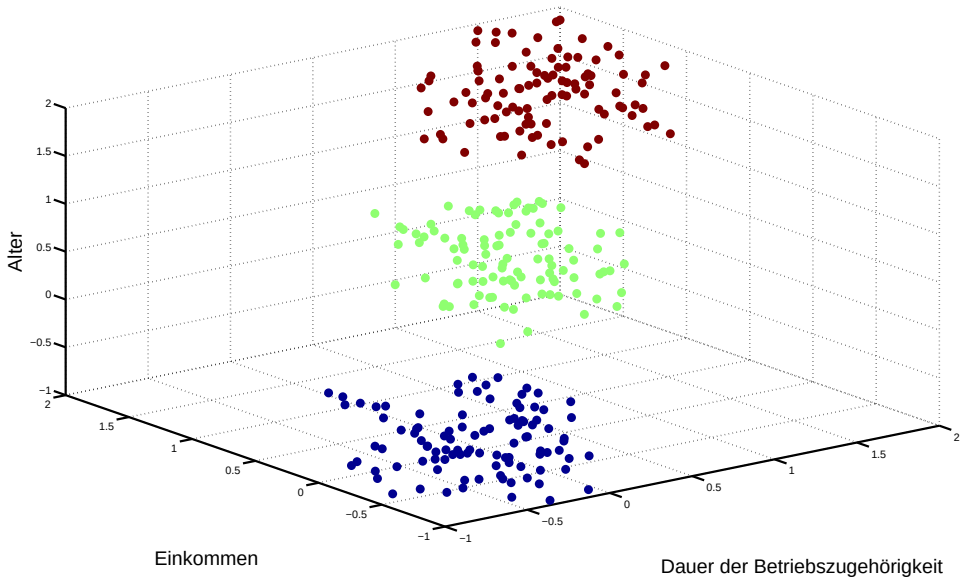
1 Hierbei ist relevant: Es handelt sich rein um zufallsgenerierte Daten, welche wenn nur zufällig einen realen Sachverhalt widerspiegeln.

Abbildung 3.1: Beispiel 3D-Plot - Ohne Cluster



Quelle: Eigene Darstellung.

Abbildung 3.2: Beispiel 3D-Plot - Mit Cluster



Quelle: Eigene Darstellung.

Mathematisch genauer ist aber der Begriff des Abstand. [1]

Damit muss nun im Folgenden spezifiziert werden, was wir unter dem Begriff Abstand und Ähnlichkeit verstehen können, wie wir diesen bestimmen und wann wir es mit der Ähnlichkeit oder Distanz (als Äquivalent zum Abstand) zu tun haben.

Relevant ist vor allem, ab welchem Messniveau unserer Daten eine geometrische Interpretation des Abstands zulässig ist.

Deshalb ist es vorab noch einmal hilfreich, sich die unterschiedlichen Skalenniveaus mit ihren jeweiligen Interpretationsmöglichkeiten ins Gedächtnis zu rufen:

- *Nominal*: Nur Aussagen über die Gleichheit von zwei Objekten sind möglich:

$$x = x' \ \vee \ x \neq x'$$

- *Ordinal*: Rangordnung sind möglich. Wenn für zwei Objekte i und j gilt: $x_i \neq x_j'$, dann gilt:

$$x_i \succ x_j' \ \vee \ x_i \prec x_j'$$

- *Intervale*: Exakte Bestimmungen der Rangabstände sind nun möglich. Ein Abstand von 4 ist also doppelt so hoch wie 2.

- *Ratio:* Zusätzlich zur Bestimmung der exakten Rangabstände existiert ein natürlicher Nullpunkt.

Je nach Skalenniveau treffen wir dabei auf unterschiedliche Kennzahlen zur Verdichtung der Homogenität zweier Fälle auf eine Kennzahl. Für nominale (und ordinale) Daten haben wir den Typus der Ähnlichkeitsmaße:

$$\begin{aligned} s_{nm} &= s_{mn} \\ s_{nm} &\leq s_{nn}, \end{aligned}$$

wobei gilt: $n, m = 1,...,N$. Damit muss die Ähnlichkeit zwischen zwei Objekten logischerweise aus beiden Betrachtungswinkeln identisch sein. Gleichzeitig gilt $s_{nm} \leq s_{nn}$, da die Homogenitätsanforderung eine höhere Intracluster- als Intercluster-Homogenität fordert. Die Ähnlichkeit eines Objekts zu sich selber muss damit logischerweise größer sein als zu irgendeinem anderen Objekt.

1 Der minimale Abstand zwischen verschiedenen Personen lässt sich aber auch als größtmögliche Ähnlichkeit begreifen.

Analog gelten für metrisches Messniveau die Distanzmaße d als Abstand zweier Fälle zueinander:

$$d_{nm} \leq d_{nm} \quad \wedge \quad d_{nm} \geq 0$$
$$d_{nn} = 0$$

Die Distanz eines Objekts zu sich selber muss natürlich Null entsprechen, die Distanz eines Objekts zu einem anderem folglich größer oder gleich Null. Die Distanz von Objekt n zu Objekt m muss logischerweise der Distanz von m zu n entsprechen.

Fasst man die Distanzen (d(n,m)) oder Ähnlichkeiten (s(n,m)) aller Objekte paarweise zusammen, erhält man eine Distanz- ($\mathbf{D} = d_{nm}$) oder Ähnlichkeitsmatrix ($\mathbf{S} = s_{nm}$).

Distanzmaße finden ihre Anwendung bei metrischen Variablen, da nur dort eine geometrisch exakte Erfassung (4 ist doppelt so viel Abstand wie 2) zulässig ist.[1] Haben wir es mit nominalen Variablen zu tun, betrachten wir i.d.R. die Ähnlichkeit als Anteil der Übereinstimmung zweier Objekte auf ihren Ausprägungen.

Bevor wir zu den konkreten Maßzahlen übergehen, müssen wir jedoch vorab das Konzept der Ähnlichkeits- und Distanzmatrix spezifizieren.

3.1 Das Konzept der Distanz- und Ähnlichkeitsmatrix

Der Homogenitätsvergleich der metrischen Kennzahlen beruht immer auf dem Konzept der Distanz. Wir wollen unsere erhobene **Rohdatenmatrix** von Objekten mit Merkmalsausprägungen:

	Merkmal 1	$\cdots$	Merkmal k	$\cdots$	Merkmal p
Objekt 1	x_{11}	$\cdots$	x_{1k}	$\cdots$	x_{1p}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
Objekt i	x_{i1}	$\cdots$	x_{ik}	$\cdots$	x_{ip}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
Objekt n	x_{n1}	$\cdots$	x_{nk}	$\cdots$	x_{np}

dafür in eine **Distanzmatrix** umwandeln:

1 Eine gewisse Aufweichung wird später bei ordinalen Variablen vorgenommen.

	Objekt 1	$\cdots$	Objekt i	$\cdots$	Objekt n
Objekt 1	d_{11}	$\cdots$	d_{1i}	$\cdots$	d_{1n}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
Objekt i	d_{i1}	$\cdots$	d_{ii}	$\cdots$	d_{in}
$\vdots$	$\vdots$		$\vdots$		$\vdots$
Objekt n	d_{n1}	$\cdots$	d_{ni}	$\cdots$	d_{nn}

Zur Bewertung der Abstände/Distanzen zweier Objekte i und n auf ihren Merkmalsausprägungen k und p dienen uns die, gleich auch formal aufgeführten, Minkowski-Metriken als Kennzahlen.

Da in unserer Distanzmatrix natürlich gilt $d_{n1} = d_{1n}$, ist die Marix symmetrisch an ihrer Hauptdiagonalen, welche nur Nullen enthält. Der Grund ist logisch: die Distanz eines Objekts zu sich selber kann nur Null sein. Damit ergibt sich als reduzierte Matrix **D**:

	Objekt 1	$\cdots$	Objekt i	$\cdots$	Objekt n
Objekt 1	0				
$\vdots$	$\vdots$	$\ddots$			
Objekt i	d_{i1}	$\cdots$	0		
$\vdots$	$\vdots$			$\ddots$	
Objekt n	d_{n1}	$\cdots$	d_{ni}	$\cdots$	0

Für unsere *nominalskalierten* Variablen lässt sich durch dasselbe Prinzip eine **Ähnlichkeitsmatrix** konstruieren:

	Objekt 1	$\cdots$	Objekt i	$\cdots$	Objekt n
Objekt 1	1				
$\vdots$	$\vdots$	$\ddots$			
Objekt i	s_{i1}	$\cdots$	1		
$\vdots$	$\vdots$			$\ddots$	
Objekt n	s_{n1}	$\cdots$	s_{ni}	$\cdots$	1

Unsere Hauptdiagonale besitzt nun durchgehend den Wert 1 und unsere Kennzahlen sind eine andere, da wir nun die **Ähnlichkeiten** und nicht mehr die **Distanzen** betrachten. Weiterhin können die Werte für unsere Kennzahl s der Ähnlichkeitsmatrix **S** nur zwischen 0 und 1 schwanken. Bei 0 gibt es keinerlei Übereinstimmungen zwischen zwei Objekten auf ihren kategorialen Ausprägungen, eine 1 ist gleichbedeutend mit vollständiger (prozentualer) Übereinstimmung. Das grundlegende Prinzip der Konstruktion ist aber zwischen beiden Varianten (Ähnlichkeits- und Distanzmatrix) identisch. Wir berechnen paarweise für alle

Objekte diese Werte, die Hauptdiagonale enthält nur den Wert 1, da die Ähnlichkeit einer Person mit sich selber zu 100 Prozent übereinstimmt.

Da wir nun das Konzept der Ähnlichkeits- und Distanzmatrix haben, können wir uns auf eine genauere Betrachtung der Berechnung der Maßzahlen fokussieren. Unser eigentliches Ziel ist nämlich die Zuordnung der Objekte zu verschiedenen Clustern; die Kennzahlen dienen uns dabei als Hilfsmittel zur Bewertung. Auf Basis ebendieser lassen sich unsere Untersuchungseinheiten mit Hilfe bestimmter Algorithmen strukturieren und fusionieren.

3.2 Ähnlichkeitsmaße für nominalskalierte (binäre) Variablen

Im Rahmen dieser Einführung werden die nominalen Kennzahlen vorgestellt, welche auf dem Prinzip des 'Matching Koeffizienten' (M-Koeffizient) beruhen.[1] Liegen binäre Merkmale vor, so hat man es mit Dummy-Variablen zu tun.[2] Die Gesamtanzahl an Übereinstimmungen ($a_{nm} + e_{nm}$) zwischen zwei Objekten/Personen (I_n und I_m) lässt sich mit einer Kontingenztabelle überprüfen:

I_n/I_m	1	0	
1	a_{nm}	c_{nm}	$a_{nm} + c_{nm}$
0	b_{nm}	e_{nm}	$b_{nm} + e_{nm}$
	$a_{nm} + b_{nm}$	$c_{nm} + e_{nm}$	p

Auf Basis dieser Kontingenztabelle lässt sich der verallgemeinerte 'Matching-Koeffizient' für die Berechnung von Ähnlichkeitsmaße für nominalskalierte Variablen beschreiben als:

$$s_{nm} = \frac{a_{nm} + \delta e_{nm}}{a_{nm} + \delta e_{nm} + \lambda(b_{nm} + c_{nm})} \quad , \tag{3.1}$$

mit δ als Gewichtungsfaktor für die Bedeutung des simultanen Nicht-Auftretens des Merkmals (0/0) und λ als Gewichtungsfaktor für die Bedeutung der beiden Zellen, für welche zwei Objekte die nicht übereinstimmenden Ausprägungen ((0/1) und (1/0)) haben. Oftmals sind die Gewichtungsfaktoren aber einfach: $\lambda = \delta = 1$, da keine explizite Gewichtung benötigt wird oder gewollt ist. Diese spezifischen Gewichtungsfaktoren sind also optional. Eine Übereinstimmung ist damit sowohl vorhanden, wenn beide Personen das Merkmal aufweisen[3], als auch, wenn sie simultan das jeweilige Merkmal nicht aufweisen.

1 Weitere Maßzahlen sind z.B. der Tanimoto-Koeffizient oder der Russel & Rao (RR)-Koeffizient. Diese stellen bestimmte Erweiterungen des Koeffizienten durch Gewichtungen dar (welche hier im Folgenden allgemein vogestellt werden).

2 In der Regel: 1 = Ausprägung liegt vor, 0 = Ausprägung liegt nicht vor.

3 Z.B. Geschlecht = 1 für Merkmal Weiblich vorhanden mit dem Komplementärereignis Geschlecht = 0 für Männlich.

Die Gewichtung beider Varianten ist dabei identisch. Die Gewichtungsfaktoren sind dabei $\lambda = \delta = 1$. Formal ausgedrückt:

$$s_{nm} = \frac{a_{nm} + e_{nm}}{p} \tag{3.2}$$

Möchte man die Übereinstimmungen (also $I_n = I_m = 1$ oder 0) höher gewichten, um ein exakteres Ähnlichkeitsmaß zu erhalten, kann man zusätzlich einen Gewichtungsfaktor u implementieren und wie folgt vorgehen:

$$s_{nm} = \frac{u(a_{nm} + e_{nm})}{u(a_{nm} + e_{nm}) + (1 - u)(b_{nm} + c_{nm})} \tag{3.3}$$

Der Faktor u ist dabei der Gewichtungsfaktor für das simultane Auftreten des Merkmals bei beiden Objekten. $(1-u)$ als Komplementärereignis ist die Gewichtung für das simultane Auftreten des Nichtvorhandenseins des Merkmals. Die spezifischen Gewichtungsfaktoren sind dabei wieder $\lambda = \delta = 1$.

Stehen nur die positiven Übereinstimmungen im Zentrum des Interesses und ist das simultane Nicht-Auftreten des Merkmals irrelevant, ergibt sich:

$$s_{nm} = \frac{a_{nm}}{a_{nm} + b_{nm} + c_{nm}} \tag{3.4}$$

Dabei wird also der Gewichtungsfaktor $\lambda = 0$ gesetzt.

3.3 Nominalskalierte, mehrstufige Variablen

Nominalskalierte Variablen werden somit auf der Grundlage ihrer Ähnlichkeit zueinander mit dem allgemeinen Ähnlichkeitskoeffizienten erfasst. Ähnlichkeit zweier Objekte auf einem nominalen Messniveau mit mehr als zwei Ausprägungen lässt sich, basierend auf dem Prinzip für zweistufige Merkmale, dabei mit dem verallgemeinerten M-Koeffizienten erfassen:

$$s_{nm} = \frac{u_{nm}}{p} \tag{3.5}$$

Wir betrachten also die Übereinstimmungen u_{nm} in Relation zu allen Kombinationen (p, z.B. vier Ausprägungen auf der Variable Freizeitverhalten) der Ausprägungen $\mathbf{x}_n$ und $\mathbf{x}_m$ der beiden Objekte n und m. Zu beachten ist dabei die Skaleninvarianz; wir haben nominalskalierte Variablen und können diese beliebig transformieren, solange die Zuordnung zu den Merkmalen genau (d.h. trennbar) bleibt. Der Wertebereich der Kennzahl ist analog wie bei den normalen Matching-Koeffizienten $0 \leq s_{nm} \leq 1$. Die Anzahl der

Ausprägungen der nominalen Variablen wird dabei als irrelevant betrachtet.[1]

Damit reduziert sich bei nominalen Variablen die Messung der Ähnlichkeit zweier Objekte auf die Betrachtung ihrer Übereinstimmungen der Ausprägungen in prozentualer Sichtweise. Der Wertebereich der Maße für nominalskalierte Variablen liegt damit logischerweise zwischen 0 (keine Übereinstimmungen) und 1 (nur Übereinstimmungen). Transformationen der Datenstruktur sind (beinahe) beliebig möglich; es muss nur sichergestellt sein, dass die Ausprägungen (Kategorien) der Variablen unterscheidbar bleiben. Optional lässt sich natürlich jedes Ähnlichkeitsmaß s_{ij} zweier Objekte i und j in ein Distanzmaß umformen:

$$d_{nm} = 1 - s_{nm} \tag{3.6}$$

3.4 Distanzmaße für ordinalskalierte Variablen

Nominalskalierte Ähnlichkeitsmaße basieren nur auf der Betrachtung von Übereinstimmung und Nicht-Übereinstimmung als Kategorien. Bei ordinalen Variablen treffen wir nun auf eine Rangfolge der Variablenausprägungen. Zwei Objekte sind damit umso ähnlicher, je näher die jeweiligen Ausprägung hinsichtlich der Rangordnung beieinander liegen. Eine Lösung für ordinale Variablen wäre Dichotomisierung für jedes Merkmal. Die Berechnungen der Ähnlichkeiten wären dann mit den obigen Maß für dichotome Variablen möglich. Der Nachteil dieser Vorgehensweise ist ein Informationsverlust, da wir nicht mehr direkt alle Ausprägungen und ihre Rangfolgen simultan betrachen können. Die zweite Lösung wäre die Betrachtung der Ausprägungen als quasi-metrisch, indem man Werte 1,...,n vergibt (und damit die Rangfolge bildet und betrachtet). Damit haben wir aber eine vorgegaukelte Genauigkeit, die eigentlich nicht vorliegt. Wir betrachten ordinale Rangfolgen als geometrisch exakte Abstände/Distanzen (da wir die Distanzmaße als Grundlage nehmen), obwohl die Daten diese Informationen nicht widerspiegeln.

Der Umgang mit ordinalen Variablen stellt damit das Sorgenkind der Clusteranalyse dar. Eine wirklich adäquate Behandlung ist mit den gängigen Maßen nicht möglich. Da aber gerade die Sozialwissenschaften durch einen methodisch nicht zu verachtenden Fokus auf Daten der Umfrageforschung häufig mit ordinalen Variablen konfrontiert sind, bleiben momentan nur diese beiden Lösungen:

1. Dichotomisierung von k Merkmalen durch k Dummies (ohne Vergleichsgruppe) und Anwendung nominaler (Ähnlichkeits-)Kennzahlen.

1 Es lässt sich über eine modifizierte Version aber auch eine Gewichtung einführen, damit Übereinstimmungen bei vielen Ausprägungen einer Variablen als höher betrachtet werden.

2. Auffassung der Variablen als quasi-metrisch (die Schwelle sollte mindestens bei fünf und optimaler bei sieben Ausprägungen liegen) und Anwendung der gleich folgenden metrischen Distanzmaße.

Eine eigene Kategorie für Variablen auf ordinalem Messniveau existiert nicht. Dies sollte immer im Hinterkopf sein, um Restriktionen bei zu weitreichenden Interpretationen der Ergebnisse bewusst zu sein.
Als Empfehlung dient an dieser Stelle aber trotzdem: Umkodierung der ordinalen Variablen in Rangfolgen, diese als quasi-metrisch betrachten und damit Distanzmaße zur Berechnung anwenden.

3.5 Distanz- und Ähnlichkeitsmaße für metrische Variablen

Bei metrischen Daten können wir uns nun der vollen Wirkungsweise geometrischer Interpretationen bedienen, da die Abstände zwischen den Ausprägungen exakt angebbar sind. Haben wir bei nominalen Variablen wegen der eingeschränkten Interpretation uns mit der reinen, prozentualen Ähnlichkeit zweier Objekte begnügt, können wir nun die exakte (geometrische) Distanz zwischen zwei Personen in einem n-dimensionalen Raum angeben.

Ein allgemeines, gebräuchliches Distanzmaß in der mathematischen Algebra ist die Minkowski-q-Metrik:

$$d(q,p)_{ij} = \left[\sum_{i=1}^{n} |x_{ik} - x_{jk}|^p\right]^{\frac{1}{p}} , \tag{3.7}$$

wobei p für den Metrikparameter der Minkowski-Metrik steht und q die Dimensionszahl angibt (z.B. drei für drei Variablen im R^3), x_{ik} und x_{jk} stehen für die konkreten Ausprägungen der Objekte i und j auf der k-ten Variable. Um Abweichungen nach unten **und** nach oben betrachten zu können, wird der Betrag betrachtet.

Der Faktor $\frac{1}{p}$ ist dabei gleichbedeutend mit der p-ten Wurzel, weswegen man manchmal auch auf die folgende Schreibweise trifft:

$$d(q,p)_{ij} = \sqrt[p]{\sum_{i=1}^{n}(x_{ik} - x_{jk})^p} \tag{3.8}$$

Verallgemeinert in Vektorschreibweise lautet die Gleichung:

$$d_p(\mathbf{x}_i,\mathbf{x}_j) = \left[\sum_{i=1}^{n} |x_{ik} - x_{jk}|^p\right]^{\frac{1}{p}} = ||\mathbf{x}_i - \mathbf{x}_j||_p \tag{3.9}$$

Einschränkend für die Minkowski-Metriken ist die Skaleninvarianz; d.h. die absoluten

Größen der gemessenen Distanzen zwischen zwei Objekten sind abhängig von den gemessenen Skaleneinheiten (bei der Erhebung). Ein direkter Vergleich der Distanzen zwischen zwei (gemessenen) Variablen ist deswegen ohne standardisierende Mittel nicht möglich: Das Einkommen in Euro ist nicht direkt mit dem Alter vergleichbar.

Standardisierung erfolgt dabei nach gängiger Vorgehensweise durch eine Normierung der Messwerte (z.B. z-Standardisierung):

$$\tilde{x_{ni}} = \frac{x_{ni} - \bar{x_i}}{s_i^{(q)}} \tag{3.10}$$

mit dem arithmetischem Mittel:

$$\bar{x_i} = \frac{1}{N} \sum_{n=1}^{N} x_{ni} \tag{3.11}$$

und der Standardabweichung:

$$s_i^{(q)} = \left(\frac{1}{N} \sum_{n=1}^{N} |x_{ni} - \bar{x_i}|^p \right)^{\frac{1}{p}} \tag{3.12}$$

Unser Endresultat nach eingesetzten Werten und (wenn benötigter) Standardisierung ist eine konkrete Kennzahl für den Abstand zwischen zwei Punkten bzw. zwischen zwei Objekten/Personen. Diese Kennzahl lässt sich paarweise für alle Untersuchungseinheiten bestimmen und zu einer 'Distanzmatrix' zusammenfassen, worauf aufbauend wir mit verschiedenen Verfahren unsere Cluster bestimmen können.

3.5.1 Spezialfälle der Minkowski-Metrik

Das euklidische Distanzmaß

Als vielfach gängiger Minkowski-Parameter wird $p = 2$ gesetzt. In dem Fall erhält man den (gebräuchlichen) euklidischen Abstand:

$$d(q,2)_{ij} = \sqrt[2]{\sum_{i=1}^{n} (x_{ik} - x_{jk})^2} \tag{3.13}$$

analog zu:

$$d_2(\mathbf{x}_i,\mathbf{x}_j) = \left[\sum_{i=1}^{n} |x_{ik} - x_{jk}|^2 \right]^{\frac{1}{2}} = ||\mathbf{x}_i - \mathbf{x}_j||_2 \tag{3.14}$$

Da gilt $\sqrt[2]{x} = \sqrt{x}$, heben sich anschließend die $\sqrt{x}$ und x^2 auf. Wir machen dies, um die totalen Abstände nach unten *und* nach oben betrachten zu können in normierten (positiven) Einheiten. Das Prinzip ist insofern analog zur Methode der kleinsten Quadrate

der linearen Regressionsanalyse. Als Kennzahl für den Abstand zwischen zwei Objekten auf *allen* relevanten Variablen erhalten wir damit eine Kennzahl als Teilmenge der positiven, reelen Zahlen (R^+). Da wir unsere Distanzen quadrieren, können wir keine Werte kleiner Null erhalten.[1] Die Vektorschreibweise impliziert dabei, das wir direkt alle Ausprägungen x_k der beiden Objekte i und j in einen Vergleich einbeziehen. Dadurch kann eine hohe Distanz auf der einen Variable durch eine niedrige Distanz auf einer anderen Variable ausgeglichen werden.

Ausgeschrieben wäre dies für zwei Objekte i und j auf ihren n Ausprägungen:

$$d_2(\mathbf{x}_i,\mathbf{x}_j) = \left\| \begin{pmatrix} x_{i1} \\ ... \\ x_{in} \end{pmatrix} - \begin{pmatrix} x_{j1} \\ ... \\ x_{jn} \end{pmatrix} \right\|_2$$

mit unserem Distanzmaß als Bewertung einer simultanen Betrachtung aller Ausprägungen von Objekt i ($x_{i1}...x_{in}$) in Relation zu Objekt j und seinen Ausprägungen der Merkmale ($x_{ji1}...x_{jn}$). Führen wir dies für alle Untersuchungseinheiten durch, erhalten wir die bereits angesprochene Distanzmatrix (**D**) i.d.R. für metrisch skalierte Variablen oder Ähnlichkeitsmatrix (**S**) für nominale Kategorien.

Die City-Block-Metrik

Setzt man alternativ den Parameter der Minkowski-Metrik p = 1, erhält man die City-Block-Metrik:

$$d_1(\mathbf{x}_i,\mathbf{x}_j) = \left[\sum_{i=1}^{n} |x_{ik} - x_{jk}|^1 \right]^{\frac{1}{1}} = ||\mathbf{x}_i - \mathbf{x}_j||_1 \tag{3.16}$$

Was lässt sich als Fazit für die Minkowski-q-Metriken festhalten?

1. Es handelt sich um Distanzmaße. Am ähnlichsten sind sich die Objekte, welche zueinander (bei paarweiser Betrachtung) die *geringste* Distanz aufweisen.

2. Durch Berechnung einer Distanzmatrix aus den Datenmatrizen lassen sich Clustereinteilungen aller Elemente vornehmen, da die Angabe der Distanz möglich ist.

3. Minkowski-Metriken haben relativ strikte Ansprüche an das Datenniveau, i.d.R. sollte dies metrisch sein, da sich dann nur der mathematisch definierte Abstand als geometrische Größe erfassen lässt. Unter gewissen Bedingungen lassen sich aber auch

1 Was auch technisch nicht möglich wäre, da die Wurzel einer negativen Zahl bei geradem Wurzelexponenten nicht definiert ist.

ordinale Daten (bei genügend hohen Ausprägungen) als quasi-metrisch auffassen. Wichtig ist, die Skaleninvarianz zu berücksichtigen. Transformationen der Daten dürfen nicht das Abstandsverhältnis berühren.[1]

3.5.2 Die Mahalabonis-Distanz

Trifft man auf lineare Korrelation der Variablen untereinander, bietet es sich an, die Mahalabonis Distanz zu verwenden:

$$\mathrm{d_M}(\mathbf{x_i},\mathbf{x_j}) = (\mathbf{x_i},\mathbf{x_j})\mathbf{F}^{-1}(\mathbf{x_i},\mathbf{x_j})^{\mathrm{T}} \quad , \tag{3.17}$$

mit $\mathbf{x_i}$ und $\mathbf{x_j}$ als Spaltenvektoren der Variablenausprägungen und $\mathbf{F}$ als bekannte Kovarianzmatrix.

Die Mahalabonis-Distanz bietet eine Möglichkeit, korrelative Effekte zwischen den Variablen herauszurechnen. Zuvor korrelierte Merkmale werden dabei erst durch Datentransformation modifiziert, so dass Unkorreliertheit entsteht. Danach wird die quadrierte euklidische Distanz berechnet, welche der Mahalabonis-Distanz entspricht:

$$||\tilde{\mathbf{x}}_n - \tilde{\mathbf{x}}_m||^2 \;=\; (\mathbf{x}_n - \mathbf{x}_m)'\mathbf{K}^{-1}(\mathbf{x}_n - \mathbf{x}_m) \tag{3.18}$$

1 Die Addition einer reelen Zahl ist erlaubt. Eine Multiplikation mit zwei aber z.B. nicht; dies würde die Abstände beeinflussen.

KAPITEL 4

Algorithmen clusteranalytischer Verfahren

Die Bestimmung der Ähnlichkeits- und Distanzmaße hat uns nur eine Einschätzung geliefert, wie die Objekte zueinander stehen. Diese Abstände lassen sich dabei in einer Distanz- bzw. Ähnlichkeitsmatrix darstellen. Die Frage ist nun, wie wir darauf basierend eine Einteilung der Objekte in unterscheidbare Cluster hinbekommen. Hierzu existieren bestimmte Cluster-Algorithmen, welche, auf verschiedenen Prinzipien basierend, eine Zuordnung der Objekte zu Clustern auf der Grundlage unserer Distanz- und Ähnlichkeitsmaße vornehmen.

Kategorien der Algorithmen clusteranalytischer Verfahren sind:

1. hierarchisches Clustern,
2. partitionierende Verfahren,
3. 'Fuzzy Clustering',
4. graphentheoretische Cluster.

Im Rahmen dieser Einführung stehen im Folgenden vor allem die ersten beiden Kategorien im Vordergrund.

4.1 Hierarchische Clusterverfahren

Der hier erst vorgestellte Subtyp der hierarchischen Clusterbildung lässt sich dabei in zwei Subtypen der Vorgehensweise unterscheiden:

- agglomerative Verfahren
- divisive Verfahren

Das Ziel der *agglomerativen Verfahren* ist das sukzessive Zusammenfassen der ungruppierten Fälle. Dabei werden immer zwei weitere Objekte (oder Cluster) zu einem neuen

Cluster fusioniert. Das Endresultat ist ein letzter großer Cluster, welcher alle Untersuchungseinheiten enthält. Wir senken quasi schrittweise unsere Homogenitätsanforderung an die Cluster.[1] Ein einmal erstelltes Cluster kann damit also *nicht* wieder aufgehoben werden, sondern nur noch mit weiteren Clustern fusioniert werden.

Vice versa gehen *divisive Verfahren* vor. Alle Objekte gehören zu Beginn zu einem großen Cluster und werden sukzessiv nacheinander getrennt. Agglomerative Verfahren gehen damit von unten nach oben vor; jede Untersuchungseinheit bildet am Anfang exakt einen eigenen Cluster. Am Ende haben wir einen Cluster, welcher alle Objekte enthält. Divise Verfahren gehen von oben nach unten vor, alle Einheiten bilden am Anfang einen großen Cluster und werden nach und nach in kleinere gesplittet. Divisive Verfahren können als Zwischenschritte dabei auch immer wieder bestehende Clustergruppierungen aufbrechen und neuordnen.

Damit weisen die agglomerativen Verfahren zu Beginn des Iterationsprozesses eine perfekte Intracluster-Homogenität auf. Jeder Cluster enthält exakt ein Element. Die Bildung der folgenden Cluster erfolgt dann durch das Zusammenfassen der nächsten beiden jeweils ähnlichsten Objekte zu einem neuen Cluster. Dieser Prozess wiederholt sich, bis alle Fälle abschließend nur noch zu einem Cluster gehören. Die Schritte sind somit für agglomerative Verfahren:

1. Jedes Objekt bildet ein selbstständiges Cluster. Clusterzahl K ist identisch mit Objektzahl N.

2. Clusterpaar mit der größten Ähnlichkeit (oder geringste Unähnlichkeit) wird mit Hilfe einer Ähnlichkeits-/Distanzmatrix gesucht (das Paar mit dem geringsten Distanzmaß). Clusterpaar wird zu einem neuen Cluster zusammengefasst (K-1 Cluster nun).

3. Neuberechnung der Distanzmatrix mit der berücksichtigten Transformation zweier Cluster. Die ähnlichsten Cluster werden miteinander fusioniert.

4. Prozess setzt sich fort bis Anzahl der Cluster K = 1 ist

Im Folgenden werden alle Algorithmen vorgestellt, welche gebräuchlich in der Praxis und zusätzlich in SPSS[2] als weitverbreitetes Statistikprogramm implementiert sind.

1 Am Anfang bildet ja jede Untersuchungseinheit einen eigenen Cluster. Damit erfüllen wir perfekt die Homogenitätsanforderung innerhalb der Cluster.

2 Die Clusteranalyse ist aber auch in jedem anderen Statistikprogramm wie Stata, R oder Mathlab implementiert. Die Beispiele hier werden aber mit SPSS angeführt, da sich dieses Programm immer noch der weitverbreitesten Beliebtheit erfreut.

Dabei werden wir mit der folgenden Datenmatrix aus Abbildung 4.1 alle Verfahren durchrechnen[1] und uns dabei das Konzept des Dendogramms anschauen. ID steht für die durchnummerierte Kennziffer jeder erhobenen Person und v1 - v4 für vier zufallsgenerierte Variablen. Wie man erkennen kann, haben die Personen mit der ID 1 bis 5 auf allen Variablen gemessene Werte zwischen 1 und 3, die Personen mit der ID 6 bis 10 erhobene Messwerte zwischen 5 und 8. Damit ist intuitiv vorab klar, dass eine Lösung von zwei Clustern das Optimum bilden dürfte. Somit können wir uns rein auf den Einfluss der folgenden, verschiedenen agglomerativen Verfahren konzentrieren.

Tabelle 4.1: Beispieldatensatz

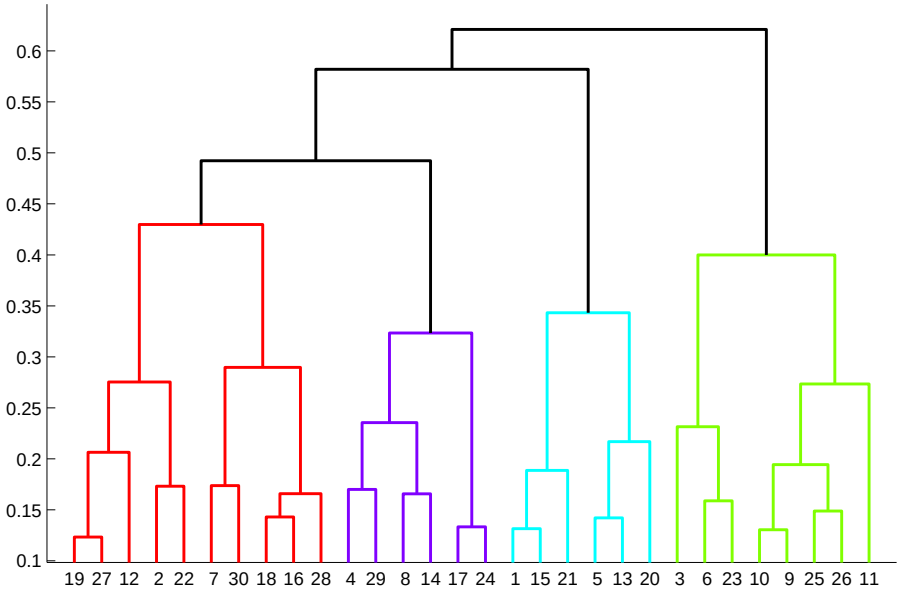
ID	1	2	3	4	5	6	7	8	9	10
v1	1	2	3	2	1	8	8	5	8	6
v2	2	2	1	3	2	5	7	6	8	7
v3	1	3	1	1	2	7	7	7	5	8
v4	3	1	3	1	1	6	8	8	6	8

1 Die Datenmatrix findet sich transponiert im Anhang A. Für die Rechnungen diese einfach in SPSS kopieren.

4.2 Grafische Visualisierung hierarchisch-agglomerativer Verfahren - Das Dendrogramm

Die Fusionierungsschritte hierarchisch-agglomerativer Verfahren lassen sich sehr gut durch ein Dendrogramm visualisieren. Dabei sind, wie in der Abbildung 4.1 dargestellt, alle Schritte des Fusionierungsprozesses unserer Objekte sehr gut sichtbar.

Abbildung 4.1: Beispiel Dendrogramm



In diesem Fall liegen keine reellen Daten vor, sondern 30 zufallsgenerierte Fälle[1]. Beispielhaft erkennt man sehr gut das Prinzip des Dendrogramms als grafische Veranschaulichung der Fusionierung unserer Cluster. Nehmen wir z.B. an, wir haben metrische Daten vorliegen, welche basierend auf einem agglomerativen Verfahren (hier dem 'average'-Algorithmus) mit der euklidischen Distanz fusioniert wurden. Die X-Achse zeigt uns unsere Objekte (hier durchnummeriert von 1 bis 30) und die Y-Achse zeigt uns die Werte des Abstandsmaßes (hier die quadrierte euklidische Metrik) an. Die Fusionierungen auf den unteren Stufen (kleiner 0.2) sind die ersten Schritte gewesen. Wir haben relativ ähnliche Personen zu der ersten Ebene an Clustern kombiniert. Bis zu einem Wert von ca. 0.4 folgen weitere Kombinationen von Clustern, welche farblich gekennzeichnet sind. An dieser Stelle existieren

1 Grafik und zufallsgenerierte Fälle sind jeweils mit Matlab erstellt. Es handelt sich um ein sehr umfangreiches und lohnendes Mathematik-Programm, das aber einer höheren Einarbeitungszeit bedarf. Bei Interesse ist für Angehörige der Universität Duisburg-Essen eine kostenlose Netzwerklizenz über den Fachbereich Mathematik zugänglich.

noch fünf Cluster (jedes in einer anderen Farbe). Nach den nächsten beiden Schritten erhalten wir nur noch einen Cluster, wie unser agglomeratives Verfahren dies vorschreibt.

Da dies natürlich *nicht* unsere gewünschte Lösung ist[1], legen wir einen Schwellenwert fest, bis zu dem eine Fusionierung der Cluster passieren soll. Bei uns ist dies ein Wert von ca. 0.4 für die euklidische Distanz. Ab diesem Wert wird dann jede weitere Fusionierung gestoppt, da die Elemente zu 'unähnlich' werden, also ihre Distanz zu hoch wird. Wir nehmen die Einteilung der Objekte in Cluster dann an diesem Schwellenwert vor; unsere Lösung wären die vier farbigen Cluster.

Hierbei wird auch noch einmal deutlich, dass die Clusteranalyse ein iteratives Verfahren ist. Eine wirklich exakte Lösung existiert nicht und muss vielmehr theoriegeleitet erfolgen. Eine Unterteilung von 30 Personen in ein einziges Cluster macht genauso wenig Sinn wie die Unterteilung in 20 Cluster.

Als Ergänzung könnte sich auch eine theoriegeleitete Lösung anbieten: Wir geben (unserem Statistikprogramm) vor, dass wir nur exakt eine bestimmte Anzahl an Clustern präferieren. Wird dieser Schwellenwert der Fusionierung erreicht, wird abgebrochen. Alternativ geben wir einen Toleranzbereich an (z.B. drei bis fünf Cluster), wonach abgebrochen werden soll.[2]

Es sollte generell bei der grafischen Analyse mittels Dendrogramm beachtet werden, dass dieses von der Wahl des Ähnlichkeits- oder Distanzmaßes abhängt. Das Ward-Verfahren z.B. weist eine andere Fusionsstruktur aufweisen, als das 'Complete-Linkage'. Ein positiver Aspekt ist aber die relativ individuelle Wahlfreiheit der Homogenitätsgrenze durch unsere Maße. Wir können selbst entscheiden, ob wir in unserem Beispiel die Grenze bei 0.3, 0.5 oder erst bei 0.8 ziehen. Rechtfertigen müssen wir uns maßgeblich vor unseren theoretischen Überlegungen.

Da wir nun das Konzept des Dendrogramms kennen, können wir uns auf die verschiedenen Algorithmen hierarchischer Clusterlösungen konzentrieren. Das 'Single-Linkage'-Verfahren, als erster Vertreter der agglomerativen Verfahren, wird dabei das erste Vorzustellende sein.

1 Würde unser Ziel die Kombination aller Elemente in ein großes Cluster sein, hätten wir dies mit der Erstellung unseres Datensatzes bereits erledigt.

2 Beide Varianten sind z.B. bei SPSS mittels der hierarchischen Clusteranalyse möglich.

4.3 Clusteranalytische Verfahren in SPSS

Als Anmerkung vorweg: alle Syntaxvarianten für die clusteranalytischen Verfahren in SPSS finden sich auch im Anhang A.

Clusteranalytische Verfahren in SPSS lassen sich über das Menu wie folgt aufrufen:

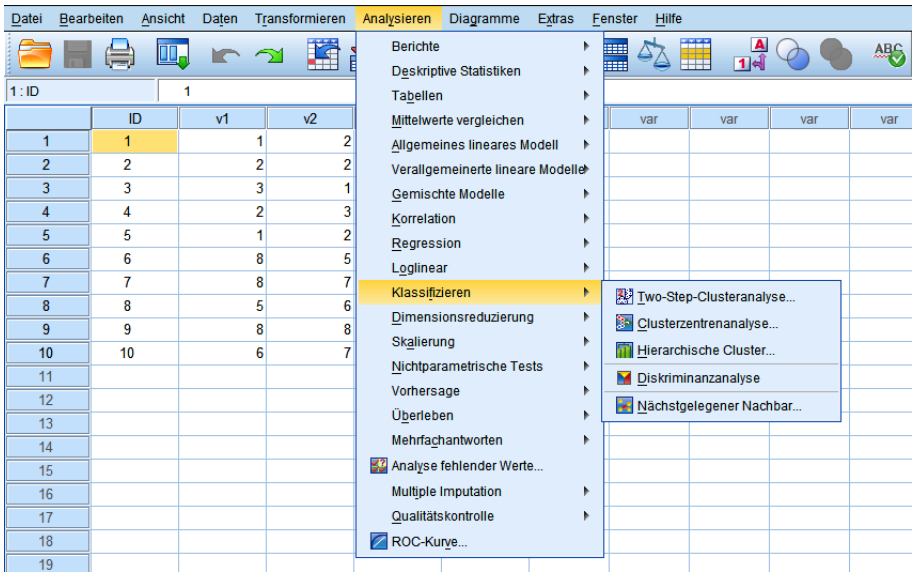


Abbildung 4.2: SPSS-Auswahlmenü Clusteranalyse

Die Option Hierarchische Cluster... steht logischerweise für die hierarchischen Clusterverfahren. Es bietet sich aber direkt an, die Steuerung von SPSS über die Syntax vorzunehmen.[1]

Für unsere vier Variable v_1, v_2, v_3 und v_4 lässt sich eine Clusteranalyse über die Syntax nach dem folgenden Schema aufrufen:

```
  CLUSTER ausgewählte Variablen
/METHOD gewählte Methode
/MEASURE= gewähltes Distanz-/Ähnlichkeitsmaß
/PRINT SCHEDULE
/PLOT VICICLE.
```

Ein Beispiel für die Methode wäre SINGLE für das 'Single-Linkage'-Verfahren ('nearest neighbour') und ein Beispiel für das Distanzmaß wäre SEUCLID für die quadrierte euklidische Distanz als metrisches Maß. Haben wir ein nominales Niveau, bietet sich

1 Alternativ sind aber natürlich auch alle nun folgenden Optionen über die Menüführung einstellbar.

z.B. JACCARD(1,0) für den Jaccard-Koeffizienten an (ein Gesamtüberblick über die integrierten Distanz- und Ähnlichkeitsmaße in SPSS lässt sich gut über die Menüführung inspizieren). Wird /PLOT VICICLE modifziert zu /PLOT DENDOGRAM VICICLE gibt SPSS zusätzlich das Denodgramm als grafische Visualisierung der Zuordnungsschritte aus. Mit der zusätzlichen Angabe von /PRINT DISTANCE wird die berechnete Distanzmatrix mit ausgegeben, die Erweiterung von /PRINT SCHEDULE zu /PRINT SCHEDULE CLUSTER(X) erweitert die Zuordnungsmatrix um die konkreten Zuordnungen der Beobachtungen zu den jeweiligen Clustern als Tabellenoutput. Wollen wir die Clusterzuordnungen aller einzelnen Fälle als zusätzliche Variable in unserem Datensatz abspeichern, rufen wir den Befehl mit /SAVE CLUSTER(X) auf. X steht dabei jeweils für die Anzahl der gewünschten Cluster, hierbei sollte man sich immer in Erinnerung halten, dass die clusteranalytischen Verfahren nie eine eindeutige Lösung liefern. Die Vorgabe der Anzahl der Cluster kann damit relativ selber bestimmt werden (z.B. gibt die Theorie eine Typologie von vier verschiedenen Wählergruppen vor - wir wählen dann z.B. vier Cluster um der Theorie gerecht zu werden). Es gibt aus statistischer Sicht oftmals keine (direkte) optimale Clusterlösung, SPSS sortiert die Fälle solange, bis sie ein einziges großes Cluster bilden. Die grafische Inspektion des Dendogramms und/oder die analytische Interpretation der Zuordnungsübersicht sollte also vor der Abspeicherung der Zuordnungen passieren. Damit kann auf dieser Grundlage eine, aus theoretischer und logischer Sicht, optimale Clusteranzahl gewählt werden.
Haben wir die Option /SAVE CLUSTER(2) angewählt, dann speichert uns SPSS eine neue Variable ab, welche für jedes Objekt die Zuordnung zu den zwei vorgegebenen Clustern enthält. Mit diesen Größen können wir dann weitergehende Verfahren rechnen, da wir nun eine Gruppenzugehörigkeit extrahiert haben.
Implementieren wir alle diese Modifikationen für eine hierarchische Clusteranalyse in unsere ursprüngliche Syntax, ergibt sich für den 'Single-Linkage'-Algorithmus:

```
  CLUSTER v1 v2 v3 v4
/METHOD SINGLE
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

Alle diese Modifikationen sind natürlich auch über die Menüführung auswählbar. Hierbei sollte aber möglichst dann nicht abschließend einfach Ok ausgewählt werden, sondern

die Option Einfügen. Hiermit schreibt SPSS die ausgewählten Operationen erst in die Syntaxdatei. Der Vorteil: man bekommt ein besseres Gefühl was gemacht wird und kann sich die Syntaxdateien abspeichern um einerseits eine Übersicht über die durchgeführten Prozeduren zu behalten und für potentielle Dritte die Transparenz des Vorgehens zu erhöhen.
Bei ungleichen Skalierungen metrischer Variablen bei der Erhebung kann es notwendig sein, die Werte vorab zu standardisieren um eine Vergleichbarkeit der Distanzmaße und damit der Clusterergebnisse zu gewährleisten. Die Auswahl zur Standardisierung funktioniert angenehm über die Menüauswahl Methode (z.B. als z-Transformation). Die Syntax ändert sich dann für unser Beispiel stark, da erst die transformierten Werte in eine externe Matrix abgespeichert und anschließend geladen werden:

```
   PROXIMITIES v1 v2 v3 v4
/MATRIX OUT('SPEICHERORT')
/VIEW=CASE
/MEASURE=SEUCLID
/PRINT NONE
/STANDARDIZE=VARIABLE Z.
CLUSTER
/MATRIX IN('SPEICHERORT')
/METHOD SINGLE
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
ERASE FILE='SPEICHERORT.
```

Der letzte Befehl ERASE FILE ist dabei nur dazu da, die temporär abgespeicherte und standardisierte Matrix wieder zu löschen. Natürlich ist es aber auch einfach möglich, die Werte der Variablen vor der Clusteranalyse zu z-transformieren.

Schauen wir uns vorab einmal die quadrierte euklidische Distanz mit SPSS[1] als Di-

1 Als Anmerkung: SPSS gibt normalerweise auch die Distanzen oberhalb der Hauptdiagonale aus. Da diese aber identisch sind mit denen unterhalb der Hauptdiagonale (da $d_{nm} = d_{mn}$), sind sie hier der Übersichtlichkeit halber nicht mit aufgeführt.

stanzmatrix[1] an (hier: ohne Standardisierung, da unsere Variablen bereits ein identisches Skalenniveau haben), so erhalten wir[2] die Abbildung 4.3. Auch hier lässt sich schon sehr

Abbildung 4.3: Distanzmatrix als SPSS-Ausgabe

Näherungsmatrix

Fall		Quadriertes euklidisches Distanzmaß									
		1: 1	2: 2	3: 3	4: 4	5: 5	6: 6	7: 7	8: 8	9: 9	10: 10
1:	1	,000									
2:	2	9,000	,000								
3:	3	5,000	10,000	,000							
4:	4	6,000	5,000	9,000	,000						
5:	5	5,000	2,000	10,000	3,000	,000					
6:	6	103,000	86,000	86,000	101,000	108,000	,000				
7:	7	135,000	126,000	122,000	137,000	148,000	8,000	,000			
8:	8	93,000	90,000	90,000	103,000	106,000	14,000	10,000	,000		
9:	9	110,000	101,000	99,000	102,000	119,000	13,000	9,000	21,000	,000	
10:	10	124,000	115,000	119,000	130,000	135,000	13,000	5,000	3,000	18,000	,000

Dies ist eine Unähnlichkeitsmatrix

gut ablesen, dass wir zwei Gruppen von Personen zu haben scheinen. Bei den Fällen 1 bis 5 haben wir sehr niedrige Distanzen im Bereich von $d_{nm} < 10$. Für die Fälle 6 bis 10 haben wir eine Distanz von: $d_{nm} < 21$. Betrachten wir aber die Distanzen über diese beiden Fallgruppen, landen wir bereits im Bereich $86 < d_{nm} < 148$.
Dies liefert uns schon einen eindeutigen Hinweis, dass zwei Gruppen von Objekten vorliegen. Wir sind nun an dem Schritt angekommen, wo wir unsere Datenmatrix in eine Distanzmatrix[3] umgewandelt haben.

Bei den nun kommenden Algorithmen wollen wir uns anschauen, wie diese einzeln vorgehen, um unsere Objekte auf Grundlage der Distanzmatrix in Cluster zu unterteilen. Dabei folgt nur noch die Ausgabe der Ergebnisse als grafische Lösung eines Dendograms mit SPSS inklusive der Angaben zur Auswahl der Verfahren. Zur Übung ist es deshalb sinnvoll, diese selbst mit der obigen Datenmatrix durchzuführen.

Nach der grundlegenden Einführung in die Menü- und Syntaxsteuerung von SPSS für die hierarchischen Clusterverfahren, können wir uns nun die jeweiligen Verfahren genauer

1 SPSS gibt Distanzmatrizen immer als Näherungsmatrizen mit dem Hinweis 'Dies ist eine Unähnlichkeitsmatrix' aus. Da wir mit der quadrierten euklidischen Distanz den Abstand zweier Objekte untersuchen, haben wir es folglich mit Unähnlichkeiten zu tun.

2 Die **Ausgabe der Distanzmatrix** lässt sich bei der Durchführung der Clusteranalyse unter der Option Statistiken auswählen. Alternativ kann innerhalb der Syntax-Eingabe die Ausgabe mit /PRINT DISTANCE eingeschaltet werden.

3 Möglich wäre natürlich auch eine Ähnlichkeitsmatrix für nominale Variablen.

anschauen.

4.4 Agglomerative (hierarchische) Verfahren in SPSS

Charakteristisch für alle agglomerativen (hierarchischen) Verfahren ist eine *perfekte* Intra-Cluster-Homogenität zu Beginn des Fusionsprozesses; alle Objekte bilden jeweils ein eigenes Cluster. Diese Anforderung wird durch den Fusionierungsprozess schrittweise aufgegeben, bis nur noch ein einziges Cluster mit allen Objekten existiert.
Die optimalen Clusterzuordnungen befinden sich irgendwo dazwischen und können mit einem Dendogram visualisiert werden.

Das agglomerative Verfahren wird bei folgenden clusteranalytischen Methoden angewandt:

- Single- und Complete-Linkage
- Mittelwertverfahren
- Zentroid- und Ward-Verfahren

Diese Verfahren lassen sich deswegen auch als hierarchisch agglomerative Verfahren bezeichnen. Weiterhin sind die agglomerativen Verfahren deterministische Verfahren, denn jedes Objekt wird eindeutig einem Cluster zugeordnet.

4.4.1 Single-Linkage-Verfahren

Single-Linkage Clustering (auch 'nearest neighbour' oder 'minimum distance method') definiert die Distanz D(X,Y) zwischen zwei Clustern X und Y als den Abstand zwischen den beiden nahesten Objekten:

$$D(X,Y) = \min(d(x,y)) \quad , \tag{4.1}$$

mit d(x,y) als Distanz zwischen den beiden Elementen $x \in X$ und $y \in Y$. Diese Distanz zwischen den beiden Clustern soll damit minimal sein. Referenzpunkt beim 'Single-Linkage'-Verfahren sind damit die jeweils nahesten Objekte der beiden Cluster. Damit sieht der agglomerative Ablauf für das Single-Linkage Clustering wie folgt aus:

1. Es wird für N Cluster eine N x N Matrix aufgestellt, welche die Abstände d(x,y) für alle Kombinationen von Clustern enthält.
2. Es werden die beiden Cluster r und s fusioniert, wobei gilt: d[(r),(s)] = min d[(x),(y)].

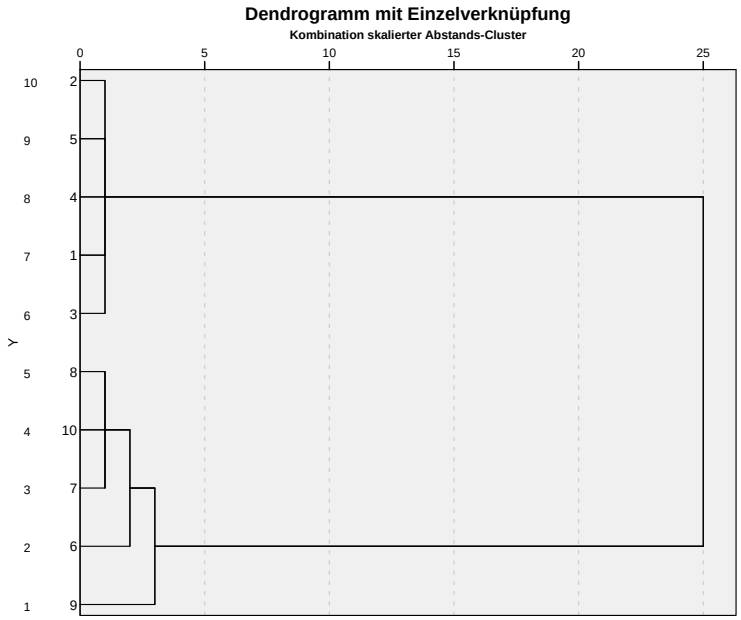
3. Es wird die Ähnlichkeitsmatrix modifiziert, durch Ersetzung der Spalten und Zeilen der Cluster r und s durch eine neue Spalte/Zeile für den (aus dem vorigen Schritt)fusionierten Cluster.

4. Neuberechnung der Ähnlichkeits-/Distanzmaße und Symbiose der nächsten beiden ähnlichsten bis Anzahl der Cluster K = 1 erreicht ist.

Was haben wir damit gemacht? Eine schrittweise, deterministische Fusionierung der immer jeweils nähesten Cluster. Am nähesten bedeutet dabei: ähnlichste Ausprägungen auf unseren Variablen (z.B. Einkommen und Alter) zwischen den *nähesten* Objekten der Cluster. Das Endresultat ist ein einziger Cluster, dies zeigt das agglomerative Vorgehen. Problematisch ist hierbei, das wir eine Kettenbildung produzieren. Wir fusionieren immer nur auf der Grundlage der beiden größten gemeinsamen Nenner, unsere Intra-Clusterhomogenitätsanforderung ist äußerst gering. Dadurch ziehen wir nach und nach Ketten von Objekten zusammen. Dies führt dazu, dass zwei stark heterogene Cluster doch wieder fusioniert werden können, da in beiden Clustern sich nur zwei Elemente sehr ähneln. Ein Vorteil ist die Identifikation von Ausreißern, welches auch die Hauptanwendung begründet. Objekte mit extremen Unterschieden zu *allen* anderen Objekten werden erst am Ende fusioniert, da ja alle Objekte erst einen eigenen Cluster bilden.

In SPSS lässt sich das Single-Linkage Verfahren (nach der Auswahl der hierarchischen Verfahren) unter dem Untermenüpunkt 'Methode' als 'Nächstgelegener Nachbar' auswählen. Dabei können wir den Algorithmus und das Proximitätsmaß unter der Kategorie 'Methode' spezifizieren. Die Ausgabe des Dendogramms muss erst noch vorab unter dem Menüpunkt Diagramme aktiviert werden.
Alternativ erfolgt die Spezifikation `/METHOD SINGLE` innerhalb der Syntax, wobei wir als Distanzmaß den quadrierten euklidischen Abstand über `/MEASURE=SEUCLID` wählen und die Ausgabe des Dendogramm durch `/PLOT DENDOGRAM VICICLE` aktivieren. Führen wir nun mit unserem Beispieldatensatz eine Clusteranalyse durch, erhalten wir als grafische Lösung das Dendogramm der Abbildung 4.4.

Abbildung 4.4: Dendogramm 'Single-Linkage'



Was zeigt sich hierbei mit dem 'Single-Linkage' Algorithmus? Die Fälle mit den ID's 1 bis 5 sind sehr homogen in ihren Ausprägungen. Sie bilden ein klares Cluster mit keinen 'Wellen' der Fusionierung. Das zweite Cluster ist in sich nicht mehr so homogen. Beziehen wir uns auf unsere Distanzmatrix für unsere Objekte, wird klar warum: Die paarweisen Distanzen der Fälle mit der ID 6 bis 10 sind höher. Hierbei fällt besonders die ID 9 auf, welche die höchsten Distanzen zu allen anderen Objekten aufweist. Wir erinnern uns: Charakteristisch für die agglomerativen Verfahren ist, dass alle Objekte zu Beginn ein eigenes Cluster bilden und die Fusionierung immer sukzessiv für die beiden Objekte mit den geringsten Distanzen zueinander erfolgt. Da die ID 9 relativ gesehen zu beinahe allen anderen Fällen die höchste Distanz besitzt, wird er erst als letzter Fall mit der Gruppe 6 bis 10 fusioniert. Hier *könnte* ein Hinweis auf einen Ausreißer vorliegen mit dem 'Single-Linkage' vorliegen (wofür aber trotzdem die Distanz hier nicht zu hoch erscheint).

Das 'Single-Linkage' Verfahren eignet sich vor allem zur Identifzierung von Ausreißerwerten als Vorabtest. Da immer die nächsten Nachbarn identifiziert werden, werden starke Ausreißer erst am Ende fusioniert. Ist das Ziel möglichst homogene Cluster zu erreichen, dann eignet sich das 'Single-Linkage' Verfahren deswegen aber nicht.

Ein weiterer nützlicher Output bei SPSS ist die 'Zuordnungsübersicht' der Abbildung 4.5.

Abbildung 4.5: Zuordnungsübersicht 'Single-Linkage' in SPSS

Zuordnungsübersicht

	Zusammengeführte Cluster			Erstes Vorkommen des Clusters		
Schritt	Cluster 1	Cluster 2	Koeffizienten	Cluster 1	Cluster 2	Nächster Schritt
1	2	5	2,000	0	0	3
2	8	10	3,000	0	0	4
3	2	4	3,000	1	0	6
4	7	8	5,000	0	2	7
5	1	3	5,000	0	0	6
6	1	2	5,000	5	3	9
7	6	7	8,000	0	4	8
8	6	9	9,000	7	0	9
9	1	6	86,000	6	8	0

Diese Übersicht zeigt uns alle vorgenommenen Fusionierungsschritte an. Die Spalte Koeffizient gibt die Distanz der beiden fusionierten Cluster an ('nächster Nachbar'!). Die geringste Distanz weisen die Objekte 2 und 5 zueinander auf; Sie werden im ersten Schritt fusioniert. Wir haben nun einen ersten Cluster mit zwei Objekten und der rechte Teil der Spalte zeigt uns an, wonach mit dem Schritt 3 ein weiteres Objekt mit diesem Cluster fusioniert wird. Dies setzt sich logisch für zwei große Cluster bis zu dem Schritt 8 fort. Bis hierhin wurden immer nur Fusionen im Bereich der ID 1 bis 5 *oder* 6 bis 10 vorgenommen. Der Schritt 9 fusioniert nun mit einer gemessenen Distanz von 86 alle Objekte zu einem letzten großen Cluster (wie unser Dendrogramm anzeigt). Hier wird eindeutig klar, dass wir zwei getrennte Cluster besitzen. Die Fusionierung im letzten Schritt führt zu einer unverhältnismäßigen Erhöhung der Distanz der zwei nähesten Objekte ('Single-Linkage').

Wie sind die Einträge unter „Erstes Vorkommen des Clusters" zu interpretieren? Schauen wir uns den Schritt 3 an: Dort werden die Cluster 2 (als Cluster 1) und 4 (als Cluster 2) fusioniert. Dem Cluster 2 wurde aber bereits im 1. Schritt des Fusionierungsprozesses der Cluster 5 zugeordnet. Die Kategorie „Erstes Vorkommen des Clusters" verweist damit für das erste Cluster auf den Fusionierungsschritt 1 um darzustellen, dass für diesen Cluster vorab bereits ein Fusionierungsschritt vorgenommen wurde. Für die Schritte 1 und 2 findet sich logischerweise kein Verweis, da hier noch ursprüngliche Cluster (also einzelne Beobachtungen!) fusioniert werden. Für den Schritt 4 werden 7 und 8 fusioniert, hierbei zeigt die Zuordnungsübersicht dann für den Cluster 2 den Hinweis auf den 2. Schritt an, da dort bereits der Cluster 8 mit dem Cluster 10 fusioniert wurde.

Also: SPSS gibt als Labelkennzeichnung immer die Nummer an, welche bei einem Fusionierungsschritt als Cluster 1 betrachtet wird. Erfolgt später dann eine weitere Zuordnung dieses Clusters zu einem anderen (als Cluster 2 dann) oder wird ein anderer Cluster diesem hinzugefügt (Cluster 1), dann verweist SPSS damit auf den Schritt, durch den dieser

Cluster zuletzt entstanden ist.

4.4.2 Der 'Complete-Linkage' Algorithmus

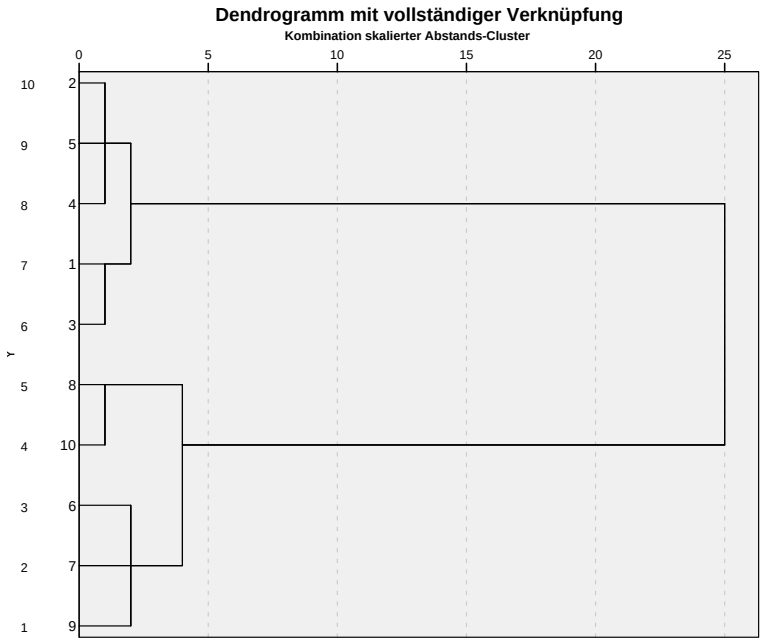
Das 'Complete-Linkage'-Verfahren (Entferntester Nachbar) basiert auf demselben Prinzip (vgl. Schritte 1-4) wie das 'Single-Linkage' mit einer Modifikation bei Schritt 2:

$$D(X,Y) = \max(d(x,y)) \tag{4.2}$$

Damit bestimmt das Complete-Linkage-Verfahren die zu fusionierenden Cluster über den Maximal-Abstand zwischen den Elementen zweier Cluster. Fusioniert werden immer die Cluster mit dem *geringsten* Maximal-Abstand. Mit 'Complete-Linkage' betrachten wir simultan alle Objekte zweier Cluster bzgl. ihrer Distanz (da nur die größte Distanz zur Geltung kommt). Die Schwierigkeit ist, dass wir weit entfernte Klassen erst spät fusionieren. Es bietet sich also an, ein 'Single-Linkage' vorzuschalten, um Ausreißer vorab zu identifizieren, da das 'Complete-Linkage' Verfahren anfällig für Ausreißer ist.

In der Menüführung von SPSS erfolgt die Auswahl analog zum 'Single-Linkage' mit der Modifikation bei der Methode auf 'Entferntester Nachbar', über die Syntax erfolgt die Auswahl des 'Complete'-Algorithmus durch `/METHOD COMPLETE.` Wenden wir als Methode 'Complete-Linkage' für unseren Beispieldatensatz an, sehen wir bereits folgende Veränderung des Dendrogramms der Abbildung 4.6.

Abbildung 4.6: Dendogramm 'Complete-Linkage



Bzgl. unserer Zuordnungsübersicht haben sich gegenüber dem 'Single-Linkage'-Verfahren (vgl. Abbildung 4.5) ab dem Schritt vier Modifikationen der Zuordnung ergeben, wie die Abbildung 4.7 zeigt.

Abbildung 4.7: Zuordnungsübersicht 'Complete-Linkage' in SPSS

Zuordnungsübersicht

	Zusammengeführte Cluster			Erstes Vorkommen des Clusters		
Schritt	Cluster 1	Cluster 2	Koeffizienten	Cluster 1	Cluster 2	Nächster Schritt
1	2	5	2,000	0	0	3
2	8	10	3,000	0	0	8
3	2	4	5,000	1	0	6
4	1	3	5,000	0	0	6
5	6	7	8,000	0	0	7
6	1	2	10,000	4	3	9
7	6	9	13,000	5	0	8
8	6	8	21,000	7	2	9
9	1	6	148,000	6	8	0

Der 'Complete-Linkage' Algorithmus stellt eine starke Fokussierung auf die Homogenität der Cluster und eignet sich daher, wenn eine hohe Homogenität der Cluster eine zentrale Forderung ist.

Single- und Complete-Linkage haben die schwächsten Anforderungen an das Datenmaterial von allen Verfahren. Unsere Ähnlichkeits- oder Distanzmatrix für die Berechnung der Distanzmaße zwischen den Objekten muss nicht zwingend metrisches Messniveau aufweisen. Es können einfach ordinale oder nominale Kategorien aus einer quantitativen Befragung vorliegen. Relevant ist nur die Wahl einer messniveau-adäquaten Kennzahl. Damit bewegen wir uns bei beiden Verfahren auch im Bereich der nichtmetrischen und mehrdimensionalen Skalierung.

Die **folgenden Mittelwertverfahren** stellen in gewisser Form Verbesserungen und Erweiterungen der beiden Basismodelle 'Single'- und 'Complete'-Linkage dar, indem sie nicht mehr nur die direkten Nachbarn der Cluster betrachten, sondern durch Mittelwertbildung die Cluster als Gesamtheit in Beziehung zu setzen. Wie gleich deutlich wird, erfordern diese aber i.d.R. metrisches Messniveau.

4.4.3 Das Average-Linkage-Verfahren

Mit dem 'Average-Linkage'-Verfahren (als eigentliche Gruppe der Mittelwertverfahren) wird zwischen zwei Clustern C_k und C_j nicht mehr nur die minimalste oder maximalste Distanz zwischen zwei Objekten betrachtet, sondern simultan für alle Objekte. Dazu wird

in jedem Cluster der Durchschnittswert der Distanzen gebildet und über alle Cluster paarweise betrachtet:

$$D(C_k,C_j) = \frac{1}{n_k n_j} \sum_{n \in C_k} \sum_{m \in C_j} d_{nm} \tag{4.3}$$

mit n_k und n_j als Anzahl der Objekte in beiden Clustern.[1]

Wir fusionieren dann diejenigen Cluster, welche als Mittelwert der Distanzen die geringsten Distanzen zueinander aufweisen:

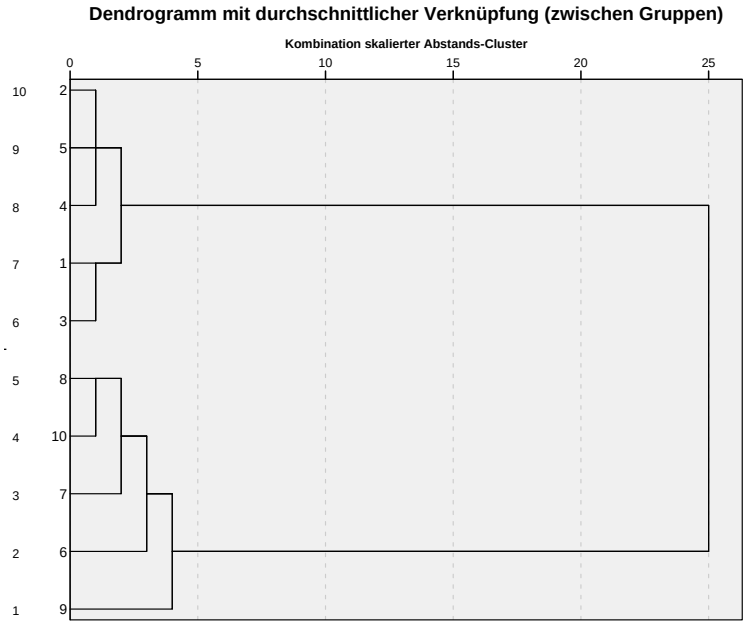
$$D(C_k,C_j) = \min \left\{ \frac{1}{n_k n_j} \sum_{n \in C_k} \sum_{m \in C_j} d_{nm} \right\} \tag{4.4}$$

Es werden also die Cluster hinsichtlich ihrer mittleren Distanz zueinander betrachtet. Dabei handelt es sich quasi um eine Kompromisslösung zwischen den Vor- und Nachteilen von 'Single-Linkage' und 'Complete-Linkage'. Das 'Average-Linkage-Verfahren' ist damit aber nur anwendbar, wenn intervallskalierte Merkmale vorliegen. Ansonsten wäre die Betrachtung des Mittels der Distanzen nicht zulässig.

In der Menüführung von SPSS erfolgt die Auswahl analog zu den beiden vorherigen Algorithmen mit der Modifikation der Methode auf 'Linkage zwischen den Gruppen', bzw. der Modifikation innerhalb der Syntax-Steuerung durch `/METHOD BAVERAGE`. Mit dem 'Average-Linkage ergibt sich für unseren Beispieldatensatz als Visualisierung durch das zugehörige Dendogram Abbildung 4.8.

1 Damit haben wir eine exaktere Vorgehensweise als zur Bestimmung eines arithmetischen Mittelwerts der Distanzen zwischen *allen* Objekten zweier Cluster.

Abbildung 4.8: Dendrogramm 'Average-Linkage'



Das wir nun durchschnittliche Betrachtungen haben wird auch noch einmal an den Fusionierungsschritten der Cluster deutlich (vgl. Abbildung 4.9). Da wir die *mittleren* Abweichungen betrachten, haben wir dort Dezimalzahlen. Folglich sollte für den 'Average-Linkage'-Algorithmus auch das metrische Messniveau als Voraussetzung beachtet werden! Die genauen Abstände und Fusionierungsschritte des Dendograms zeigt uns dabei wieder die dazugehörige Zuordnungsübersicht der Abbildung 4.9.

Abbildung 4.9: Zuordnungsübersicht 'Average-Linkage' in SPSS

Zuordnungsübersicht

	Zusammengeführte Cluster			Erstes Vorkommen des Clusters		
Schritt	Cluster 1	Cluster 2	Koeffizienten	Cluster 1	Cluster 2	Nächster Schritt
1	2	5	2,000	0	0	3
2	8	10	3,000	0	0	5
3	2	4	4,000	1	0	6
4	1	3	5,000	0	0	6
5	7	8	7,500	0	2	7
6	1	2	8,167	4	3	9
7	6	7	11,667	0	5	8
8	6	9	15,250	7	0	9
9	1	6	111,520	6	8	0

4.4.4 Das Zentroid-Verfahren

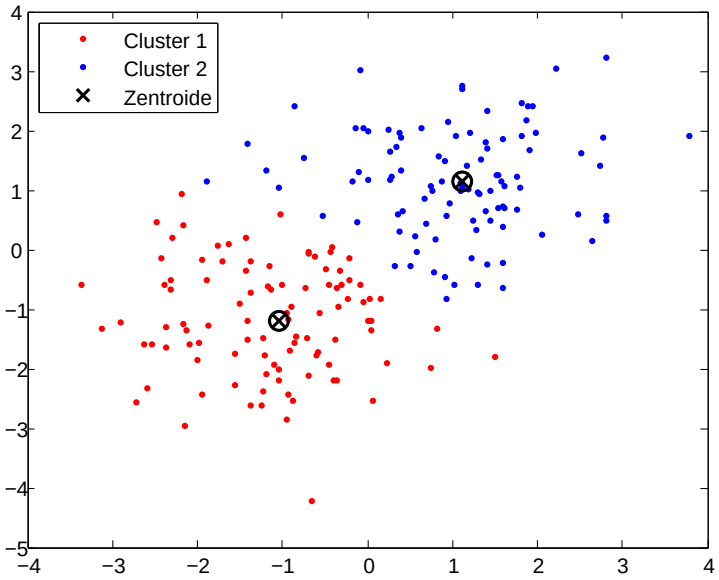
Eine Anwendung des Zentroid-Verfahrens ist klar begrenzt auf metrische Variablen, da wir für jeden Cluster das arithmetische Mittel als Klassenschwerpunkt aller internen Objekte eines Clusters berechnen:

$$\bar{\mathbf{x}_k} := \frac{1}{n_k} \sum_{n \in C_k} \mathbf{x}_n \tag{4.5}$$

Damit steht der Clusterschwerpunkt (als Mittelwertsvektor aller Variablen für alle Objekte des Clusters) als Repräsentant für den jeweiligen Cluster.

Grafisch veranschaulicht sich die Idee des Zentroid-Verfahrens zweier Cluster analog zu Abbildung 4.10.

Abbildung 4.10: Grafisches Beispiel Zentroid-Verfahren



Dabei nehmen wir (nur in diesem beispielhaften Fall!) die Cluster-Einteilung bereits als gegeben an (rot (C_k) und blau (C_j)). Die Zentroide bilden die Klassenschwerpunkte als arithmetisches Mittel der Realisierungen unserer Zufallsvariable(n). Im Gegensatz zum 'Average-Linkage'-Verfahren betrachten wir also *nicht* das Mittel der Distanzen, sondern das Mittel unserer konkreten Variablenausprägungen. Sind diese zusätzlich (z-)standardisiert an, lassen sich diese auch direkt vergleichen. Spätestens hier wird damit auch deutlich, dass die Datenlage metrisch oder dichotomisiert (wenn nominal) sein muss. Ordinale Variablen lassen sich aber hier ab ca. sieben Ausprägungen als quasi-metrisch auffassen.[1]

1 Dies ist aber keine statistische Eigenschaft, sondern ergibt sich aus der allgemeinen Anwendung in der Forschungspraxis!

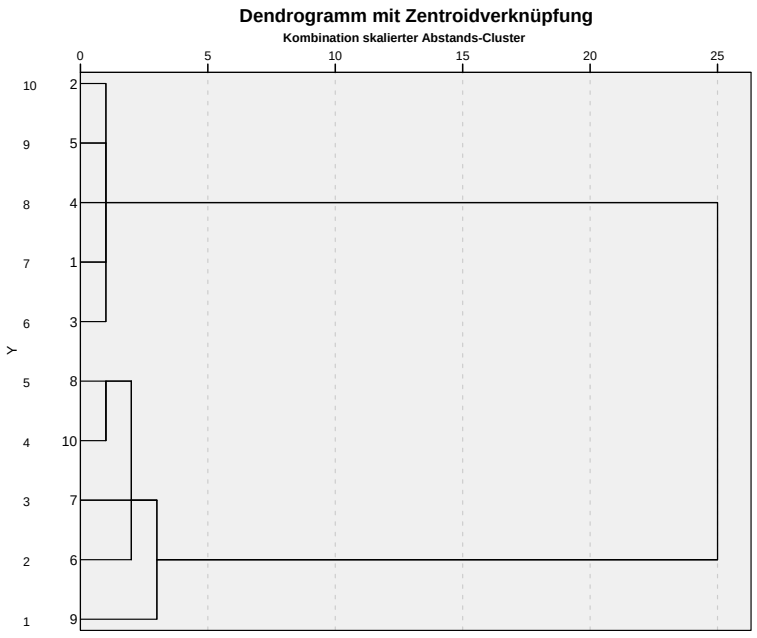
Wie fusionieren wir mit dem Zentroid-Verfahren? Die Bewertung des Abstands zweier Cluster C_k und C_j ergibt sich über:

$$D(C_k,C_j) = \min_{k \neq j} ||\bar{\mathbf{x}}_k - \bar{\mathbf{x}}_j||^2 \tag{4.6}$$

Mit $\bar{\mathbf{x}}_k$ als Vektor der arithmetischen Mittel aller Variablen für den Cluster k und $\bar{\mathbf{x}}_j$ für den Cluster j. Durch die quadrierte Differenz können wir damit direkt die absoluten Abweichungen aller Mittelwerte der beiden Cluster simultan bestimmen.
Damit fusionieren wir beim Zentroid-Verfahren also diejenigen Cluster, welche die geringste Distanz bezüglich der Klassenschwerpunkte, also ihrer arithmetischen Mittel, aufweisen. Da es sich um ein agglomeratives Verfahren handelt, haben wir natürlich im ersten Fusionierungsschritt nur die einzelnen Objekte vorliegen. Das arithmetische Mittel ist gleich der Variablenausprägung. Erst ab dem zweiten Schritt wird das arithemtische Mittel ($\bar{x}_i = \frac{1}{N} \sum x_{ni}$) berechnet.

Die Modifikation in SPSS für den Zentroid-Algorihmus erfolgt unter der Methode 'Zentroid-Clustering' oder alternativ über die den Syntax-Befehl `/METHOD CENTROID`. Abbildung 4.11 zeigt das Dendogram unseres Beispiels.

Abbildung 4.11: Dendrogramm Zentroid



Die Zuordnungsschritte zeigt Abbildung 4.12.

Abbildung 4.12: Zuordnung Zentroid

Zuordnungsübersicht

Schritt	Zusammengeführte Cluster		Koeffizienten	Erstes Vorkommen des Clusters		Nächster Schritt
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	2	5	2,000	0	0	3
2	8	10	3,000	0	0	6
3	2	4	3,500	1	0	5
4	1	3	5,000	0	0	5
5	1	2	5,806	4	3	9
6	7	8	6,750	0	2	7
7	6	7	9,667	0	6	8
8	6	9	11,938	7	0	9
9	1	6	104,400	5	8	0

Die Anforderung des Zentroid-Verfahrens an die Clusterbildung ist, dass die Cluster nur im Mittel ähnlich sein müssen. Dadurch kann man bestimmte Ausreißer ignorieren. Dieses Prinzip der mittleren Ähnlichkeit/Distanz gilt sowohl für das 'Zentroid'-Verfahren als auch für das 'Average-Linkage'-Verfahren.
Die Unterscheidung ist nur: Das 'Average-Linkage'-Verfahren betrachtet die mittlere Abweichung der Distanzen, dass 'Zentroid'-Verfahren die mittlere Abweichung der Variablenausprägungen.

4.4.5 Das Ward-Verfahren

Das Ward-Verfahren (auch 'Minimum-Varianz-Methode') als Algorithmus beruht auf folgender Idee: Fusioniere die beiden Cluster, welche die minimalste Erhöhung der Varianz im neuen Cluster (durch das Hinzufügen weiterer Beobachtungen) erzeugen:

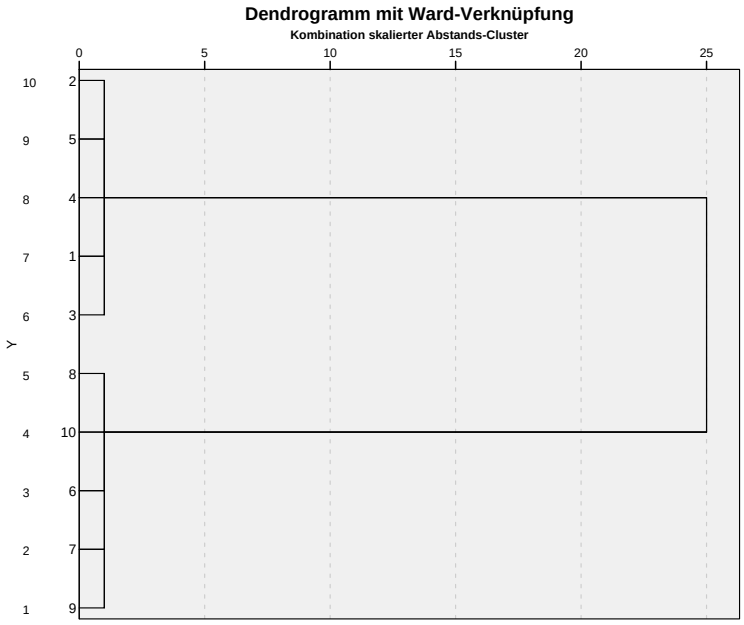
$$D_W(C_j,C_k) = \frac{n_j n_k}{n_j + n_k} ||\bar{x}_j - \bar{x}_k||^2 \tag{4.7}$$

Dies entspricht einem minimalen Zuwachs der Fehlerquadratsumme durch die Fusion. Dabei ist es möglich, einen Schwellenwert für die maximale $\triangle QS_{Fehler}$ festzulegen. Damit lässt sich für das Ward-Verfahren auch sagen: Der auftretende Homogenitätsverlust durch die Fusionierung zweier Cluster soll minimiert werden. Die Anwendung des Verfahrens nach Ward ist damit beschränkt auf metrische Daten. In der Praxis erzeugt das Verfahren nach Ward sehr homogene Gruppen und stellt das leistungsstärkste Verfahren unter den agglomerativen Verfahren dar.

Für die Anwendung des Ward-Algorithmus muss wieder nur unter den hierarchischen Clusterverfahren die Modifikation unter der Kategorie 'Methode' auf 'Ward-Methode' verändert werden, die Änderung der Syntax erfolgt in `/METHOD WARD`.

Für unser Beispieldatensatz gibt SPSS das Denodgramm von Abbildung 4.13 aus.

Abbildung 4.13: Dendrogramm 'Ward'



Wie man an dem Dendrogramm erkennen kann, hat der Ward-Algorithmus das Minimalziel der Varianzreduktion bereits im ersten Schritt realisiert. Es ergibt sich somit eine sehr schnelle Trennleistung unserer beiden Gruppen in zwei getrennte Cluster, wobei keine 'Wellen' der Fusionierungen auftreten, was die hohe Leistungsfähigkeit des Ward-Verfahren

Standardmäßig wird natürlich auch für das Ward-Verfahren noch einmal die Zuordnungsübersicht ausgegeben (Abbildung 4.14).

Abbildung 4.14: Zuordnungsübersicht 'Ward'

Zuordnungsübersicht

Schritt	Zusammengeführte Cluster		Koeffizienten	Erstes Vorkommen des Clusters		Nächster Schritt
	Cluster 1	Cluster 2		Cluster 1	Cluster 2	
1	2	5	1,000	0	0	3
2	8	10	2,500	0	0	8
3	2	4	4,833	1	0	7
4	1	3	7,333	0	0	7
5	6	7	11,333	0	0	6
6	6	9	17,333	5	0	8
7	1	2	24,300	4	3	9
8	6	8	35,600	6	2	9
9	1	6	296,600	7	8	0

4.5 Divisive Verfahren - der k-Means-Algorithmus in SPSS

Divise Verfahren wählen einen Fusionierungsweg in die entgegengesetzte Richtung zu agglomerativen Verfahren. Zu Beginn bilden alle Untersuchungseinheiten einige wenige große Cluster, welches schrittweise in immer kleinere Cluster unterteilt wird. Dabei werden immer die Gruppen von Einheiten getrennt, welche am unähnlichsten sind, bzw. die größte Distanz aufweisen.

Ein immer noch zentrales Problem diviser Verfahren ist die hohe benötigte Rechenleistung, da nicht nur eine Distanzmatrix benötigt wird, sondern jeweils eine für alle *potentiellen* Kombinationen von Fällen (da wir einen großen Cluster nun nach und nach auflösen). Für unseren kleinen Beispieldatensatz ist dies kein Problem, bei 1.000 Beobachtungen wird die Sache aber schon sehr rechenintensiv. In der Praxis sind deswegen eher agglomerative Verfahren relevant. Aufgrund der immer schnelleren Rechenleistungen bekommen divisive Verfahren aber eine zunehmend höhere Bedeutung. Als gängigster Vertreter wird hier der k-Means-Algorithmus und seine Anwendung in SPSS vorgestellt.

Der k-Means-Algorithmus[1] stellt das wichtigste und leistungsstärkste divisive Verfahren dar. Elementar und Abgrenzend zu den agglomerativen Verfahren ist, dass die Anzahl der Cluster dem Algorithmus von Beginn an vorgegeben wird und anschließend iterativ bestätigt und optimiert wird.

Die zu optimierende Zielfunktion des 'k-Means-Verfahrens' ist:

$$\min \sum_{j=1}^{k} \sum_{i=1}^{n} ||x_{i,j} - c_j||^2 \tag{4.8}$$

1 Der Algorithmus wird manchmal auch als 'Lloyd's Algorithmus' bezeichnet, jedoch eher in der Informatik.

mit $||x_{i,j} - c_j||^2$ als Abstand eines Datenpunktes (Person/Objekt) vom jeweiligen Clusterzentrum. Das doppelte Summenzeichen ($\sum_{j=1}^{k} \sum_{i=1}^{n}$) zeigt, dass sowohl über alle n Zielpersonen als auch über alle k Startcluster der Optimierungsalgorithmus läuft. Dieser Abstand von den Clusterzentren (also die Streuung) soll mit dem 'k-Means-Algorithmus' minimal werden. Damit hat der 'k-Means'-Algorithmus mit den Mittelwertsverfahren gemeinsam, dass die Fusionierung der Cluster ebenfalls ausgehend von den Clusterzentren erreicht wird.

Dabei zeigt sich auch schon die Aufwendigkeit des Verfahrens im Vergleich zu den agglomerativen: Der Algorithmus wird immer wieder Objekte zu Gruppen zusammenfügen und ihre Zentroide (Mittelwerte) berechnen. So wird überprüft, ob diese Gruppe als Startgruppe (Cluster) adäquat ist, also ihre interne Varianz und damit die Distanz relativ gering ist. Dieser Vorgang wird solange wiederholt, bis die Anzahl definierter Clusterzentren der vorgegebenen Clusteranzahl k entspricht. Deshalb ist der 'k-Means' auch kein hierarchisches und aufteilendes, sondern ein partitionierendes Verfahren.

Die Schritte des 'k-Means-Algorithmus' sind:

1. Gebe die Anzahl zu erstellender Cluster k vor.
2. Der Algorithmus verteilt diese Clusterpunkte als Clusterzentren auf die Datenpunkte und erstellt aus den Clusterzentren einen Mittelwertsvektor.
3. Die Zuteilung aller Objekte erfolgt zu denjenigen Cluster(-zentren), zu denen die Distanz (z.B. euklidischer Abstand oder Mahalabonis) minimal ist.
4. Nach der Neuordnung werden die Clusterzentren wieder neu bestimmt (die *Gesamtzahl* der Cluster ist dabei immer *konstant*).
5. Die Schritte 3 und 4 werden nun solange wiederholt, bis sich die Clusterzentren nicht mehr grundlegend verändern, ein festgelegtes Konvergenzkriterium der Veränderung der Zentren unterschritten wird oder die maximale Anzahl der (vorgegebenen) Iterationsschritte erreicht wird. Auf der Grundlage dieser entgültigen Clusterzentren werden dann alle Fälle den Clustern zugeordnet.

Insgesamt lässt sich der 'k-Means-Algorithmus' als divisives Verfahren damit als sehr rechenaufwendig auffassen, da immer wieder Neuzuordnungen zwischen den Clustern stattfinden. Agglomerative Verfahren ordnen Objekte *einmalig* und *endgültig* Clustern zu und divisive Verfahren ordnen immer wieder neu (aber trotzdem deterministisch, also eindeutig). Bei agglomerativen Verfahren lassen sich auch Bereiche von Clusterlösungen angeben, bei dem 'k-Means-Algorithmus' muss dies für jede gewünschte Clusteranzahl einzeln wiederholt werden.

Eine potentielle Lösung ist natürlich, den 'k-Means-Algorithmus' bei großen Datensätzen nur über eine Stichprobe laufen zu lassen, um iterativ die optimale Clusteranzahl zu bekommen. Liegt eine Entscheidung fest (z.B. vier verschiedene Cluster), kann der 'k-Means' dann auf den gesamten Datensatz angewandt werden, was die benötigte Rechenleistung vorab gut reduziert.

Da SPSS die Clusterpunkte automatisch auf die Datenstruktur verteilt, ist der 'k-Means-Algorithmus' anfällig für kleine Gruppen von Ausreißerwerten! Diesen wird i.d.R. ein eigener Cluster zugeordnet um der Datenstruktur gerecht zu werden. Sind die Ausreißerwerte aber nicht durch den natürlichen Datengenerierungsprozess bedingt, sondern z.B. auf Messfehler zurückzuführen, dann ergibt sich dabei eine Verfälschung der Ergebnisse. Eine Inspektion der Verteilung der Beobachtungen auf den Variablen sollte also vorher stattfinden.

Bei SPSS lässt sich das 'k-Means-Verfahren' über die Option Klassifizieren → Clusterzentrenanalyse durchführen. Dabei muss man die Anzahl der zu iterierenden Cluster vorgeben. Gleichzeitig lässt sich ein Schwellenwert für die maximale Anzahl an Iterationen oder/und ein Konvergenzwert für die Änderung der Clusterzentren angeben. Wird eine optimale Lösung (erneute Iteration ergab keine grundlegende Veränderung der Clusterzentren mehr) schon vorher erreicht, bricht die Iteration ab und die Clusterzuordnungen werden ausgegeben. Der Verlauf der Änderungen der Clusterzentren zwischen den Iterationen ist dabei dokumentiert, die Clusterzuordnungen der Fälle und ihr jeweiliger Abstand vom Clusterzentrum lassen sich auch für das 'k-Means-Verfahren' in separate Variablen abspeichern und stehen damit für weitere Analysen zur Verfügung.

Eine beispielhafte Syntaxangabe für unseren Datensatz wäre:

```
  QUICK CLUSTER v1 v2 v3 v4
/MISSING=LISTWISE
/CRITERIA=CLUSTER(2) MXITER(10) CONVERGE(0)
/METHOD=KMEANS(NOUPDATE)
/SAVE CLUSTER DISTANCE
/PRINT INITIAL CLUSTER DISTAN.
```

/MISSING=LISTWISE sorgt hier für einen listenweisen Fallausschluss bei fehlenden Werten, eine Modifikation zu /MISSING=PAIRWISE für einen paarweisen Fallusschluss. Die Option /CRITERIA=CLUSTER(2) MXITER(10) CONVERGE(0) gibt die Vorgaben für den 'k-Means-Algorithmus' an. CLUSTER(2) gibt eine zu erzielende Clusteranzahl

von zwei vor[1], MXITER(10) gibt SPSS an, dass maximal 10 Iterationsschritte durchgeführt werden sollen[2] und CONVERGE(0) bedeutet schließlich, dass ein vorzeitiger Abbruch der Iteration erfolgen soll, wenn sich die Clusterzentren um 0 bei einer Neuzuordnung verändern. /SAVE CLUSTER DISTANCE ist die Anweisung an SPSS, die Clusterzugehörigkeiten sowie die Distanz vom Clusterzentrum für jedes Objekt in eine jeweils neue Variable zu schreiben. Die Option /PRINT INITIAL CLUSTER DISTAN gibt tabellarische Informationen über die anfänglichen Clusterzentren (INITIAL), die Clusterzugehörigkeit jedes Objektes sowie seine Distanz zum Clusterzentrum (CLUSTER - was wir vorher auch abgespeichert haben) sowie die Distanzen der Cluster*zentren* zueinander (DISTAN). Potentiell kann durch die zusätzliche Angabe von /PRINT ANOVA eine ANOVA-Tabelle ausgegeben werden, hieraus lassen sich die desktiptiven Bedeutungen der einzelnen Variablen für die Clusterzuordnungen interpretieren.

Insgesamt ergibt sich über unsere obige Syntax als Output (zusätzlich der ANOVA-Tabelle) die Abbildung 4.15 der nächsten Seite.

1 Das dies eine optimale Lösung für unsere Daten ist, wissen wir ja bereits durch die agglomerativen Verfahren.

2 Die benötigte Rechenzeit kann hierdurch beschränkt werden.

Abbildung 4.15: SPSS-Ausgaben 'k-Means'

Anfängliche Clusterzentren

	Cluster	
	1	2
v1	3	8
v2	1	7
v3	1	7
v4	3	8

Anzahl der Fälle in jedem Cluster

Cluster	1	5,000
	2	5,000
Gültig		10,000
Fehlend		,000

Iterationsprotokoll[a]

	Änderung in Clusterzentren	
Iteration	1	2
1	2,059	1,356
2	,000	,000

a. Konvergenz wurde aufgrund geringer oder keiner Änderungen der Clusterzentren erreicht. Die maximale Änderung der absoluten Koordinaten für jedes Zentrum ist ,000. Die aktuelle Iteration lautet 2. Der Mindestabstand zwischen den anfänglichen Zentren beträgt 11,045.

Cluster-Zugehörigkeit

Fallnummer	Cluster	Distanz
1	1	1,562
2	1	1,625
3	1	2,059
4	1	1,428
5	1	1,200
6	2	2,245
7	2	1,356
8	2	2,245
9	2	2,764
10	2	1,800

Clusterzentren der endgültigen Lösung

	Cluster	
	1	2
v1	2	7
v2	2	7
v3	2	7
v4	2	7

Distanz zwischen Clusterzentren der endgültigen Lösung

Cluster	1	2
1		10,218
2	10,218	

ANOVA

	Cluster		Fehler		F	Sig.
	Quadrate	df	der	df		
v1	67,600	1	1,350	8	50,074	,000
v2	52,900	1	,900	8	58,778	,000
v3	67,600	1	1,000	8	67,600	,000
v4	72,900	1	1,200	8	60,750	,000

Die F-Tests sollten nur für beschreibende Zwecke verwendet werden, da die Cluster so gewählt wurden, daß die Differenzen zwischen Fällen in unterschiedlichen Clustern maximiert werden. Dabei werden die beobachteten Signifikanzniveaus nicht korrigiert und können daher nicht als Tests für die Hypothese der Gleichheit der Clustermittelwerte interpretiert werden.

Damit gibt uns SPSS für das 'k-Means'-Verfahren Informationen über:

1. Die **anfänglichen Clusterzentren** für *jede* Variable, wobei SPSS die Cluster über die Datenpunkte verteilt setzt und danach neu ordnet ('k-Means' ist iterativ!)

2. Ein **Iterationsprotokoll**, welches die Veränderung der Clusterzentren während der Iterationsschritte zeigt. Bei unserem klein gehaltenen und eindeutigem Datensatz erfolgt bereits nach dem 2. Schritt ein Abbruch, da sich die Clusterzentren nicht mehr verändert haben (0,00). Der Abbruch ist damit hier *nicht* passiert, weil wir unsere vorgegebenen Iterationsschritte abgearbeitet haben (MXITER(10)), sondern weil das Konvergenzkriterium von einer minimalen Veränderung der Clusterzentren von Null erreicht wurde (CONVERGE(0))

3. Die **Anzahl der Fälle in jedem Cluster**, womit die absolute Anzahl der endgültigen Zuordnungen der Clusterlösung angezeigt werden

4. Die **Cluster-Zugehörigkeit**: Hier wird für jede Beobachtung die Cluster-Zugehörigkeit und seine Distanz zum jeweiligen Clusterzentrum angegeben

5. Die **Clusterzentren der endgültigen Lösung**, was analog zu 1. interpretierbar ist und somit das Ausmaß der Veränderung während des Zuordnungsprozesses widerspiegelt

6. Die **Distanz zwischen Clusterzentren der endgültigen Lösung**, wobei natürlich die Angaben diagonal gespiegelt sind, die Distanz vom Cluster 1 zu 2 ist identisch mit der Distanz des Cluster 2 zu 1

7. Und schließlich noch die (optionale) **ANOVA-Tabelle**

Sowohl für die Mittelwertverfahren als auch den 'k-Means'-Algorithmus gilt: Nominale Variablen lassen sich als (0,1)-Regressoren in die Analyse integrieren, ordinale Variablen können ab ca. 7 Ausprägungen als quasi-metrisch aufgefasst und integriert werden.

KAPITEL 5

Clusteranalytische Verfahren mit R

Ziel dieses Kapitels ist nicht nur eine Anleitung für eine Clusteranalyse in R zu geben, sondern auch für die Anwendung von R zu werben und zu sensibilisieren. Auch wenn R auf den ersten Blick nicht sehr intuitiv erscheint und die reine Syntaxführung abschreckt: Bei einem Preis von exakt 0 Euro gibt es kein anderes Programm, welches an dieses Preis-Leistungsverhältnis herankommt.
Da R unter der GNU-Lizenz läuft, ist es komplett frei verfügbar und wird von Wissenschaftlern und Praktikern auf der ganzen Welt fortlaufend weiterentwickelt. Auch kompliziertere und speziellere Verfahren sind oftmals als erstes in R implementiert und können als zusätzliche Packete geladen werden.
Weitere Vorteile sind die enorme Schnelligkeit des Programms und die hochwertigen grafischen Ausgaben, welche beliebig modifzierbar sind.
Es existieren unzählige kostenlose Einführungen in verschiedenste Verfahren mit R (oder allgemeine Einführungen), welche zum größten Teil frei verfügbar über das Web erreichbar sind, kaum ein anderes Programm kann bei dieser Effizienz und Beratungsdichte mithalten. Als persönliche und umfangreiche Empfehlung lohnt sich ein Blick in das Skript von Andreas Handl. Lässt man sich also auf die Syntaxführung ein, wird man mit einem effizienten und leistungsstarken Programm belohnt.
Für einen leichteren Einstieg in R sei noch RStudio, welches eine grafische Benutzeroberfläche für R bietet, sowie das Statistiklabor der FU Berlin empfohlen. Falls der Einstieg direkt über R erfolgen soll , dann eignet sich als perfekte Ergänzung noch Tinn-R. Mit letzterem lässt sich der R-Code visuell gesondert und besser darstellen und mit frei wählbaren Shortcuts direkt an das gestartete R senden.
Um die Empfehlungen auch vollständig zu machen: Als Alternative zu Word eignet sich das Textsatzungssystem LaTeX besonders, hierzu ist der einfachste Weg erst MikTeX zu installieren (was im Hintergrund den Code dann übersetzt) und als Editor für die Setzung

des Code das TeXnicCenter. Für LaTeX existieren ebenfalls mehr als genug Einführungen im Web. Als guter Einstieg eignet sich besonders das Dokument Diplomarbeit mit LaTeX der Fernuni Hagen.
Die Ausgaben von R lassen sich auch sehr gut automatisch in LaTeX durch das Packet Sweave[1] setzen.

Da unzählige Einführungen in R im Netz existieren, wird hier auf eine umfangreichere Einführung verzichtet. Die nachfolgenden Befehle um Daten für eine Clusteranalyse zu generieren und diese durchzuführen werden aber alle jeweils erklärt und können so gut nachgerechnet werden.

Anstatt den R-Code aus dem laufenden Skript zu kopieren, sei hier auf den Anhang B („R-Code und Angaben“) verwiesen. Der R-Code lässt sich daraus einfach in R kopieren und damit ausgeführt werden. Einen Datensatz zu laden ist nicht notwendig, da alle Daten für das Beispiel selbst generiert werden, was wir nun auch tun wollen.

5.1 Datengenerierung in R

Damit wir überhaupt eine Clusteranalyse mit R rechnen können, brauchen wir erst einmal ein paar Daten. Um im Rahmen dieser Einführung optimale Lösungen hinzubekommen, generieren wir uns diese mit R selber. Dabei wollen wir im Folgenden Gruppen an Hand ihres Einkommens und ihrer Verweildauer im Bildungssystem generieren.
Dazu wollen wir uns als erstes eine Gruppe von 100 Beobachtungen generieren, welche einen Niedriglohnsektor darstellen sollen:

```
> inc1 <- rnorm(100, mean = 10000, sd = 5000)
> inc1
```

Als kurzer Einschub: R arbeitet objektorientiert, wir definieren alle Variablen in Objekten durch die Angabe eines $<-$ (und speichern sie dadurch ab). Hier haben wir die generierten Werte direkt in das Objekt *inc1* gespeichert um auf diese später zurückgreifen zu können. Gibt man R den Namen eines gespeicherten Objektes an, erhält man die Ausgabe der Daten.
Die Funktion *rnorm* ist hierbei für R die Anweisung, normalverteilte Größen zu generieren. Hierbei haben wir $\mu = 10000$ und $\sigma = 5000$ festgelegt, damit weichen unsere Beobachten *durchschnittlich* um €5000 vom gewählten Mittelwert ab.
Der Mittelwert und die Standardabweichung unserer generierten Daten lassen sich in R aufrufen durch:

1 Entwickelt von Prof. Leisch der LMU München.

```
> mean(inc1)

[1] 9851.736

> sd(inc1)

[1] 4713.245
```

und weichen damit natürlich von μ und σ ab, da wir nur eine Stichprobengröße von 100 gewählt haben.[1]

Weiterhin generieren wir uns noch eine Gruppe mittlerer Einkommen ($\mu = 40000$ bei $\sigma = 5000$):

```
> inc2 <- rnorm(100, mean = 40000, sd = 5000)
> inc2
```

Und schauen uns auch noch dort kurz den Mittelwert und die Standardabweichung der gezogenen Stichprobe an:

```
> mean(inc2)

[1] 40590.59

> sd(inc2)

[1] 5237.641
```

Um uns schließlich noch eine dritte Gruppe mit einer hohen Einkommensklasse zu erstellen. Dazu wählen wir ein $\mu = 70000$ bei einem σ von 5000:

```
> inc3 <- rnorm(100, mean = 70000, sd = 5000)
> inc3
```

Und werfen ebenfalls einen kurzen Blick auf Mittelwert und Standardabweichung:

```
> mean(inc3)

[1] 69416.05
```

1 Weswegen natürlich auch die individuell generierten Werte abweichen werden!
Zur Probe: Einfach die generierten Beobachtungen mal auf 1000, 10000 und 100000 erhöhen, wir nähern uns dann den asymptotischen Kennwerten an. Hier zeigt sich auch die Schnelligkeit von R, je nach Rechenleistung kann man sich schnell auch Fälle im Millionenbereich generieren ohne lange auf Ergebnisse warten zu müssen.

```
> sd(inc3)

[1] 5006.808
```

Damit haben wir nun bereits drei Objekte in *R* erstellt: $inc1$, $inc2$ und $inc3$, welche unsere Einkommensdaten der jeweiligen Klasse darstellen.

Weiterhin wollen wir aber als zweite Größe noch den Bildungsgrad unserer Beobachtungen später haben. Dazu erstellen wir uns wieder drei Gruppen von Daten, welche die Verweildauer im Bildungssystem in Jahren enthalten sollen.[1]

Unsere erste Klasse (welche später dem Niedriglohnsektor zugeordnet wird) enthält wieder 100 Beobachtungen, welche zwischen 6 und 10 variieren:

```
> school1 <- round(runif(100, min = 6, max = 10))
> school1
```

$round$ rundet die generierten Zahlen dabei auf ganze Größen. Wir speichern die generierten Zahlen durch die Angabe von $school1 <-$ wieder direkt ab um später auf diese zugreifen zu können. Die Beobachtungen dieser Gruppe haben also die Möglichkeit zwischen 6 und 10 Jahren im Bildungssystem zu verweilen.

Unsere zweite Gruppe soll 12 oder 13 Jahre im Bildungssysten verweilen und wird später der Klasse der mittleren Einkommen zugeordnet werden:

```
> school2 <- round(runif(100, min = 12, max = 13))
> school2
```

Und schließlich benötigen wir für die höheren Einkommen noch 100 Beobachtungen, welche zwischen 16 und 18 Jahren im Bildungssystem verbracht haben:

```
> school3 <- round(runif(100, min = 16, max = 18))
> school3
```

An diesem Punkt haben wir nun also sechs Objekte in R definiert: $inc1$, $inc2$, $inc3$, $school1$, $school2$ und $school3$.
Die einzelnen Objekte (mit ihren jeweils 100 Beobachtungen) müssen wir nun noch zusammenführen, wozu wir als erstes die drei Klassen der Einkommensdaten zu einer Variable zusammenfassen:

```
> inc <- c(inc1, inc2, inc3)
```

1 Damit wir auch bei beiden Größen eine nette metrische Vergleichsgröße haben.

Der Befehl c steht dabei für 'combine', womit wir die Fälle aneinander hängen und aus den drei Variablen mit jeweils 100 Beobachtungen eine Variable mit allen 300 Beobachtungen generieren.
Das identische Prozedere führen wir mit den Bildungsdaten durch:

```
> school <- c(school1, school2, school3)
```

Um schließlich beide Variablen (genauer: Vektoren) zu einem Datensatz (also einer 300 x 2 Matrix) zu vereinen:

```
> data <- cbind(inc, school)
```

Der Befehl $data.frame()$ definiert in R ein Objekt als Datensatz, um auf diesen später besser zugreifen zu können:

```
> data <- data.frame(data)
> data
```

Der Befehl $attach(Name\ des\ Datensatz)$ 'hängt' quasi den Datensatz in den Speicher von R, so das auf die Variablen des Datensatzes später bequem zugegriffen werden kann:

```
> attach(data)
```

Die folgende Ausgabe signalisiert uns, dass unsere beiden Variablen nun in der 'globalen Umgebung' von R enthalten sind:

```
The following object(s) are masked _by_ '.GlobalEnv':

    inc, school
```

Zur Kontrolle kann man sich auch noch einmal alle Variablen des Arbeitsspeichers anschauen:

```
> ls()
```

Nun wollen wir uns noch Indikatoren basteln, welche für jede Beobachtung signalisieren, zu welcher generierten Gruppe diese gehört.
Dazu eignet sich der $ifelse$-Befehl. Wir greifen auf unsere Einkommensdaten durch $data\$inc$ zu (die wir vorher 'attached' haben) und vergeben die Kategorie $high.dummy$, wenn das Einkommen größer als 60000 ist. Der if-Teil der $ifelse$-Bedingung vergibt dann eine 3 auf der Variablen. Ist das Einkommen jedoch für eine Person niedriger, dann vergibt der *else*-Teil der Bedingung eine 0:

```
> high.dummy <- ifelse(data$inc > 60000, 3, 0)
> high.dummy
```

Analog gehen wir vor um erst alle Werte unterhalb von 60000 mit einer 2 zu belegen (und dem Rest eine 0):

```
> mid.dummy <- ifelse(data$inc < 60000 & data$inc > 30000, 2, 0)
> mid.dummy
```

Um schließlich diejenigen Werte, welche kleiner als 30000 sind, mit einer 1 zu kennzeichnen (und analog alle anderen wieder 0):

```
> low.dummy <- ifelse(data$inc < 30000, 1, 0)
> low.dummy
```

Binden wir die drei Vektoren/Variablen aneinander:

```
> cbind(low.dummy, mid.dummy, high.dummy)
```

Erhalten wir direkt den folgenden Output:

```
       low.dummy mid.dummy high.dummy
  [1,]         1         0          0

[100,]         1         0          0
[101,]         0         2          0

[200,]         0         2          0
[201,]         0         0          3

[300,]         0         0          3
```

Und können somit hier kontrolliert erkennen, dass wir nun drei Variablen vorliegen haben, welche jeweils mit unterschiedlichen Zahlen unsere ursprünglich generierten Gruppen enthalten. Die ersten 100 Fälle erhalten eine 1, der 101-te bis 200-te Fall eine 2 auf der zweiten Variablen und die Fälle 201 bis 300 eine 3 auf der dritten Variable.
Was wir aber wollen, ist eine Variable, welche die Informationen der Zugehörigkeit zu den Gruppen für alle Fälle direkt enthält. Dazu erstellen wir einfach ein neues Objekt *categories*, wozu wir alle drei erstellten Variablen addieren. Die Abrufung von *categories* zeigt uns, dass wir nun eine Variable mit den Kategorien 1, 2 und 3 haben.

```
> categories <- low.dummy + mid.dummy + high.dummy
> categories
```

R bietet aber zusätzlich noch die Möglichkeit, durch den Befehl *factor* Objekte als kategoriale Variablen zu definieren. Die Klammer enthält dann als erstes den Namen der umzuwandelnden Variablen und mit dem Befehl *labels* die Kennzeichnung der Kategorien, *c* definiert damit quasi den Vektor der Bezeichnungen[1], welche für die nacheinander folgenden Werte vergeben werden sollen:

```
> categories = factor(categories, labels = c("Low", "Mid", "High"))
```

Schauen wir uns die Variable *categories* nun an, sehen wir, dass die Gruppen nun benannt sind und z.B. für grafische Visualisierungen nun besser unterschieden werden können:

```
> categories
```

Wir benutzen nun wieder den *cbind*-Befehl, um an unseren ursprünglichen Datensatz *data* noch die Variable *categories* anzuhängen:

```
> data <- cbind(data, categories)
```

Schauen wir uns die Namen des *data.frame data* an, dann sehen wir, dass *categories* nun enthalten ist und wir auf diese Variable nun auch zugreifen können:

```
> names(data)
```

```
[1] "inc"        "school"     "categories"
```

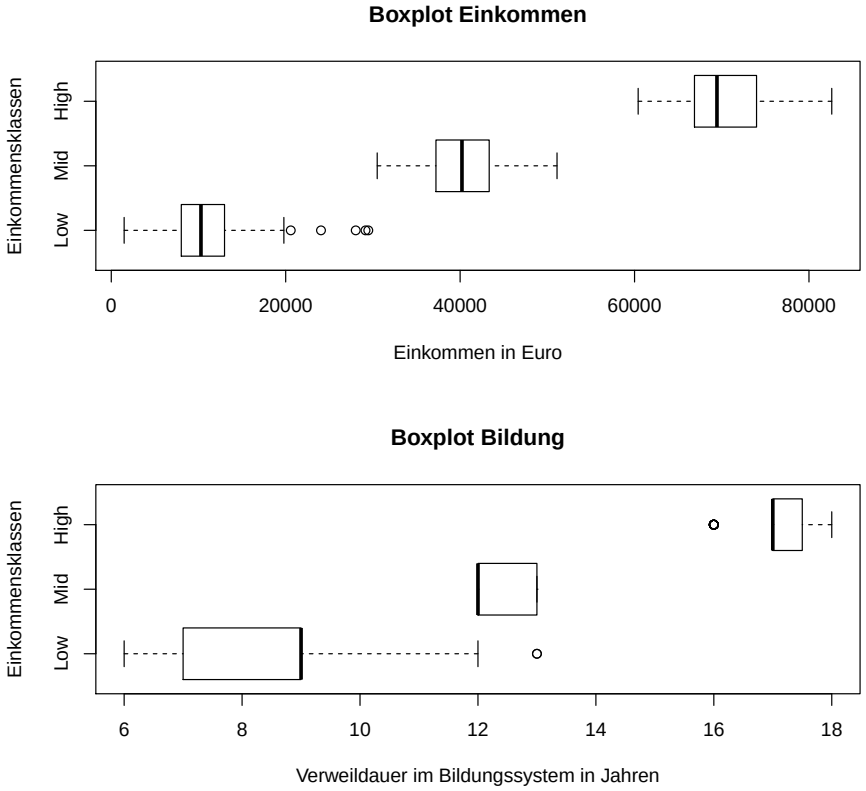
Durch unsere vorgenommenen Kategorisierungen können wir nun unsere Einkommens- und Bildungsdaten sehr gut grafisch darstellen. Hier wird durch *boxplot* ein Boxplot-Diagramm verwendet, *main* gibt den Titel der Grafik an und *ylab* und *xlab* enthalten die Achsenbeschriftungen. Durch *inc categories* unterteilen wir unsere Einkommensdaten nach den vorgenommenen Kategorisierungen. Der Befehl *par* gibt R vor, dass nun mehrere Grafiken auf einer Zeile zwei Spalten unterteilt werden sollen (damit wir unsere beiden Boxplots nebeneinander darstellen können):

```
> par(mfrow=c(1,2))
> boxplot(inc~categories,main="Boxplot Einkommen",
+ ylab="Einkommen in Euro",xlab="Einkommensgruppierungen")
> boxplot(school~categories,main="Boxplot Bildung",
+ ylab="Verweildauer im Bildungssystem in Jahren",xlab="Einkommensgruppierungen")
```

Grafisch erkennen wir dadurch noch einmal sehr gut unsere drei Gruppen an Beobachtungen, welche wir später in Cluster unterteilen wollen (Abbildung 5.1).

1 Zwingend durch „Bezeichnung des Faktors“ anzugeben!

Abbildung 5.1: Boxplots Einkommen und Bildung

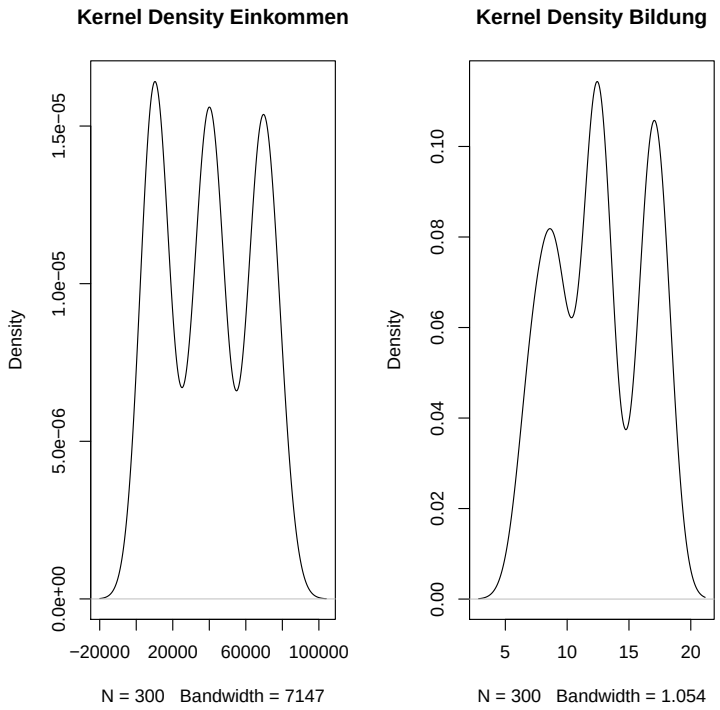


Ausreißerwerte für potentielle unklare Zuordnungen sind auch nur relativ wenig vorhanden, hauptsächlich einige Fälle der untersten Einkommensklasse brechen nach oben aus.
Wollen wir uns eher eine 'nahtlose' Dichtefunktion der Verteilungen anschauen, so eignen sich Kerndichteschätzer als grafische Darstellung (Abbildung 5.2).

```
> par(mfrow=c(1,2))
> plot(density(inc),main="Kernel Density Einkommen")
> plot(density(school),main="Kernel Density Bildung")
```

Auch hier sind unsere drei Gruppen von Beobachtungen als lokale Erhebungen klar erkennbar, verdeutlicht aber im Gegensatz zu der Darstellung der Boxplots, dass die Übergänge zwischen den Gruppen mehr oder weniger fließend verlaufen.

Abbildung 5.2: Kerndichteschätzer für Einkommen und Bildung



Als letzten Schritt sollten wir unseren Datensatz vorsichtshalber im R-Datenformat sichern, wobei der Speicherort natürlich noch genauer spezifiziert werden sollte:

```
> save.image("C:\\Speicherort")
```

Unsere Datengenerierung ist hierbei zu Ende, wir können mit den vorhandenen Daten nun im Folgenden eine einfache und einführende Clusteranalyse mit R rechnen. Der Weg erscheint bei R etwas langwierig, wir haben uns hier aber auch selber Daten generiert und auf keine bestehenden Datensätze zurückgegriffen, um später exakte Lösungen bei der Clusteranalyse zu bekommen.
Der Vorteil von R liegt aber auf der Hand: mit diesem (kostenlosen!) Statistikprogramm erlangt man die volle Kontrolle über eine schier unerschöpfliche Basis statistischer Verfahren. R benötigt zwar eine relativ hohe Einarbeitungszeit und ist nur über Kommandos zu steuern, dafür erlangt man aber auch die volle Kontrolle und wird aus didaktischer Sicht auch viel mehr gezwungen mit der konkreten Anwendung der Verfahren auseinanderzusetzen, da ein 'click and drop' nicht möglich ist.
Ein weiterer Vorteil ist die hohe Basis qualitativ hochwertiger grafischer Outputs. Hier gibt es kaum Begrenzungen, da immer wieder Nutzer von R neue Packete mit optimierten Varianten schreiben.

5.2 Clusteranalyse in R

Um eine Clusteranalyse in R durchführen zu können, müssen wir als erstes das Packet 'cluster' laden, welches die Befehle für eine Clusteranalyse in R enthält. Der Befehl $library$ lädt bereits installierte Packete, bei einer neu installierten R Version, muss dieses aber erst installiert werden. Der Befehl $utils :::menuInstallPkgs()$ ruft das Installationsmenü für optionale Packete auf (Alternative: Menüführung), wo das Packet $cluster$ ausgewählt werden kann.

```
> library(cluster)
```

Da wir in unserem $data.frame(data)$ noch eine kategoriale Variable haben, schreiben wir uns die Einkommens- und Bildungsdaten erst noch in ein neues $data.frame$ mit dem Namen $cluster$:

```
> cluster<-data.frame(data$inc,data$school)
```

5.2.1 Hierarchische/agglomerative Clusteranalyse in R

Da unsere Datenmatrix nun nur noch Variablen mit einem metrischen Messniveau enthält, können wir nun durch den $daisy$ Befehl unsere Distanzmatrix für die hierarchischen Verfahren berechnen:

```
> dist.euclid<-daisy(cluster,metric="euclidean",stand=TRUE)
```

Durch die Angabe von *dist.euclid* geben wir wieder einen Namen vor, worunter die Distanzmatrix in R abgespeichert werden soll. Da wir $metric = \text{“}euclidean\text{“}$ angegeben haben, nimmt R als Distanzmaß den euklidischen Abstand.[1] Die Option $stand = TRUE$ ist hier besonders wichtig, sie standardisiert die Werte vor der Clusteranalyse. Dies ist hier notwendig, da die Skalierungsbereiche des Einkommens und der Jahre der Schulbildung komplett unterschiedlich ist. Eine Distanz von 1 auf beiden Variablen hat eine vollkommen andere Bedeutung! Würden wir z.B. das Einkommen in Euro und die Konsumausgaben in Euro skaliert haben, wäre eine Standardisierung natürlich obsolet und zudem schwieriger direkt zu interpretieren.

Da wir die Distanzmatrix nun unter *dist.euclid* abgespeichert haben, können wir nun unseren Cluster-Algorithmus anwenden:

1 Es sind aber natürlich auch andere Distanzmaße möglich wie $metric = \text{“}manhattan\text{“}$ oder $method = \text{“}minkowski\text{“}, p$ für die allgemeine Minkowski-Metrik mit dem Parameter p. Für nominale Variablen lässt sich der Simple-Matching-Koeffizienten mit $metric = \text{“}gower\text{“}$ auswählen.

```
> dendogramm<-hclust(dist.euclid,method="average")
```

hclust ist dabei die Angabe für eine hierarchische Clustermethode, *dist.euclid* gibt den Namen des Objekts an, wo die Distanz-/Ähnlichkeitsmatrix abgespeichert wurde. Unter *method* = "*average*" ist die Variante des Algorithmus definiert, hier wären auch alle anderen in diesem Skript aufgeführten Möglichkeiten denkbar.[1] Auch hier geben wir wieder ein Objekt an, worunter die Ergebnisse des Algorithmus abgespeichert werden sollen (hier: *dendogramm*).

Mit den abgespeicherten Ergebnissen können wir nun sehr gut die Ergebnisse der Clusteranalyse als Dendogramm plotten:

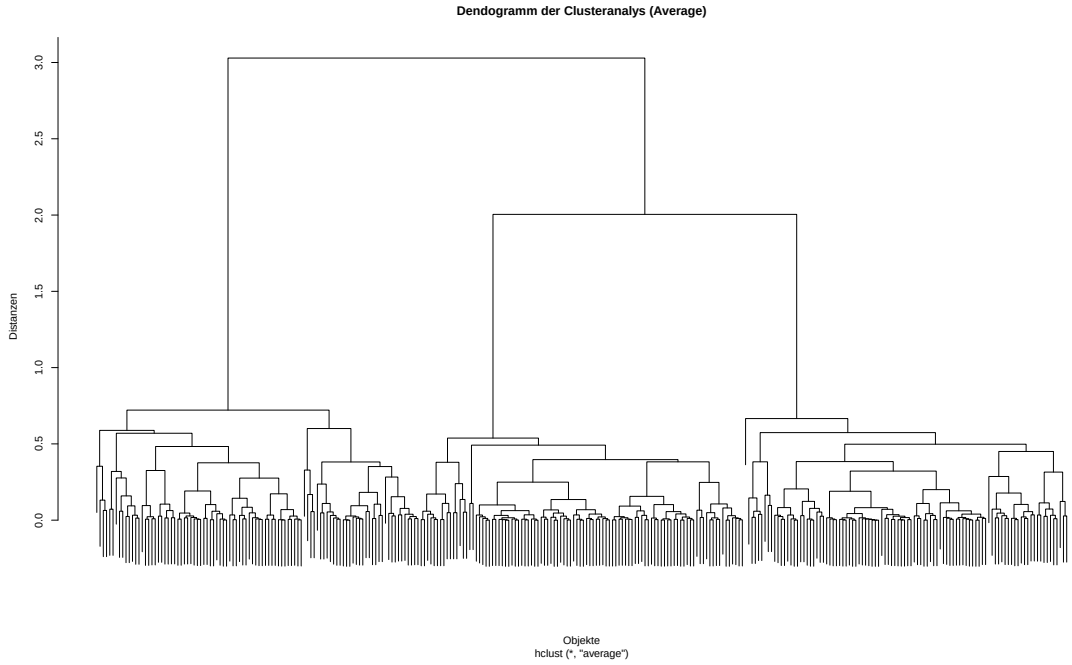
```
> plot(dendogramm,xlab="Objekte",ylab="Distanzen",
> + main="Dendogramm der Clusteranalys (Average)",labels=FALSE)
```

xlab und *ylab* geben hier wieder die Bezeichnungen der Achsen an, *main* den Titel der Grafik (alles optional). Da wir in unserer Clusteranalyse ja 300 Fälle haben, macht es Sinn die Option *labels* = *FALSE* zu aktivieren. Hiermit wird bei der grafischen Ausgabe die Bezeichnungen der Fälle an der x-Achse unterdrückt, was ab einer bestimmten Anzahl an Fällen die Übersichtlichkeit des Dendogramms steigert.
Der Befehl *dendogramm* bedeutet natürlich nicht, dass hier ein Dendogramm geplottet werden soll, sondern wir haben ja mit dem Befehl vorher einfach nur die Ergebnisse des Algorithmus unter diesem Namen abgespeichert (*abcd* als Name wäre ebenso möglich).

Den grafischen Output des Dendogramms durch R zeigt die Abbildung 5.3 der nächsten Seite.

1 Als folgende Bezeichnungen für R: "*average*","*single*","*complete*","*centroid*","*ward*".

Abbildung 5.3: Dendogramm Clusteranalyse Einkommen und Bildung



Da das Ziel der Clusteranalyse i.d.R. ist, die Gruppenzuordnungen für weitergehende Analyseverfahren zu erhalten, macht es Sinn die Zuordnungen innerhalb des Datensatzes für jedes einzelne Objekt abzuspeichern.

```
> cluster.hierarch_3<-cutree(dendogramm,k=3)
> cluster.hierarch_3
```

Für den Befehl *cutree* muss dabei das Objekt angegeben werden, welches das Ergebnis des Algorithmus enthält (hier *dendogramm*), sowie mit $k =$ die Anzahl der gewünschten Cluster (hier unsere drei ursprünglich generierten Gruppen).
Bindet man anschließend noch den Vektor der Clusterzuordnungen mit dem *cbind* Befehl an den ursprünglichen Datensatz an:

```
> data<-cbind(data,cluster.hierarch_3)
```

Lassen sich für unsere Einkommensvariable *inc* und unsere Dauer der Jahre im Schulsystem *school* z.B. sehr gut die jeweiligen Mittelwerte und Standardabweichungen, unterteilt nach Clustern, mit dem *tapply*-Befehl berechnen:

```
> tapply(inc,cluster.hierarch_3,mean)
```

```
       1        2        3
10294.05 40287.88 69966.64

> tapply(inc, cluster, sd)

       1        2        3
4420.252 5591.132 5518.532

> tapply(school,cluster.hierarch_3,mean)

    1     2     3
 8.00 12.58 16.90
```

Schauen wir uns abschließend noch die Anzahl der Personen in den einzelnen Clustern an:

```
> table(cluster.hierarch_3)

cluster
  1   2   3
100 100 100
```

In diesem Fall hat R also allen Clustern eine identische Anzahl an Personen zugeordnet.

Der Fall $k = 2$:

```
> cluster.hierarch_2<-cutree(dendogramm,k=2)
> data<-cbind(data,cluster.hierarch_2)
> tapply(inc,cluster.hierarch_2,mean)

cluster2
  1   2
100 200
```

Dementsprechend modifzieren sich auch unsere Mittelwerte der Variablen in den jeweiligen Clustern zu:

```
> tapply(school, cluster.hierarch_2, mean)

        1         2
 9963.199 54774.109

> tapply(school, cluster.hierarch_2, mean)
```

```
     1      2
 7.990 14.745
```

Erzeugt hingegen einen Cluster mit 100 Fällen (die untere Einkommensklasse mit niedriger Bildung) und ein weiteres Cluster mit der mittleren *und* hohen Einkommens- sowie Bildungsklasse. Warum das so sicher ist? Mit einem Blick auf das Dendogramm aus Abbildung 5.3 auf der Höhe der standardisierten Distanz von ca. 2.5 dürfte der Grund schnell klar werden.

5.2.2 Der k-Means-Algorithmus in R

Um den k-Means-Algorithmus in R auszuführen, benötigen wir den *kmeans*-Befehl, welchen wir auf unseren *cluster* Datensatz mit den Einkommens- und Schuljahren anwenden wollen.
Vorab müssen wir uns aber noch einmal in Erinnerung rufen, wie der k-Means-Algorithmus vorgeht: Wir geben k Cluster vor, welche gefunden werden sollen. Unser Statistikprogramm verteilt davon ausgehend zufällig Clusterzentren auf die Daten um danach die Zuteilungen zu modifzieren bis die Zuteilung so optimal ist, dass keinerlei Veränderungen der Clusterzentren zwischen zwei Iterationsschritten mehr auftreten. Die Optimierung finden dabei sowohl über die Abstände zu allen anderen Objekten als auch zu den Clusterzentren gleichzeitig statt.
Wenn wir uns aber unsere beiden Variablen kurz anschauen:

```
> summary(school)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   6.00    9.00   13.00   12.45   16.00   18.00
```

```
summary(inc)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  -4505   13140   40710   40020   67270   81110
```

Wird schnell klar, dass beide Variable eine unterschiedliche Skala haben. Das Einkommen streut viel stärker und besitzt viel extremere Maximalwerte, womit die Skalen nicht vergleichbar sind, dass k-Means Verfahren aber *nicht* skaleninvariant ist. Würden wir die Werte nun vorab nicht standardisieren, dann würde deshalb das Gewicht der Schuldaten für die Distanzmessung marginal werden.[1]

1 Bei den hierarchischen Verfahren mussten wir nur nicht standardisieren, weil wir die Option innerhalb der Syntax automatisch angeben konnten.

Wir standardisieren beide Variablen also vorab über eine z-Standardisierung der Form:

$$Z = \frac{X - \mu}{\sigma}$$

Was sich innerhalb der R-Syntax realisieren lässt als:

```
> school.stand<-(school-mean(school))/sd(school)
> inc.stand<-(inc-mean(inc))/sd(inc)
```

Und führen die beiden Variablen in einen neuen Datensatz für das 'k-Means-Clustering' zusammen:

```
> cluster.k<-cbind(inc.stand,school.stand)
> cluster.k
```

Auf diesen Datensatz können wir nun unseren Algorithmus laufen lassen, da beide Variablen durch die Standardisierung nun eine identische Skala besitzen. Die Ergebnisse speichern wir direkt in das Objekt *clusterzentren* ab:

```
> clusterzentren <- kmeans(cluster.k, centers = 3)
> clusterzentren
```

Der Befehl *centers* = 3 gibt dem k-Means die Anzahl der zu erstellenden Cluster vor (die der Algorithmus ja benötigt).
Der Aufruf der Ergebnisse durch *clusterzentren* liefert folgende Ergebnisse:

1. Der erste Teil beschreibt die Clusterlösung des k-Means, hier wurden 3 Cluster mit jeweils 100 Beobachtungen erstellt:

   ```
   K-means clustering with 3 clusters of sizes 100, 100, 100
   ```

2. Worauf folgend die jeweiligen Mittelwerte auf unseren beiden (nun standardisierten) Variablen in den Clustern angegeben werden:

   ```
   Cluster means:
      inc.stand    school.stand
   1 -1.199457343 -1.190380261
   2  0.009780812  0.003498546
   3  1.189676531  1.186881715
   ```

   Hierbei wird wieder deutlich, dass wir ziemlich gut unsere ursprünglich generierten Gruppierungen getroffen haben. Der Cluster 1 enthält sowohl ein unterdurchschnittliches Einkommens- als auch Bildungsniveau, der Cluster 2 bildet das mittlere Zentrum

mit standardisierten Werten um 0 auf beiden Variablen[1]. Im Cluster 3 hingegen weichen die Beobachtungen um durchschnittlich ca. 1.2 Standardabweichungen nach oben ab, damit liegen für unseren Fall in Cluster 2 die Beobachtungen mit hohen Einkommens- und Bildungswerten vor, der Cluster 3 enthält die mittlere Klasse.
Hierbei als wichtiger Hinweis: Da das k-Means Verfahren iterativ mit zufälliger Anordnung arbeitet, muss die engültige Clusterlösung nicht in der bereits optimalen Reihenfolge wie hier erscheinen! Es kann z.B. passieren, dass Cluster 1 und 3 in der Anordnung vertauscht sind, Cluster 3 enthält dann die niedrigen Einkommensbezieher und das Cluster 1 die hohen. In jedem Fall aber: die Cluster werden immer nahezu gleichgut unterscheidbar sein.

3. Weiterhin wird uns (der hier verkürzte) Vektor der Zuordnungen der Objekte zu den Clustern ausgegeben:

```
Clustering vector:
  [1] 1 1 ...
 [98] 1 1 1 2 2 ...
[199] 2 2 3 3 ...
[299] 3 3
```

Damit haben wir sogar eine exakte Trennleistung hinbekommen. Die ersten 100 Fälle sind in den Cluster 1, die Beobachtungen 101 bis 200 in den Cluster 2 und die Beobachtungen 201 bis 300 in den Cluster 3 eingeordnet. Dies spiegelt exakt unsere ursprüngliche Verteilung wider.

4. Da der k-Means-Algorithmus so arbeitet, dass die Abstände von den Clusterzentren minimal werden sollen (also die Streuung minimal werden soll), gibt R als zusätzliche Information die Streuungsquadratsumme innerhalb jedes Clusters an:

```
Within cluster sum of squares by cluster:
[1] 15.068483  6.746320  8.202253
 (between_SS / total_SS =  95.9 %)
```

Da wir die Variablen vorab standardisiert haben, lässt sich auch die Streuungsquadratsumme direkt vergleichen. Das der Cluster 1 hier die größte Streuung aufweist, hat eine einfache Erklärung: Die Anzahl der Jahre im Bildungssystem streut hier gegenüber den anderen beiden Gruppen höher (was wir bei der Datengenerierung

1 Zur Erinnerung: Die z-Transformation zentriert mit einem Mittelwert von 0 und einer Standardabweichung von 1.

so gewählt haben). Auf den Einkommensdaten hingegen streuen alle drei Gruppen gleich, da wir für die Generierung der Daten jeweils ein $\sigma = 5000$ angegeben hatten.

5. Als letzte Komponente werden uns noch die Objekte des k-Means angezeigt, auf die wir nun zugreifen können:

```
Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
    "betweenss"    "size"
```

Dies wollen wir abschließend auch machen, indem wir wieder unsere Clusterzuordnungen der jeweiligen Beobachtungen mit unserem ursprünglichen Datensatz $cluster.k$ kombinieren und ebendiesen überschreiben:

```
> clusterzentren$cluster
> cluster.k<-data.frame(cbind(cluster.k,clusterzentren$cluster))
> cluster.k
> names(cluster.k)
```

Mit dem zusätzlichen Befehl $data.frame$ haben wir uns wieder die kombinierten Variablen als Datensatz definiert, um auf diese Objekte wieder direkt zugreifen zu können.
Damit schreiben wir uns nun wieder aus den Clusterzuordnungen des k-Means eine neue Variable, welche wir direkt als Faktor (kategoriale Variable) umdefinieren. Die verschiedenen Clusterzuordnungen sollen dabei sozioökonomische Gruppen widerspiegeln:

```
> sozioökonomische.gruppe=factor(clusterzentren$cluster,
> +labels=c("Niedrig","Mittel","Hoch"))
> sozioökonomische.gruppe
```

Diese Zuordnung hängen wir nun wieder an unseren *ursprünglichen* Datensatz $data$ an:

```
> data<-cbind(data,sozioökonomische.gruppe)
> names(data)
> data
```

Und schauen uns auch für diese Zuordnungen noch einmal die Mittelwerte der beiden Variablen, getrennt nach den Clusterzuordnungen an:

```
tapply(inc, sozioökonomische.gruppe, mean)
```

```
 Niedrig   Mittel     Hoch
10565.64 40528.95 69765.20
```

```
tapply(school, sozioökonomische.gruppe, mean)
```

```
Niedrig  Mittel    Hoch
   7.94   12.49   17.00
```

Womit wir wieder sehen, dass wir ziemlich genau an unseren ursprünglichen Kennwerten wieder sind.
Die Clusteranalyse wäre an diesem Punkt in R abgeschlossen, wir haben die Zuordnungen der Objekte zu den Clustern abgespeichert und könnten nun wieder weitere Verfahren rechnen, welche diese Gruppenzuordnungen benötigen.

Als letztmaliger Hinweis noch einmal: Alle hier generierten Daten werden *immer* bei der eigenen Durchführung der Befehle in den Kennziffern abweichen! Wir haben hier nur (normalverteilte) Zufallswerte, welche mit an Sicherheit grenzender Wahrscheinlichkeit nicht zweimal identisch sein werden!

Literaturverzeichnis

[1] **Abonyi, János/Feil, Balázs:** *Cluster Analysis for Data Mining and System Identification.* Basel/Boston/Berlin: Birkhäuser Verlag AG, 2007.

[2] **Bacher, Johann/Pöge, Andreas/Wenzig,Knut** *Clusteranalyse. Anwendungsorientierte Einführung in Klassifikationsverfahren*, 3. Auflage. München [u.a.]: Oldenbourg Wissenschaftsverlag, 2010.

[3] **Backhaus, Klaus:** *Multivariate Analysemethoden. Eine anwendungsorientierte Einführung*, 11., über. Auflage. Berlin [u.a.]: Springer, 2006.

[4] **Fahrmeir, Ludwig/ Hamerle, Alfred/ Tutz, Gerhard (Hrsg.):** *Multivariate statistische Verfahren*, 2. über. Auflage Berlin/New York: Walter de Gruyter, 1996.

[5] **Mirkin, Boris:** *Mathematical Classification and Clustering.* Dordrecht/Boston/London: Kluwer Academic Publishers, 1996.

[6] **Schendera, Christian F.G.:** *Clusteranalyse mit SPSS: Mit Faktorenanalyse.* München: Oldenburg Wissenschaftsverlag, 2010.

ANHANG A

Beispieldatensatz und SPSS-Syntax der Beispiele

Beispieldatensatz:

ID	v1	v2	v3	v4
1	1	2	1	3
2	2	2	3	1
3	3	1	1	3
4	2	3	1	1
5	1	2	2	1
6	8	5	7	6
7	8	7	7	8
8	5	6	7	8
9	8	8	5	6
10	6	7	8	8

Syntax Single-Linkage:

```
  CLUSTER v1 v2 v3 v4
/METHOD SINGLE
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

Syntax Single-Linkage (standardisiert):

```
  PROXIMITIES v1 v2 v3 v4
/MATRIX OUT('SPEICHERORT')
/VIEW=CASE
/MEASURE=SEUCLID
/PRINT NONE
/STANDARDIZE=VARIABLE Z.
CLUSTER
/MATRIX IN('SPEICHERORT')
/METHOD SINGLE
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
ERASE FILE='SPEICHERORT'.
```

Complete-Linkage:

```
  CLUSTER v1 v2 v3 v4
/METHOD COMPLETE
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

Average-Linkage:

```
  CLUSTER v1 v2 v3 v4
/METHOD BAVERAGE
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

Zentroid-Verfahren:

```
  CLUSTER v1 v2 v3 v4
/METHOD CENTROID
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

Ward-Verfahren:

```
  CLUSTER v1 v2 v3 v4
/METHOD WARD
/MEASURE=SEUCLID
/PRINT SCHEDULE CLUSTER(2)
/PRINT DISTANCE
/PLOT DENDROGRAM VICICLE
/SAVE CLUSTER(2).
```

k-Means:

```
  QUICK CLUSTER v1 v2 v3 v4
/MISSING=LISTWISE
/CRITERIA=CLUSTER(2) MXITER(10) CONVERGE(0)
/METHOD=KMEANS(NOUPDATE)
/SAVE CLUSTER DISTANCE
/PRINT INITIAL ANOVA CLUSTER DISTAN.
```

ANHANG B

R-Code und Angaben

Als Tipp: Einfach alles rauskopieren und in R laufen lassen, da die #-ten alles
geschriebene auskommentieren.
Generierung der Einkommensdaten:

```
inc1 <- rnorm(100, mean = 10000, sd = 5000)
inc1
mean(inc1)
sd(inc1)
inc2 <- rnorm(100, mean = 40000, sd = 5000)
inc2
mean(inc2)
sd(inc2)
inc3 <- rnorm(100, mean = 70000, sd = 5000)
inc3
mean(inc3)
sd(inc3)
```

Generierung der Anzahl der Jahre der Schulbesuche:

```
school1 <- round(runif(100, min = 6, max = 10))
school1
school2 <- round(runif(100, min = 12, max = 13))
school2
school3 <- round(runif(100, min = 16, max = 18))
school3
```

Zusammenführung der Einkommens- und Schuldaten:

```
inc <- c(inc1, inc2, inc3)
school <- c(school1, school2, school3)
data <- cbind(inc, school)
data <- data.frame(data)
data
```

Einkommenskategorisierungen:

```
attach(data)
ls()
high.dummy <- ifelse(data$inc > 60000, 3, 0)
high.dummy
mid.dummy <- ifelse(data$inc < 60000 & data$inc > 30000, 2, 0)
mid.dummy
low.dummy <- ifelse(data$inc < 30000, 1, 0)
low.dummy
cbind(low.dummy, mid.dummy, high.dummy)
categories <- low.dummy + mid.dummy + high.dummy
categories
categories = factor(categories, labels = c("Low", "Mid", "High"))
categories
data <- cbind(data, categories)
names(data)
```

Grafische Visualisierungen:

```
par(mfrow=c(1,2))
boxplot(inc~categories,main="Boxplot Einkommen",
ylab="Einkommen in Euro",xlab="Einkommensgruppierungen")
boxplot(school~categories,main="Boxplot Bildung",
ylab="Verweildauer im Bildungssystem in Jahren",xlab="Einkommensgruppierungen")
par(mfrow=c(1,2))
plot(density(inc),main="Kernel Density Einkommen")
plot(density(school),main="Kernel Density Bildung")
```

Ab hier beginnt die eigentliche Clusteranalyse
Nachfolgender Befehl ruft das Installationsprogramm für die Packete auf - cluster
installieren!

```
utils:::menuInstallPkgs()
library(cluster)
cluster<-data.frame(data$inc,data$school)
```

Beginn der hierarchischen Clusteranalyse

```
dist.euclid<-daisy(cluster,metric="euclidean",stand=TRUE)
dendogramm<-hclust(dist.euclid,method="average")
plot(dendogramm,xlab="Objekte",ylab="Distanzen",
main="Dendogramm der Clusteranalys (Average)",labels=FALSE)
cluster.hierarch_3<-cutree(dendogramm,k=3)
cluster.hierarch_3
data<-cbind(data, cluster.hierarch_3)
tapply(inc, cluster.hierarch_3, mean)
tapply(school, cluster.hierarch_3, mean)
table(cluster.hierarch_3)
cluster.hierarch_2<-cutree(dendogramm,k=2)
data<-cbind(data,cluster.hierarch_2)
tapply(inc, cluster.hierarch_2, mean)
tapply(school, cluster.hierarch_2, mean)
table(cluster.hierarch_2)
```

Beginn des k-Means Algorithmus

```
summary(school)
summary(inc)
school.stand<-(school-mean(school))/sd(school)
inc.stand<-(inc-mean(inc))/sd(inc)
cluster.k<-cbind(inc.stand,school.stand)
clusterzentren<-kmeans(cluster.k,centers=3)
clusterzentren
clusterzentren$cluster
cluster.k<-data.frame(cbind(cluster.k,clusterzentren$cluster))
cluster.k
names(cluster.k)
```

```
sozioökonomische.gruppe=factor(clusterzentren$cluster,
+labels=c("Niedrig","Mittel","Hoch"))
sozioökonomische.gruppe
data<-cbind(data,sozioökonomische.gruppe)
names(data)
data
tapply(inc, sozioökonomische.gruppe, mean)
tapply(school, sozioökonomische.gruppe, mean)
```

Abbildungsverzeichnis