Datenanalyse mit R mosaic

Karsten Lübke

2018-07-27

Vorbemerkungen

- R unterscheidet zwischen Groß- und Kleinbuchstaben
- R verwendet den Punkt `.` als Dezimaltrennzeichen
- Fehlende Werte werden in R durch `NA` kodiert
- Eine Ergebniszuweisung erfolgt über `<-`
- Hilfe zur Funktion `foo`: `?foo`

Innerhalb von `mosaic`:

```
analysiere(y ~ x | z , data = Daten)
```

d. h., modelliere `y` in Abhängigkeit von `x` getrennt bzw. bedingt für `z` aus dem Datensatz `Daten`.[1]

Zusatzpakete müssen vor der ersten Benutzung einmalig installiert und geladen werden:

```
# Einmalig installieren
install.packages("mosaic")
# Laden, einmalig in jeder Sitzung
library(mosaic)
```

Daten

Einlesen

```
# csv Datensatz einlesen
Daten <- read.csv2("Pfad/Datei")
# xlsx Datensatz einlesen
library(readxl) # Paket zum xlsx Import
Daten <- read_excel("Pfad/Datei")
```

Datenhandling

Paket `dplyr` (mit `mosaic` geladen)

```
filter()    # Beobachtungen filtern
select()    # Variablen wählen
mutate()    # Variablen verändern/ erzeugen
summarise() # Beobachtungen zusammmenfassen
group_by()  # Beobachtungen gruppieren
%>%         # Übergabe von Ergebnissen
```

Datenanalyse

Grafische Verfahren

```
bargraph()   # Balkendiagramm
histogram()  # Histogramm
bwplot()     # Boxplot
xyplot()     # Streudiagrmm
mosaicplot() # Mosaikplot
# Plot von Mittelwert und Konidenzintervall:
gplots::plotmeans()
```

Kennzahlen

```
inspect()  # Datenübersicht
tally()    # Tabellierung, Häufigkeiten
prop()     # Anteile
diffprop() # Differenz zweier Anteile
favstats() # Kennzahlübersicht
mean()     # Arithmetischer Mittelwert
diffmean() # Differenz zweier Mittelwerte
cor()      # Korrelationskoefizient
```

Verteilungen, Simulation

Normalverteilung

```
xpnorm() # Verteilungsfunktion Normalverteilung
xqnorm() # Quantilsfunktion Normalverteilung
```

Randomisierung, Simulationen

```
set.seed() # Zufallszahlengenerator setzen
rflip()    # Münzwurf
do() *     # Wiederholung (Schleife)
sample()   # Stichprobe ohne Zurücklegen
resample() # Stichprobe mit Zurücklegen
shuffle()  # Permutation
```

Inferenz / Modellierung

Testverfahren

```
prop.test()   # Binomialtest (approximativ)
xchisq.test() # Chi-Quadrat Unabhängigkeitstest
t.test()      # t-Test
aov()         # Varianzanalyse (ANOVA)
```

Modellierung

```
lm()                        # Lineare Regression
glm(, family="binomial")    # Logistische Regression
plotModel()                 # Modell zeichnen
anova()                     # ANOVA Tabelle
residuals()                 # Residuen
fitted()                    # Angepasste Werte
predict()                   # Vorhersagen
prcomp()                    # Hauptkomponentenanalyse
kmeans()                    # K-Means Clusteranalyse
```

[1] Beim Mac ist `~` die Tastenkombination `alt+n`, `|` die Tastenkombination `alt+7`

Beispielanalyse

Vorbereitung

```
library(mosaic)    # mosaic laden
data(KidsFeet)     # Interner Datensatz
?KidsFeet          # Hilfe zum Datensatz
inspect(KidsFeet) # Deskriptive Daten
```

Eine kategoriale Variable

```
bargraph( ~ domhand, data = KidsFeet)
tally( ~ domhand, data = KidsFeet)
prop( ~ domhand, success = "L", data = KidsFeet)
```

Eine metrische Variable

```
histogram( ~ length, data = KidsFeet)
favstats(~ length, data = KidsFeet)
```

Zwei kategoriale Variablen

```
mosaicplot(biggerfoot ~ domhand, data = KidsFeet)
tally(biggerfoot ~ domhand, data = KidsFeet)
xchisq.test(biggerfoot ~ domhand, data = KidsFeet)
```

Zwei numerische Variablen

```
xyplot(width ~ length, data = KidsFeet)
cor(width ~ length, data = KidsFeet)
cor.test(width ~ length, data = KidsFeet)
```

Zwei Stichproben: kategorial

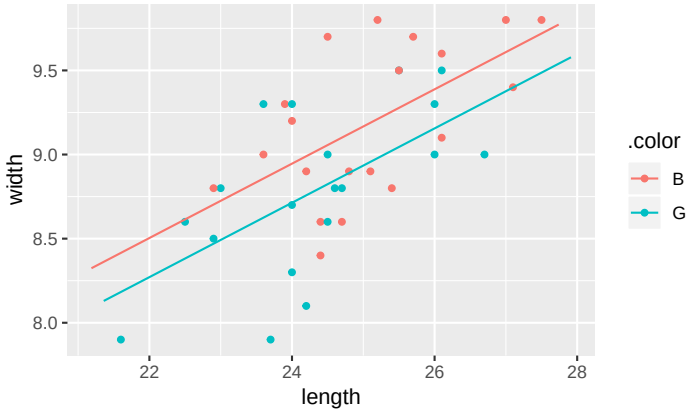
```
bargraph( ~ domhand | sex, data = KidsFeet)
prop(domhand ~ sex, success = "L",
     data = KidsFeet)
prop.test(domhand ~ sex, success = "L",
          data = KidsFeet)
```

Zwei Stichproben: numerisch

```
histogram( ~ length | sex, data = KidsFeet)
bwplot(length ~ sex, data = KidsFeet)
favstats(length ~ sex, data = KidsFeet)
t.test(length ~ sex, data = KidsFeet)
```

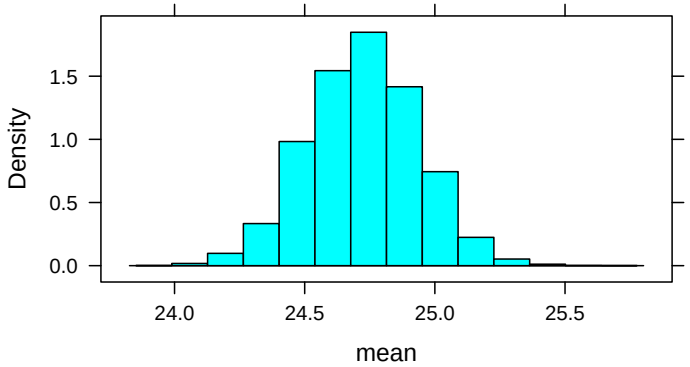
Lineare Regression

```
erglm <- lm(width ~ length + sex, data = KidsFeet)
plotModel(erglm)
summary(erglm)
anova(erglm)
```



Bootstrap[2]

```
set.seed(1896)
Bootvtlg <- do(10000) *
  mean(~ length, data = resample(KidsFeet))
histogram( ~ mean, data = Bootvtlg)
```

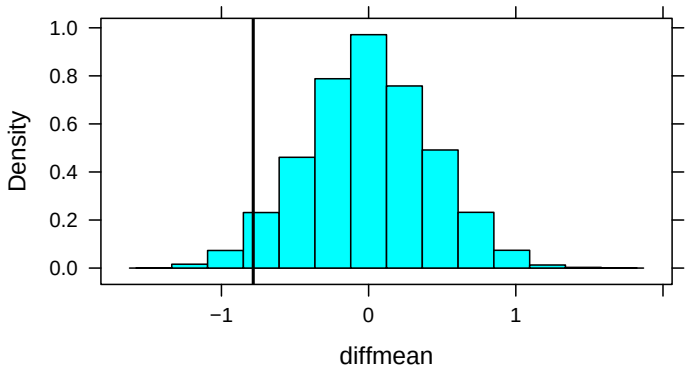



```
# Bootstrap Konfidenzintervall
quantile( ~ mean, probs = c(0.025, 0.975),
          data = Bootvtlg)
```

```
##     2.5%    97.5%
## 24.30513 25.13333
```

Permutationstest

```
set.seed(1896)
mdiff <- diffmean(length ~ sex, data = KidsFeet)
Nullvtlg <- do(10000) *
  diffmean(length ~ shuffle(sex), data = KidsFeet)
histogram( ~ diffmean, v=mdiff, data = Nullvtlg)
```



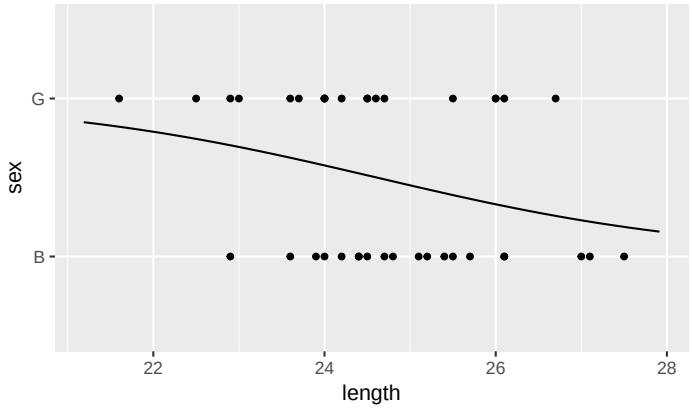

```
# Permutationstest p-Wert, H_0: MW gleich
prop( ~ abs(diffmean)>=abs(mdiff), data = Nullvtlg)
```

```
## prop_TRUE
##    0.0646
```

[2] Datensatz hier eher zu klein für Bootstrap Perzentile

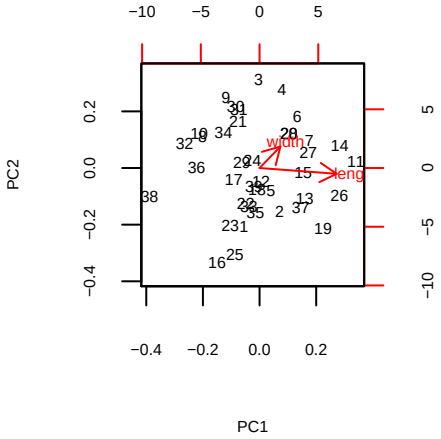
Logistische Regression

```
ergglm <- glm(sex ~ length, family = binomial,
              data = KidsFeet)
plotModel(ergglm)
summary(ergglm)
anova(ergglm)
# Odds Ratio
exp(coef(ergglm))
```



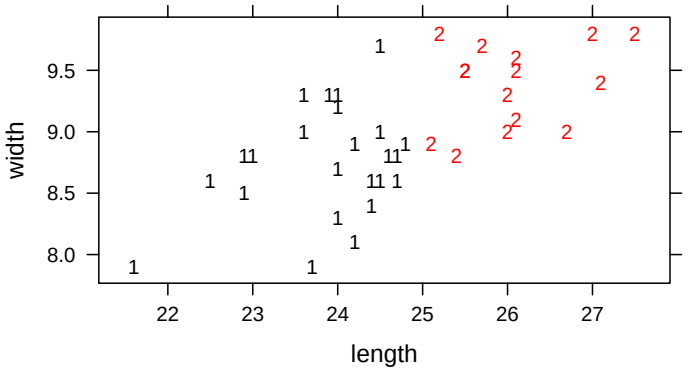
Hauptkomponentenanalyse (PCA)

```
ergpca <- prcomp( ~ length + width,
                  data = KidsFeet)
plot(ergpca) # Screeplot
summary(ergpca)
biplot(ergpca)
```



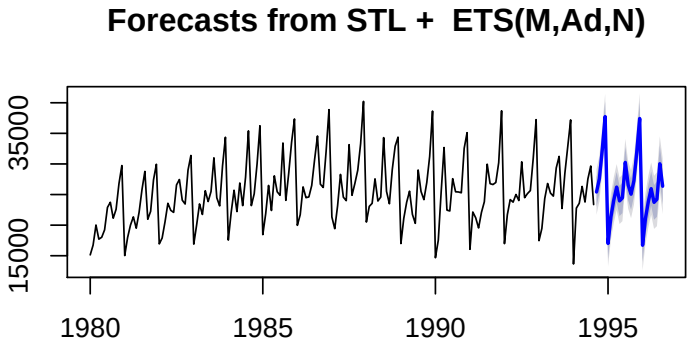
Clusteranalyse

```
set.seed(1896)
# Numerische Variablen auswählen
KidsFeetnumeric <- KidsFeet %>%
  select(length, width)
ergkmeans <- kmeans(KidsFeetnumeric,
                    centers = 2) # k=2
ergkmeans
```



Zeitreihenzerlegung[3]

```
library(forecast) # Paket forecast laden
data(wineind)     # Interner Datensatz
?wineind          # Hilfe zum Datesatz
wineind
plot(wineind)
# Zerlegung
ergstl <- stl(wineind, s.window = 11)
plot(ergstl)
# Vorhersagen
predstl <- predict(ergstl)
plot(predstl)
```



- Lizenz Creative Commons Attribution-ShareAlike 3.0 Unported.

- R Version: 3.5.0
- mosaic Version: 1.3.0

[3] Anderer Beispieldatensatz.