Übungszettel Indexerstellung

Peter Zezula (pzezula@uni-goettingen.de)

Links

Übungszettel als PDF-Datei zum Drucken

Übungszettel mit Lösungen:

Lösungszettel als PDF-Datei zum Drucken

Der gesamte Übugszettel als .Rmd-Datei (Zum Downloaden: Rechtsklick > Speichern unter…)

Hinweise zur Bearbeitung

1. Bitte beantworten Sie die Fragen in einer .Rmd Datei. Sie können Sie über `Datei > Neue Datei > R Markdown...` eine neue R Markdown Datei erstellen. Den Text unter dem *Setup Chunk* (ab Zeile 11) können Sie löschen. Unter diesem Link können Sie auch unsere Vorlage-Datei herunterladen (Rechtsklick > Speichern unter…).
2. Informationen, die Sie für die Bearbeitung benötigen, finden Sie auf der Website der Veranstaltung
3. Zögern Sie nicht, im Internet nach Lösungen zu suchen. Das effektive Suchen nach Lösungen für R-Probleme im Internet ist tatsächlich eine sehr nützliche Fähigkeit, auch Profis arbeiten auf diese Weise. Die beste Anlaufstelle dafür ist der R-Bereich der Programmiererplattform Stackoverflow
4. Auf der Website von R Studio finden Sie sehr hilfreiche Übersichtszettel zu vielen verschiedenen R-bezogenen Themen. Ein guter Anfang ist der Base R Cheat Sheet

Erste Schritte

Setzen Sie das gewünschte Arbeitsverzeichnis, wir empfehlen auf dem Teaching-Server ein (Unter-)Verzeichnis in Ihrem persönlichen Bereich, z. B. `P:/mv`. Achten Sie dort auch darauf, dass das Verzeichnis: `P:/R/library` bei Ihnen existiert.

Erzeugen Sie im Anschluss ein Datenobjekt `dd` aus dem Datensatz `exercises` aus der `library(ALA)`. Sie können den Datensatz alternativ auch direkt laden mit der Syntax: `dd <- readRDS(gzcon(url("http://md.psych.bio.uni-`

Erläuterungen zu dem Datensatz finden Sie unter https://rdrr.io/rforge/ALA/man/exercise.html. Machen Sie sich die Struktur des Datensatzes klar.

Sollten Sie die `library(ALA)` nicht installiert haben und installieren wollen, können Sie das über den Befehl `install.packages("ALA", repos="http://R-Forge.R-project.org")` erledigen. Die Website zum Package findet sich unter https://rdrr.io/rforge/ALA/man/ALA-package.html

Laden Sie das `tidyverse`-Paket (`library(tidyverse)`).

Visualisierung

Im folgenden Code-Chunk wird eine Visualisierung der Daten erstellt, die für die folgenden Aufgaben hilfreich sein mag. Lassen Sie den Chunk laufen und verschaffen Sie sich einen Eindruck der Verhältnisse.

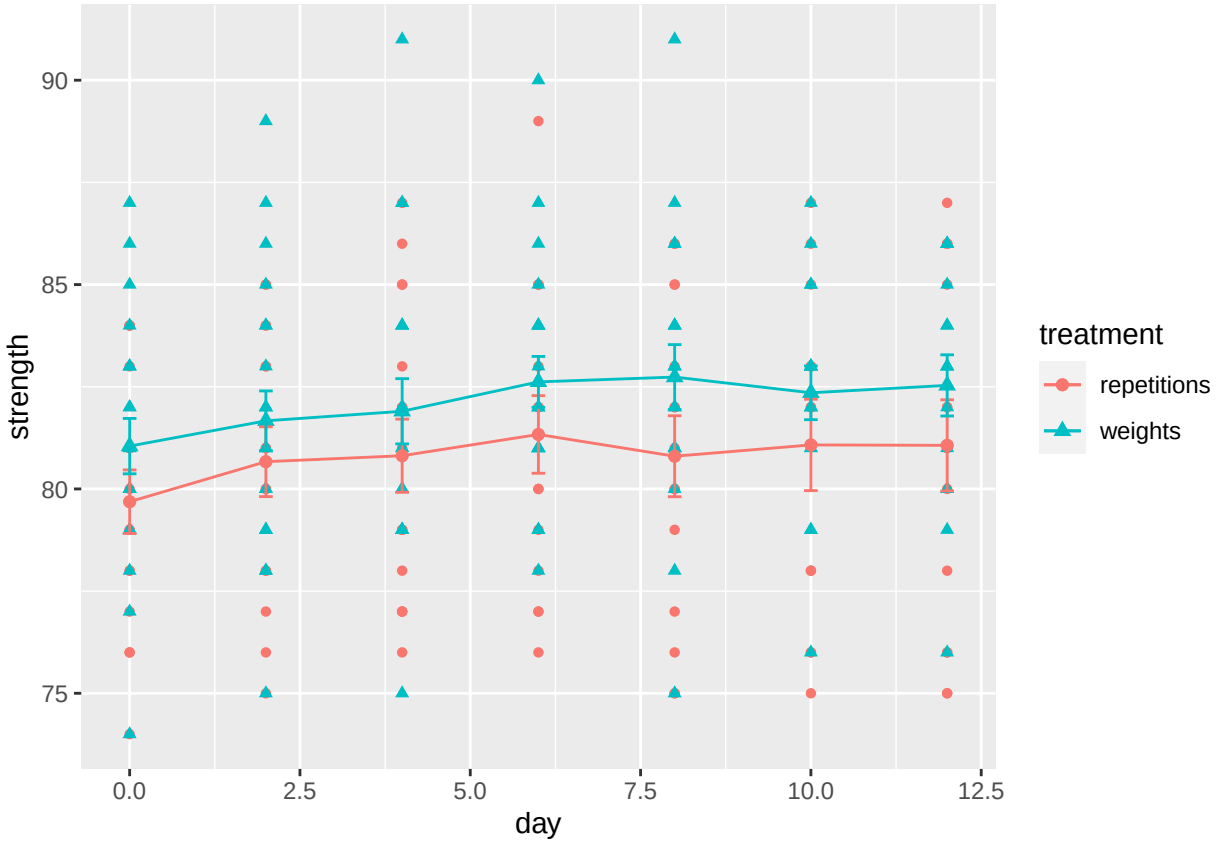
```
# if ALA is installed:
# dd <- ALA::exercise
# we can load exercise data also via:
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))

# we load tidyverse
library(tidyverse)

# we do a visualization to better understand the data
dd %>% ggplot(aes(x=day, y=strength, color=treatment, shape=treatment, group=treatment)) +
   geom_point() +
   # geom_line() +
   stat_summary(fun = mean, geom = "point", size=2) +
   stat_summary(fun = mean, geom = "line") +
   stat_summary(fun.data = mean_se, geom = "errorbar", width=0.2)
```

```
## Warning: Removed 20 rows containing non-finite values (stat_summary).
## Removed 20 rows containing non-finite values (stat_summary).
## Removed 20 rows containing non-finite values (stat_summary).

## Warning: Removed 20 rows containing missing values (geom_point).
```



Aufgabe 1: Trainingsmotivation

Überlegen Sie sich einen Index, mit dem Sie versuchen, die individuelle Trainingsmotivation der Personen zu charakterisieren, die an der Studie teilgenommen haben.

Die Trainingsmotivation soll durch eine Zahl ausgedrückt werden, die zwischen 0 (keinerlei Motivation) und 100 (volle Motivation) schwankt.

Erklären Sie die Idee zu Ihrer Lösung. Erklären Sie auch eventuelle Stärken und Schwächen Ihrer Idee.

Führen Sie die Berechnung des von Ihnen vorgeschlagenen Index durch. Speichern Sie den Index in dem Datenobjekt `dd` unter dem Namen `t_motivation`.

Die Idee ist, die Anzahl der tatsächlich absolvierten Trainingseinheiten ins Verhältnis zu setzen mit der Anzahl der maximal möglichen. Der Quotient soll dann umgerechnet werden, um die geforderte Skalierung zu haben.

Ein erster Check zeigt, dass nur wenige `strength`-Werte für Trainingseinheiten nicht vorliegen, es gibt lediglich 1, max. 2 Missings pro `id`. Die Daten lassen unklar, ob NAs durch mangelnde Motivation, also durch Nicht-Teilnahme zustandekommen oder durch andere Einflüsse, wie z. B. fehlerhafte Messung von `strength` bzw. technisches Versagen bei der Messung. 0 Missings bei einer `id` würde interpretiert als 100% Motivation, 7 Missings als alle Einheiten sind nicht absolviert bei einer `id`. Dann allerdings ist die Frage, warum eine `id` überhaupt im Datensatz auftauchen sollte.

Zu beachten ist, dass die Daten im Long-Format vorliegen.

```
library(tidyverse)
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))
# we can get the NAs per id by
table(is.na(dd$strength), dd$id)
```

```
##
##         1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
##   FALSE 6 7 7 7 6 7 7 7 5  7  7  7  7  6  6  6  7  7  5  7  7  7  6  5  7  5  6  5  6  7  7  7  5  7
##   TRUE  1 0 0 0 1 0 0 0 2  0  0  0  0  1  1  1  0  0  2  0  0  0  1  2  0  2  1  2  1  0  0  0  2  0
##
##         36 37
##   FALSE  7  7
##   TRUE   0  0
```

```
# we see, that there isn't too much variation ;-)
# base R {}todo

# tidyverse
# n is the number of training days per id
# n_miss is the number of missing strenght values per id
dd %>% dplyr::group_by(id) %>% dplyr::summarize(n_miss = sum(is.na(strength)), n = n()) -> dd.t
# we look at the variation
# we could look at dd.t by view(dd.t)
# vector of missings
dd.t$n_miss
```

```
##  [1] 1 0 0 0 1 0 0 0 2 0 0 0 0 1 1 1 0 0 2 0 0 0 1 2 0 2 1 2 1 0 0 0 2 0 0 0 0
```

```r
# frequencies
table(dd.t$n_miss)
```

```
##
##  0  1  2
## 23  8  6
```

```r
# we calculate the relative frequency
dd.t %>% dplyr::mutate(rel_miss = n_miss/n)
```

```
## # A tibble: 37 x 4
##    id    n_miss     n rel_miss
##    <fct>  <int> <int>    <dbl>
##  1 1          1     7    0.143
##  2 2          0     7    0
##  3 3          0     7    0
##  4 4          0     7    0
##  5 5          1     7    0.143
##  6 6          0     7    0
##  7 7          0     7    0
##  8 8          0     7    0
##  9 9          2     7    0.286
## 10 10         0     7    0
## # ... with 27 more rows
```

```r
# we do the calibration
dd.t %>% dplyr::mutate(rel_miss = n_miss/n, t_motivation = 100 - (rel_miss * 100)) -> dd.t
```

Aufgabe 2: Trainingserfolg

Überlegen Sie sich einen Index, mit dem Sie versuchen, den individuellen Trainingserfolg der Personen zu charakterisieren, die an der Studie teilgenommen haben.

Die Einheit, in der der Trainingserfolg ausgedrückt wird, ist nicht festgelegt, soll aber die Personen vergleichbar machen.

Erklären Sie die Idee zu Ihrer Lösung. Erklären Sie auch eventuelle Stärken und Schwächen Ihrer Idee.

Führen Sie die Berechnung des von Ihnen vorgeschlagenen Index durch. Speichern Sie den Index in dem Datenobjekt `dd` unter dem Namen `t_result`.

Idee: Von der maximalen `strength` wird die minimale `strength` abgezogen. Der Unterschied ist der Trainingserfolg

Schwäche ist, dass die Entwicklung vernachlässigt wird. Auch eventuelle Ausreißer der Messung an den Endpunkten werden nicht berücksichtigt. Großer Nachteil ist, dass die Reihenfolge vernachlässigt wird. D. h., der Index funktioniert nur, wenn es über die Trainingstage hinweg wirklich einen Zuwachs an `strength` gegeben hat.

```r
library(tidyverse)
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))

# base R
```

```r
# we loop over the subjecs
# we select all strength values of an id by conditional slicing
for (subj in unique(dd$id)){
  dd$t_result[dd$id == subj] <- max(dd$strength[dd$id == subj], na.rm = T) - min(dd$strength[dd$id == su
}

# dplyr
# we use group_by to repeat by id
dd  %>%
  dplyr::group_by(id) %>%
  dplyr::mutate(t_result = max(strength, na.rm = T) - min(strength, na.rm=T)) -> dd.t

# tidyverse with conversion long to wide, by this we can check the results
dd %>% dplyr::group_by(id) %>%
  dplyr::filter(!is.na(strength)) %>%
  dplyr::mutate(t_result = max(strength, na.rm=T) - min(strength, na.rm=T)) %>%
  tidyr::pivot_wider(names_from = day, values_from = strength) -> dd.t
```

Aufgabe 3: Kraft-Zuwachsgeschwindigkeit

Überlegen Sie sich einen Index, der für jede Person `dd$id` ausdrückt, wie schnell die Kraft aufgebaut wird. Der Index soll die individuelle Zuwachsgeschwindigkeit der Kraft `dd$strength` pro Tag ausdrücken. Er soll positiv sein, wenn die Kraft zunimmt und negativ, wenn die Kraft abnimmt im Laufe des Trainings. Die Skalierung bleibt Ihnen überlassen.

Erklären Sie die Idee zu Ihrer Lösung. Erklären Sie auch eventuelle Stärken und Schwächen Ihrer Idee.

Führen Sie die Berechnung des von Ihnen vorgeschlagenen Index durch. Speichern Sie den Index in dem Datenobjekt `dd` unter dem Namen `t_growth`.

Ideen

1. `dd$t_result` geteilt durch die Anzahl der Trainingstage Vorteil ist, dass der maximale Unterschied erklärt wird Problematisch ist, dass die Entwicklung vernachlässigt wird, z. B. bei U-förmigem Verlauf und dass Ausreißer voll durchschlagen. Sehr problematisch ist, dass die Richtung verloren geht.
2. Steigung einer Regression, die die `dd$strength` vorhersagt aus `dd$day`. Vorteil: Entwicklung wird gemittelt Nachteil: nur lineare Entwicklung wird berücksichtigt

```r
library(tidyverse)
# base R
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))
# we repeat the solution above to have dd$t_result
for (subj in unique(dd$id)){
  dd$t_result[dd$id == subj] <- max(dd$strength[dd$id == subj], na.rm = T) - min(dd$strength[dd$id == su
}
# we calculate day min and max and filter out all days without strength value
dd <- dd[!is.na(dd$strength),]
for (subj in unique(dd$id)){
  dd$d_max[dd$id == subj] <- max(dd$day[dd$id == subj], na.rm = T)
  dd$d_min[dd$id == subj] <- min(dd$day[dd$id == subj], na.rm = T)
  dd$t_growth[dd$id == subj] <- dd$t_result[dd$id == subj] / (dd$d_max[dd$id == subj] - dd$d_min[dd$id =
}
```

```r
# dplyr
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))
# idea 1, we rebuild the t_result and then divide it
dd  %>% dplyr::group_by(id) %>%
  dplyr::mutate(t_result = max(strength, na.rm = T) - min(strength, na.rm=T)) %>%
  dplyr::filter(!is.na(strength)) %>%
  dplyr::mutate(d_max = max(day, na.rm=T)) %>%
  dplyr::mutate(d_min = min(day, na.rm=T)) %>%
  dplyr::mutate(t_growth1 = t_result / (d_max - d_min)) -> dd.t

# idea 2: we appy a linear regression within every subject and look at the slope
dd  %>% dplyr::group_by(id) %>%
  dplyr::mutate(t_growth2 = lm(strength ~ day)$coefficients["day"]) -> dd.t

# we can do both and compare the results in wide format
dd  %>%
  dplyr::arrange(id, day) %>%
  dplyr::group_by(id) %>%
  dplyr::mutate(t_result = max(strength, na.rm = T) - min(strength, na.rm=T)) %>%
  dplyr::filter(!is.na(strength)) %>%
  dplyr::mutate(d_max = max(day, na.rm=T)) %>%
  dplyr::mutate(d_min = min(day, na.rm=T)) %>%
  dplyr::mutate(t_growth1 = t_result / (d_max - d_min)) %>%
  dplyr::mutate(t_growth2 = lm(strength ~ day)$coefficients["day"]) %>%
  tidyr::pivot_wider(names_from = day, values_from = strength) -> dd.t
# we look at it with reordered columns
dd.t %>% dplyr::relocate(c("0", "2", "4", "6", "8", "10", "12"))
```

```
## # A tibble: 37 x 14
## # Groups:   id [37]
##      `0`   `2`   `4`   `6`   `8`  `10`  `12` id    treatment   t_result d_max d_min t_growth1 t_grow
##    <int> <int> <int> <int> <int> <int> <int> <fct> <fct>          <int> <dbl> <dbl>     <dbl>     <dl
##  1    79    NA    79    80    80    78    80 1     repetitions        2    12     0    0.167     0.0
##  2    83    83    85    85    86    87    87 2     repetitions        4    12     0    0.333     0.3
##  3    81    83    82    82    83    83    82 3     repetitions        2    12     0    0.167     0.0
##  4    81    81    81    82    82    83    81 4     repetitions        2    12     0    0.167     0.0
##  5    80    81    82    82    82    NA    86 5     repetitions        6    12     0    0.5       0.4
##  6    76    76    76    76    76    76    75 6     repetitions        1    12     0    0.0833   -0.0
##  7    81    84    83    83    85    85    85 7     repetitions        4    12     0    0.333     0.2
##  8    77    78    79    79    81    82    81 8     repetitions        5    12     0    0.417     0.3
##  9    84    85    87    89    NA    NA    86 9     repetitions        5    12     0    0.417     0.1
## 10    74    75    78    78    79    78    78 10    repetitions        5    12     0    0.417     0.3
## # ... with 27 more rows
```

Aufgabe 4: Kraft-Zuwachsgeschwindigkeit der Trainingsart

Obige Kraft-Zuwachsgeschwindigkeit soll auch für die beiden Trainingsarten aus der Spalte `dd$treatment` als Gruppenwert ermittelt werden. Er soll die gleiche Skalierung haben, wie der individuelle Wert

Erklären Sie die Idee zu Ihrer Lösung. Erklären Sie auch eventuelle Stärken und Schwächen Ihrer Idee.

Führen Sie die Berechnung des von Ihnen vorgeschlagenen Index durch. Speichern Sie für jede Person `dd$id` den Index, zu deren Gruppe `dd$treatment` sie gehört, unter dem Namen `t_growth_treatment` im Datenobjekt `dd`.

Idee: Hier wird nur die Idee 2 aus, die Idee 1 beim individuellen Index hat zu viele Nachteile. Steigung einer Regression, die die `dd$strength` vorhersagt aus `dd$day`, allerdings für jede Teilgruppe aus `dd$treatment` spezifisch. Vorteil: Entwicklung wird gemittelt Nachteil: nur lineare Entwicklung wird berücksichtigt

```
# we keep on with the idea to appy a linear regression within every subgroup and look at the slope
library(tidyverse)
dd <- readRDS(gzcon(url("http://md.psych.bio.uni-goettingen.de/mv/data/div/exercise.rds")))

# base R
# subgroup weights
dd.t <- dd[dd$treatment == "weights",]
m_weights <- lm(strength ~ day, data = dd.t)
m_weights$coefficients["day"]
```

```
##       day
## 0.1260087
```

```
# subgroup repetitions
dd.t <- dd[dd$treatment == "repetitions",]
m_repetitions <- lm(strength ~ day, data = dd.t)
m_repetitions$coefficients["day"]
```

```
##        day
## 0.09029065
```

```
# dplyr

dd  %>% dplyr::group_by(treatment) %>%
  dplyr::mutate(t_growth_treatment = lm(strength ~ day)$coefficients["day"]) -> dd.t

dd.t %>% tidyr::pivot_wider(names_from = day, values_from = strength) -> dd.t
```

Bemerkung

Achtung: Wir verwenden die lineare Regression hier **nicht** als statistisches Modell, sondern nur als Werkzeug, um einen Kennwert, die Steigung, zu ermitteln. Statistisch läge im vorliegenden Datensatz ein Messwiederholungsdesign vor, was ein anderes statistisches Vorgehen notwendig machen würde.

Version: 27 April, 2022 15:10