

Datenanalyse mit R mosaic und ggformula

Karsten Lübke
2019-09-05

Vorbemerkungen

- R unterscheidet zwischen Groß- und Kleinbuchstaben
- R verwendet den Punkt . als Dezimaltrennzeichen
- Fehlende Werte werden in R durch NA kodiert
- Eine Ergebnisuweisung 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", type = "binary")
# Laden, einmalig in jeder Sitzung
library(mosaic)
```

Daten

Einlesen:

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

Datenhandling mit Paket dplyr (mit mosaic geladen):

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

Logik:

```
== ; !=      # Gleichheit bzw. Ungleichheit
> ; >= ; <= ; < # größer bzw. kleiner (gleich)
& ; |        # und bzw. oder
```

Arithmetik:

```
+ ; - ; * ; : # Grundrechenarten
^ ; sqrt(x)   # Potenz bzw. Quadratwurzel
exp(x) ; log(x) # e^x bzw. ln(x)
abs(x)        # Absolutbetrag
```

Datenanalyse

Grafische Verfahren:

```
gf_bar()      # Säulendiagramm
gf_histogram() # Histogramm
gf_boxplot()  # Boxplot
gf_point()    # Streudiagramm
mosaicplot()  # Mosaikplot (nicht ggformula)
```

Kennzahlen:

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

Verteilungen, Simulation

Normalverteilung:

```
xpnorm()      # Verteilungsfunktion Normalverteilung
xqnorm()      # Quantilsfunktion Normalverteilung
gf_qq()       # QQ-Plot (allgemein)
```

Randomisierung, Simulationen:

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

Modellierung

```
lm()          # Lineare Regression
glm(, family="binomial") # Logistische Regression
plotModel()   # Modell zeichnen
residuals()   # Residuen
fitted()      # Angepasste Werte
predict()     # Vorhersagen
```

Inferenz

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

¹Beim 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) # Datenübersicht
```

Eine kategoriale Variable:

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

Eine metrische Variable:

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

Zwei kategoriale Variablen:

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

Zwei numerische Variablen:

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

Zwei Stichproben: kategorial:

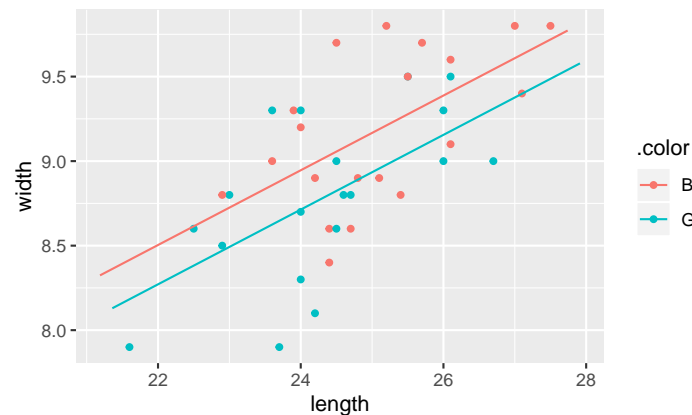
```
gf_bar(~ domhand | sex, data = KidsFeet)
prop(domhand ~ sex, success = "L",
     data = KidsFeet)
prop.test(domhand ~ sex, success = "L",
          data = KidsFeet)
```

Zwei Stichproben: numerisch:

```
gf_histogram(~ length | sex, data = KidsFeet)
gf_boxplot(length ~ sex, data = KidsFeet)
favstats(length ~ sex, data = KidsFeet)
t.test(length ~ sex, data = KidsFeet)
```

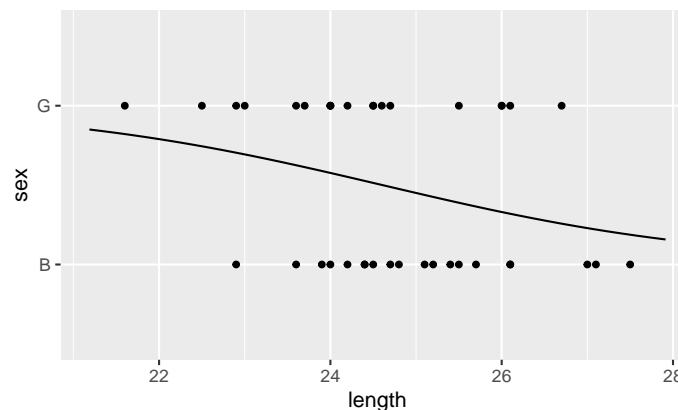
Lineare Regression:

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



Logistische Regression:

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

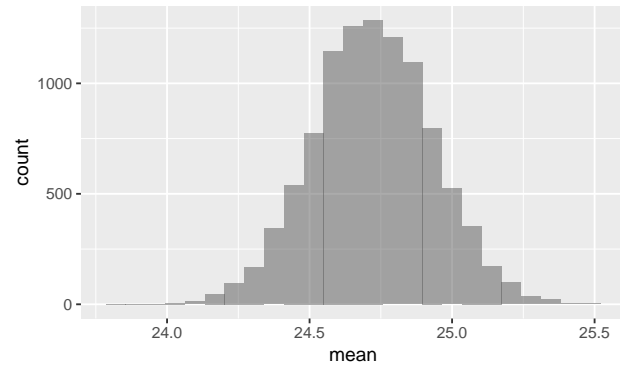


Datenhandling:

```
# Variablen selektieren
KidsFeet.length <- KidsFeet %>%
  select(length)
# Beobachtungen auswählen
KidsFeet.boys <- KidsFeet %>%
  filter(sex == "B")
# Variablen erzeugen
KidsFeet.in <- KidsFeet %>%
  mutate(length.in = 0.394*length)
# Bedingungen
KidsFeet.grouped <- KidsFeet %>%
  mutate(length.grouped = case_when(
    length > 25 ~ "large",
    length <= 25 ~ "not large"))
```

Bootstrap:²

```
set.seed(1896)
# Simuliere Ziehen zufälliger Stichprobe
Bootvtlg <- do(10000) *
  mean(~ length, data = resample(KidsFeet))
# Bootstrap-Verteilung
gf_histogram( ~ mean, data = Bootvtlg)
```

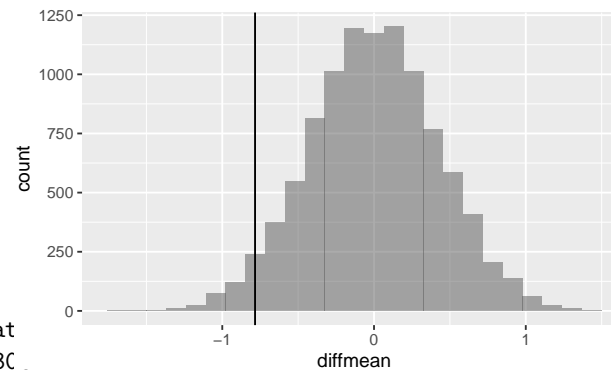


```
# Bootstrap Konfidenzintervall
confint(Bootvtlg)
```

```
##   name   lower upper level   method estimat
## 1 mean 24.31019 25.1334 0.95 percentile 24.7230
```

Permutationstest:

```
set.seed(1896)
# Beobachtete Differenz
est.diff <- diffmean(length ~ sex,
  data = KidsFeet)
# Simuliere zufällige Zuordnung:
Nullvtlg <- do(10000) *
  diffmean(length ~ shuffle(sex),
  data = KidsFeet)
# Verteilung unter H_0
gf_histogram( ~ diffmean, data = Nullvtlg) %>%
  gf_vline(xintercept = ~ est.diff)
```



```
# Permutationstest p-Wert
prop( ~ abs(diffmean) >= abs(est.diff),
  data = Nullvtlg)
```

```
## prop_TRUE
##      0.066
```

-
- Lizenz Creative Commons Attribution-ShareAlike 3.0 Unported.
 - R Version: 3.6.1
 - mosaic Version: 1.5.0

²Datensatz hier eher zu klein für Bootstrap Perzentile