Einfaches Datenmanagement in R

Achim Zeileis

2009-02-20

1 Daten einlesen

Datensätze werden in R typischerweise als Objekte der Klasse `"data.frame"` dargestellt. In diesen entsprechen die Zeilen den Beobachtungen und die Spalten den verschiedenen Variablen. Es gibt eine Vielzahl an Möglichkeiten solche Daten in R einzulesen. Zwei der wichtigsten werden hier kurz vorgestellt. Ein einfaches Format zum Datenaustausch ist ‚csv' (comma-separated values), das auch von Programmen wie **Excel** importiert und exportiert werden kann. In deutschsprachigen Windows-Versionen steht in R der Befehl `read.csv2` zur Verfügung. Dieser returniert einen `"data.frame"`, der dann einem Objekt zugewiesen werden kann. Beispielsweise für den Bookbinders Book Club Datensatz:

```
R> BBBClub <- read.csv2("BBBClub.csv")
```

Für weitere Details der Benutzung kann man, wie bei jeder Funktion, die Hilfeseite per `?read.csv2` (bzw. völlig äquivalent `help(read.csv2)`) aufrufen.

Eine andere Möglichkeit, Daten einzulesen, die wir im Rahmen dieser LV häufig zum Datenaustausch nutzen werden, ist das eigene binäre Datenformat von R, das üblicherweise in ‚.rda' oder ‚.RData' Dateien abgespeichert wird. In diesem koennen beliebig viele Objekte beliebiger Typen (also nicht nur `"data.frame"`s) abgespeichert werden. Sie können mit `load` geladen werden, wodurch die darin enthaltenen Objekte direkt verfügbar sind, also keinem Objekt zugewiesen werden müssen. Beispielsweise macht der Befehl

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

den `"data.frame"` `BBBClub` direkt verfügbar und erzeugt damit also in diesem speziellen Fall dasselbe Ergebnis wie der obige Befehl.

Als zweiten Beispieldatensatz verwenden wir die GSA (Guest Survey Austria) Daten, die mit

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

geladen werden.

2 Auswahl von Teildatensätzen

Häufig enthalten Datensätze viel mehr Informationen als man zur Analyse bestimmter Sachverhalte bzw. zur Beantwortung konkreter Fragestellungen benötigt. Um nicht mit unnötig umfangreichen Daten arbeiten zu müssen ist es in der Regel empfehlenswert vor Beginn der eigentlichen Analyse einen geeigneten Teildatensatz auszuwählen und mit diesem alle weiteren Analysen durchzuführen.

In einem Datensatz entsprechenden die Spalten verschiedenen Merkmalen und die Zeilen die Beobachtungen, die zu einem bestimmten Merkmalsträger gehören. In beiden Richtungen kann man

Teilmengen auswählen. Zunächst wählt man die Beobachtungen/Merkmalsträger aus, beispielsweise die, die eine bestimmte Eigenschaft erfüllen (nur die Frauen, oder nur Kunden, die mehr als einen bestimmten Betrag ausgegeben haben, u.ä.). Dann läßt man die Variablen weg, die nicht für die Analyse benötigt werden. Ein besonderes Problem sind fehlende Werte, die oft gesondert behandelt werden müssen.

Zusammenfassend werden also Teildatensätze in folgenden Schritten ausgewählt:

1. Auswahl von Beobachtungen (Zeilen)
2. Auswahl der relevanten Variablen (Spalten)
3. Behandlung von fehlenden Werten (`NA`s)

2.1 Selektion von Beobachtungen

Zur Auswahl von Beobachtungen eines Datensatzes verwendet man in aller Regel logische Ausdrücke wie etwa „alle Kunden, die mindestens USD 200 ausgegeben haben“. Dies wird mit dem R-Befehl `subset(data.frame, logical)` gemacht. Das erste Argument ist immer ein `data.frame` und das zweite ein Vektor mit logischen Ausdrücken (die also entweder `TRUE` oder `FALSE` sind). Als Vergleichsoperatoren stehen in R u.a. Gleichheit (`==`), Ungleichheit (`!=`), größer (`>` und `>=`) und kleiner (`<` und `<=`) zur Verfügung. Außerdem können diese mit Hilfe der logischen Operatoren ‚und‘ (`&`) und ‚oder‘ (`|`) verknüpft werden.

Einige einfach Illustrationen sind:

```
R> x <- c(3, 5, 2, 9, 1)
R> x > 4

[1] FALSE  TRUE FALSE  TRUE FALSE

R> x > 6 | x < 2

[1] FALSE FALSE FALSE  TRUE  TRUE

R> x >= 3 & x <= 5

[1]  TRUE  TRUE FALSE FALSE FALSE
```

Wenn wir aus dem Datensatz `BBBClub` beispielsweise alle Beobachtungen auswählen wollen, die zu weiblichen Kunden gehören, die mindestens USD 200 ausgegeben haben, so können wir das entweder in zwei Schritten tun

```
R> bbbc <- subset(BBBClub, gender == "female")
R> bbbc <- subset(bbbc, amount >= 200)
```

oder völlig äquivalent auch in einem Schritt

```
R> bbbc <- subset(BBBClub, gender == "female" & amount >= 200)
```

Damit hat sich die Anzahl der Zeilen von 1,300 reduziert auf 228. Die Anzahl der Zeilen kann man entweder mit `nrow` abfragen oder mit `dim`, das zusätzlich die Anzahl der Spalten angibt.

```
R> dim(BBBClub)
```

```
[1] 1300   11

R> dim(bbbc)

[1] 228  11

R> nrow(bbbc)

[1] 228
```

Zusätzlich kann man Zeilen auch über die Auswahl von konkreten Indizes, d.h. Zeilennummern auswählen. Dies wollen wir anhand einer einfachen Matrix veranschaulichen:

```
R> x <- matrix(1:16, ncol = 4)
R> x

     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    2    6   10   14
[3,]    3    7   11   15
[4,]    4    8   12   16

R> x[c(1, 3), ]

     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    3    7   11   15

R> x[-2, ]

     [,1] [,2] [,3] [,4]
[1,]    1    5    9   13
[2,]    3    7   11   15
[3,]    4    8   12   16
```

D.h. man indiziert Matrizen (und auch Datensätze) mit `[i, j]`, wobei `i` die Zeilennummer(n) angibt und `j` die Spaltennummer(n). Wird `i` oder `j` unspezifiziert gelassen, so werden alle Zeilen bzw. Spalten benutzt. Bei negativen Indizes werden die entsprechenden Zeilen bzw. Spalten weggelassen.

2.2 Selektion von Variablen

Die Selektion von relevanten Variablen (oder das Weglassen der irrelevanten Variablen) funktioniert ganz einfach über die Indizierung mit `[i, j]` wie bereits im letzten Abschnitt erläutert.

```
R> x[, c(2, 4)]

     [,1] [,2]
[1,]    5   13
[2,]    6   14
[3,]    7   15
[4,]    8   16
```

```
R> x[, -3]

     [,1] [,2] [,3]
[1,]    1    5   13
[2,]    2    6   14
[3,]    3    7   15
[4,]    4    8   16
```

Ein Datensatz hat außerdem immer Spaltennamen, d.h. Variablennamen, die anstatt der Spaltennummern benutzt werden können. Der `bbbc` Datensatz hat folgende Spaltennamen:

```
R> names(bbbc)

 [1] "choice" "gender" "amount" "freq"   "last"   "first"  "child"  "youth"
 [9] "cook"   "diy"    "art"
```

die äquivalent auch mit `colnames(bbbc)` abgefragt werden können. Wenn wir nun den Zusammenhang von Kaufentscheidung (`choice`) und der Anzahl bereits gekaufter Kunstbücher untersuchen wollten, könnten wir den entsprechenden Teildatensatz mit

```
R> bbbc <- bbbc[, c("choice", "art")]
```

oder wiederum äquivalent mit `bbbc <- bbbc[, c(1, 11)]`. Der resultierende Datensatz hat jetzt nur noch zwei Spalten:

```
R> dim(bbbc)

[1] 228   2

R> ncol(bbbc)

[1] 2
```

2.3 Fehlende Werte

Nicht immer sind alle Merkmale bei allen Merkmalsträgern erhoben worden, etwa wegen eines Messfehlers, weil die Daten verloren gegangen sind oder weil das Merkmal unbeobachtbar war. Mit dem Fehlen von Werten muß man sich bei der Analyse von Daten ebenfalls auseinandersetzen: dabei ist die defensivste Strategie, die Beobachtungen wegzulassen, bei denen nicht alle Merkmale wegzulassen. Dabei kann aber unter Umständen wertvolle Information verloren gehen, weshalb in vielen Anwendungen Anstrengungen unternommen werden, fehlende Werte zu imputieren, also durch geeignete Werte zu ersetzen. Hierfür ist häufig Hintergrundwissen hilfreich. In bestimmten Situation kann auch das Fehlen einer Beobachtung selbst von Interesse sein.

In R werden fehlende Werte durch `NA` (für not available) repräsentiert und es gibt verschiedene Funktionen, die speziell für deren Handhabung bereitgestellt werden. Besonders wichtig ist die Funktion `na.omit`, die in dem übergebenen Datensatz alle Zeilen wegläßt, in denen es mindestens einen fehlenden Wert gibt.

Als Beispiel benutzen wir den `GSA` Datensatz, anhand dessen wir einmal alle drei Schritte bei der Selektion von Teildatensätzen durchgehen wollen. Hier sollen nur die Touristen betrachtet werden, die das Burgenland besucht haben. Von diesen wollen wir den Zusammenhang von Einkommen und Ausgaben untersuchen; fehlende Werte sollen weggelassen werden.

```
R> dim(GSA)
```

```
[1] 14571    52

R> gsa <- subset(GSA, province == "Burgenland")
R> dim(gsa)

[1] 1077   52

R> gsa <- gsa[, c("income", "expenditure")]
R> dim(gsa)

[1] 1077    2

R> gsa <- na.omit(gsa)
R> dim(gsa)

[1] 810   2
```

Hierbei ist es besonders wichtig, daß die fehlenden Werte erst ganz am Schluss weggelassen werden, da sonst auch fehlende Werte in möglicherweise irrelevanten Variablen dazu führen, daß einige Beobachtungen weggelassen werden, obwohl `income` und `expenditure` verfügbar wären.