Datenanalyse mit R
Workshop

Tag 2

Christopher Harms

Data Scientist

mail@christopherharms.de

Überblick
Themen

- Wiederholung & Zusammenfassung Tag 1
- Wo gibt es Hilfe und mehr Infos?
- tidyverse
- ANOVA
- ggplot2

Zusammenfassung: Tag 1

Daten laden

```
# Für Komma-separierte Daten

df <- read.csv("Datendatei.csv", encoding =
  "UTF-8")

# Für Semikolon-separierte Daten (z.B. Excel)

df <- read.csv2("Datendatei.csv", encoding =
  "UTF-8")

# SPSS-Daten lesen

library(foreign)

df <- read.spss("Datendatei.sav",
  to.data.frame = TRUE)
```

Neue Variablen berechnen

```
# Neue Variable aus bestehenden berechnen

df$Alter <- 2018 – df$Geburtsjahr

# Aus reinen Zahlenangaben (1, 2, ...) eine
  Faktor erstellen

df$Gruppe <- factor(df$Priming, labels =
  c("Cognitive", "Affective"), levels = c(1,
  2))

# Aus Textangaben einen Faktor erstellen
  (z.B. wenn der Text schon in den Daten
  steht):

df$Geschlecht <- as.factor(df$Geschlecht)
```

Zusammenfassung: Tag 1

Spalten auswählen

```
# Spalten mittels Position (Index)
  auswählen
df[,c(1, 2, 3)]
df[,1:3]
# Spalten mittels Namen auswählen
df[,c("Spalte.1", "Spalte.2")]
# Spalten mittels Position (Index)
  abwählen
df[,-c(1, 2)]
```

Zeilen auswählen (filtern)

```
# Zeilen über Zeilennummer filtern
df[c(1, 5, 6),]
# Zeilen über Bedingung filtern:
df[df$Alter > 18,]
# Zeilen über mehrere Bedingungen filtern:
df[df$Alter > 18 & df$Geschlecht ==
  "männlich",]
# Beispiel: Fehlende Werte recodieren
df[df$Outcome == -99,]$Outcome <- NA
```

Zusammenfassung: Tag 1

Deskriptive Statistiken (univariat)

```r
# Anzahl Zeilen (N)

nrow(df)

# Anzahl Zeilen mit bestimmter Bedingung:

nrow(df[df$Geschlecht == "männlich",])

# Mittelwert

mean(df$Alter)

# Standardabweichung

sd(df$Alter)
```

```r
# Median

median(df$Alter)

# Varianz

var(df$Alter)

# Wenn eine Spalte fehlende Werte (NA)
  enthält, müssen diese bei der Analyse
  entfernt werden:

mean(df$Alter, na.rm = TRUE)

mean(df[complete.cases(df),]$Alter)
```

Zusammenfassung: Tag 1

Deskriptive Statistiken (bivariat)

```r
# Einfache Pearson-Korrelation zwischen
  zwei Variablen

cor(df$Alter, df$Reaktionszeit)

# Falls es fehlende Werte gibt, so dass
  nur vollständige Beobachtungen verwendet
  werden:

cor(df$Alter, df$Reaktionszeit, use =
  "complete.obs")
```

```r
# Spearman-Rangkorrelation

cor(df$Alter, df$Reaktionszeit, method =
  "spearman")
```

Zusammenfassung: Tag 1

Signifikanztests: Korrelation

```
# Korrelation auf Signifikanztesten

cor.test(df$Alter, df$Reaktionszeit)

# Falls es fehlende Werte gibt, so dass
  nur vollständige Beobachtungen verwendet
  werden:

cor.test(df$Alter, df$Reaktionszeit, use =
  "complete.obs")

# Für Spearman-Rangkorrelation

cor.test(df$Alter, df$Reaktionszeit,
  method = "spearman")
```

Signifikanztest: t-Test

```
# Mittelwert zweier Gruppen vergleichen

t.test(df[df$Gruppe == 1,]$Outcome,
  df[df$Gruppe == 2,]$Outcome)

# Mittelwert zweier Antworten vergleichen
  (d.h. verbundene Stichprobe)

t.test(df$Outcome.t0, df$Outcome.t1,
  paired = TRUE)
```

Zusammenfassung: Tag 1

Lineare Regression / Lineares Modell

```
# Lineares Modell erstellen (einfache, multiple
  Regression – alle Prädiktoren werden
  gleichzeitig aufgenommen)

lin.regr <- lm(Outcome ~ Gruppe + Alter, data =
  df)

# Einfache Zusammenfassung

lin.regr

# Ausführlichere Zusammenfassung

summary(lin.regr)

# Zusammenfassung als ANOVA-Tabelle (Type-1-SS)

aov(lin.regr)
```

Pakete installieren und laden

```
# Einzelnes Paket aus CRAN installieren

install.packages("ggplot2")

# Mehrere Pakete aus CRAN installieren

install.packages(c("psych", "foreign",
  "ggplot2"))

# Zuvor installierte Pakete laden

library(ggplot2)
```

Zusammenfassung: Tag 1

Datensätze speichern

```
# Als CSV mit Komma-Trennung speichern

write.csv(df, file = "Dateiname.csv",
  fileEncoding = "UTF-8")

# Als CSV mit Semikolon-Trennung speichern
  (um es z.B. in Excel wieder öffnen zu
  können)

write.csv2(df, file = "Dateiname.csv",
  fileEncoding = "UTF-8")
```

Hilfe aufrufen

```
# Hilfe-Seite für eine einzelne Funktion

?t.test

# Nach einem Funktionsnamen suchen (auch
  außerhalb der aktuell geladenen Pakete)

??ggplot

# Hilfe nach Begriff durchsuchen

help.search("ANOVA")
```

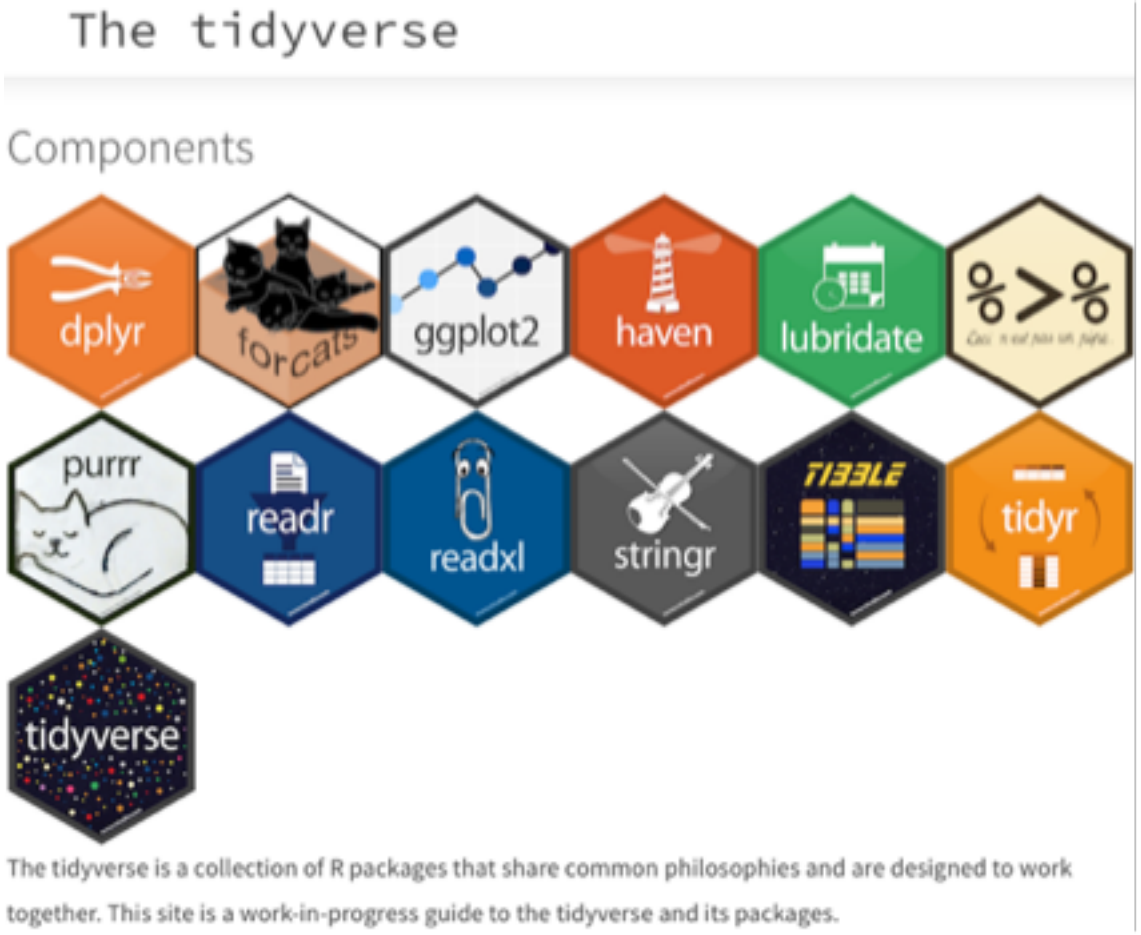
Hilfe zur Selbsthilfe … weitere Anhaltspunkte und Literatur

- **Google** & **StackOverflow** bieten Akuthilfe
- **Base R Cheat Sheet:** https://www.rstudio.com/wp-content/uploads/2016/10/r-cheat-sheet-3.pdf
  (gibt es auch für viele weitere Pakete, z.B. `dplyr` – einfach mal googlen!)
- **R 4 Data Science** (R4DS): http://r4ds.had.co.nz
- **A Psychologist's Guide to R:** https://github.com/seanchrismurphy/A-Psychologists-Guide-to-R
- **Data Camp:** https://www.datacamp.com

Datenaufbereitung und –transformation mit `tidyverse`

Datentransformation mit tidyverse

- Das `tidyverse` ist eine Sammlung verschiedener Pakete, um saubere Datentransformationen durchzuführen
- Ermöglicht vieles auf schnellere Weise als Basis R
- https://www.tidyverse.org/packages/



Analyse-Beispiel

1. Daten aus Excel-Tabelle laden
2. Daten aufbereiten (Alter berechnen, “falsche“ Angaben rausfiltern)
3. ANOVA berechnen (Package `ezANOVA`)
4. Daten transformieren für RM-ANOVA
5. (Nächster Abschnitt:) Diagramme mit `ggplot2`

Analyse-Beispiel mit tidyverse

Daten einlesen

```
# Notwendige Pakete laden

library(tidyverse)

library(readxl)

# Datensatz aus der Excel-Datei laden

df <- read_xlsx(„Primingstudie.xlsx“)

# Anzeigen, was in den Daten steckt

str(df)

head(df)
```

Daten aufbereiten

```
df <- df %>%

  # Alter berechnen und „Priming“ als
  Faktor anlegen

  mutate(Alter = 2018 – Geburtsjahr,

         Priming = factor(Priming, levels
  = c(„Cognitive“, „Affective“)) %>%

  # Zu junge und zu alte Probanden
  rausfiltern

  filter(Alter > 18 | Alter < 90)
```

Analyse-Beispiel mit tidyverse

ANOVA rechnen

```
# Paket „ez“ laden

library(ez)

# Einfache ANOVA

ezANOVA(df,

  dv = Wahrgenommene.Nähe,

  between = c(Priming, Geschlecht),

  wid = Probandennr.,

  type = 3)
```

Daten transformieren

```
df <- df %>%

  gather(key = "name", value = "value", …)
  %>%

  mutate(time = case_when(…)) %>%

  select(-name) %>%

  rename(reaction.time = value) %>%

  mutate(time = as.factor(time))

# Und anschließend ist eine RM-ANOVA für
  die Reaktionszeiten möglich
```

Daten visualisieren

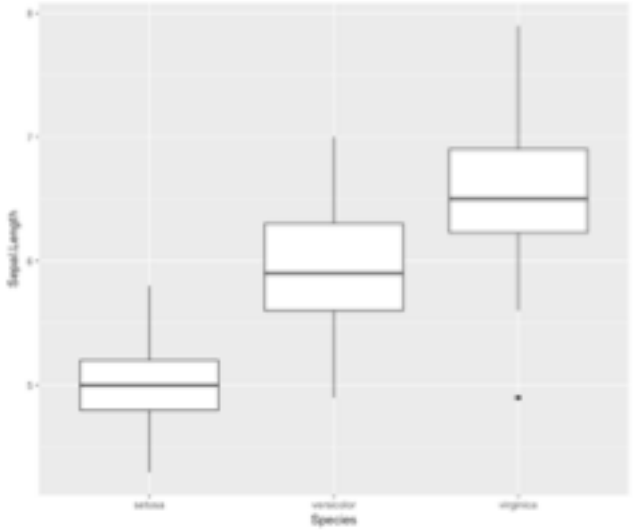
Zwei Möglichkeiten für Visualisierungen

Base R

```
> hist(df$age)
> plot(cars$speed,      dist)
> plot(iris)
```

ggplot2

```
> library(ggplot2)
> ggplot(data = iris, aes(x = Species, y =
  Sepal.Length)) + stat_boxplot()
```

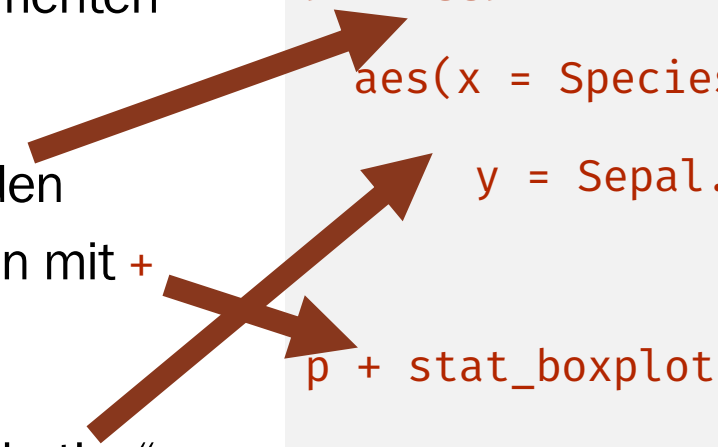


ggplot2: Grundlagen

`iris` ist ein in R verfügbarer Beispiel-Datensatz

- Plots werden aus verschiedenen Elementen „zusammengebaut“
- `ggplot()` erstellt dabei die Basis – den Hintergrund quasi – Elemente werden mit `+` einfach hinzugefügt
- Wesentlicher Baustein sind die “Aesthetics“ `aes()`

```
p <- ggplot(iris,
  aes(x = Species,
      y = Sepal.Length))


p + stat_boxplot()
```



ggplot2: Aesthetics (Auswahl)

- X-Achse: `x`
- Y-Achse: `y`
- Farbe: `color` / `colour`
- Form: `shape`
- Größe: `size`
- Breite: `width`
- Höhe: `height`
- Transparenz: `alpha`

- Welche Aesthetics verfügbar sind und tatsächlich eine Auswirkung haben, hängt von der jeweiligen Darstellung ab, die gewählt wird

ggplot2: Darstellungen (Auswahl)

- Für einige Elemente/Ebene werden die Rohdaten benutzt – für andere müssen wir vorher die Daten aggregieren (z.B. Mittelwerte)
- Punkte: `geom_point()`
- Balken: `geom_bar()`
- Linien: `geom_line()`
- Fehlerbalken: `geom_errorbar()` / `geom_errorbarh()`

```
ggplot(iris, aes(x = Petal.Width, y =
  Sepal.Length)) +

  geom_point()
```

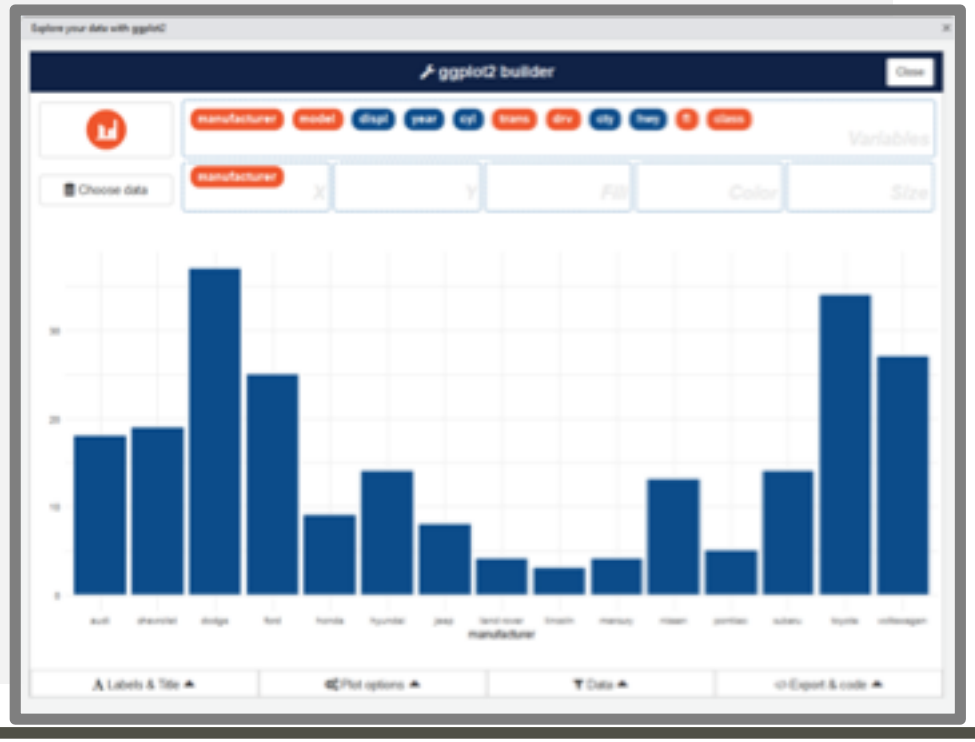
ggplot2: Darstellungen (Auswahl)

- Die Pakete tidyr und dplyr helfen uns bei der Vorbereitung der Daten

```
df.plot <- df %>%
  group_by(Priming, Geschlecht) %>%
  summarise(mean.nähe =
  mean(Wahrgenommene.Nähe),
     sd.nähe = sd(wahrgenommene.nähe),
     n.nähe = n())
ggplot(df.plot, aes(x = Priming, y =
  mean.nähe, color = Geschlecht)) +
  geom_point() +
  geom_line(aes(group = Geschlecht))
```

RStudio-AddIn: esquisse

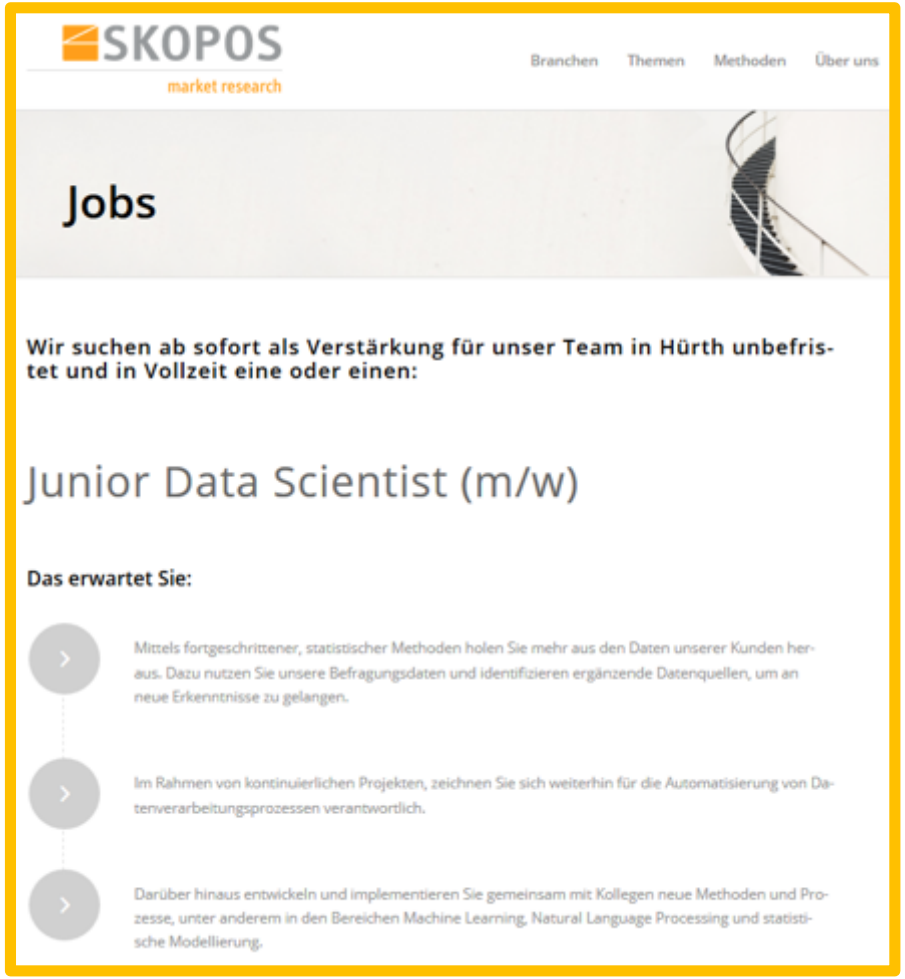
```
install.packages("devtools")

devtools::install_github("dreamRs/esquisse
  ")
```



- Funktioniert nicht immer einwandfrei, ist aber auch noch in der Entwicklung
- https://github.com/dreamRs/esquisse

Spaß an R und Datenverarbeitung gefunden?

- „Data Science“ ist der hippe neue Name für Statistik und ihre Anwendung in der Praxis
- Bei SKOPOS (Marktforschungsinstitut) bauen wir diesen Bereich gerade auf!
- **Interesse? Lust auf ein Praktikum oder Berufseinstieg?**
  christopher.harms@skopos.de
- http://www.skopos.de



Fortgeschrittenes

Fortgeschrittene Themen und Pakete

- Manuskripte direkt in R mit `RMarkdown`, `knitr` und `papaja` erstellen
- Power-Analysen mit `pwr`
- Arbeiten mit Open Data & saubere Codes
- Bayes Faktoren mit `BayesFactor` und `ReplicationBF` berechnen
- Faktorenanalysen und Strukturgleichungsmodelle mit `lavaan`
- Mehrebenenmodelle mit `lmer` und `brms`
- Interaktive Apps mit Shiny erstellen: http://shiny.rstudio.com
- Eigene Pakete erstellen: http://r-pkgs.had.co.nz