

Statistische Software (R)

Paul Fink, M.Sc., Eva Endres, M.Sc.

Institut für Statistik

Ludwig-Maximilians-Universität München

Datensatz-Aufbereitung



Datenbeispiel

- chickwts:** Datensatz über das Gewicht in *Gramm* von 71 Küken, gefüttert mit 6 verschiedenen Beimischungen
- ges:** Vektor des Geschlechts der Küken (selbst gebaut)

Erzeugung der notwendigen Variablen

```
> daten <- chickwts  
> ges <- factor(sample(c("m", "f"), size = 71, replace = TRUE))
```

Übersicht über Variablennamen im Datensatz:

```
> names(daten)  
[1] "weight" "feed"
```

Zugriff auf Variablen in `data.frame`

- Listenzugriff mit Name der Variable

```
daten$weight
```

- Matrixzugriff mit Name der Variable

```
daten[, "weight"]
```

- Matrixzugriff mit Index der Variable

```
daten[, 1]
```

Vorteil von Matrixzugriff: Man kann auch mehrere Variablen gleichzeitig herausholen

Zugriff auf bestimmte Beobachtungen

Indizes bekannt: Matrixzugriff mit Indizes

```
daten[c(1, 4, 20), ]
```

Indizes unbekannt: Beobachtungen nach Kriterien finden

- Matrixzugriff:

```
daten[daten$feed == "casein", ]
```

- Funktion `subset()`:

```
subset(daten, feed == "casein")
```

Logische Operatoren & Verknüpfungen

<code>==, !=</code>	gleich, ungleich
<code>>, >=</code>	größer, größer gleich
<code><, <=</code>	kleiner, kleiner gleich
<code>!</code>	Negation (nicht)
<code>%in%</code>	enthalten in

<code>&, &&</code>	und
<code> , </code>	oder
<code>xor()</code>	entweder oder (ausschließend)

<code>TRUE,</code>	wahr
<code>FALSE,</code>	falsch

Nur '`&&`' und '`||`' sind **nicht** vektorwertig.

Die Funktion `subset()`

Befehlssyntax:

```
subset(Data.frame-Objekt, Kriterium an Variablen)
```

Beispiele:

```
subset(daten, feed == "casein")
```

```
subset(daten, feed %in% c("casein", "linseed"))
```

```
subset(daten, (feed == "casein") & (weight > 240))
```

Spart Schreibarbeit bei komplizierteren Kriterien

Neue Variablen hinzufügen

- Listenzugriff:

```
daten$gender <- ges
```

- `data.frame()` Funktion:

```
daten <- data.frame(daten, ges)
```

- `cbind()` Funktion:

```
daten <- cbind(daten, ges)
```

- Matrixzugriff:

```
daten[, 3] <- ges
```

Falls `length(ges) < nrow(daten)` \implies Recycling-Regel

Achtung: Bestehende Variablen werden überschrieben!

Variablen aus `data.frame` löschen

- Listenzugriff:

```
daten$gender <- NULL
```

- Mit Matrixzugriff herausfiltern

```
daten <- daten[, -3]
```


Ändern von Variablen in einem data.frame

Vorgehen anhand von Beispiel:

Das Gewicht soll in kg umgewandelt werden

```
# 1. Variable aus data.frame in Hilfsobjekt speichern  
h <- daten[, "weight"]  
# 2. Änderungen am Hilfsobjekt durchfuehren  
h <- h / 1000  
# 3. Hilfsobjekt als Variable in data.frame aufnehmen  
daten[, "weight"] <- h
```

oder einfacher:

```
# Alle Schritte laufen intern ab  
daten <- within(daten, {weight <- weight / 1000})
```

Weitere nützliche Funktionen

Beobachtungen der Größe nach ordnen:

```
daten[order(V1, V2, usw. ), ]
```

daten wird nach *V1* geordnet, wenn Werte gleich, dann nach *V2*, usw.

Beobachtungen aggregieren:

```
aggregate(zuaggVar ~ nachFaktorVar, FUN =  
Funktion, data = daten)
```

Zum Beispiel für das Gruppenmittel die Funktion **mean()**.

Schreibarbeit sparen:

```
with(daten, ... )
```

Innerhalb von ... stehen einem direkt die Variablennamen von **daten** zur Verfügung

Aufgaben

1. Lesen Sie den Datensatz *nba.asc* aus dem Datenarchiv des Instituts für Statistik (<http://www.statistik.lmu.de/service/datenarchiv/nba/nba.html>) in R ein als `data.frame` mit Namen *nba*.
2. Wandeln Sie die Variable *Datum* in den Datentyp `Date` um.
Hinweis: `?as.Date`
3. Berechnen Sie `hdiff` als neue Variable in *nba* die Differenz der erzielten Punkte der Auswärtsmannschaft von der Heimmannschaft.
4. Berechnen Sie die binäre Variable `siegh` in *nba*, die angibt, ob die Heimmannschaft gewonnen hat (1: Sieg, 0: Niederlage)