Einführung in R(Studio)

Explorative Datenanalyse mit R

The R Bootcamp @ CSS

Dezember 2019

R ist eine Programmiersprache

Eine Programmiersprache ist eine **formale Sprache** die eine Reihe Instruktionen für alle möglichen Ziele spezifiziert. Programmiersprachen bestehen aus **instruktionen für einen Computer** und werden genutzt um **Algorithmen zu implementieren**.

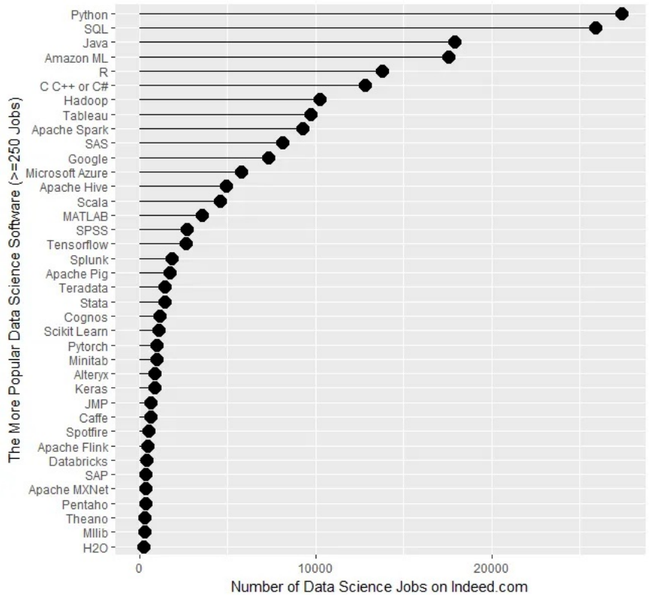
Algorithmus in Worten

1. Lade Daten
2. Extrahiere Variablen
3. Führe Analyse durch
4. Zeige Resultate

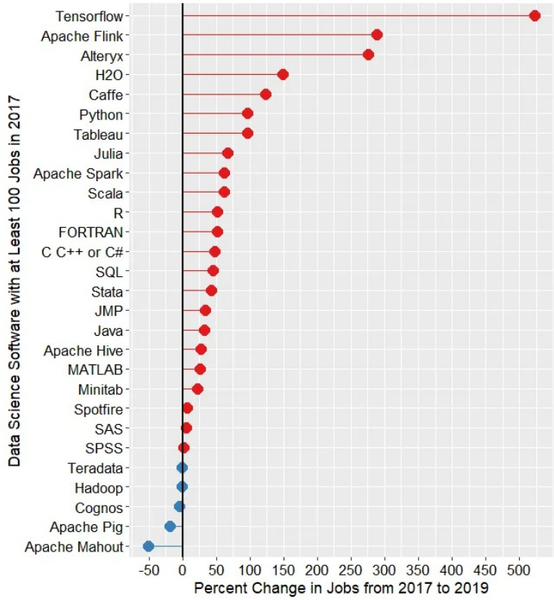
Implementation in R

```
data <- read.table(link)
variables <- data[,c('gruppe','variable')]
analysis <- lm(variable ~ gruppe, data = variables)
summary(analysis)
```

Warum R?



siehe r4stats.com



siehe r4stats.com

R wird relevant bleiben

R ist de facto die **lingua franca** für Statistik und Datenanalyse.

Pro

1. **Open-source** und umsonst.
2. **Community** (e.g., **stackoverflow**)
3. **Erweiterbarkeit** (**CRAN**)
4. **Tidyverse**
5. **RStudio**
6. **Produktivität**: **Latex**, **Markdown**, **GitHub**

Ehemals Contra

1. **Unschön**e Sprache wird überarbeitet (**Tidyverse**)
2. **Langsam**e Elemente werden ersetzt (**Rcpp**)
3. **Brücken** zu externen Tools/Sprachen (**rPython**, **tensorflow**)

Komponenten von R

R

adapted from cei.org

RStudio

adapted from rstudio.com

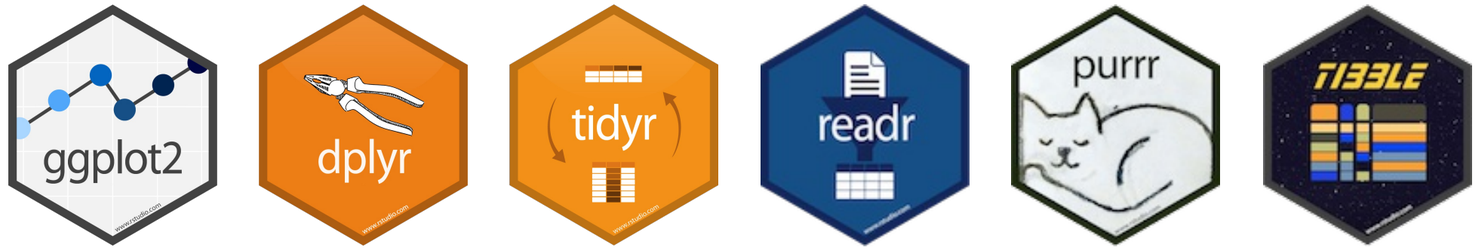
R Packages

adapted from towardsdatascience.com

Das mächtige `tidyverse`

Das `tidyverse` ist im Kern eine Sammlung hoch-performanter, nutzerfreundlicher Pakete, die speziell für eine effizientere Datenanalyse entwickelt wurden.

1. `ggplot2` für Grafiken.
2. `dplyr` für Datenverarbeitung.
3. `tidyr` für Datenverarbeitung.
4. `readr` für Daten I/O.
5. `purrr` für funktionales Programmieren.
6. `tibble` für moderne `data.frames`.



10 grundlegende R Lektionen

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

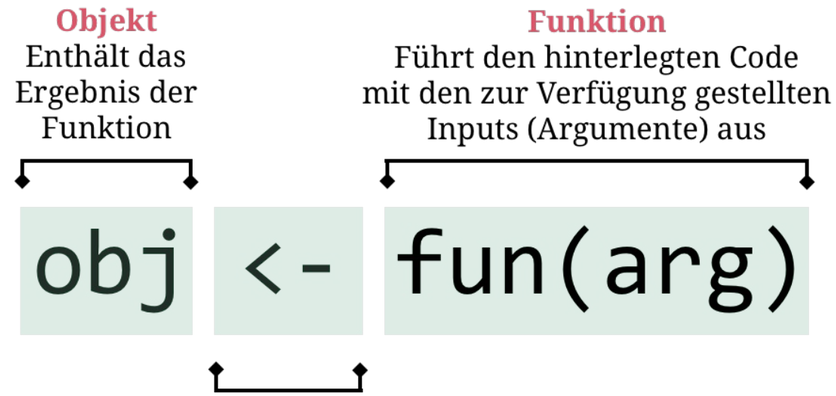
10 grundlegende R Lektionen

Syntax

1. **Der Assignment Operator <-**
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio



10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Printe das Objekt
print(eins_zwei_drei)
```

```
## [1] 1 2 3
```

```
# Printe das Objekt
eins_zwei_drei
```

```
## [1] 1 2 3
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Berechne den Mittelwert
mean(eins_zwei_drei)
```

```
## [1] 2
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Berechne den Mittelwert
ergebnis <- mean(eins_zwei_drei)

# Benutze Ergebnis
ergebnis * 10
```

```
## [1] 20
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Printe das Objekt
eins_zwei_drei
```

```
## [1] 1 2 3
```

```
# Addiere 100
eins_zwei_drei + 100
```

```
## [1] 101 102 103
```

```
# Printe das Objekt
eins_zwei_drei
```

```
## [1] 1 2 3
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Printe das Objekt
eins_zwei_drei
```

```
## [1] 1 2 3
```

```
# Ändere das Objekt
eins_zwei_drei <- eins_zwei_drei + 100

# Printe das Objekt
eins_zwei_drei
```

```
## [1] 101 102 103
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. **<- kreiert/verändert Objekte**
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Ein Objekt namens eins_zwei_drei
eins_zwei_drei <- c(1, 2, 3)

# Ergänze um weitere Zahl
und_vier <- c(eins_zwei_drei, 4)

# Printe das Objekt
und_vier
```

```
## [1] 1 2 3 4
```

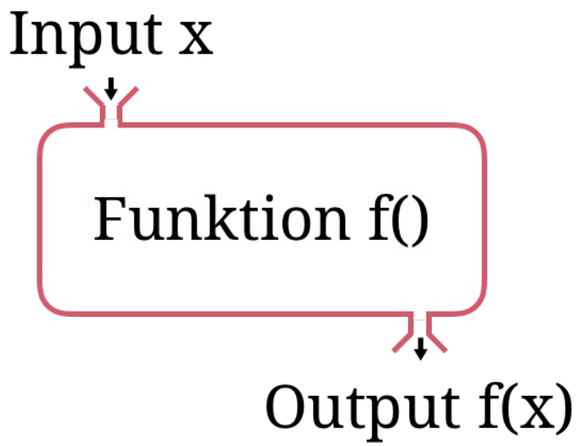
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. **Alles passiert durch Funktionen**
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio



10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. **Alles passiert durch Funktionen**
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Funktion c()
eins_zwei_drei <- c(1, 2, 3)

# Funktion `+`()
eins_zwei_drei + 100
```

```
## [1] 101 102 103
```

```
# Funktion print()
eins_zwei_drei
```

```
## [1] 1 2 3
```

```
# Funktion mean()
mean(x = eins_zwei_drei)
```

```
## [1] 2
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. **Funktionen haben (Default) Argumente**
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Kein Argument
mean()
```

```
## Error in mean.default(): argument "x" is missing, with n
```

```
# Ein (notwendiges) Argument
mean(c(1, 2, 3))
```

```
## [1] 2
```

```
# Hinzufügen eines fehlenden Werts (NA)
mean(c(1, 2, 3, NA))
```

```
## [1] NA
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. **Funktionen haben (Default) Argumente**
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Kein Argument
mean()
```

```
## Error in mean.default(): argument "x" is missing, with n
```

```
# Ein (notwendiges) Argument
mean(c(1, 2, 3))
```

```
## [1] 2
```

```
# Ändere den Default zur Entfernung des NAs
mean(c(1, 2, 3, NA), na.rm = TRUE)
```

```
## [1] 2
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. **Argumente erwarten Klassen**
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Berechne Mittelwert von c(1, 2, 3)
mean(c(1, 2, 3))
```

```
## [1] 2
```

```
# Berechne Mittelwert von c('1', '2', '3')
mean(c("1", "2", "3"))
```

```
## Warning in mean.default(c("1", "2", "3")): argument is n
```

```
## [1] NA
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. **Argumente erwarten Klassen**
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```r
# Klasse von c(1, 2, 3)
class(c(1, 2, 3))
```

```
## [1] "numeric"
```

```r
# Klasse von c('1', '2', '3')
class(c("1", "2", "3"))
```

```
## [1] "character"
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
?mean
```

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

x	An R object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for trim = 0, only.
trim	the fraction (0 to 0.5) of observations to be trimmed from each end of x before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
na.rm	a logical value indicating whether NA values should be stripped before the computation proceeds.
...	further arguments passed to or from other methods.

Value

If trim is zero (the default), the arithmetic mean of the values in x is computed, as a numeric or complex vector of length one. If x is not logical (coerced to numeric), numeric (including integer) or complex, NA_real_ is returned, with a warning.

If trim is non-zero, a symmetrically trimmed mean is computed with a fraction of trim observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

weighted.mean, mean.POSIXct, colMeans for row and column means.

Examples

```
x <- c(0:10, 50)
xm <- mean(x)
c(xm, mean(x, trim = 0.10))
```

[Package *base* version 3.6.0 Index]

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
?mean
```

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

`x`	An `R` object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
`trim`	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
`na.rm`	a logical value indicating whether NA values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Mittelwert von c(1, 2, 3)
mean(c(1, 2, 3))
```

```
## [1] 2
```

```
# Mittelwert von c(TRUE, FALSE, TRUE)
mean(c(TRUE, FALSE, TRUE))
```

```
## [1] 0.6667
```

```
# Mittelwert von c("1", "2", "3")
mean(c("1", "2", "3"))
```

```
## Warning in mean.default(c("1", "2", "3")): argument is n
```

```
## [1] NA
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
?mean
```

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

`x`	An `R` object. Currently there are methods for numeric/logical vectors and date, date-time and time interval objects. Complex vectors are allowed for `trim = 0`, only.
`trim`	the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of trim outside that range are taken as the nearest endpoint.
`na.rm`	a logical value indicating whether NA values should be stripped before the computation proceeds.
`...`	further arguments passed to or from other methods.

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Mittelwert von c(1, 2, 3, NA)
mean(c(1, 2, 3, NA), na.rm = TRUE)
```

```
## [1] 2
```

```
# Mittelwert von c(1, 2, 3, NA)
mean(c(1, 2, 3, NA), na.rm = "TRUE")
```

```
## [1] 2
```

```
# Mittelwert von c(1, 2, 3, NA)
mean(c(1, 2, 3, NA), na.rm = "ABCD")
```

```
## Error in if (na.rm) x <- x[!is.na(x)]: argument is not i
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Klasse von TRUE
class(TRUE)
```

```
## [1] "logical"
```

```
# Klasse von "TRUE"
class("TRUE")
```

```
## [1] "character"
```

```
# Klasse von "ABCD"
class("ABCD")
```

```
## [1] "character"
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. **Findet Hilfe mit ?**

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Wandle "TRUE" in logical um
as.logical("TRUE")
```

```
## [1] TRUE
```

```
# Wandle "ABCD" in logical um
as.logical("ABCD")
```

```
## [1] NA
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

Kurze Pause?

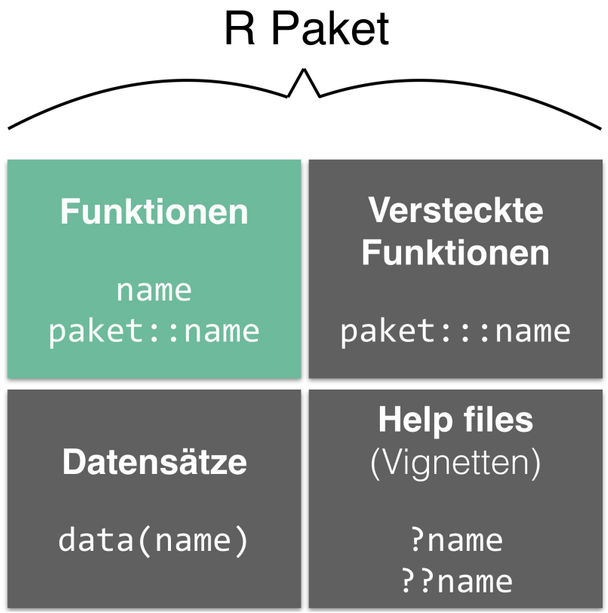
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. **Funktionen leben in Paketen**
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio



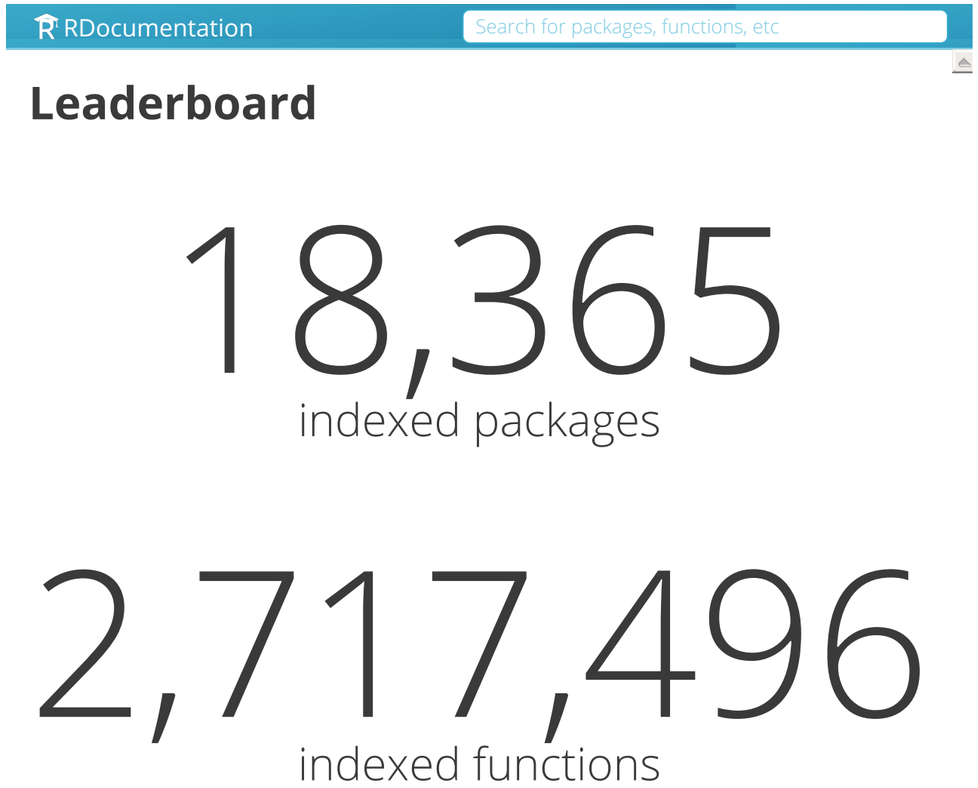
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. **Funktionen leben in Paketen**
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio



10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. **Funktionen leben in Paketen**
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

Installiere Pakete **einmal** mit `install.packages()`

```
install.packages("tidyverse")
```

Lade existierende Pakete **jedes mal** mit `library()`

```
library(tidyverse)
```

```
── Attaching packages ─────────────────────────────────────
tidyverse 1.2.1 ──
✔ ggplot2 3.1.0     ✔ purrr   0.2.5
✔ tibble  2.0.1     ✔ dplyr   0.7.6
✔ tidyr   0.8.1     ✔ stringr 1.3.1
✔ readr   1.1.1     ✔ forcats 0.3.0
── Conflicts ────────────────────────────────────────
se_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
Warning message:
package 'tibble' was built under R version 3.5.2
```

10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. **Lernt von Fehlern und Warnungen**
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

```
# Message: Interessant
basel <- type_convert(baselers)
```

```
## Parsed with column specification:
## cols(
##   sex = col_character()
## )
```

```
# Warning: Wichtig
result <- mean('NA')
```

```
## Warning in mean.default("NA"): argument is not numeric o
```

```
# Error: Beheben
lenth(x = 1)
```

```
## Error in lenth(x = 1): could not find function "lenth"
```


10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. **Lernt von Fehlern und Warnungen**
9. Benutzt R in RStudio
10. Benutzt Projekte in Rstudio

Fehler	Beschreibung
`'could not find function'`	Typo oder Paket nicht geladen
`'error in eval'`	Ein Objekt wird verwendet, dass nicht existiert.
`'cannot open()'`	Typo oder fehlende Ordner im Pfad.
`'no applicable method'`	Funktion erwartet andere Klassen.
package errors	Fehler im installieren, kompilieren, oder laden von Paketen.

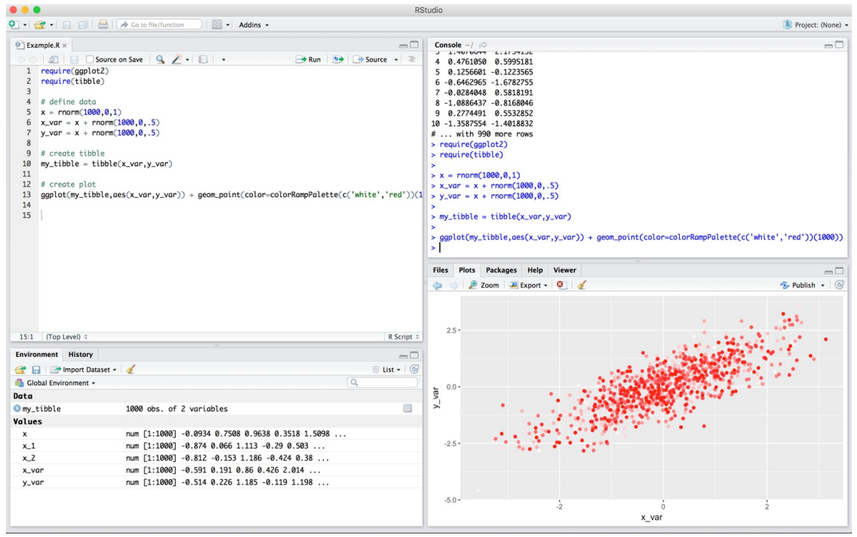
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. **Benutzt R in RStudio**
10. Benutzt Projekte in Rstudio



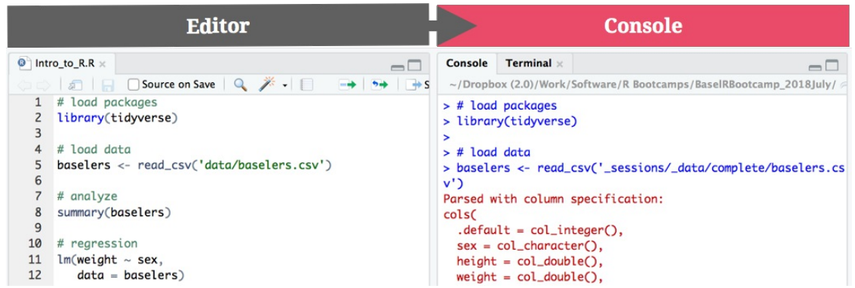
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. **Benutzt R in RStudio**
10. Benutzt Projekte in Rstudio



Shortcut für **schicke Code zur Console**:

⌘/ctrl + ⏎

Shortcut für **führe Chunk wiederholt aus**:

⌘/ctrl + ⇧ + p

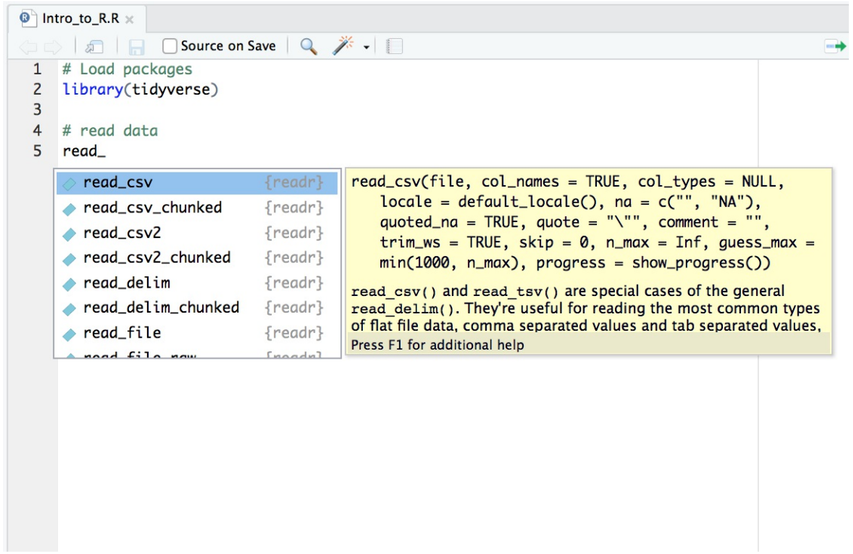
10 grundlegende R Lektionen

Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. **Benutzt R in RStudio**
10. Benutzt Projekte in Rstudio



10 grundlegende R Lektionen

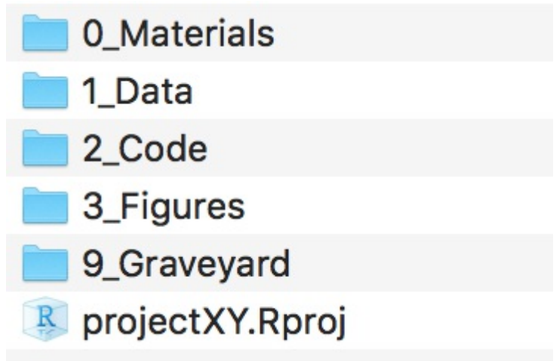
Syntax

1. Der Assignment Operator <-
2. <- kreiert/verändert Objekte
3. Alles passiert durch Funktionen
4. Funktionen haben (Default) Argumente
5. Argumente erwarten Klassen
6. Findet Hilfe mit ?

Produktivität

7. Funktionen leben in Paketen
8. Lernt von Fehlern und Warnungen
9. Benutzt R in RStudio
10. **Benutzt Projekte in Rstudio**

Projekte helfen bei...
workspace und history speichern ● projektspezifische Optionen setzen ● Dateien finden ● Versionskontrolle ● etc.



Downloads

Interactive