1 Der Datentyp factor in R

2 Beispieldatensatz

```
> patienten<-data.frame(
      c(28,61,79,54,NA,48),
      c(2,2,3,2,1,1),
      c("Pillen","Pillen","Massage","Gymnastik","Massage","Gymnastik"),

c("10.1.2003","30.12.1999","1.1.2000","27.2.2003","1.3.2003","30.8.2001"),
      c(TRUE,FALSE,FALSE,FALSE,TRUE,FALSE))
> names(patienten)<-c("Alter","Schweregrad","Behandlung","Datum","Erfolg")
> patienten
  Alter Schweregrad Behandlung      Datum Erfolg
1    28           2     Pillen  10.1.2003   TRUE
2    61           2     Pillen 30.12.1999  FALSE
3    79           3    Massage   1.1.2000  FALSE
4    54           2  Gymnastik  27.2.2003  FALSE
5    NA           1    Massage   1.3.2003   TRUE
6    48           1  Gymnastik  30.8.2001  FALSE
```

alle nicht numerischen oder logischen Vektoren werden automatisch zum Datentyp factor:

```
> lapply(patienten,is.factor)
$Alter
[1] FALSE
$Schweregrad
[1] FALSE
$Behandlung
[1] TRUE
$Datum
[1] TRUE
$Erfolg
[1] FALSE
```

```
> attach(patienten)
```

```
> Behandlung
[1] Pillen    Pillen    Massage   Gymnastik Massage   Gymnastik
Levels: Gymnastik Massage Pillen
```

alphabetisch sortiert!

```
> as.character(Behandlung)
[1] "Pillen"    "Pillen"    "Massage"   "Gymnastik" "Massage"   "Gymnastik"
> as.numeric(Behandlung)
[1] 3 3 2 1 2 1
levels(Behandlung)
```

```
[1] "Gymnastik" "Massage"   "Pillen"
> codes(Behandlung)
[1] 3 3 2 1 2 1
> labels(levels(Behandlung))
[1] "1" "2" "3"
```

Obacht!

```
> labels(Behandlung)
[1] "1" "2" "3" "4" "5" "6"
> unclass(Behandlung)
[1] 3 3 2 1 2 1
attr(,"levels")
[1] "Gymnastik" "Massage"   "Pillen"
```

Mit levels kann man die gültigen Werte einschränken. Werte, die nicht in den levels vorkommen, werden auf NA gesetzt.

vgl. Dalgaard S. 16/17

Faktoren und der Datentyp "numeric"

```
> factor(Alter)
[1] 28   61   79   54   <NA> 48
Levels: 28 48 54 61 79
> is.numeric(factor(Alter))
[1] FALSE
> mode(factor(Alter))
[1] "numeric"
> as.numeric(factor(Alter))
[1]  1  4  5  3 NA  2
```

Ein Faktor ist als numerischer Vektor gespeichert, die numerischen Werte stimmen aber nicht mit den Levels überein, auch wenn diese ebenfalls Zahlen sind.

```
Wie bekommt man die ursprünglichen Zahlenwerte?
as.numeric(as.character(factor(Alter)))
[1] 28 61 79 54 NA 48
oder
as.numeric(levels(factor(Alter)))[as.integer(factor(Alter))]
(letzteres ist Rechenzeit-effizienter)
```

Behandlung von NA

```
> test<-factor(Alter,exclude=NULL)
> test
[1] 28   61   79   54   <NA> 48
Levels: 28 48 54 61 79 NA
> as.numeric(test)
[1] 1 4 5 3 6 2
```

ordered factor

```
> as.ordered(Schweregrad)
[1] 2 2 3 2 1 1
Levels: 1 < 2 < 3
```

> Schweregrad<- ordered(Schweregrad,levels=1:3,labels=c("leicht","mittel","schwer"))

```
> Schweregrad
[1] mittel mittel schwer mittel leicht leicht
Levels: leicht < mittel < schwer
Beachte levels und labels im Vergleich zur Definition
> levels(Schweregrad)
[1] "leicht" "mittel" "schwer"
> labels(levels(Schweregrad))
[1] "1" "2" "3"
> as.numeric(Schweregrad)
[1] 2 2 3 2 1 1
```

2.1 Datum

```
> Datum
[1] 10.1.2003  30.12.1999 1.1.2000   27.2.2003  1.3.2003   30.8.2001
Levels: 1.1.2000 1.3.2003 10.1.2003 27.2.2003 30.12.1999 30.8.2001
> as.ordered(Datum)
[1] 10.1.2003  30.12.1999 1.1.2000   27.2.2003  1.3.2003   30.8.2001
Levels: 1.1.2000 < 1.3.2003 < 10.1.2003 < 27.2.2003 < 30.12.1999 < 30.8.2001
```

das ist die "alphabetische Ordnung"

```
> strptime(Datum,format="%d.%m.%Y")
Error in strptime(Datum, format = "%d.%m.%Y") :
        invalid ‘x’ argument
```

strptime akzeptiert einen factor-Wert offensichtlich nicht zum Umwandeln in Datum

```
> strptime(as.character(Datum),format="%d.%m.%Y")
[1] "2003-01-10" "1999-12-30" "2000-01-01" "2003-02-27" "2003-03-01"
"2001-08-30"
> Datum2<-strptime(as.character(Datum),format="%d.%m.%Y")
> is.factor(Datum2)
[1] FALSE
> is.numeric(Datum2)
[1] FALSE
> class(Datum2)
[1] "POSIXt"  "POSIXlt"
```

Datumdarstellung funktioniert in Splus anders. Die Splus-Darstellung findet man in R in der chron-Library

```
> as.ordered(Datum2)
Error in unique.default(x) : unique() applies only to vectors
> as.ordered(as.character(Datum2))
[1] 2003-01-10 1999-12-30 2000-01-01 2003-02-27 2003-03-01 2001-08-30
6 Levels: 1999-12-30 < 2000-01-01 < 2001-08-30 < 2003-01-10 < ... < 2003-03-01
```

Die richtige Ordnung erhält man wegen der Darstellung Y-m-d

```
> patienten$Datum<-as.ordered(as.character(Datum2))
```

so sieht jetzt unser Frame aus:

```
> patienten
  Alter Schweregrad Behandlung      Datum Erfolg
1    28      mittel     Pillen 2003-01-10   TRUE
2    61      mittel     Pillen 1999-12-30  FALSE
3    79      schwer    Massage 2000-01-01  FALSE
4    54      mittel  Gymnastik 2003-02-27  FALSE
5    NA      leicht    Massage 2003-03-01   TRUE
6    48      leicht  Gymnastik 2001-08-30  FALSE
> lapply(patienten,class)
$Alter
NULL
$Schweregrad
[1] "ordered" "factor"
$Behandlung
[1] "factor"
$Datum
[1] "ordered" "factor"
$Erfolg
NULL
```

in Splus erscheint bei $Alter "integer" und bei $Erfolg "logical"

```
> lapply(patienten,mode)
$Alter
[1] "numeric"
$Schweregrad
[1] "numeric"
$Behandlung
[1] "numeric"
```

$Datum

[1] "numeric"

```
$Erfolg
[1] "logical"
```

aber

> is.numeric(Datum)

[1] FALSE

nochmal Obacht!

> as.numeric(Datum)

[1] 4 1 2 5 6 3

logische Abfragen und arithmetische Operationen

```
> Behandlung==1
[1] FALSE FALSE FALSE FALSE FALSE FALSE
> Behandlung-1
[1] NA NA NA NA NA NA
Warning message:
"-" not meaningful for factors in: Ops.factor(Behandlung, 1)
in Splus keine Warnung
> Behandlung>1
[1] NA NA NA NA NA NA
Warning message:
">" not meaningful for factors in: Ops.factor(Behandlung, 1)
in Splus keine Warnung
> Behandlung==Pillen
Error: Object "Pillen" not found
> Behandlung=="Pillen"
[1]  TRUE  TRUE FALSE FALSE FALSE FALSE
> as.numeric(Behandlung)==1
[1] FALSE FALSE FALSE  TRUE FALSE  TRUE
identisch mit
> codes(Behandlung)==1
[1] FALSE FALSE FALSE  TRUE FALSE  TRUE
aber es funktionieren Character-Operationen:
> paste(Behandlung,"1")
[1] "Pillen 1"    "Pillen 1"    "Massage 1"   "Gymnastik 1" "Massage 1"
[6] "Gymnastik 1"
```

speziell bei ordered factors:

```
> Schweregrad
[1] mittel mittel schwer mittel leicht leicht
Levels: leicht < mittel < schwer
> Schweregrad>1
[1] NA NA NA NA NA NA
> Schweregrad>leicht
Error: Object "leicht" not found
> Schweregrad>"leicht"
[1]  TRUE  TRUE  TRUE  TRUE FALSE FALSE
> Schweregrad[2:3]>Schweregrad[4:5]
[1] FALSE  TRUE
```

3 Replacement

Zuweisung eines Werts, der bereits in Levels enthalten ist:

```
> test<-Behandlung
test[test=="Pillen"]<-"Massage"
> test
[1] Massage   Massage   Massage   Gymnastik Massage   Gymnastik
Levels: Gymnastik Massage Pillen
```

Zuweisung eines Werts, der nicht in Levels enthalten ist:

```
test<-Behandlung
> test[test=="Pillen"]<-"Creme"
Warning message:
invalid factor level, NAs generated in: "[<-.factor"(*tmp*, test == "Pillen",
value = "Creme")
> test
[1] <NA>      <NA>      Massage   Gymnastik Massage   Gymnastik
Levels: Gymnastik Massage Pillen
```

Es funktioniert, wenn man zuerst die Levels neu definiert:

```
test<-Behandlung
levels(test)<-c(levels(test),"Creme")
#---hier bringt Splus eine Fehlermeldung, weil widersprüchliche Zahl von
Levels
test[test=="Pillen"]<-"Creme"
test
[1] Creme     Creme     Massage   Gymnastik Massage   Gymnastik
Levels: Gymnastik Massage Pillen Creme
```

hier ist die alphabetische Ordnung verlorengegangen!

wiederholtes Anwenden von factor schmeißt nicht gebrauchte leves weg:

(Beispiel aus help-File)

```
> ff<-factor(substring("statistics",1:10,1:10),levels=letters)
> ff
 [1] s t a t i s t i c s
Levels: a b c d e f g h i j k l m n o p q r s t u v w x y z
> codes(ff)
 [1] 19 20  1 20  9 19 20  9  3 19
> factor(ff)
 [1] s t a t i s t i c s
Levels: a c i s t
> codes(factor(ff))
 [1] 4 5 1 5 3 4 5 3 2 4
```

merging von factor-Vektoren:

```
> test<-c("Creme",Behandlung)
> test
[1] "Creme" "3"     "3"     "2"     "1"     "2"     "1"
```

Umwandlung in Character, weil gemischte Datentypen!

```
> as.numeric(test)
[1] NA  3  3  2  1  2  1
Warning message:
NAs introduced by coercion
> factor(test)
[1] Creme 3     3     2     1     2     1
Levels: 1 2 3 Creme
```

aber:

```
> levels(Behandlung)
[1] "Gymnastik" "Massage"   "Pillen"
> labels(levels(Behandlung))
[1] "1" "2" "3"
> as.numeric(factor(test))
[1] 4 3 3 2 1 2 1
```

es hilft auch nichts, zunächst "Creme" in factor umzuwandeln:

```
> test<-c(as.factor("Creme"),Behandlung)
> test
[1] 1 3 3 2 1 2 1
> is.factor(test)
[1] FALSE
> test<-factor(c(as.factor("Creme"),Behandlung))
> test
[1] 1 3 3 2 1 2 1
Levels: 1 2 3
```

Vor dem Mergen muss man zunächst Faktoren in Character zurückverwandeln:

```
> test2<-c("Creme",a.character(Behandlung))
> factor(test2)
[1] Creme     Pillen    Pillen    Massage   Gymnastik Massage   Gymnastik
Levels: Creme Gymnastik Massage Pillen
> as.numeric(factor(test2))
[1] 1 4 4 3 2 3 2
```

4 Spaltenweises Mergen von Data Frames

Füge zweites Datum in das Data Frame ein

(2 Tage nach dem ursprünglichen Datum; für Berechnungen im Datum benötigt man die chron-library)

```
> Datum2<-
c("12.1.2003","1.1.2000","3.1.2000","29.2.2003","3.3.2003","1.9.2001")
> Datum2<-strptime(as.character(Datum2),format="%d.%m.%Y")
> patienten<-cbind(patienten,Datum2)
> patienten
  Alter Schweregrad Behandlung      Datum Erfolg     Datum2
1    28           2     Pillen 2003-01-10   TRUE 2003-01-12
2    61           2     Pillen 1999-12-30  FALSE 2000-01-01
3    79           3    Massage 2000-01-01  FALSE 2000-01-03
4    54           2  Gymnastik 2003-02-27  FALSE 2003-03-01
5    NA           1    Massage 2003-03-01   TRUE 2003-03-03
6    48           1  Gymnastik 2001-08-30  FALSE 2001-09-01
> lapply(patienten,class)
$Alter
NULL
$Schweregrad
NULL
$Behandlung
[1] "factor"
$Datum
[1] "ordered" "factor"
$Erfolg
NULL
$Datum2
[1] "POSIXt"  "POSIXct"
```

Beim Mergen wird die Variablenklasse beibehalten!

(anders als in früheren Versionen von R und Splus)

Aber Obacht mit cbind:

> testframe<-cbind(1:6,Behandlung)

```
> testframe
       Behandlung
[1,] 1          3
[2,] 2          3
[3,] 3          2
[4,] 4          1
[5,] 5          2
[6,] 6          1
> apply(testframe,2,mean)
```

```
[1] 3.5 2.0
```

es wurde ohne Warnung in numeric umgewandelt!

Was für ein Objekt ist testframe?

```
> class(testframe[,2])
NULL
> is.data.frame(testframe)
[1] FALSE
```

Bei cbind ist offensichtlich in eine numerische Matrix umgewandelt worden.

Neuer Versuch:

```
> testframe<-data.frame(1:6,Behandlung)
> testframe
  X1.6 Behandlung
1    1     Pillen
2    2     Pillen
3    3    Massage
4    4  Gymnastik
5    5    Massage
6    6  Gymnastik
> apply(testframe,2,mean)
      X1.6 Behandlung
        NA         NA
Warning messages:
1: argument is not numeric or logical: returning NA in: mean.default(newX[,
i], ...)
2: argument is not numeric or logical: returning NA in: mean.default(newX[,
i], ...)
```

aber

```
> lapply(testframe,mean)
$X1.6
[1] 3.5
$Behandlung
[1] NA
Warning message:
argument is not numeric or logical: returning NA in: mean.default(X[[2]], ...)
```

Fazit: Behandelt man ein Data Frame als Matrix, geht die Klassierung als Faktor verloren. Verwende statt cbind data.frame.

Äquivalent sind

```
> testframe<-data.frame(1:6,Behandlung)
```

und

```
> testframe<-cbind(1:6,data.frame(Behandlung))
```

Was passiert bei Mittelwertbildung?

```
> lapply(patienten,mean)
```

```
$Alter
[1] NA
numerisch, aber mit NA
$Schweregrad
[1] 1.833333
```
bei ordered factor wird Mittelwert gebildet!
```
$Behandlung
[1] NA
$Datum
[1] NA
Mittelwertbildung bei Faktoren nicht sinnvoll
$Erfolg
[1] 0.3333333
Mittelwertbildung nach Umcodierung von TRUE in 1 und FALSE in 0
$Datum2
[1] "2001-11-02 15:50:00 Westeuropäische Normalzeit"
hier war Datum nicht in factor umgewandelt worden
Warning messages:
1: argument is not numeric or logical: returning NA in: mean.default(X[[3]],
...)
2: argument is not numeric or logical: returning NA in: mean.default(X[[4]],
...)
>
```

5 Import mit read.table

Excel-File als txt (tab-delimited) speichern, importieren mit

```
test<-read.table("c:\\temp\\test.txt",header=TRUE)
```

(sonst versteht R die Spaltennamen als erste Datenzeile)

Dann geschieht die Umwandlung so wie bei der direkten Eingabe

Bei Splus6 geschieht der Import über das Menü aus Excel-Files korrekt, aber wenn in der 1. Zeile ein NA ist, wandelt Splus diese Spalte immer in factor um.

Plotten von Faktoren

plot(x,y) x als Faktor ruft plot.factor auf, dieses wiederum versucht boxplot oder barplot zu erzeugen

testx_factor(c(rep("Gruppe2",10),rep("Gruppe1",10),rep("Gruppe3",10)))

```
testy_runif(30)
plot(testx,testy) gibt boxplot!
plot(testx,testy,type="p")
Plot-Methode plot.factor - erzeugt boxplot oder barplot!
Abhilfe schafft
plot(unclass(testx),testy,axes=F)
axis(1,at=unique(testx),labels=levels(testx))
axis(2)
box()
```

```
#---Addition von Tagen habe ich mit strptime nicht rausgekriegt
library(chron)
>
datum.orig_c("10.1.2003","30.12.1999","1.1.2000","27.2.2003","1.3.2003","30.8.2
001")
> dates(datum.orig,format="d.m.y")
[1] 10.01.03 30.12.99 01.01.00 27.02.03 01.03.03 30.08.01
> dates(datum.orig,format="d.m.y")+2
[1] 12.01.03 01.01.00 03.01.00 01.03.03 03.03.03 01.09.01
> ordered(dates(datum.orig,format="d.m.y")+2)
[1] 12064 10957 10959 12112 12114 11566
Levels: 10957 < 10959 < 11566 < 12064 < 12112 < 12114
> class(dates(datum.orig,format="d.m.y"))
[1] "dates" "times"
Datum2_dates(datum.orig,format="d.m.y")+2

#---Datentyp wird erhalten, nicht mehr zwangsweise in factor umgewandelt
testframe_data.frame(patienten,Datum2)
> lapply(testframe,class)
$Alter
NULL
$Schweregrad
NULL
$Behandlung
[1] "factor"
$Datum
[1] "factor"
$Erfolg
NULL
$Datum2
[1] "dates" "times"
```

Damit sind Probleme behoben, die alte Versionen von R und Splus brachten!