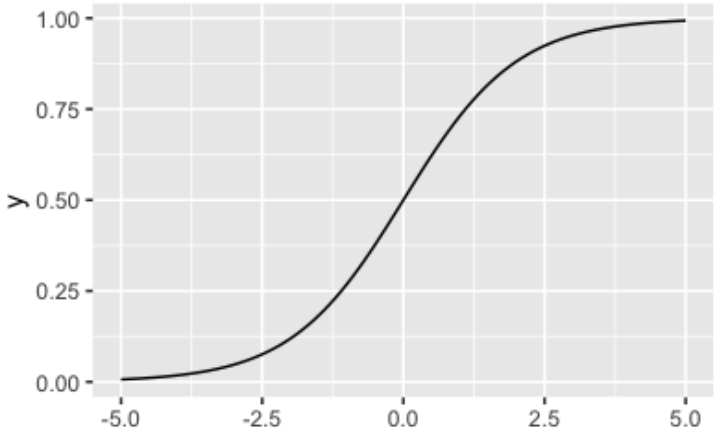
Lösungen zu den Aufgaben

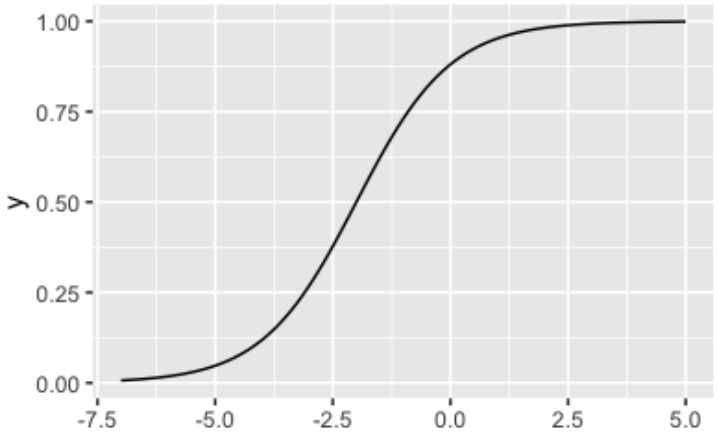
1. **Aufgabe**

   Geben Sie die Funktion für jedes Diagramm an!

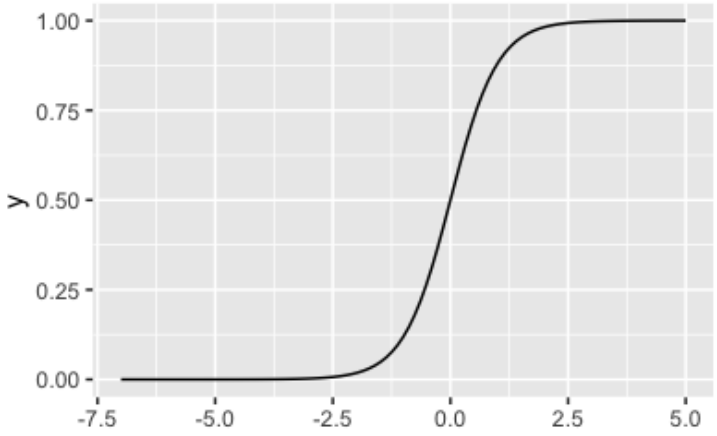
   a. Diagramm A



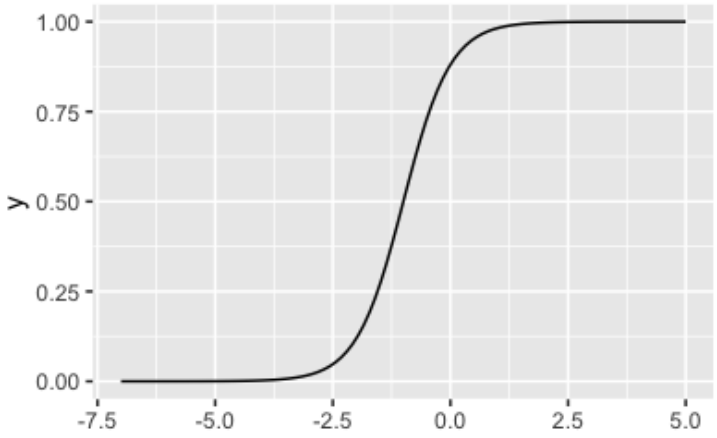
   b. Diagramm B



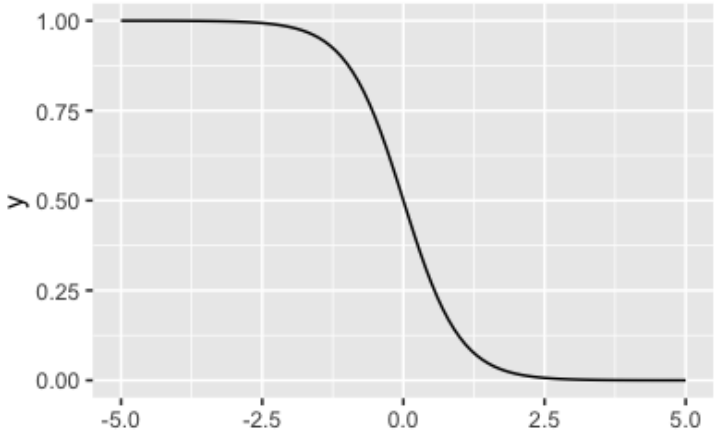
   c. Diagramm C



   d. Diagramm 4



e. Diagramm 5



Lösung

a. Diagramm A:

```
## (exp(x)/(1 + exp(x)))
```

`exp(x)` bezeichent die e-Funktion: e^x, mit $e \approx 2.7178$, der Eulerschen Zahl.

Oder anders gesagt:

$$Pr(y = 1) = \mathscr{L}(x)$$

wobei $\mathscr{L}$ die *logistische Funktion* bezeichnet (im Folgenden mit `logist` abgekürzt).

Oder:

$$Pr(y = 1) = \mathscr{L}^{-1}(x)$$

b. Diagramm B:

```
## logist(2 + x)
```

c. Diagramm C:

```
## logist(2 * x)
```

d. Diagramm D:

```
## logist(2 + 2 * x)
```

e. Diagramm E:

```
## logist(-2 * x)
```

2. **Aufgabe**

Rechnen Sie von Logits in Inv-Logits um!

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.

a. 10
b. 4
c. -5

Lösung

Die Umrechnungsformel lautet:

$$p = \frac{e^x}{1 + e^x}$$

Beispiel für $x = 1$:

```
e <- 2.7178  # Eulersche Zahl, gerundet
(e^1) / ((1+e^1))
```

```
## [1] 0.73
```

Oder man nutzt die “fertige” Funktion zum Umrechnen:

```
invlogit <- plogis
```

```
invlogit(1)
```

```
## [1] 0.73
```

a. 1
b. 0.98
c. 0.01

3. **Aufgabe**

Eine logistische Regression wurde an einen Datensatz angepasst. Es ergaben sich folgende Koeffizienten (jeweils Punktschätzer):

```
Konstante = -1.9 x = 0.7 z = 0.7
```

`x` ist ein metrischer Prädiktor mit einem Range von 0 bis 10; `z` ist eine binäre Variable mit den Werten 0 und 1.

Visualisieren Sie die Kurven in einem Diagramm für

a. $\mathscr{L}$ vs. x
b. $Pr(y = 1)$ vs. x

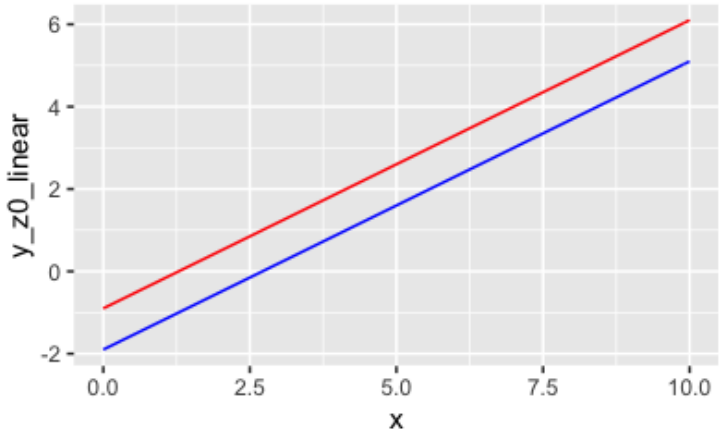
Lösung

Wir definieren die Variablen:

```
d <-
  tibble(
    x = seq(from = 0, to = 10, by = 0.1),
    z = 1,
    y_z0_linear = -1.9 + 0.7 * x + 0*z,
    y_z1_linear = -1.9 + 0.7 * x + 1*z,
    p_y_z0 = plogis(y_z0_linear),
    p_y_z1 = plogis(y_z1_linear)
  )
```

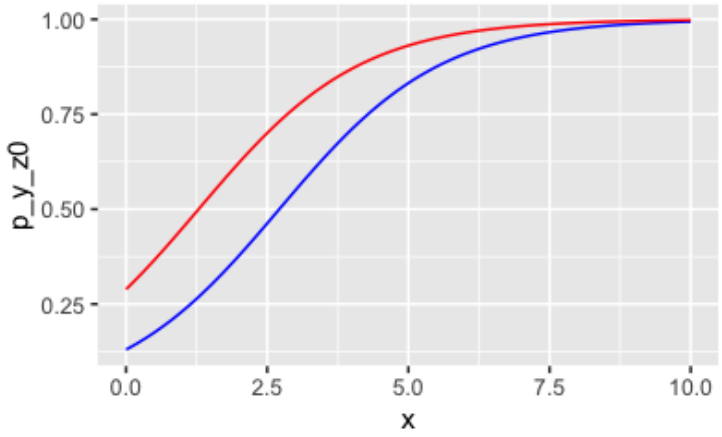
Hier ist das Diagramm mit *Logits* auf der Y-Achse:

```
d %>%
  ggplot() +
  aes(x = x) +
  geom_line(aes(y = y_z0_linear), color = "blue") +
  geom_line(aes(y = y_z1_linear), color = "red")
```



Hier ist das Diagramm mit *Wahrscheinlichkeit* auf der Y-Achse:

```
d %>%
  ggplot() +
  aes(x = x) +
  geom_line(aes(y = p_y_z0), color = "blue") +
  geom_line(aes(y = p_y_z1), color = "red")
```



4. **Aufgabe**

Forschungsfrage: Ist der Zusammenhang von Körpergröße und ‘Mann’ positiv? Gehen also höhere Werte in Körpergröße `height` einher mit einer höheren Wahrscheinlichkeit, dass es sich um einen Mann `m` handelt?

Berechnen Sie ein Bayes-Modell und geben Sie die Punktschätzer der Modellkoeffizienten an - einmal für `height` in Inches und einmal in Zentimeter.

Hinweise:

- Geben Sie `1` ein für “ja” und `0` für nein.
- Gleichen sich Koeffizienten bis einschließlich der 1. Dezimale, so gelten sie als “gleich”.
- Nutzen Sie Standardwerte von `stan_glm()` für Ihr Modell.
- Entfernen Sie fehlende Werte wo nötig.
- Gehen Sie von unzentrierten Daten aus, soweit nicht anders angegeben.

a. Handelt es sich bei der Transformation (von Inches in Zentimeter) um eine lineare Transformation?
b. Ändert sich der Punktschätzer des Achsenabschnitts durch die Transformation?
c. Ändert sich der Punktschätzer des Regressionsgewichts durch die Transformation?
d. Bei zentrierten Daten: Ändert sich der Punktschätzer des Regressionsgewichts durch die Transformation?

Lösung

```
d <- read_csv(
"https://vincentarelbundock.github.io/Rdatasets/csv/openintro/speed_gender_height.csv")
```

Bereiten wir die Daten vor:

```
d2 <- d %>%
  select(gender, height) %>%
  drop_na() %>%
  mutate(male = ifelse(gender == "male", 1, 0),
         height_cm = height * 2.54,
         height_c = height - mean(height),
         height_cm_c = height_cm - mean(height_cm))
```

Modell in Inches:

```
m1 <- stan_glm(male ~ height,
               family = binomial(link="logit"),
               data = d2, refresh = 0)
coef(m1)
```

```
## (Intercept)      height
##      -35.35        0.52
```

Modell in Zentimetern:

```
m2 <- stan_glm(male ~ height_cm,
               family = binomial(link="logit"),
               data = d2, refresh = 0)
coef(m2)
```

```
## (Intercept)   height_cm
##       -35.4         0.2
```

Der Achsenabschnitt ändert sich nicht, da `x=0` vor und nach der Transformation gleich bleibt: 0 Inches sind 0 Zentimeter.

Der Regressionskoeffizient ändert sich, da eine Einheit auf der X-Achse jetzt nicht mehr ein Inch, sondern ein Zentimeter ist, entsprechend ändert sich auch der Wert in der Y-Achse.

Zentriertes Modell in Inches:

```
m3 <- stan_glm(male ~ height_c,
               family = binomial(link="logit"),
               data = d2, refresh = 0)
coef(m3)

## (Intercept)    height_c
##       -1.10        0.52
```

Zentriertes Modell in Zentimetern:

```
m4 <- stan_glm(male ~ height_cm_c,
               family = binomial(link="logit"),
               data = d2, refresh = 0)
coef(m4)

## (Intercept) height_cm_c
##        -1.1         0.2
```

Hier (bei zentrierten Prädiktoren) ändert sich der Achsenabschnitt *nicht* durch die Transformation, da vor und nach der Transformation `x=0` weiterhin `x=0` ist. Das Regressiongewicht, β, ändert sich.

```
sol_a = 1
sol_b <- 0
sol_c <- 1
sol_d <- 1
```

a. Ja; Transformation der Art $y = a + bx$ nennt man *linear*.
b. Nein; der Punktschätzer des Achsenabschnitts ändert sich nicht durch die Transformation.
c. Ja; der der Punktschätzer des Regressionsgewichts ändert sich durch die Transformation.
d. Ja; Bei zentrierten Daten ändert sich der Punktschätzer des Regressionsgewichts durch die Transformation.

5. **Aufgabe**

Betrachten Sie den Datensatz `mtcars`. Die Forschungsfrage betreffe den Einfluss vom Spritverbrauch, x, UV, auf die Wahrscheinlichkeit, dass es sich um ein Auto mit Automatik-Schaltung, y, AV, handelt.

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Berechnen Sie ein lineares Modell auf Basis der Bayes-Statistik.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.

a. Was ist der Punktschätzer (Median) in Logits für den Prädiktor?
b. Was ist der Standardfehler (MAD_SD) in Inv-Logits für den Prädiktor?
c. Was ist die Wahrscheinlichkeit für `am=1` bei einem Auto mit mittlerem Spritverbrauch, $\overline{mpg}$? Berichten Sie den Punktschätzer!

Lösung

a.

Berechnen wir den Koeffizienten (Punktschätzer) für β:

```
m1 <-
  stan_glm(am ~ mpg, data = mtcars,
           refresh = 0,
           family = binomial(link = "logit"))

coef(m1)

## (Intercept)         mpg
##       -6.78        0.32

sol_a <- coef(m1)[2] %>% round(2)
sol_a

##  mpg
## 0.32
```

b.

```
sol_b <- se(m1)[2]%>% invlogit() %>% round(2)
sol_b

##  mpg
## 0.53
```

c.

```
sol_c <-
  predict(m1, newdata = tibble(mpg = mean(mtcars$mpg))) %>%
  invlogit() %>%
  round(2)

sol_c

##    1
## 0.39
```

a. 0.32
b. 0.53
c. 0.39

6. **Aufgabe**

Betrachten Sie den Datensatz `mtcars`. Die Forschungsfrage betreffe den Einfluss vom Spritverbrauch, x, UV, auf die Wahrscheinlichkeit, dass es sich um ein Auto mit Automatik-Schaltung, y, AV, handelt.

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Berechnen Sie ein lineares Modell auf Basis der Bayes-Statistik.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.
- Gehen Sie von einem Modell mit standardisiertem Prädiktor aus.

a. Wie groß ist die Ungewissheit (95%-PI) für den Koeffizienten β in Logits? Geben Sie die Breite an.
b. Wie groß ist die Ungewissheit (95%-PI) für y wenn $x = 1$, in Wahrscheinlichkeit? Geben Sie die Breite an.
c. Wie groß ist die Ungewissheit (95%-PI) für den Achsenabschnitt β in Logits? Geben Sie die Breite an.

Lösung

a. *Wie groß ist die Ungewissheit (95%-PI) für den Koeffizienten β in Logits? Geben Sie die Breite an. Geben Sie die Breite an.*

```
m1 <-
  mtcars %>%
  mutate(mpg_z = scale(mpg)) %>%
  stan_glm(am ~ mpg_z, data = .,
           refresh = 0,
           family = binomial(link = "logit"))

m1_post_intverval <-
  posterior_interval(m1, prob = .95, pars = "mpg_z")
m1_post_intverval
```

```
##       2.5% 97.5%
## mpg_z 0.77   3.4
```

Berechnen wir die Breite des Intervalls:

```
sol_a <- m1_post_intverval[2] - m1_post_intverval[1]
sol_a <-
  sol_a %>% round(2)
sol_a
```

```
## [1] 2.7
```

b. *Wie groß ist die Ungewissheit (95%-PI) für y wenn $x = 1$, in Wahrscheinlichkeit? Geben Sie die Breite an.*

Mit `posterior_epred()` bekommt man Stichproben für den erwarteten Wert laut Post-Verteilung (für bestimmte Prädiktorwerte) als Wahrscheinlichkeit ausgegeben.

Die Stichproben fassen wir dann anhand der Quantile für 2.5% und 97.5% zusammen.

```
new <- tibble(mpg_z = 1)

pred_x1 <-
  posterior_epred(m1, newdata = new) %>%
  as_tibble() %>%
  summarise(pi95 = quantile(`1`, prob = c(0.025, .975)))

pred_x1
```

```
## # A tibble: 2 × 1
##    pi95
##   <dbl>
## 1 0.511
## 2 0.961
```

Jetzt kennen wir die Intervallgrenzen des 95%-PI. Die Breite errechnet sich als Differenz der beiden Werte:

```
sol_b <-
  pred_x1 %>%
  summarise(pi95_width = diff(pi95) %>% round(2))

sol_b
```

```
## # A tibble: 1 × 1
##   pi95_width
##        <dbl>
## 1       0.45
```

```
sol_b <- sol_b %>% pull(pi95_width)
```

Alternativ könnte man auch mit `posterior_linpred()` arbeiten. Dann bekommt man die die Vorhersagen in der Logit-Skala, die man dann zum Schluss in Wahrscheinlichkeit umrechnet.

```
invlogit <- plogis

m1_post_interval_x1 <-
  posterior_linpred(m1, newdata = new) %>%
```

```
  as_tibble()

sol_b2 <-
  m1_post_interval_x1 %>%
  summarise(pi95_logit = quantile(`1`, prob = c(0.025, .975))) %>%
  mutate(pi95_invlogit = invlogit(pi95_logit)) %>%
  summarise(pi95_width = diff(pi95_invlogit)) %>%
  pull(pi95_width) %>%
  round(2)

sol_b2
```

```
## 97.5%
##  0.45
```

c. *Wie groß ist die Ungewissheit (95%-PI) für den Achsenabschnitt β in Logits? Geben Sie die Breite an.*

```
new <- tibble(mpg_z = 0)

pred_x0 <-
  posterior_epred(m1, newdata = new) %>%
  as_tibble() %>%
  summarise(pi95 = quantile(`1`, prob = c(0.025, .975)))

pred_x0
```

```
## # A tibble: 2 × 1
##    pi95
##   <dbl>
## 1 0.212
## 2 0.606
```

Jetzt kennen wir die Intervallgrenzen des 95%-PI. Die Breite errechnet sich als Differenz der beiden Werte:

```
sol_c <-
  pred_x0 %>%
  summarise(pi95_width = diff(pi95) %>% round(2))

sol_c
```

```
## # A tibble: 1 × 1
##   pi95_width
##        <dbl>
## 1       0.39
```

```
sol_c <- sol_c %>% pull(pi95_width)
```

a. 2.67
b. 0.45
c. 0.39

7. **Aufgabe**

Betrachten Sie den Datensatz `mtcars`. Die Forschungsfrage betreffe den Einfluss vom Spritverbrauch, x, UV, auf die Wahrscheinlichkeit, dass es sich um ein Auto mit Automatik-Schaltung, y, AV, handelt.

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Berechnen Sie ein lineares Modell auf Basis der Bayes-Statistik.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.

Was ist der größte (statistische) Effekt des Prädiktors (in Wahrscheinlichkeit)?

Lösung

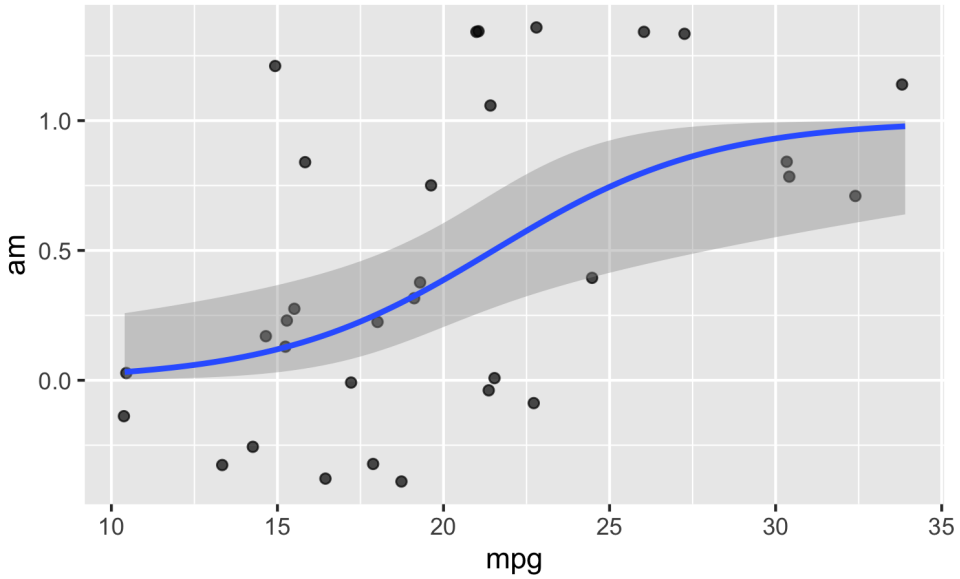
a.

```
data(mtcars)
m1 <-
  stan_glm(am ~ mpg, data = mtcars,
           refresh = 0,
           family = binomial(link = "logit"),
           chains = 1)

coef(m1)

## (Intercept)         mpg
##       -6.82        0.31
```

Wir ziehen hier nur ein mal die Stichproben (`chains=1`), um Rechenzeit zu sparen. Bei diesem einfachen Modell ist nicht zu erwarten, dass es irgendwo Schwierigkeiten gibt.

Plotten wir das Modell zur Orientierung. Eine einfache Methode zum Plotten geht so:

```
mtcars %>%
  ggplot(aes(x = mpg, y = am)) +
  geom_jitter(width = 0.1, alpha = .7) +
  geom_smooth(method = "glm", method.args=list(family="binomial"))
```



Im Gegensatz zu einer Geraden ist die Steigung bei einer Kurve wie unserer Ogive unterschiedlich in Abhängigkeit vom X-Wert.

`geom_smooth()` geht hier von einem Logit-Modell mit uniformen Prioris aus, was nicht den Standard von `rstan_glm` gleicht. Der Hintergrund ist, dass `geom_smooth()` auf Basis der Frequentistischen Statistik arbeitet, `rstanarm` hingegen Bayesianisch. Bei Frequentistischen Modellen wird - übersetzt man sie in Bayes-Modelle - immer eine uniforme Priori-Verteilung verwendet.

Wir könnten diese Aufgabe durch einfaches Abschätzen oder Ausprobieren lösen, indem wir die vorhergesagten Änderungen in Y für verschiedene Punktepaare (x_1, x_2) berechnen, wobei bei jedem Punktepaar der X-Abstand 1 beträgt. Anders gesagt könnten wir die Steigung der Ogive an verschiedenen Stellen berechnen.

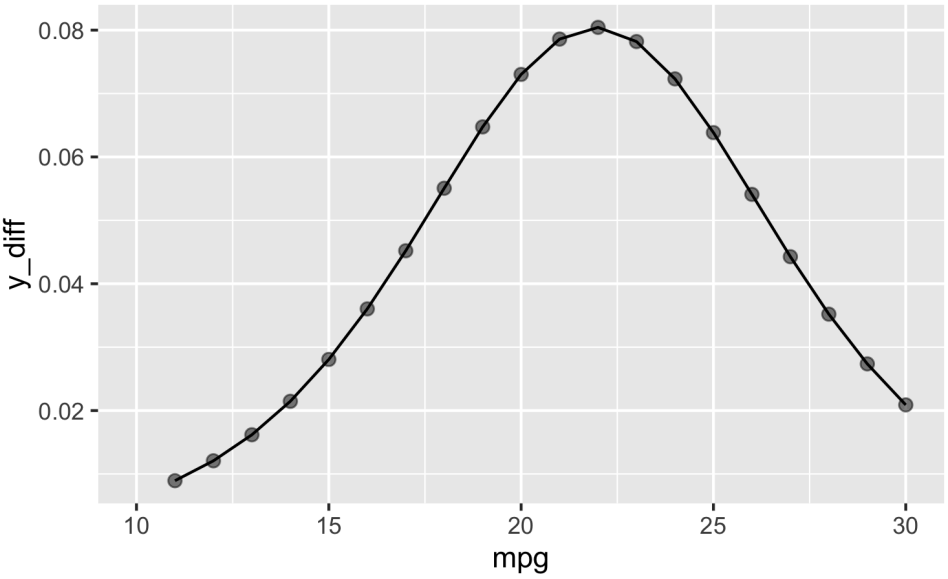
```
preds_df <-
  tibble(
    mpg = 10:30,
```

```
    y_logits = predict(m1, newdata = tibble(mpg = mpg)),
    y_pr = invlogit(y_logits),
    y_lag = lag(y_pr),
    y_diff = y_pr - y_lag
  )

preds_df %>%
  select(mpg, y_diff)
```

```
## # A tibble: 21 × 2
##      mpg   y_diff
##    <int>    <dbl>
##  1    10 NA
##  2    11  0.00894
##  3    12  0.0121
##  4    13  0.0162
##  5    14  0.0215
##  6    15  0.0281
##  7    16  0.0360
##  8    17  0.0452
##  9    18  0.0551
## 10    19  0.0647
## # … with 11 more rows
```

Das visualisieren wir mal:

```
preds_df %>%
  ggplot(aes(x = mpg, y = y_diff)) +
  geom_line() +
  geom_point(size = 2, alpha = .5)
```



Wir sehen also, dass der maximale Zuwachs an Prozentpunkten bei ca. 0.08 liegt, was bei ca `mpg=22` der Fall ist.

Diesen Wert nehmen wir als Lösung:

```
sol <-
  max(preds_df$y_diff, na.rm = TRUE) %>%
  round(2)

sol
```

```
## [1] 0.08
```

Alternativ könnten wir uns auch folgenden Sachverhalt vor Augen führen:

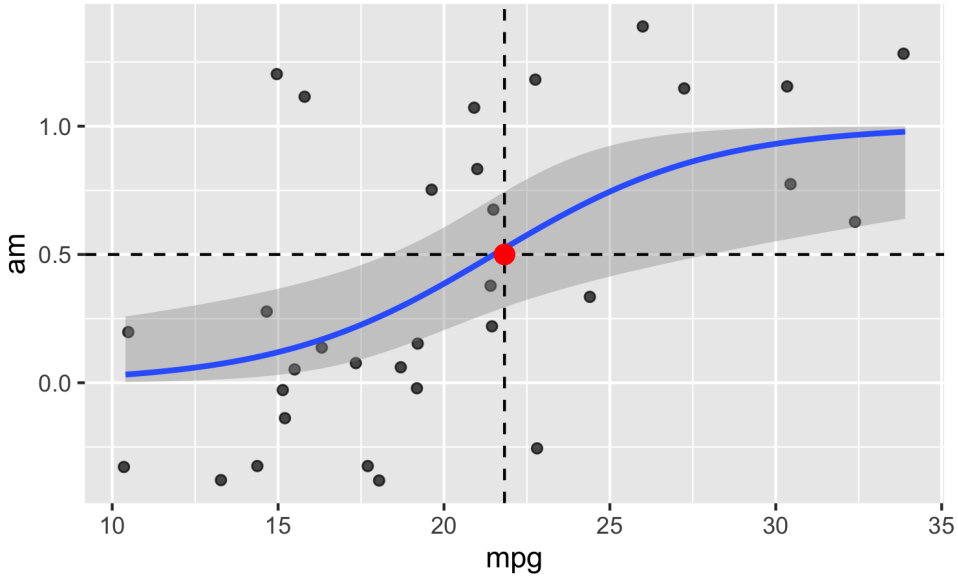
Die Kurve ist am steilsten in ihrem Zentrum (Wendepunkt); vgl. diese Animnation.

Der Wendepunkt liegt bei $Pr(y = 1) = 1/2$, also 50%. Damit gilt auch $\alpha + \beta x = 0$, denn $\mathscr{L}^{-1}(0) = 1/2$.

An dieser Stelle ist die Steigung der Ogive maximal und beträgt $\beta_{\mathscr{L}}/4$ Prozentpunkte, vgl. ROS S. 220 (Kap. 13.2), also ein Viertel der Steigung in Logits.

Die obere Grenze der Steigung in Prozentpunkten liegt also bei $\beta/4$, wobei β in Logits gemessen ist.

```
mtcars %>%
  ggplot(aes(x = mpg, y = am)) +
  geom_jitter(width = 0.1, alpha = .7) +
  geom_smooth(method = "glm", method.args=list(family="binomial")) +
  geom_hline(yintercept = 1/2, linetype = "dashed") +
  geom_vline(xintercept = -coef(m1)[1] / coef(m1)[2], linetype = "dashed") +
  geom_point(x = -coef(m1)[1] / coef(m1)[2],
             y = 1/2,
             color = "red",
             size = 3,
             alpha = .3)
```



In unserem Fall:

```
max_steigung_in_pr <-
  coef(m1)[2] / 4

max_steigung_in_pr
```

```
##    mpg
## 0.078
```

Wir hätten den steilsten Punkt auch so bestimmen können:

$$\begin{aligned} -6.8 + 0.32x &= 0 \\ 0.32x &= 6.8 \\ x &= 6.8/0.32 \approx 21 \end{aligned}$$

8. **Aufgabe**

Betrachten Sie den Datensatz `mtcars`. Die Forschungsfrage betreffe den Einfluss vom Spritverbrauch, x_1, UV1, sowie von der Zahl der Zylinder (als Faktor-Variable), x_2, UV2, auf die Wahrscheinlichkeit, dass es sich um ein Auto mit Automatik-Schaltung, y, AV, handelt. Modellieren Sie dabei einen Interaktionseffekt.

Aufgaben:

a. Überschneiden sich die Graphen der drei Gruppen?
b. Visualisieren Sie das Modell! Einmal mit der Y-Achse skaliert in Logits…
c. … einmal mit der Y-Achse skaliert in Wahrscheinlichkeiten.

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Berechnen Sie ein lineares Modell auf Basis der Bayes-Statistik.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.

Lösung

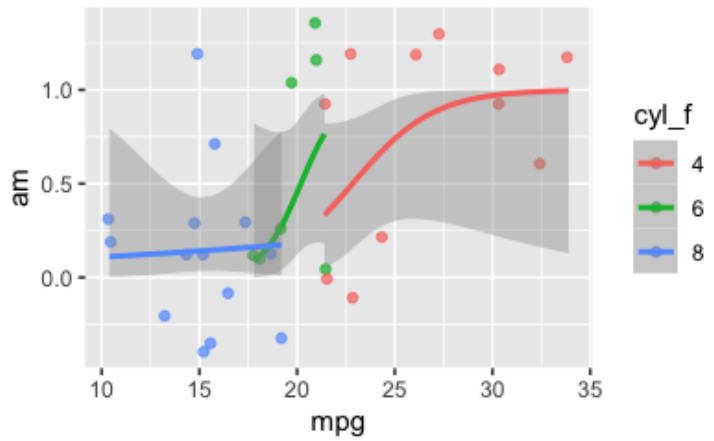
a. *Überschneiden sich die Graphen der drei Gruppen?*

```
data(mtcars)
mtcars2 <-
  mtcars %>%
  mutate(cyl_f = factor(cyl))

m1 <-
  stan_glm(am ~ mpg + cyl_f + mpg:cyl_f,
           data = mtcars2,
           refresh = 0,
           family = binomial(link = "logit"))

coef(m1)
```

```
## (Intercept)         mpg      cyl_f6      cyl_f8  mpg:cyl_f6  mpg:cyl_f8
##      -7.621       0.347      -1.145       1.235       0.080      -0.071
```

Plotten wir das Modell zur Orientierung. Eine einfache Methode zum Plotten geht so:

```
mtcars2 %>%
  ggplot(aes(x = mpg, y = am, color = cyl_f)) +
  geom_jitter(width = 0.1, alpha = .7) +
  geom_smooth(method = "glm", method.args=list(family="binomial"))
```



Dabei ist zu beachten, dass hier ein uniformer Prior (für das lineare Modell) verwendet wird.

Im Gegensatz zu einer Geraden ist die Steigung bei einer Kurve wie unserer Ogive unterschiedlich in Abhängigkeit vom X-Wert.

Ja, die Ogiven überschneiden sich.

Die Visualisierung wird nicht allen Wünschen gerecht, weil die Regressionskurve nur entlang des Wertebereichs der Daten gezeigt wird.

b. *Visualisieren Sie das Modell! Einmal mit der Y-Achse skaliert in Logits…*

Bei b) soll das Modell visualisiert werden, also los.

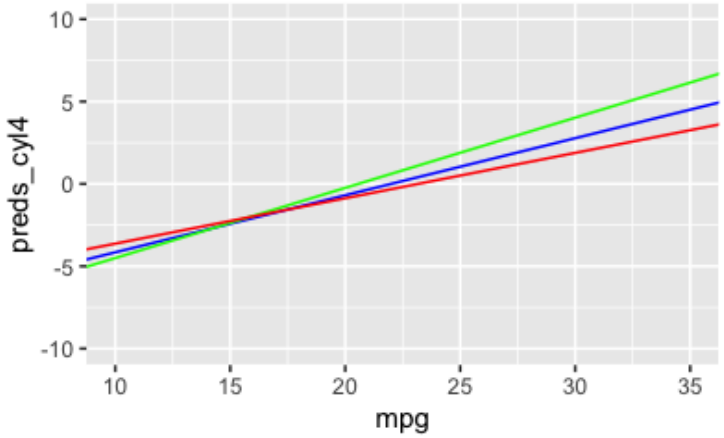
Sagen wir, wir wollen y-Werte für Werte von `mpg` von 10 bis 35 vorhersagen, für alle 3 Gruppen von `cyl`.

Sagen wir dafür entsprechend die Logit-Werte vorher:

```
preds_df <-
  tibble(
    mpg = 10:35,
    preds_cyl4 = predict(m1, newdata =
                           tibble(mpg = mpg,
                                  cyl_f = "4")),
    preds_cyl6 = predict(m1, newdata =
                           tibble(mpg = mpg,
                                  cyl_f = "6")),
    preds_cyl8 = predict(m1, newdata =
                           tibble(mpg = mpg,
                                  cyl_f = "8"))
  )
```

Alternativ könnten wir direkt die Regressionskoeffizienten nutzen, für `geom_abline()`:

```
preds_df %>%
  ggplot(aes(x = mpg,
             y = preds_cyl4)) +
  scale_x_continuous(limits = c(10, 35)) +
  scale_y_continuous(limits = c(-10, 10)) +
  geom_abline(slope = coef(m1)[2],
              intercept = coef(m1)[1],
              color = "blue") +
  geom_abline(slope = coef(m1)[2] + coef(m1)["mpg:cyl_f6"],
              intercept = coef(m1)[1] + coef(m1)["cyl_f6"],
              color = "green") +
  geom_abline(slope = coef(m1)[2] + coef(m1)["mpg:cyl_f8"],
              intercept = coef(m1)[1] + coef(m1)["cyl_f8"],
              color = "red")
```



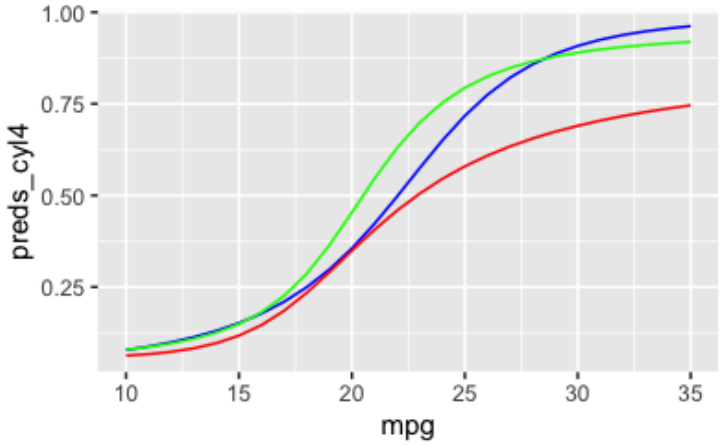
c.

```
preds_df2 <-
  tibble(
    mpg = 10:35,
    preds_cyl4 = predict(m1, newdata =
                           tibble(mpg = mpg,
```

```
                                    cyl_f = "4"),
                          type = "response"),
    preds_cyl6 = predict(m1, newdata =
                           tibble(mpg = mpg,
                                  cyl_f = "6"),
                          type = "response"),
    preds_cyl8 = predict(m1, newdata =
                           tibble(mpg = mpg,
                                  cyl_f = "8"),
                          type = "response")
  )
```

Das können wir dann visualisieren:

```
preds_df2 %>%
  ggplot(aes(x = mpg)) +
  geom_line(aes(y = preds_cyl4),
            color = "blue")  +
  geom_line(aes(y = preds_cyl6),
            color = "green")  +
  geom_line(aes(y = preds_cyl8),
            color = "red")
```



c. *… einmal mit der Y-Achse skaliert in Wahrscheinlichkeiten.*

Um die Ungewissheit zu visualisieren, müssen wir die Stichproben aus der Post-Verteilung visualisieren.

Dabei hlefen die Pakete `tidybayes` und `modelr`:

```
library(tidybayes)
library(modelr)  # für `data_grid()`
```

Zuerst erstellen wir ein “Grid” mit den Prädiktorwerten, für die wir eine Vorhersage wollen:

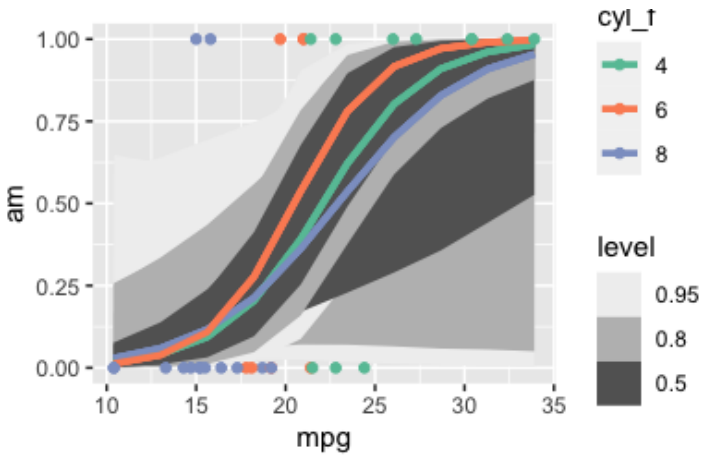
Jetzt fügen wir die Stichproben aus der Post-Verteilung von `m1` hinzu, gleich in der Wskt-Skalierung. Das ging doch mit `posterior_epred()`…

```
## # A tibble: 300 × 7
## # Groups:   mpg, cyl_f, .row [30]
##      mpg cyl_f  .row .chain .iteration .draw   .epred
##    <dbl> <fct> <int>  <int>      <int> <int>    <dbl>
##  1  10.4 4         1     NA         NA     1 0.00572
##  2  10.4 4         1     NA         NA     2 0.0691
##  3  10.4 4         1     NA         NA     3 0.0610
##  4  10.4 4         1     NA         NA     4 0.337
##  5  10.4 4         1     NA         NA     5 0.499
##  6  10.4 4         1     NA         NA     6 0.263
##  7  10.4 4         1     NA         NA     7 0.00156
##  8  10.4 4         1     NA         NA     8 0.0468
##  9  10.4 4         1     NA         NA     9 0.00153
```

```
## 10  10.4 4         1     NA         NA    10 0.000643
## # … with 290 more rows
```

Für jede der 30 Fälle (`.row`) wurden 4000 Stichproben (`draw`) gezogen, in Summe als 120,000.

Diese Stichproben(-Fülle) können wir uns jetzt komfortabel plotten lassen von `tidybayes`:



Das Diagramm sieht nicht gerade vertrauenserweckend aus… Zu viel Ungewissheit in unserem Modell.

9. **Aufgabe**

Betrachten Sie den Datensatz `mtcars`. Die Forschungsfrage betreffe den (statistischen) Einfluss von der Zahl der Zylinder, x_2, UV, auf die Wahrscheinlichkeit, dass es sich um ein Auto mit Automatik-Schaltung, y, AV, handelt. Modellieren Sie keinen Interaktionseffekt.

Hinweise:

- Runden Sie auf zwei Dezimalstellen.
- Berechnen Sie ein lineares Modell auf Basis der Bayes-Statistik.
- Orientieren Sie sich am Vorgehen aus dem Unterricht.
- Geben Sie `1` ein für “ja” und `0` für nein.
- Gleichen sich Koeffizienten bis einschließlich der 1. Dezimale, so gelten sie als “gleich”.
- Nutzen Sie Standardwerte von `stan_glm()` für Ihr Modell.
- Entfernen Sie fehlende Werte wo nötig.

Aufgaben:

a. Ist die Wahrscheinlichkeit für ein Automatik-Getriebe höher bei `cyl=4` im Vergleich zu `cyl=6`, laut dem Modell? Beziehen Sie sich auf die den Punktschätzer der Post-Verteilung.
b. Um welchen Wert unterscheidet sich die Wahrscheinlichkeit?

Lösung

a.

```
data(mtcars)

m1 <-
  stan_glm(am ~ cyl,
           data = mtcars,
           refresh = 0,
           family = binomial(link = "logit"))
```

```
coef(m1)
```

```
## (Intercept)         cyl
##        3.90       -0.71
```

Der Punktschätzer der Post-Verteilung zeigt uns: Je höher die Zylinderzahl, desto geringer der Logit (und damit die Wahrscheinlichkeit) für ein manuelles Getriebe bzw. umso höher für ein Automatik-Getriebe.

Also lautet die Antwort: nein.

```
sol_a <- 0
```

b.

Ziehen wir Stichproben aus der Post-Verteilung:

```
pred_pr <-
  posterior_epred(m1, newdata = tibble(cyl = c(4,6))) %>%
  as_tibble() %>%
  mutate(cyl4_rather_manuell = `1` > `2`)

pred_pr %>%
  slice_head(n=5)
```

```
## # A tibble: 5 × 3
##     `1`   `2` cyl4_rather_manuell
##   <dbl> <dbl> <lgl>
## 1 0.822 0.441 TRUE
## 2 0.681 0.395 TRUE
## 3 0.604 0.365 TRUE
## 4 0.693 0.299 TRUE
## 5 0.818 0.580 TRUE
```

Wir haben hier geprüft, wie oft `1` < `2` gilt. Also: wie oft die Wahrscheinlichkeit für Manuell-Getriebe (`am=1`) bei `cyl=4` höher ist als bei `cyl=6`.

```
pred_pr %>%
  summarise(cyl4_rather_manuell = mean(cyl4_rather_manuell))
```

```
## # A tibble: 1 × 1
##   cyl4_rather_manuell
##                 <dbl>
## 1                1.00
```

Diese Wahrscheinlichkeit ist (praktisch) 1, laut dem Modell.

```
sol_b <- 1
```

a. Nein, je höher die Zylinderzahl, desto geringer der Logit (und damit die Wahrscheinlichkeit) für ein manuelles Getriebe bzw. umso höher für ein Automatik-Getriebe.
b. 1

10. **Aufgabe**

Eine Studie untersuchte den Arsengehalt `arsenic` in Brunnen in Bangladesh. Die Forschis untersuchten u.a., ob die Menschen bereit waren, auf einen Brunnen zu wechseln `switch`, der nicht mit Arsen belastet war, und welche Rolle die Entfernung (in Einheiten von 100 Meter, `dist100`) zum nächsten unbelasteten Brunnen spielt.

Die Daten sind hier zu beziehen:

```
d_path <- "https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Arsenic/data/wells.csv"
```

```
d <- read_csv(d_path)
```

Hier ist das Regressionsmodell:

```
m1 <- stan_glm(switch ~ dist100 + arsenic,
               family = binomial(link = "logit"),
               data = d,
               refresh = 0)

## stan_glm
##  family:       binomial [logit]
##  formula:      switch ~ dist100 + arsenic
##  observations: 3020
##  predictors:   3
## ------
##             Median MAD_SD
## (Intercept)  0.00   0.08
## dist100     -0.90   0.10
## arsenic      0.46   0.04
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

a. Vergleichen Sie zwei Personen mit der gleichen Distanz zum nächsten unbelasteten Brunnen, wobei eine Person einen Arsenwert von 0.5 hat und die andere Person einen Wert von 1.0.

Wie groß ist der Unterschied in Wahrscheinlichkeit zwischen erster und zweiter Person, den Brunnen zu wechseln (laut dem obigen Modell)?

Beziehen Sie sich auf die Daten der Post-Verteilung.

b. Vergleichen Sie zwei Personen mit der gleichen Distanz zum nächsten unbelasteten Brunnen, wobei eine Person einen Arsenwert von 0.5 hat und die andere Person einen Wert von 1.0.

Wie groß ist der Unterschied in Wahrscheinlichkeit, dass die zweite Person den Brunnen wechselt? Geben Sie einen Punktschätzer auf Basis des Mittelwerts der Post-Verteilung an! Beziehen Sie sich auf die Daten der PPV.

a. Wie groß ist die Wahrscheinlichkeit bei Frage a?
b. Wie groß ist die Wahrscheinlichkeit bei Frage b?

Lösung

a.

```
invlogit <- plogis

preds_a <-
  posterior_linpred(m1, newdata = tibble(dist100 = c(1, 1), arsenic = c(0.5, 1)))

preds_a_summ <-
  preds_a %>%
  as_tibble() %>%
  pivot_longer(everything(),
               names_to = "arsenic_level",
               values_to = "switch_logit") %>%
  mutate(switch_p = invlogit(switch_logit)) %>%
  group_by(arsenic_level) %>%
  summarise(switch_mean = mean(switch_p),
            switch_q05 = quantile(switch_p, prob = .05),
            switch_q95 = quantile(switch_p, prob = .95),
            switch_sd = sd(switch_p))
```

```
preds_a_summ
```

```
## # A tibble: 2 × 5
##   arsenic_level switch_mean switch_q05 switch_q95 switch_sd
##   <chr>               <dbl>      <dbl>      <dbl>     <dbl>
## 1 1                   0.340      0.309      0.371    0.0191
## 2 2                   0.393      0.365      0.422    0.0174
```

Die Antwort lautet also:

```
sol_a <- preds_a_summ$switch_mean[2] - preds_a_summ$switch_mean[1]
sol_a <- sol_a %>% round(2)
sol_a
```

```
## [1] 0.05
```

0.05.

Auf dieser Basis könn(t)en wir Punktschätzer plus ihre Ungewissheit quantifizieren und visualisieren.

Mit `posterior_epded()` bekommt man das gleicher Ergebnis wie mit `posterior_linpred() %>% invlogit()`.

Man könnte auch `predict()` nutzen; dann bekommt man den Punktschätzer (auf Basis des Medians) aber ohne Angaben der Ungewissheit der Schätzung. Daher ist `predict()` nicht so nützlich.

```
preds_a <- predict(m1,
                   newdata = tibble(dist100 = c(1, 1), arsenic = c(0.5, 1)),
        type = "response")

preds_a
```

```
##    1    2
## 0.34 0.39
```

Der Unterschied beträgt ca. 0.05.

`predict()` gibt uns den *Punktschätzer`, $\widehat{a} + \widehat{b}x$. Da wir nur den Punktschätzer bekommen, fehlt uns die Einschätzung der Ungewissheit der Koeffizienten.

Vgl. ROS, S. 92, Kap. 9.2.

b. Berechnen wir die PPV:

```
preds_b <- posterior_predict(m1,
                             newdata = tibble(dist100 = c(1, 1),
                                              arsenic = c(0.5, 1)))
```

`posterior_predict` zieht Werte aus der PPV.

Was ist die Wahrscheinlichkeit für `switch` in jeder Gruppe?

```
preds_b_summ <-
  preds_b %>%
  as_tibble() %>%
  summarise(wechsler_gruppe_a = mean(`1`),
            wechsler_gruppe_b = mean(`2`)) %>%
  mutate(diff = wechsler_gruppe_b - wechsler_gruppe_a)

preds_b_summ
```

```
## # A tibble: 1 × 3
##   wechsler_gruppe_a wechsler_gruppe_b   diff
##               <dbl>             <dbl>  <dbl>
## 1             0.346             0.392 0.0455
```

Die Differenz der beiden Werte gibt uns einen Schätzwert für den Unterschied in der Wahrscheinlichkeit in den beiden Gruppen hinsichtlich des Wechselns des Brunnens.

Die Antwort für b) lautet also:

```
sol_b <-
  preds_b_summ %>%
  pull(diff)

sol_b
```

```
## [1] 0.046
```

a. Der Wert beträgt 0.05.
b. Der Wert beträgt 0.05.

11. **Aufgabe**

Eine Studie untersuchte den Arsengehalt `arsenic` in Brunnen in Bangladesh. Die Forschis untersuchten u.a., ob die Menschen bereit waren, auf einen Brunnen zu wechseln `switch`, der nicht mit Arsen belastet war, und welche Rolle die Entfernung (in Einheiten von 100 Meter, `dist100`) zum nächsten unbelasteten Brunnen spielt.

Die Daten sind hier zu beziehen:

```
d_path <- "https://raw.githubusercontent.com/avehtari/ROS-Examples/master/Arsenic/data/wells.csv"

d <- read_csv(d_path)
```

Hier ist ein Regressionsmodell mit Interaktionseffekt:

```
m2 <- stan_glm(switch ~ dist100 + arsenic + dist100:arsenic,
               family = binomial(link = "logit"),
               data = d,
               refresh = 0)

print(m2, digits = 2)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      switch ~ dist100 + arsenic + dist100:arsenic
##  observations: 3020
##  predictors:   4
## ------
##                Median MAD_SD
## (Intercept)     -0.15   0.12
## dist100         -0.58   0.21
## arsenic          0.56   0.07
## dist100:arsenic -0.18   0.10
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

Und hier das Modell mit gleichen Prädiktoren, aber zentrierten Prädiktoren:

```
d2 <-
  d %>%
  select(dist100, arsenic, switch) %>%
  drop_na() %>%
  mutate(dist100_c = dist100 - mean(dist100),
         arsenic_c = arsenic - mean(arsenic))

m3 <- stan_glm(switch ~ dist100_c + arsenic_c + dist100_c:arsenic_c,
               family = binomial(link = "logit"),
```

```
               data = d2,
               refresh = 0)

print(m3, digits = 2)
```

```
## stan_glm
##  family:       binomial [logit]
##  formula:      switch ~ dist100_c + arsenic_c + dist100_c:arsenic_c
##  observations: 3020
##  predictors:   4
## ------
##                     Median MAD_SD
## (Intercept)          0.35   0.04
## dist100_c           -0.87   0.10
## arsenic_c            0.47   0.04
## dist100_c:arsenic_c -0.18   0.10
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

Warum hat das *nicht* zentrierte Modell größere Standardfehler (MAD_sd)?

Lösung

Der Hauptgrund ist, dass im Modell `m2`, dem Modell mit unzentrierten Prädiktoren, eine hohe Korrelation zwischen einfachen Prädiktoren und dem Interaktionsterm vorliegt. Hohe Korrelation zwischen den Prädiktoren erhöht die Standardfehler. Warum ist das so? Sind zwei Prädiktoren hoch korreliert, was ist dann der Mehrwert, den zweiten Prädiktor zu kennen, wenn man den ersten kennt? Gering, vermutlich! Denn der zweite birgt ja redundante Information im Vergleich zum ersten. Aber das Modell könnte auch denken, dass der zweite Prädiktor wichtig und der erste redundant ist. Kurz gesagt: Das Modell ist verwirrt. Das schlägt sich in höheren Standardfehlern nieder. Durch das Zentrieren wird die Korrelation zwischen den einfachen Prädiktoren und dem Interaktionsterm aufgelöst bzw. zumindest deutlich verringert.

Zur Verdeutlichung: Die Korrelation zwischen einfachen Prädiktoren und Interaktionsterm:

```
library(corrr)
d %>%
  select(dist100, switch, arsenic) %>%
  mutate(dist100xarsenic = dist100 * arsenic) %>%
  correlate()
```

```
## # A tibble: 4 × 5
##   term            dist100   switch arsenic dist100xarsenic
##   <chr>             <dbl>    <dbl>   <dbl>           <dbl>
## 1 dist100          NA     -0.118     0.178         0.772
## 2 switch           -0.118 NA         0.184         0.00758
## 3 arsenic           0.178  0.184    NA             0.635
## 4 dist100xarsenic   0.772  0.00758   0.635        NA
```

Wie man sieht, ist die Korrelation hoch.

Jetzt zum Vergleich die zentrierten Daten:

```
d2 %>%
  select(dist100_c, switch, arsenic_c) %>%
  mutate(dist100_cxarsenic_c = dist100_c * arsenic_c) %>%
  correlate()
```

```
## # A tibble: 4 × 5
##   term                dist100_c  switch arsenic_c dist100_cxarsenic_c
##   <chr>                   <dbl>   <dbl>     <dbl>               <dbl>
## 1 dist100_c              NA     -0.118     0.178               0.185
## 2 switch                 -0.118 NA         0.184              -0.0373
```

```
## 3 arsenic_c               0.178  0.184    NA                   0.0394
## 4 dist100_cxarsenic_c     0.185 -0.0373    0.0394            NA
```

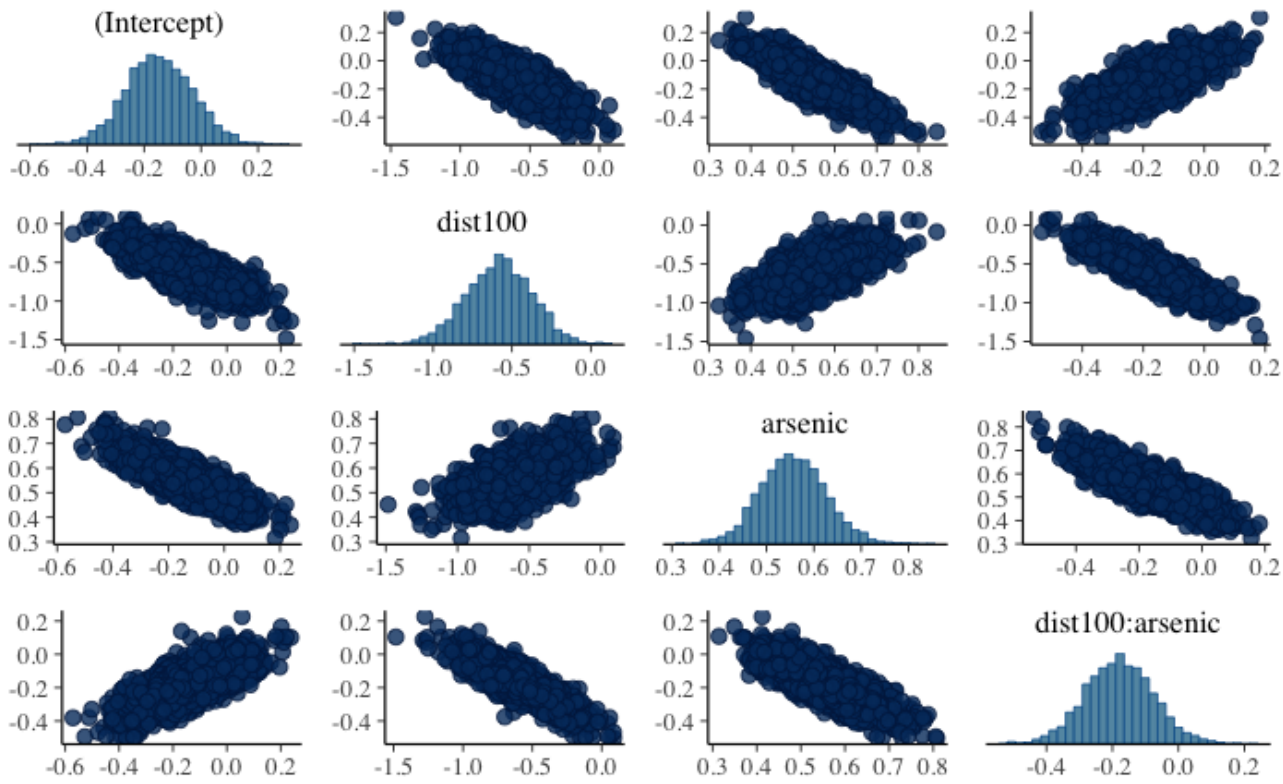
Dieses Phänomen bezeichnet man auch *Kollinearität*.

Weitere Erklärung findet sich bei McElreath 2020, Kap. 6 oder z.B. hier.

Ein ergänzender Blick auf die Korrelationen der Prädiktoren:

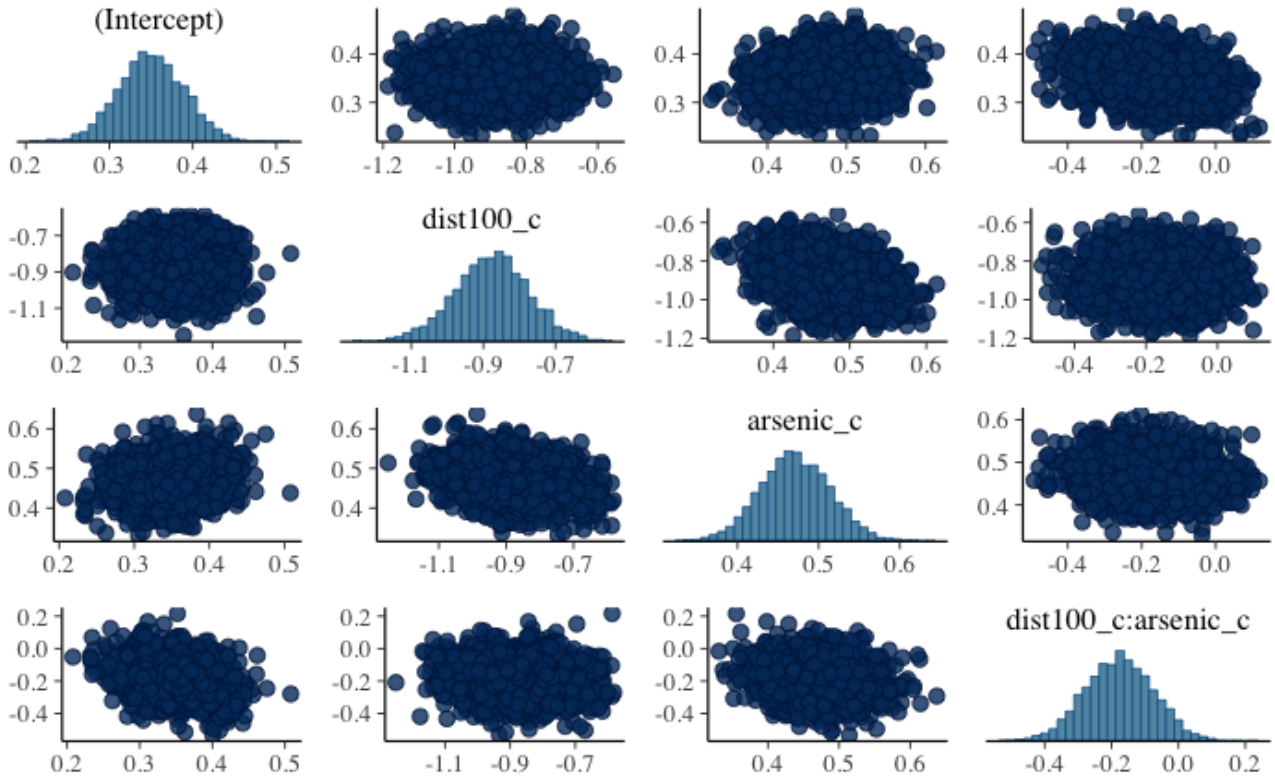
Korrelationen der Koeffizienten für `m2`:

```
library(bayesplot)
mcmc_pairs(m2)
```



Korrelationen der Koeffizienten für `m3`:

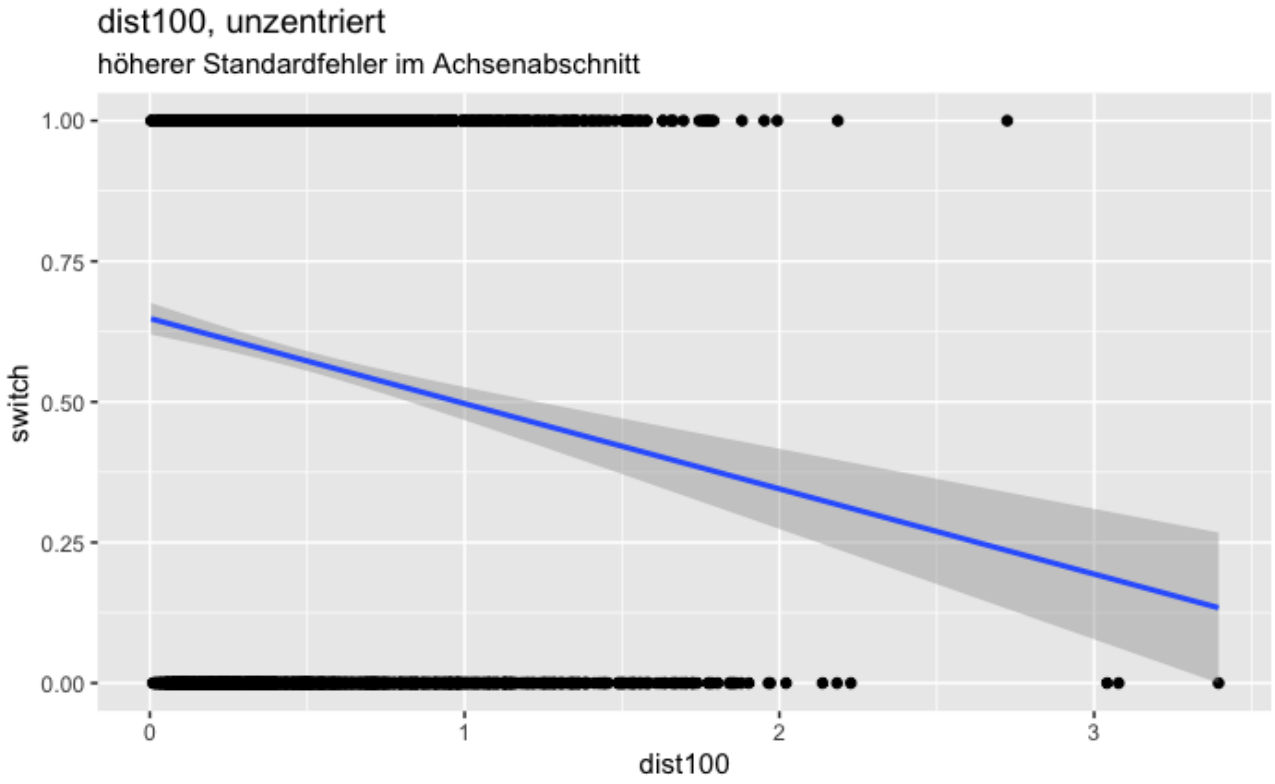
```
library(bayesplot)
mcmc_pairs(m3)
```

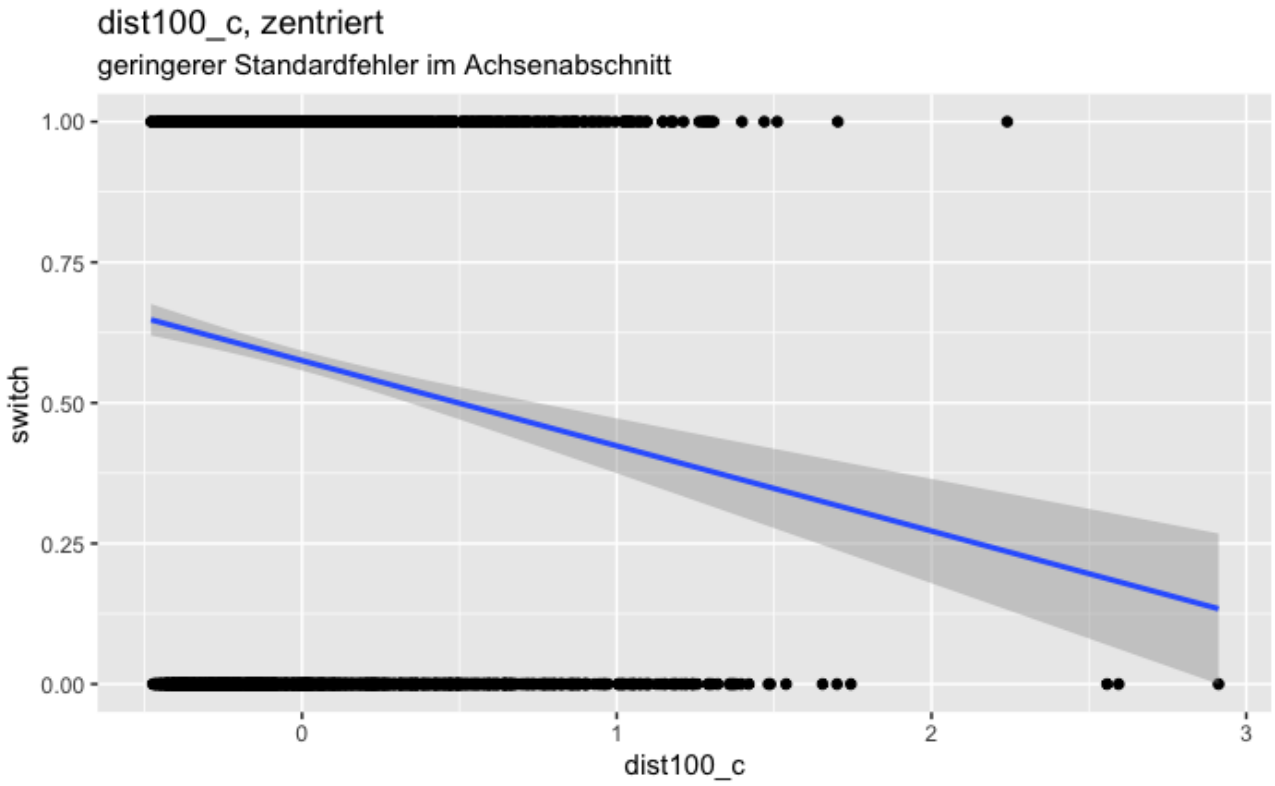


Ein zweiter, weniger wichtiger Grund für die Reduktion der Standardfehler ist der Folgende.

Liegt der Achsenabschnitt (Intercept, Konstante des Modells) am *Rand* der Daten, so ist seine Ungewissheit größer, als wenn er in der Mitte der Daten liegt:

```
d2 %>%
  ggplot(aes(x = dist100, y = switch)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "dist100, unzentriert",
       subtitle = "höherer Standardfehler im Achsenabschnitt")


d2 %>%
  ggplot(aes(x = dist100_c, y = switch)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "dist100_c, zentriert",
       subtitle = "geringerer Standardfehler im Achsenabschnitt")
```



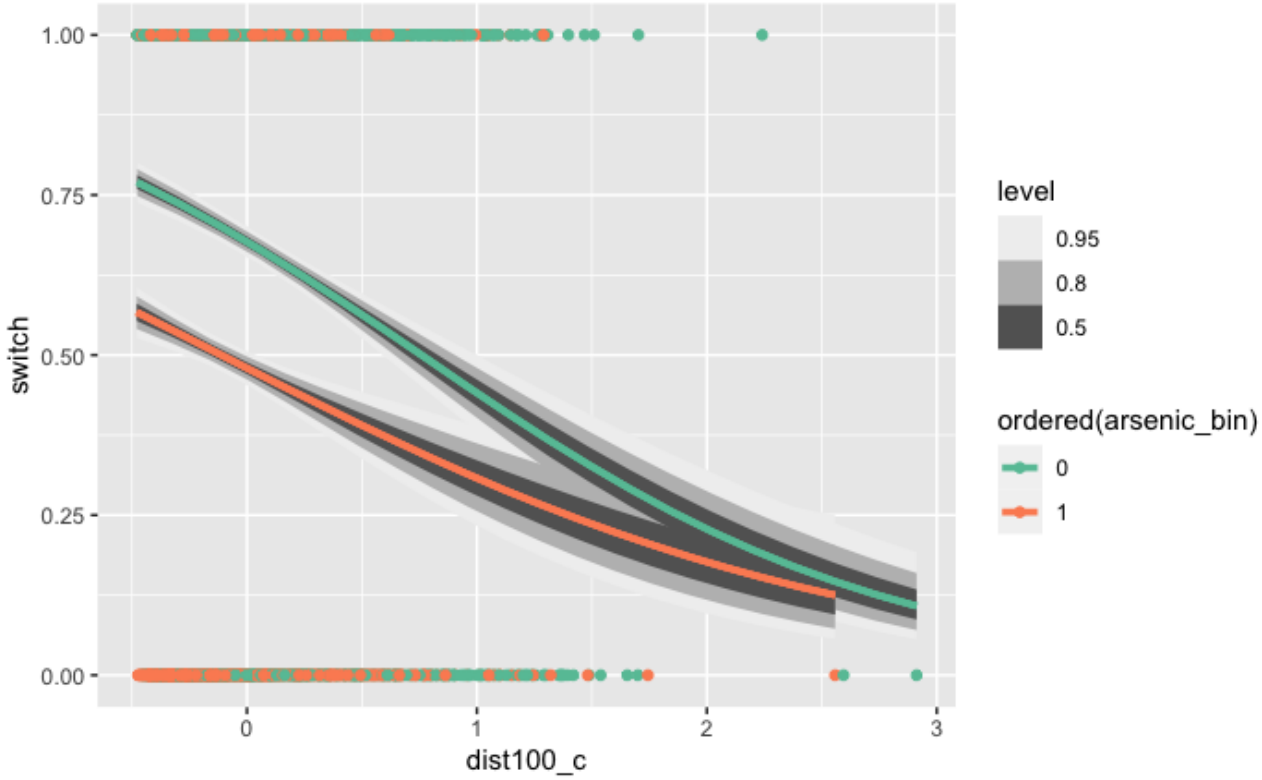


Der Effekt ist allerdings nicht groß.

Auch für ein generalisiertes lineares Modell (wie die logistische Regression) gilt dieser Grundsatz: An den Rändern der Daten (minimales bzw. maximale Prädiktorwerte) sind die Schätzungen ungenauer als im Zentrum.

```
library(tidybayes)
library(modelr)
d3 <-
d2 %>%
  mutate(arsenic_bin = ifelse(arsenic < median(arsenic), 1, 0))
```

```
m4 <- stan_glm(switch ~ dist100_c + arsenic_bin + dist100_c:arsenic_bin,
               family = binomial(link = "logit"),
               data = d3,
               refresh = 0)


d4 <-
  d3 %>%
  group_by(arsenic_bin) %>%
  data_grid(dist100_c = seq_range(dist100_c, n = 20)) %>%
  add_epred_draws(m4)

d4 %>%
  ggplot(aes(x = dist100_c, y = switch, color = ordered(arsenic_bin))) +
  stat_lineribbon(aes(y = .epred)) +
  geom_point(data = d3) +
  scale_fill_brewer(palette = "Greys") +
  scale_color_brewer(palette = "Set2")
```



Vergleichen wir dazu die Standardfehler in einem Modell ohne Interaktion, `m4`. Wir lassen die Interaktion weg, um zu prüfen, ob es Effekte auf die Standardfehler gibt, die unabhängig von der Interaktion (bzw. der dadurch erzeugten Kollinearität) sind.

```
m1 <- stan_glm(switch ~ dist100 + arsenic,
               family = binomial(link = "logit"),
               data = d,
               refresh = 0)

m4 <-
  stan_glm(switch ~ dist100_c + arsenic_c,
           family = binomial(link = "logit"),
           data = d2,
           refresh = 0)
```

Vergleichen wir die Standardfehler der Modelle. Zur Erinnerung: Der Standardfehler ist eine Möglichkeit, die Ungewissheit des Modells zu quantifizieren.

Standardfehler im *nicht*-zentrierten Modell (ohne Interaktion):

```
se(m1)
```

```
## (Intercept)     dist100     arsenic
##       0.080       0.102       0.041
```

Standardfehler im zentrierten Modell (ohne Interaktion):

```
se(m4)
```

```
## (Intercept)   dist100_c   arsenic_c
##       0.038       0.106       0.042
```

Wie wir sehen, wirkt sich die Zentrierung nur (günstig) auf den Achsenabschnitt aus, aber nicht auf den Standardfehler des Regressionskoeffizienten.