Worked Out Example

(Un-)Standardisierte Regression in R

Datengrundlage

Wir verwenden für dieses Beispiel einen synthetischen Datensatz, der die Abhängige Variable `Pro-Environmental Behaviour` und die unabhängige `Climate Anxiety` enthält. Der Datensatz besteht aus 232 Beobachtungen.

Import der Daten

```
library(readr) # für den data import
library(dplyr) # für das datawrangling

data <- readr::read_csv("https://ogy.de/t101")
```

Deskriptive Statistik

Mit folgendem Code können wir einfache Kennzahl der univariaten Deskriptivstatistik berechnen.

```
# Mittelwerte
mean(data$CA)                                                    ①
mean(data$PEB)

# Standardabweichungen
sd(data$CA)
sd(data$PEB)
```

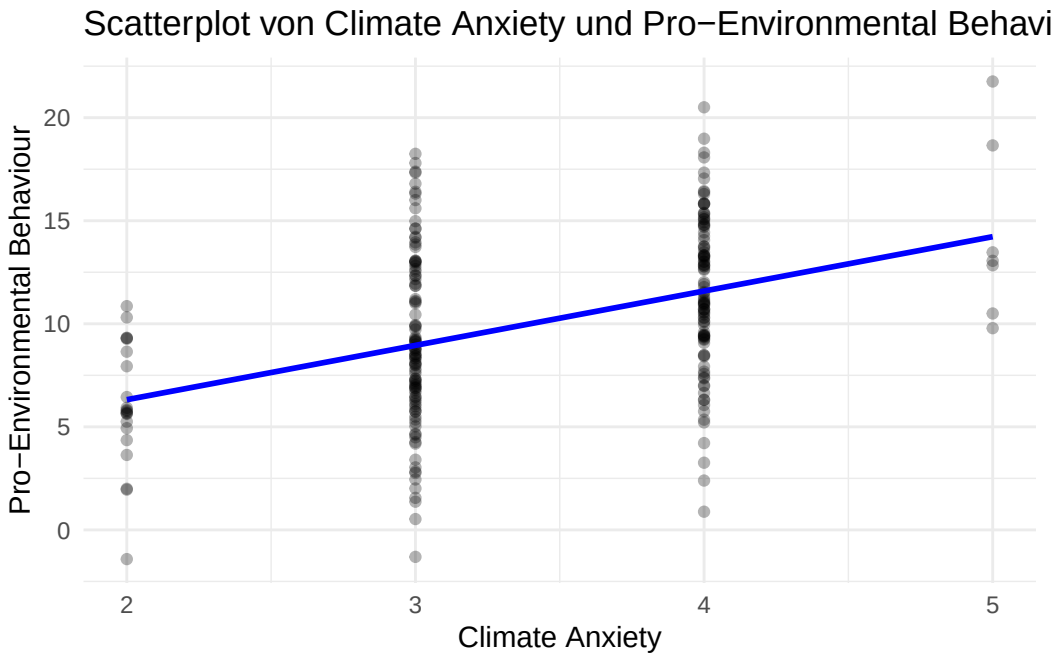
① Das Dollarzeichen `$` wird verwendet, um auf eine bestimmte Spalte in einem Dataframe zuzugreifen. In diesem Fall greift `data$CA` auf die Spalte `CA` im Dataframe `data` zu.

```
[1] 3.400862
[1] 10.01005
[1] 0.6954392
[1] 4.303129
```

Visualisierung der Daten

```
library(ggplot2) # für die visualisierung
ggplot(data,                                                                    ①
       aes(x = CA, y = PEB)) +                                                  ②
  geom_point(alpha = .3) +                                                      ③
  geom_smooth(method = "lm", se = FALSE, color = "blue") +                      ④
  labs(                                                                         ⑤
    title = "Scatterplot von Climate Anxiety und Pro-Environmental Behaviour",
    x = "Climate Anxiety",
    y = "Pro-Environmental Behaviour"
  ) +
  theme_minimal()                                                               ⑥
```

① Plotten mit der Funktion `ggplot()`, wobei das erste Argument **immer** der Datensatz ist, der geplottet werden soll.
② `aes(x = CA, y = PEB)` definiert die ästhetischen Zuordnungen für das Diagramm, wobei `CA` auf der x-Achse und `PEB` auf der y-Achse dargestellt wird.
③ `geom_point(alpha = .3)` fügt dem Diagramm Punkte hinzu, wobei `alpha = .3` die Transparenz der Punkte auf 30 % setzt.
④ `geom_smooth(method = "lm", se = FALSE, color = "#267326")` fügt eine lineare Regressionslinie hinzu, wobei `se = FALSE` bedeutet, dass kein Konfidenzintervall angezeigt wird.
⑤ `labs()` wird verwendet, um Titel und Achsenbeschriftungen hinzuzufügen.
⑥ `theme_minimal()` wendet ein minimalistisches Design auf das Diagramm an.



Durchführung der Regression

In der `lm()`-Funktion wird das lineare Regressionsmodell spezifiziert. Das erste Argument ist die Formel, die die abhängige Variable (`PEB`) links der Tilde `~` und die unabhängige Variable (`CA`) rechts der Tilde angibt. Das zweite Argument ist der Datensatz, der die Variablen enthält.

```
# lineares Regressionsmodell
lm(PEB ~ CA, data = data)
```

```
Call:
lm(formula = PEB ~ CA, data = data)

Coefficients:
(Intercept)           CA
      1.044        2.636
```

Der Outout kann wie folgt interpretiert werden:

- Der Achsenabschnitt (Intercept) beträgt etwa 1.044. Dies bedeutet, dass wenn `CA` gleich 0 ist, der vorhergesagte Wert von `PEB` etwa 1.044 beträgt.
- Der Koeffizient für `CA` beträgt etwa 2.636. Dies bedeutet, dass für jede Einheitserhöhung in `CA`, `PEB` im Durchschnitt um etwa 2.636 Einheiten zunimmt.

Durchführung der Regression mit standardisierten Variablen

Das Problem bei dieser Regression ist, dass die Einheiten der Variablen recht beleibig sind: `CA` z.B. basiert auf einer Likertskala. Angenommen andere Forschende verwenden andere Anker für die Skala, dann sind die Regressionskoeffizienten nicht mehr vergleichbar. Um dieses Problem zu lösen, können wir die Variablen standardisieren (Mittelwert = 0, Standardabweichung = 1). Dies geschieht mit der `scale()`-Funktion. Anschließend führen wir die Regression mit den standardisierten Variablen durch.

```
# Variablen standardisieren
data_std <-                                                    ①
  data %>%                                                     ②
  mutate(                                                      ③
    CA_std = scale(CA),                                        ④
    PEB_std = scale(PEB)                                       ⑤
  )
# lineares Regressionsmodell mit standardisierten Variablen
lm(PEB_std ~ CA_std, data = data_std)                          ⑥
```

① Wir erstellen ein neues Dataframe `data_std`, das die standardisierten Variablen enthält.
② Wir verwenden den Pipe-Operator `%>%`, um den Dataframe `data` an die nächste Funktion weiterzugeben (*»Nimm **data** und mache dann …«*).
③ Wir verwenden die `mutate()`-Funktion, um **neue Variablen** zu erstellen.
④ `CA_std = scale(CA)` erstellt die standardisierte Version der Variable `CA`. Typisch ist hier in dieser Syntax, dass die neue Variable links vom Gleichheitszeichen steht und die Transformationsvorschrift rechts davon.
⑤ `PEB_std = scale(PEB)` erstellt die standardisierte Version der Variable `PEB`.
⑥ Wir führen die lineare Regression mit den standardisierten Variablen durch.

```
Call:
lm(formula = PEB_std ~ CA_std, data = data_std)

Coefficients:
(Intercept)       CA_std
 -9.845e-16    4.261e-01
```

Der Output kann wie folgt interpretiert werden:

- Der Achsenabschnitt (Intercept) beträgt etwa 0. Dies ist zu erwarten, da standardisierte Variablen einen Mittelwert von 0 haben.
- Der Koeffizient für `CA_std` beträgt etwa 0.43. Dies bedeutet, dass für jede Standardabweichungserhöhung in `CA`, `PEB` im Durchschnitt um etwa 0.43 Standardabweichungen zunimmt.