Computergestützte Statistische Datenanalyse mit R

Dr. Wasilios Hariskos

Die Programmiersprache R ist beliebt

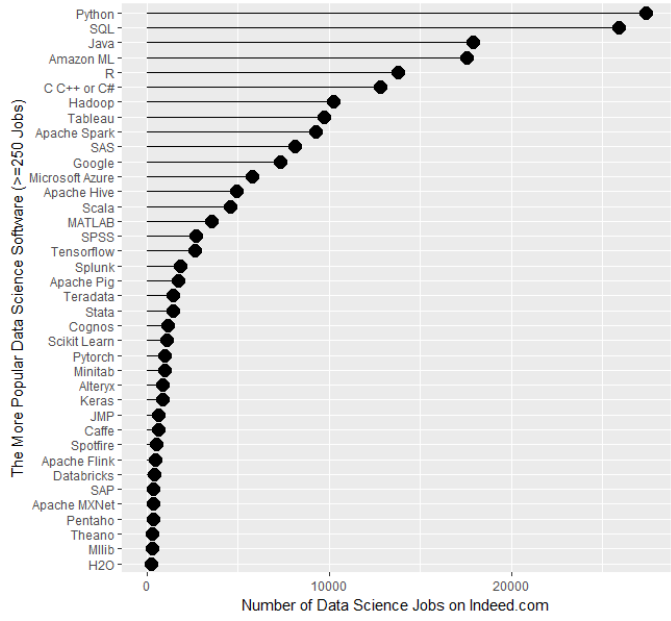


Figure 1: Quelle: http://r4stats.com/articles/popularity

Lernziele

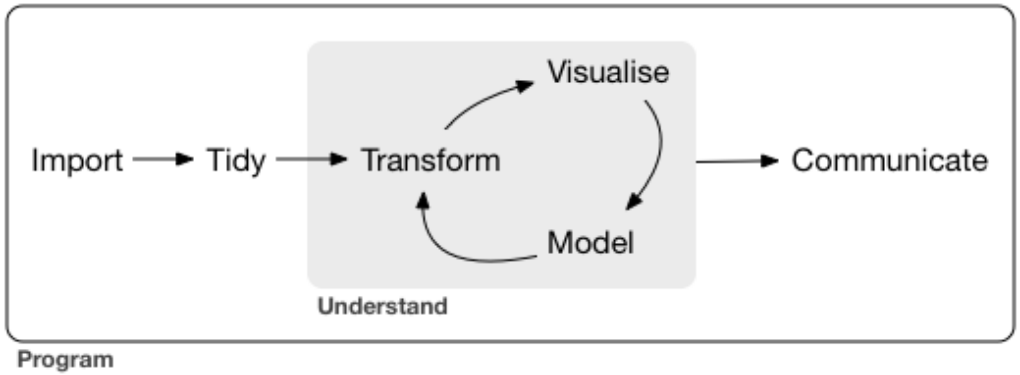


Figure 2: Quelle: https://r4ds.had.co.nz/introduction.html

- Daten *importieren* und *aufräumen* (jede Zeile ist eine Beobachtung und jede Spalte eine Variable)
- Daten *transformieren* (Beobachtungen auswählen, Variablen hinzufügen, numerische Zusammenfassungen)
- Daten *visualisieren* (um diese besser zu verstehen und interessante Fragestellungen zu finden)
- Daten *modellieren* (um Fragen wissenschatlich zu beantworten)
- Daten *kommunizieren* (in schriflichen Arbeiten oder in einer Präsentation)
- Programmieren lernen, um besser Daten computergestützt

Kernmodul: Programmieren

- Vektoren
- Matrizen
- Faktoren
- Datensätze
- Listen
- Operatoren
- Bedingte Anweisungen
- Schleifen
- Funktionen

Aufgabe: Programmieren

- Anton und Berta haben beide die Mathe und Statistik Klausuren bestanden.
- Anton hat in Mathe eine 2 bekommen und in Statistik eine 3.
- Berta hat in Mathe eine 4 bekommen und in Statistik eine 1.
- Erstellen Sie einen Datensatz mit den Variablen Name, Klausur und Note.

```r
noten_vektor <- c(2, 3, 4, 1)
print(noten_vektor)
```

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

```r
namen_vektor <- c("Anton", "Anton", "Berta", "Berta")
print(namen_vektor)
```

```
## [1] "Anton" "Anton" "Berta" "Berta"
```

```r
klausuren_faktor <- factor(x = c("Mathe", "Statistik", "Mat
print(klausuren_faktor)
```

```
## [1] Mathe     Statistik Mathe     Statistik
## Levels: Mathe Statistik
```

```r
klausur_daten <- data.frame(Name = namen_vektor,
                            Klausur = klausuren_faktor,
                            Note = noten_vektor)
print(klausur_daten)
```

```
##    Name   Klausur Note
## 1 Anton     Mathe    2
## 2 Anton Statistik    3
## 3 Berta     Mathe    4
## 4 Berta Statistik    1
```

Vertiefung: Programmieren

- Matrizen `matrix()` und Listen `list()`
- Operatoren
  - mathematische `+  -  *  /  ^`
  - vergleichend `==  !=  <  >=`
  - logische `!  &  |  xor()`
- Bedingte Anweisungen (If Statements)
  - `if (condition){Do something}`
  - `if (condition){Do smth} else {Do smth}`
  - `if (condition){Do smth} ifelse (condition) {Do smth} else {Do smth}`
- Schleifen
  - For Loop `for (variable in sequence){Do smth}`
  - While Loop `while (condition){Do smth}`
- Funktionen `function_name <- function(var){Do smth; return(new_variable)}`

Kernmodul: Datenexploration

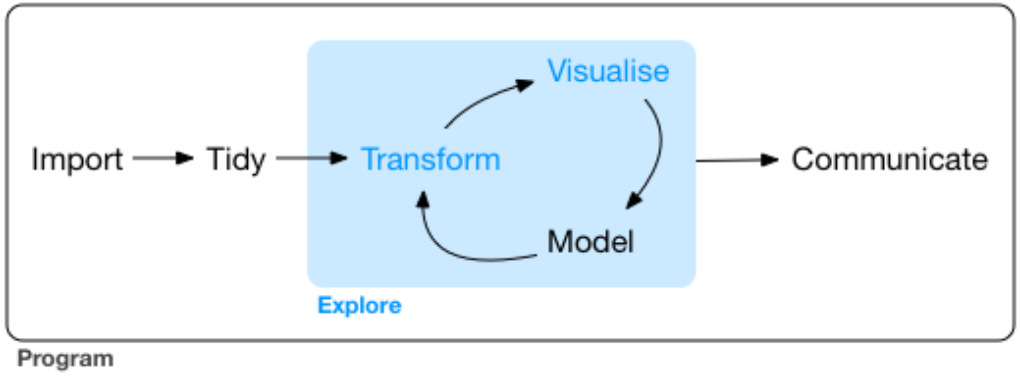


Figure 3: Quelle: https://r4ds.had.co.nz/explore-intro.html

Ein makroökonomischer Datensatz

```
print(gapminder)
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPerca
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl
##  1 Afghanistan Asia       1952    28.8  8425333       779
##  2 Afghanistan Asia       1957    30.3  9240934       821
##  3 Afghanistan Asia       1962    32.0 10267083       853
##  4 Afghanistan Asia       1967    34.0 11537966       836
##  5 Afghanistan Asia       1972    36.1 13079460       740
##  6 Afghanistan Asia       1977    38.4 14880372       786
##  7 Afghanistan Asia       1982    39.9 12881816       978
##  8 Afghanistan Asia       1987    40.8 13867957       852
##  9 Afghanistan Asia       1992    41.7 16317921       649
## 10 Afghanistan Asia       1997    41.8 22227415       635
## # ... with 1,694 more rows
```

Funktionen für die Datentransformation

- Zur Datentransformation nutzen wir das Paket `dplyr`
- Es beinhaltet folgende Funktionen (oder Verben)
  - `mutate()` fügt eine Variable als Spalte hinzu oder ersetzt eine bestehende Variable
  - `select()` wählt eine Variable aus
  - `filter()` filtert Beobachtungen (Zeilen) anhand von Kriterien
  - `summarize()` erstellt numerische Zusammenfassungen
  - `arrange()` sortiert den Datensatz anhand einer Variable
- `group_by()` erlaubt Daten nach Gruppen zu manipulieren
- Selbststudium: `vignette("dplyr")` durchlesen

Aufgabe: Datentransformation

- Berechnen Sie den Länderdurchschnitt der Lebenserwartung bei Geburt
  - gruppiert nach Kontinenten und
  - den Jahren 1952 und 2007.
- Interpretieren Sie die numerische Zusammenfassung.

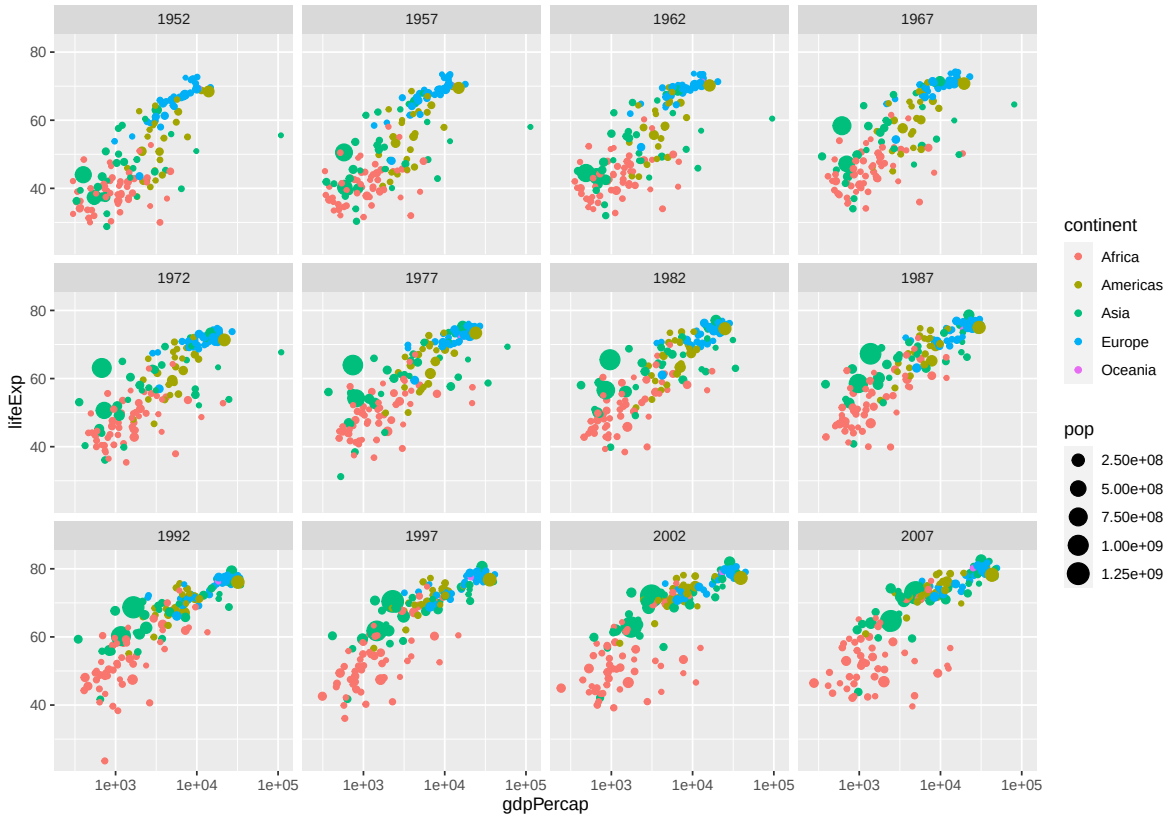
```
gapminder %>%
  filter(year %in% c(1952, 2007)) %>%
  group_by(continent, year) %>%
  summarize(avgLifeExp = mean(lifeExp))
```

```
## # A tibble: 10 x 3
## # Groups:   continent [5]
##    continent  year avgLifeExp
##    <fct>     <int>      <dbl>
##  1 Africa     1952       39.1
##  2 Africa     2007       54.8
##  3 Americas   1952       53.3
##  4 Americas   2007       73.6
```

Aufgabe: Datenvisualisierung

- Visualisieren Sie den Gapminder-Datensatz.
- Erstellen Sie für jedes Jahr ein Streudiagramm, indem Sie
  - Durschnittseinkommen auf die Dimension (Ästhetik) x-Achse abbilden,
  - Lebenserwartung auf die y-Achse abbilden,
  - Kontinent auf Farbe der Punkte abbilden und
  - Ländergröße auf Größe der Punkte.
- Interpretieren Sie die Grafik.

```
ggplot(data = gapminder) +
  geom_point(mapping = aes(x = gdpPercap,
                           y = lifeExp,
                           color = continent,
                           size = pop)) +
  facet_wrap(facets = ~year) +
  scale_x_log10()
```


1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007
lifeExp
80
60
40
1e+03
1e+04
1e+05
gdpPercap
continent
Africa
Americas
Asia
Europe
Oceania
pop
2.50e+08
5.00e+08
7.50e+08
1.00e+09
1.25e+09

Kernmodul: Datenmodellierung

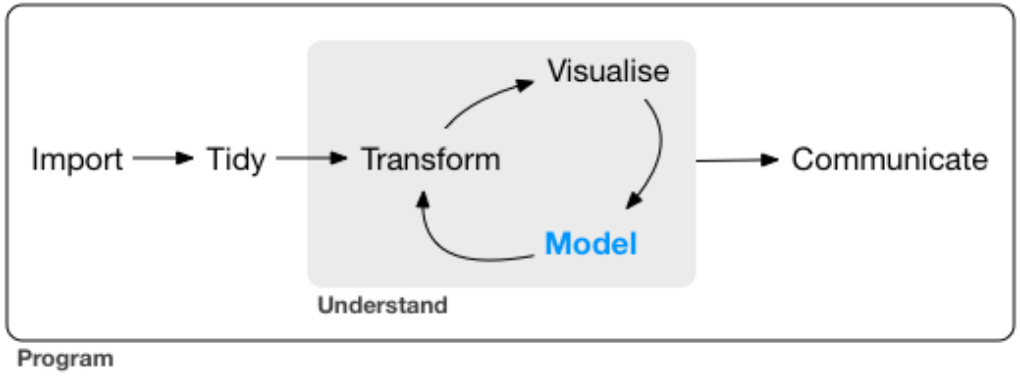


Figure 4: Quelle: https://r4ds.had.co.nz/model-intro.html

Beispiel: Datenmodellierung