Übungszettel: Einfache lineare Regression (03)

M.Psy.205, Dozent: Dr. Peter Zezula

Johannes Brachem (johannes.brachem@stud.uni-goettingen.de)

06 Mai, 2021 23:11

Deutsche Version

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.
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

Ressourcen

Da es sich um eine praktische Übung handelt, können wir Ihnen nicht alle neuen Befehle einzeln vorstellen. Stattdessen finden Sie hier Verweise auf sinnvolle Ressourcen, in denen Sie für die Bearbeitung unserer Aufgaben nachschlagen können.

Ressource	Beschreibung
Field, Kapitel 7 (7.1 - 7.5, 7.9)	Buchkapitel, das Schritt für Schritt erklärt, worum es geht, und wie man Regressionen in R durchführt. **Große Empfehlung!**
R for Data Science	Einsteiger-Buch von R-Gott Hadley Wickham. Hier wird topaktuell in die Arbeit mit R, insbesondere zur Datenaufbereitung und Visualisierung, eingeführt.

Ressource	Beschreibung
R Tutorial	Schritt-für-Schritt Einführung in das Arbeiten mit R von Christian Treffenstädt. Nützlich, falls Sie grundlegende Dinge noch einmal nachschlagen möchten

Tipp der Woche

Mit `strg` + `alt` + `c` (Windows) oder `cmd` + `alt` + `c` (Mac) können Sie direkt den Code-Chunk ausführen, in dem sich Ihr Cursor gerade befindet. Mit `strg` + `alt` + `n` (Windows) oder `cmd` + `alt` + `n` (Mac) führen Sie direkt den nächsten Chunk aus.

1) Daten einlesen

1. Setzen Sie ein sinnvolles Arbeitsverzeichnis für den Übungszettel (in der Regel der Ordner, in dem Ihre .Rmd liegt). Aber Vorsicht: Beim Rendern (Knit) geht RStudio davon aus, dass das Working-Directory das ist, in dem auch die .Rmd-Datei liegt. Dies ist besonders wichtig, wenn es um relative Links geht.
2. Laden Sie den Datensatz starwars.csv herunter (*Rechtsklick > Ziel speichern unter* oder *Rechtsklick > Verknüpfte Datei laden*) und speichern Sie ihn in Ihrem Arbeitsverzeichnis (idealerweise haben Sie noch den Ordner vom letzten Übungszettel - speichern Sie den Datensatz im Unterordner /data).
3. Laden Sie die Pakete des `tidyverse` und fügen Sie eine entsprechende Code-Zeile an den Beginn Ihres Dokuments ein.
4. Lesen Sie den Datensatz `starwars.csv` unter dem Namen `sw_data` in R ein.

Lösung

Unteraufgabe 1 Bitte folgen Sie den Anweisungen im Aufgabentext.

Unteraufgabe 2 Bitte folgen Sie den Anweisungen im Aufgabentext.

```
library(tidyverse)
```

Unteraufgabe 3 Anmerkung: `library()` und `require()` sind beides Befehle zum Laden von Paketen. `require()` ist prinzipiell für die Verwendung innerhalb von Funktionen gedacht. Siehe `?library` oder `?require` für Details.

```
# example code, works only if the local situation permits ...
# sw_data <- read_csv("data/starwars.csv")
# alternative reading from URL
sw_data <- readr::read_csv("https://md.psych.bio.uni-goettingen.de/mv/data/div/starwars.csv")
```

Unteraufgabe 4

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   name = col_character(),
##   height = col_double(),
##   mass = col_double(),
##   hair_color = col_character(),
##   skin_color = col_character(),
##   eye_color = col_character(),
##   birth_year = col_double(),
##   gender = col_character(),
##   homeworld = col_character(),
##   species = col_character()
## )
```

2) Regressionsmodell

0. Schlagen Sie für Erklärungen zur Verwendung von `lm()` in Kapitel 7.4.2 und zur Interpretation des Outputs in Kapitel 7.5 von *Discovering Statistics Using R* (Field, 2012) nach.
1. Erstellen Sie ein Regressions-Modell namens `m_height`, in dem Sie das **Gewicht** durch die **Größe** der Personen im Datensatz vorhersagen. Nutzen Sie dafür die FUnktion `lm()`.
2. Lassen Sie sich eine Zusammenfassung der Analyse mit `summary()` anzeigen.
3. Schreiben Sie mit den Werten aus dem Output aus `summary()` die Regressionsgleichung auf.
4. Interpretieren Sie die Regression
    a. Passt das Modell auf die Daten?
    b. Ist Größe ein signifikanter Prädiktor für Gewicht?

Lösung

```
m_height <- lm(mass ~ height, data = sw_data)
```

Unteraufgabe 1

```
summary(m_height)
```

Unteraufgabe 2

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
##  -61.43  -30.03  -21.13  -17.73 1260.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.8103   111.1545  -0.124    0.902
## height        0.6386     0.6261   1.020    0.312
##
## Residual standard error: 169.4 on 57 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.01792,    Adjusted R-squared:  0.0006956
## F-statistic:  1.04 on 1 and 57 DF,  p-value: 0.312
```

Unteraufgabe 3 Die Allgemeine Form für die Regressionsgleichung kann man z.B. so schreiben:

$$y_i = \beta_0 + \beta_1 \cdot x_i + \epsilon_i$$

Wir können mit unserem Modell nicht die *wahren* Werte von β_0 und β_1 bestimmen, sondern sie nur anhand unserer Daten schätzen. Deshalb gibt es eine extra Gleichung für die Schätzer:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_i$$

Das Dach über den Buchstaben gibt an, dass es sich hier im geschätzte Werte handelt. In der Schätzgleichung, wenn $\hat{y}_i$ auf der linken Seite steht, brauchen wir außerdem kein ϵ_i mehr, denn das Residuum ϵ_i ist der Unterschied zwischen y_i und $\hat{y}_i$, also zwischen vorhergesagtem, geschätztem Outcome und tatsächlichem Outcome.

Jetzt können wir die Werte aus unserem Output von `summary()` in die Gleichung oben eintragen. $\hat{\beta}_0$ ist der Intercept, $\hat{\beta}_1$ ist der Koeffizient für den Prädiktor Größe (`height`).

$$\widehat{mass}_i = -13.81 + 0.64 \cdot height_i$$

Wenn wir nicht das vorhergesagte, geschätzte Gewicht $\widehat{mass}_i$, sondern das echte, gemessene Gewicht $mass_i$ auf die linke Seite schreiben, dann fügen wir wieder das Residuum hinzu:

$$mass_i = -13.81 + 0.64 \cdot height_i + \epsilon_i$$

Beide Schreibweisen sind legitim.

Unteraufgabe 4

Teil a) Relevant ist hier der F-Test am Ende der Ausgabe von `summary()`. In diesem Fall wird der Test nicht signifikant mit $F(1, 57) = 1.04$ und $p = 0.312$. Das heißt, das Modell passt nicht signifikant besser auf die Daten als eines, das nur das mittlere Gewicht der Personen im Datensatz zur Vorhersage benutzt. Das ist das sogenannte Nullmodell, es wird immer als Vergleichspunkt für diesen F-Test genommen.

Teil b) Relevant ist hier der t-Test für den Koeffizienten des Prädiktors "Größe" (`height`) im Output von `summary()`. Der Test wird hier nicht signifikant mit $t(57) = 1.02$ und $p = 0.312$. Das heißt, der Regressionskoeffizient für Größe ist nicht signifikant verschieden von 0.

3) Scatterplot

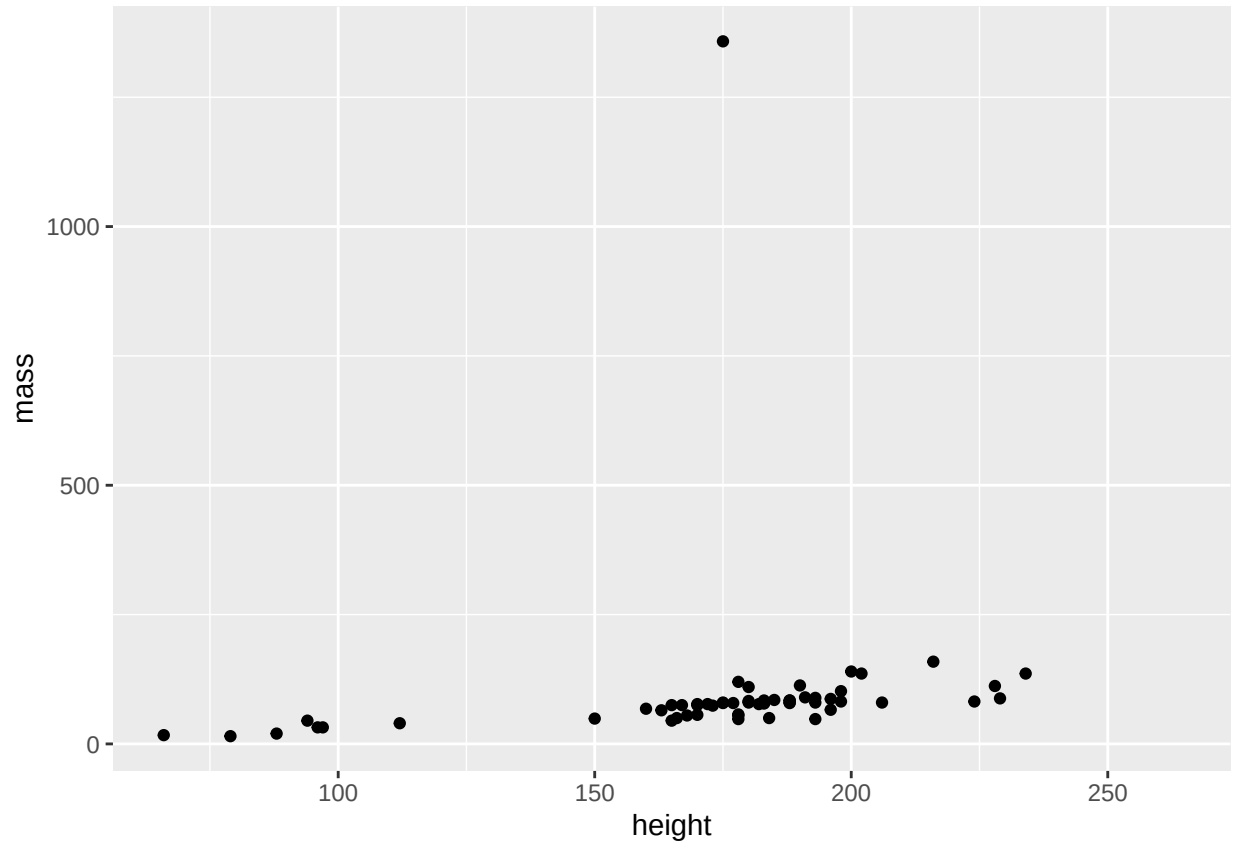
1. Erstellen Sie mit den ggplot-Befehlen, die Sie beim letzten Mal kennengelernt haben, einen Scatterplot. Auf der x-Achse sollte die Größe der Personen stehen, auf der y-Achse das Gewicht (`mass`). Hinweis: Die Werte in `mass` sind in Kilogramm angegeben.
2. Fügen Sie dem Plot eine Regressionsgerade hinzu.
3. Fügen Sie dem Plot einen aussagekräftigen Titel und y-, sowie x-Achsenbeschriftungen hinzu.
4. Geben Sie dem Plot ein *Theme*, das ihn Publikationsreif aussehen lässt.
5. Speichern Sie den Plot als .png Datei in Ihrem Arbeitsverzeichnis.

Lösung

```
# Objekt erstellen
height_plot <- ggplot(sw_data, aes(x = height, y = mass))

# Ebene hinzufügen und direkt mit abspeichern
height_plot <- height_plot + geom_point()

# Plot anzeigen
height_plot
```

Unteraufgabe 1

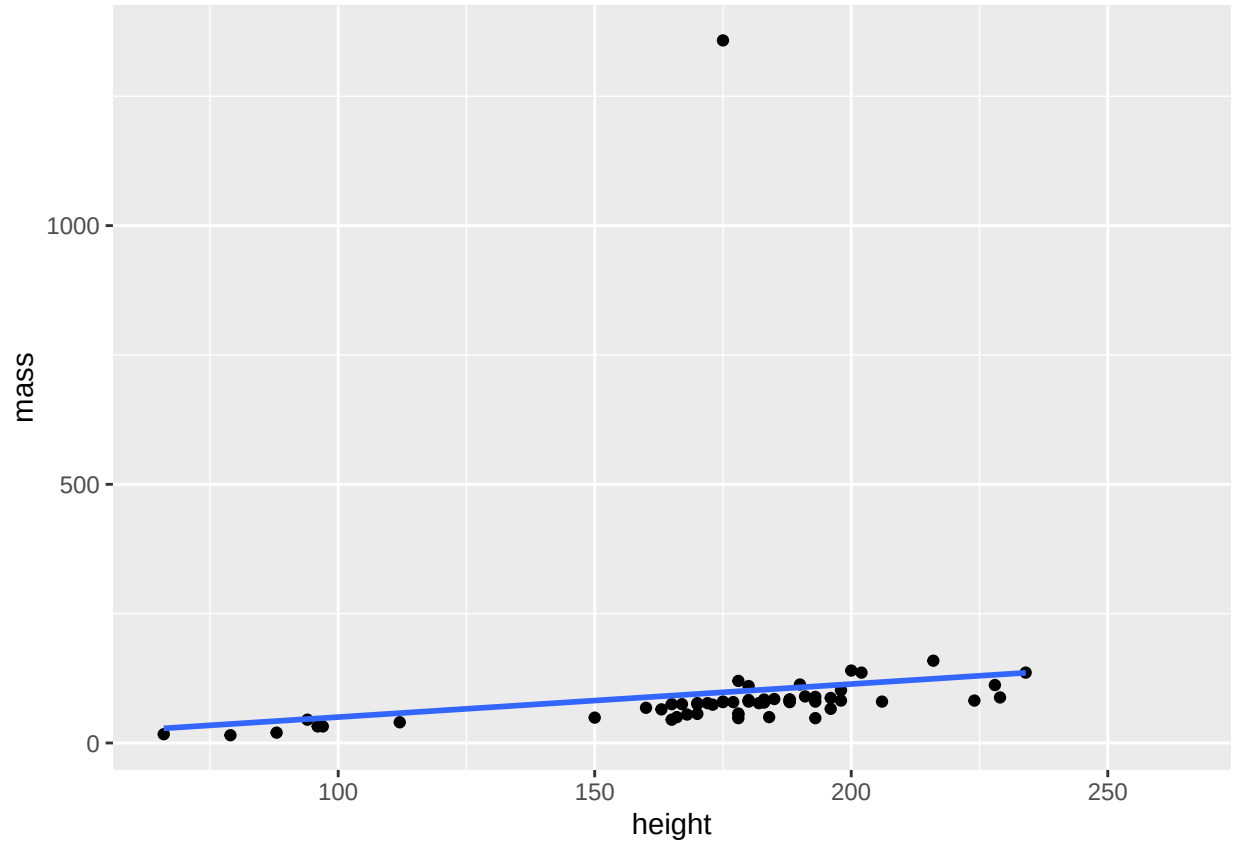
```
## Warning: Removed 28 rows containing missing values (geom_point).
```



```r
# Ebene hinzufügen und direkt mit abspeichern
height_plot <- height_plot + geom_smooth(method = "lm", se = F)

# Plot anzeigen
height_plot
```

Unteraufgabe 2

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 28 rows containing non-finite values (stat_smooth).

## Warning: Removed 28 rows containing missing values (geom_point).
```

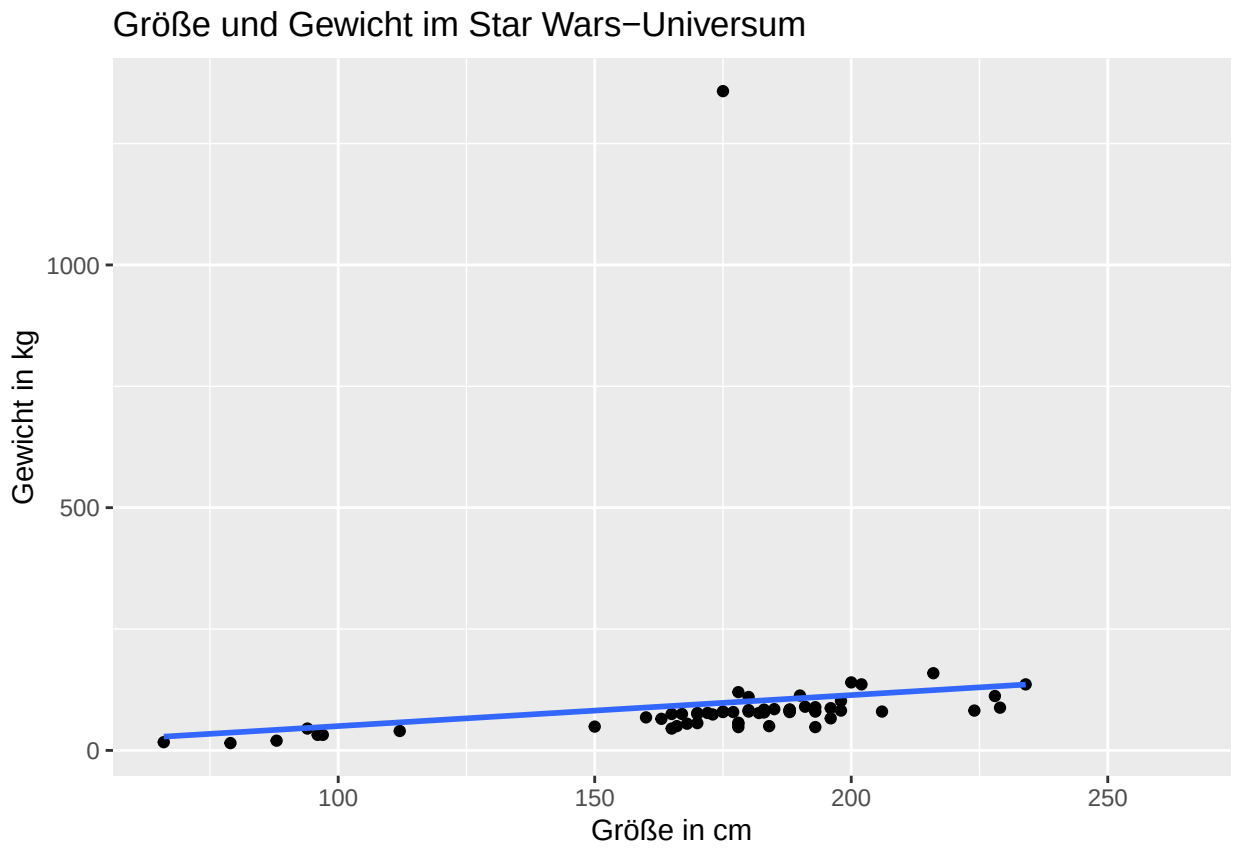



```r
# Ebene hinzufügen und direkt mit abspeichern
height_plot <- height_plot +
  labs(title = "Größe und Gewicht im Star Wars-Universum",
       x = "Größe in cm",
       y = "Gewicht in kg")
```

```
# Plot anzeigen
height_plot
```

Unteraufgabe 3

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 28 rows containing non-finite values (stat_smooth).

## Warning: Removed 28 rows containing missing values (geom_point).
```

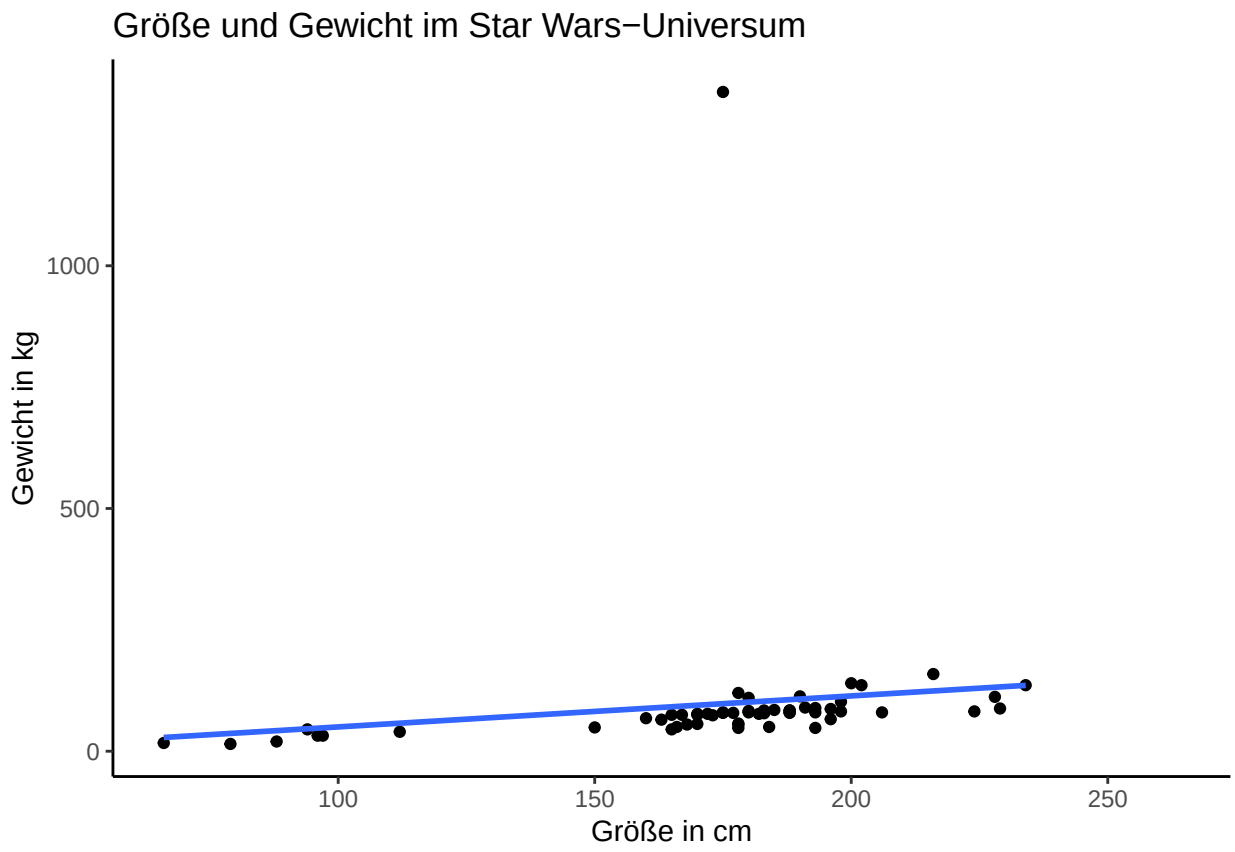


Unteraufgabe 4 Hinweis: Ein anderes *Theme* kann hier auch richtig sein. `theme_classic()` entspricht meinem Eindruck nach am ehesten den APA-Vorgaben.

```
# Ebene hinzufügen und direkt mit abspeichern
height_plot <- height_plot + theme_classic()

# Plot anzeigen
height_plot
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 28 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



Unteraufgabe 5 Hier beachten: Ich habe oben bei den Lösungen in jeder Unteraufgabe den Plot mit der neuen Ebene abgespeichert. Deshalb kann ich nun einfach das Objekt `height_plot` hier verwenden, um das Bild abzuspeichern.

```
ggsave("height_plot.png", height_plot)
```

Hier noch einmal der komplette Code für den Plot:

```
# Objekt erstellen
height_plot <- ggplot(sw_data, aes(x = height, y = mass))

# Ebenen hinzufügen und abspeichern
height_plot <- height_plot +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(title = "Größe und Gewicht im Star Wars-Universum",
       x = "Größe in cm",
       y = "Gewicht in kg") +
  theme_classic()
```

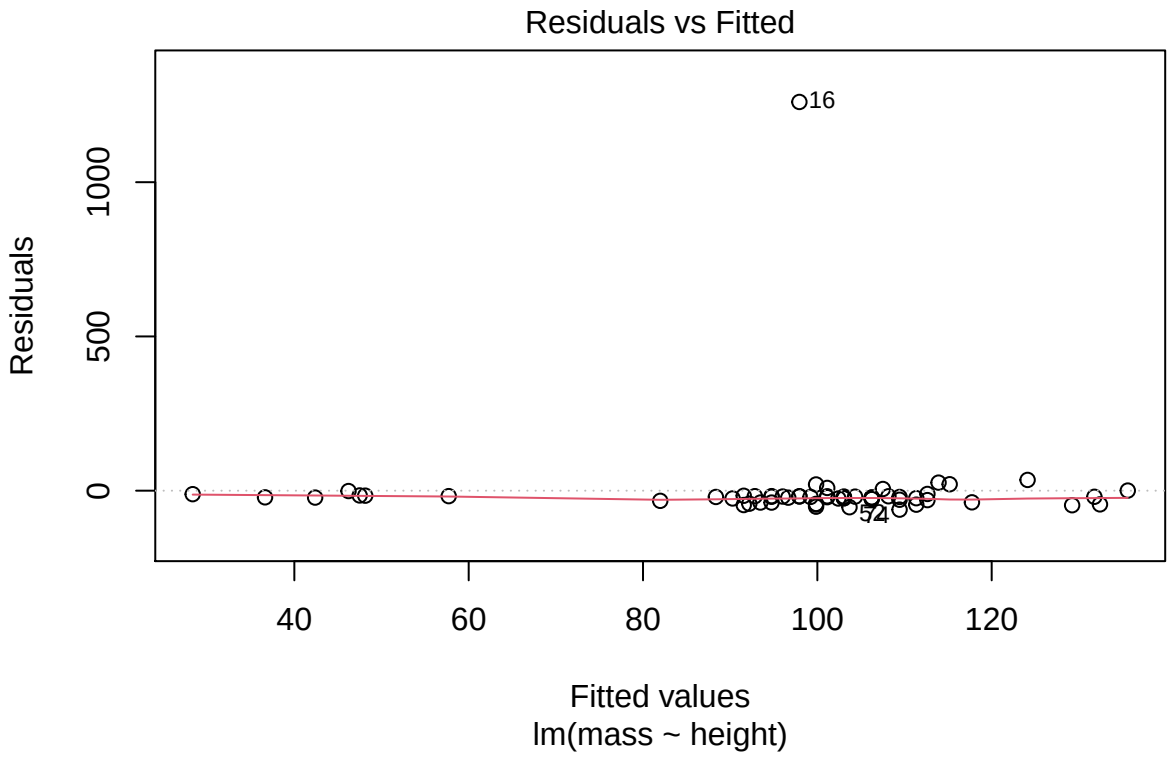
4) Regression überprüfen

Der Plot erzeugt einen Verdacht: Wird die Regression von einem einzigen Extremwert verzerrt? Wir wollen dem weiter auf den Grund gehen.

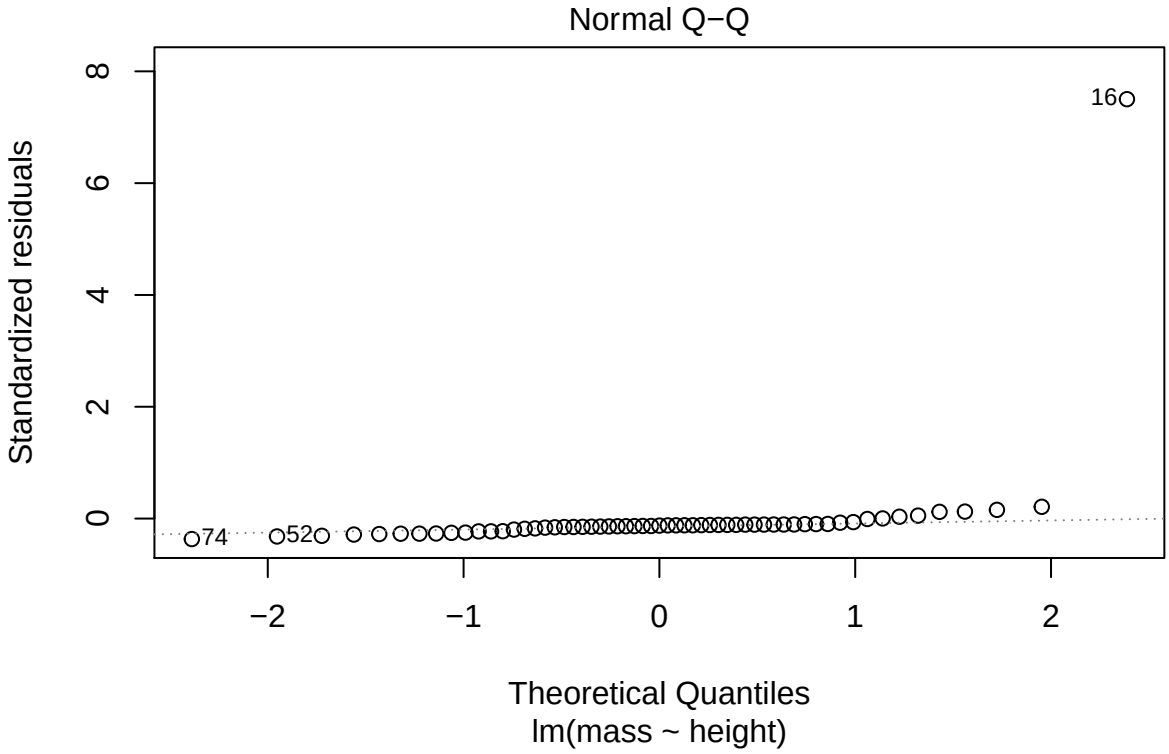
1. Wenden Sie die Funktion `plot()` auf das Regressionsmodell an, das Sie oben erzeugt haben. Folgen Sie nun den Anweisungen in der Konsole. Dort sollten Sie die Aufforderung "Drücke Eingabetaste für den nächsten Plot:" sehen. Insgesamt werden Ihnen nacheinander vier Plots angezeigt.
2. Lesen Sie diesen kurzen Artiekl. Finden Sie die folgenden Dinge für jeden Plot heraus:
    a. Welche Annahmen der Regressionsanalyse (siehe Field, 2012; Kap. 7.7.2.1) können Sie anhand des Plots überprüfen?
    b. Wie sollte der jeweilige Plot aussehen, wenn alles in Ordnung ist?
    c. Welche Muster in den Plots deuten potentiell auf Probleme mit dem Modell hin? Wenn Sie mehr erfahren möchten, oder zusätzliche Informationen benötigen, ist das Kapitel 7.7.1 in *Field (2012): Discovering Statistics Using R* hilfreich.
3. Schauen Sie sich nun noch einmal die vier Plots an. Was fällt Ihnen auf?
4. Identifizieren Sie, zu welcher Person die auffällige Beobachtung gehört. (Hinweis: In den diagnostischen Plots steht neben Extremwerten in der Regel die zugehörige Zeile im Datensatz.)

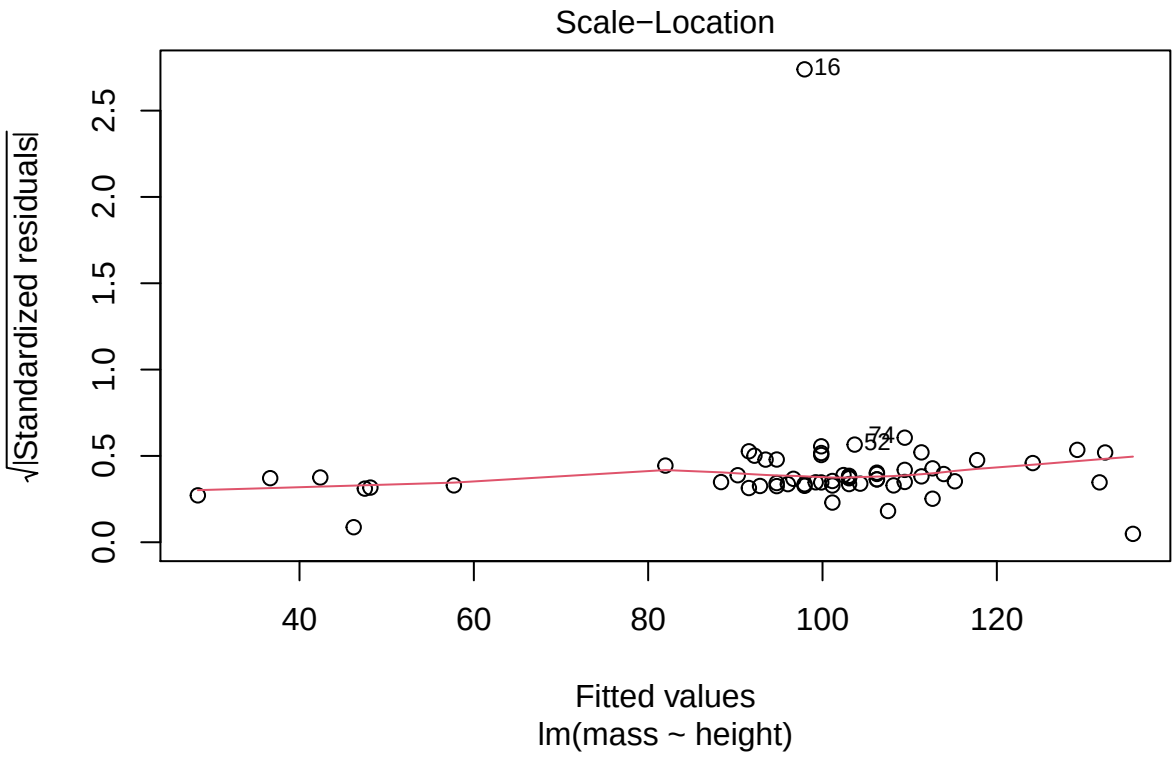
Lösung

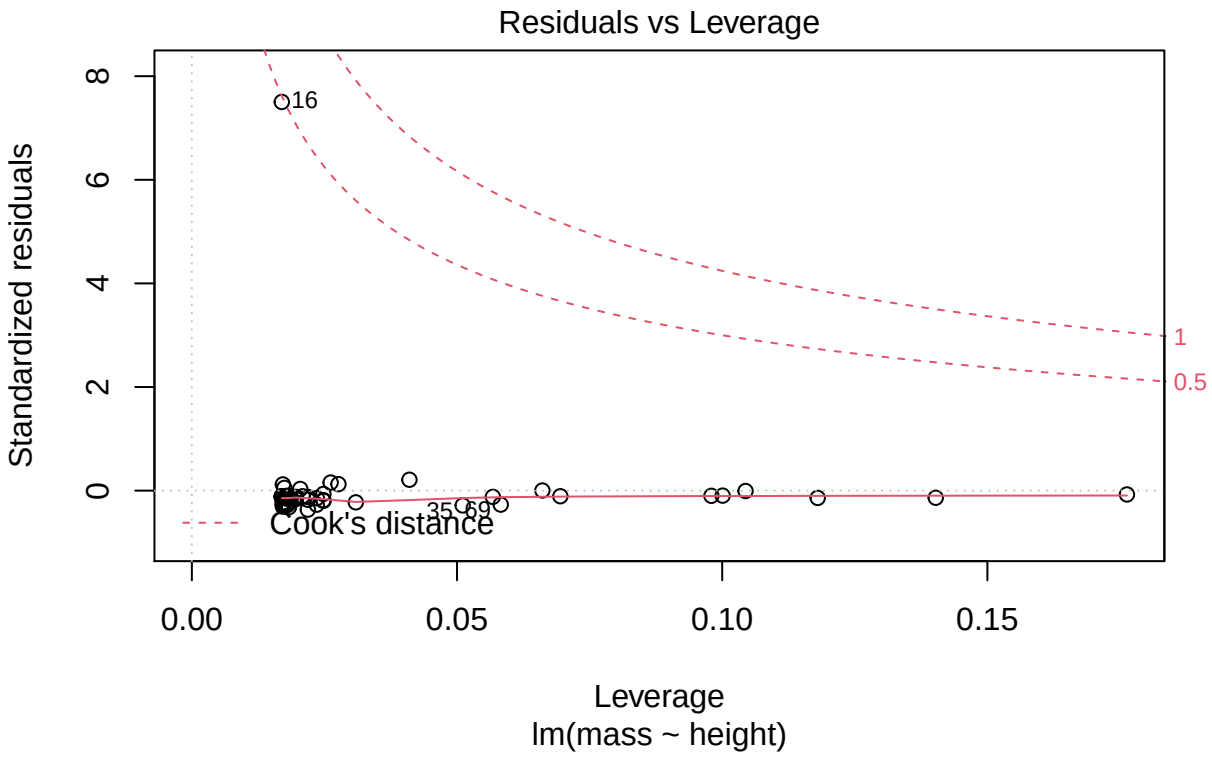
```
plot(m_height)
```



Unteraufgabe 1

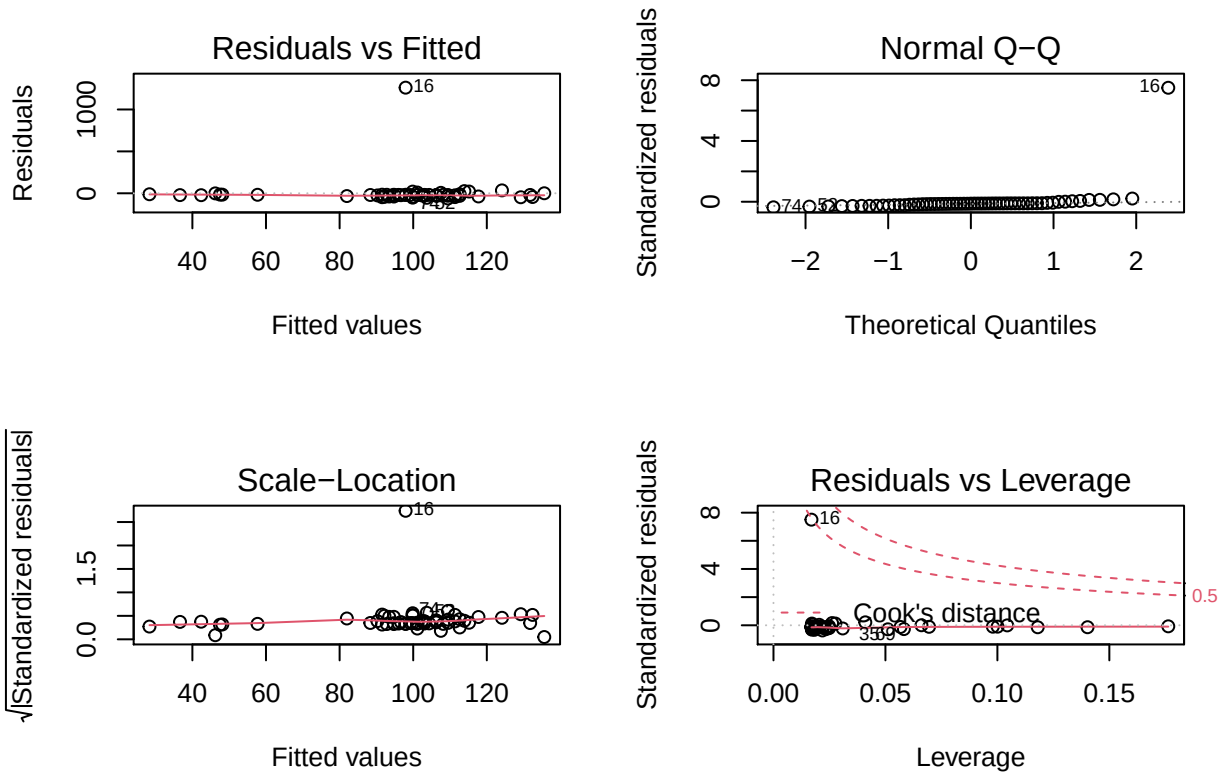



Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
16
74
52
40
60
80
100
120
Fitted values
lm(mass ~ height)



Alternativ zur Darstellung aller Plots auf einmal:

```
par(mfrow = c(2,2))
plot(m_height)
```



Unteraufgabe 2

a. Annahmen

1. Mit dem ersten Plot kann die Annahme eines linearen Zusammenhangs zwischen Prädiktor und Outcome überprüft werden.
2. Mit dem zweiten Plot kann die Annahme normalverteilter Fehler überprüft werden.
3. Mit dem dritten Plot kann die Annahme der Homoskedasdizität überprüft werden.
4. Mit dem vierten Plot wird keine Annahme überprüft. Stattdessen können besonders Einflussreiche Fälle identifiziert werden, die das Modell übermäßig beeinflussen.

b. Alles in Ordnung

1. Bei zufälliger Punktewolke.
2. Bei möglichst gerader Linie.
3. Bei möglichst horizontaler Linie und zufälliger Punktewolke.
4. Wenn keine Werte im oberen rechten oder unteren rechten Bereich, oder jenseits der gestrichelten roten Linien liegen.

c. Hinweis auf Problem

1. Bei deutlichem Muster: Hinweis auf nicht-linearen Zusammenhang zw. Prädiktor und Outcome.
2. Bei deutlichen Abweichungen von der geraden Linie: Hinweis auf nicht-normalverteilte Fehler.
3. Bei deutlichem Muster: Hinweis auf Heteroskedasdizität.
4. Bei Werten besonders weit oben rechts oder unten rechts, oder jenseits der gestrichelten roten Linien: Hinweis auf besonders starken Einfluss dieser Fälle auf das Regressionsmodell.

Unteraufgabe 3 In jedem einzelnen Plot fällt Beobachtung Nr. 16 aus der Reihe. Die anderen Werte sehen gut aus, nur diese Beobachtung scheint stark abzuweichen.

Unteraufgabe 4 Da wir wissen, dass es sich um Beobachtung, also Zeile Nr. 16 handelt, können wir einfach mit eckigen Klammern hinter dem Namen des tibbles nachsehen, was in dieser Zeile steht.

Beachten: Wenn wir nur eine Zeile auswählen wollen, müssen wür trotzdem das Schema `[<zeile>, <spalte>]` einhalten. Das heißt, wir müssen das Komma setzen! Den Wert für die Spalte können wir leer lassen. So bekommen wir z.B. alle Spalten, die in die Konsole passen, für die Zeile 16 ausgegeben.

```
sw_data[16,]
```

```
## # A tibble: 1 x 10
##   name          height  mass hair_color skin_color    eye_color birth_year gender    homeworld speci
##   <chr>          <dbl> <dbl> <chr>      <chr>         <chr>           <dbl> <chr>     <chr>     <chr>
## 1 Jabba Desili~    175  1358 <NA>       green-tan, b~ orange            600 hermaphr~ Nal Hutta Hutt
```

Jabba scheint hier also für Probleme verantwortlich zu sein. Ein Blick in seine Daten zeigt, dass das Sinn macht: Bei einer Körpergröße von 1,75m wiegt er 1,358 Tonnen.

5) Korrigierte Regression

1. Erstellen Sie eine Kopie von `sw_data` namens `sw_data_ex`, in der Sie den problematischen Fall ausschließen, den Sie oben identifiziert haben.
2. Führen Sie erneut eine Regressionsanalyse durch und lassen Sie sich den Output mit `summary()` anzeigen.
3. Erstellen Sie, wie oben, einen Scatterplot mit Regressionslinie, basierend auf den neuen Daten. (Tipp: Sie können hier sehr viel Code wiederverwenden.)
4. Vergleichen Sie die erste und die zweite Regressionsanalyse in Bezug auf
   a. Anteil erklärter Varianz (Multiples R^2)
   b. F-Test für Modellfit
   c. t-Test für den Koeffizienten $\hat{\beta}_1$
5. Interpretieren Sie die Ergebnisse der Analyse.

Lösung

Unteraufgabe 1 Wir können zum Beispiel die Funktion `filter()` für diesen Zweck benutzen. Dafür finde ich hier zunächst den genauen Namen der betroffenen Person heraus. Mit eckigen Klammern hinter dem Namen des tibble-Datensatzes kann ich Zeilen und spalten auswählen. Das funktioniert nach dem Muster `<tibble>[<zeile>, <spalte>]`.

```
sw_data[16,1]
```

```
## # A tibble: 1 x 1
##   name
##   <chr>
## 1 Jabba Desilijic Tiure
```

Jetzt kenne ich den vollen Namen und kann ihn in der `filter()` Funktion einsetzen. Ich benutze `name != "Jabba Desilijic Tiure"`, um mir nur die Beobachtungen (Zeilen) anzeigen zu lassen, bei denen der Name *ungleich* (dafür steht `!=`) *Jabba Desilijic Tiure* ist.

```
sw_data_ex <- sw_data %>% filter(name != "Jabba Desilijic Tiure")
```

Unteraufgabe 2 Modell erstellen

```
m_height_ex <- lm(mass ~ height, data = sw_data_ex)
```

Output anzeigen

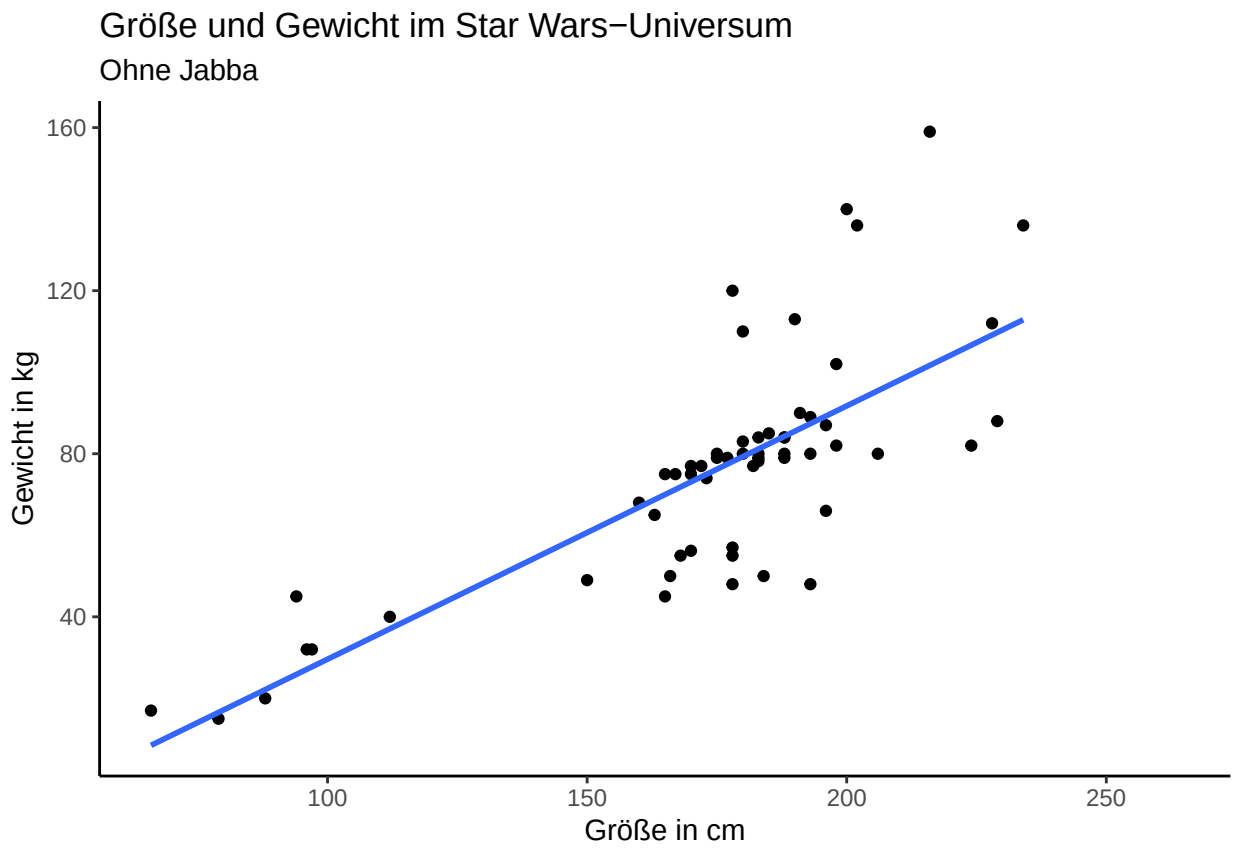
```
summary(m_height_ex)
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data_ex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.382  -8.212   0.211   3.846  57.327
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -32.54076   12.56053  -2.591   0.0122 *
## height        0.62136    0.07073   8.785 4.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.14 on 56 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.5795, Adjusted R-squared:  0.572
## F-statistic: 77.18 on 1 and 56 DF,  p-value: 4.018e-12
```

Unteraufgabe 3 Ich ändere hier im Vergleich zum Code von oben nur... 1. ... den Namen, unter dem das Objekt gespeichert wird. 2. ... den verwendeten Datensatz (`sw_data_ex`) 3. ... den Untertitel, der vorher nicht vergeben war. Hier gibt er an, dass der Plot die Daten von Jabba nicht enthält.

```
# Objekt erstellen
height_plot_ex <- ggplot(sw_data_ex, aes(x = height, y = mass))

# Ebenen hinzufügen und Plot anzeigen
height_plot_ex +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(title = "Größe und Gewicht im Star Wars-Universum",
       subtitle = "Ohne Jabba",
       x = "Größe in cm",
       y = "Gewicht in kg") +
  theme_classic()
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 28 rows containing non-finite values (stat_smooth).

## Warning: Removed 28 rows containing missing values (geom_point).
```



Unteraufgabe 4 Hierfür stellen wir am besten die beiden Outputs gegenüber:

```
summary(m_height) # Modell mit Jabba
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -61.43  -30.03  -21.13  -17.73 1260.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.8103   111.1545  -0.124    0.902
## height        0.6386     0.6261   1.020    0.312
##
## Residual standard error: 169.4 on 57 degrees of freedom
```

```
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.01792,    Adjusted R-squared:  0.0006956
## F-statistic:  1.04 on 1 and 57 DF,  p-value: 0.312
```

```
summary(m_height_ex) # Modell ohne Jabba
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data_ex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.382  -8.212   0.211   3.846  57.327
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -32.54076   12.56053  -2.591   0.0122 *
## height        0.62136    0.07073   8.785 4.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.14 on 56 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.5795, Adjusted R-squared:  0.572
## F-statistic: 77.18 on 1 and 56 DF,  p-value: 4.018e-12
```

Vergleich	Mit Jabba	Ohne Jabba
R^2	0.018	0.589
F-Test	$F(1, 57) = 1.04$, $p = 0.312$	$F(1, 56) = 77.18$, $p < .001$
t-Test für $\hat{\beta}_1$	$t(57) = 1.02$, $p = 0.312$	$t(56) = 8.79$, $p < .001$

a. Während bei Berücksichtigung von Jabba nur 1,8% der Varianz durch das Modell aufgeklärt werden konnte, stieg der Anteil der aufgeklärten Varianz bei Ausschluss von Jabba auf 58,9%.
b. Bei Berücksichtigung von Jabba passte das Modell nicht signifikant besser auf die Daten als das Nullmodell. Wenn Jabba ausgeschlossen wird, liegt eine hochsignifikante Passung vor.
c. Während die Größe bei Einschluss von Jabba kein signifikanter Prädiktor für das Gewicht ist, sagt es bei Ausschluss von Jabba hochsignifikant das Gewicht vorher.

Unteraufgabe 5 Wichtig ist hier die Unterscheidung zwischen **explorativen** und **konfirmatorischen** Ergebnissen. Da wir in unseren Analysen keine vorher definierten Hypothesen festgelegt haben, handelt es sich um explorative Ergebnisse. Diese können wir benutzen, um neue Hypothesen zu generieren, die wir dann in der Folge mit weiteren Experimenten überprüfen. Erst wenn sich in solchen, **konfirmatorischen** Studien die *vorhergesagten* Ergebnisse zeigen, liegt starke Evidenz *für* eine Hypothese oder Theorie vor.

Zunächst einmal wird offensichtlich, dass Jabba ein besonderer Fall ist, für den keine ähnliche Beziehung zwischen Größe und Gewicht gilt, wie für die anderen Charaktere im Datensatz.

Wir könnten daraus z.B. die Hypothese ableiten, dass auch die Spezies ein wichtiger Faktor für das Gewicht eines Individuums ist. Diese Hypothese könnten wir in der Folge durch die Erhebung neuer Daten konfirmatorisch testen.

Aus der Regressionsanalyse ohne Jabba können wir die Hypothese ableiten, dass für viele Charaktere im Star Wars Universum eine enge Beziehung zwischen Größe und Gewicht besteht, die es uns erlaubt, das Gewicht aufgrund der Größe vorherzusagen. Wir könnten nun weitere Daten erheben, um diese explorativen Ergebnisse konfirmatorisch zu untersuchen.

Literatur

Anmerkung: Diese Übungszettel basieren zum Teil auf Aufgaben aus dem Lehrbuch *Dicovering Statistics Using R* (Field, Miles & Field, 2012). Sie wurden für den Zweck dieser Übung modifiziert, und der verwendete R-Code wurde aktualisiert.

Field, A., Miles, J., & Field, Z. (2012). *Discovering Statistics Using R*. London: SAGE Publications Ltd.

English Version

Links

Exercise sheet in PDF

Exercise sheet with solutions included

Exercise sheet with solutions included as PDF

The source code of this sheet as .Rmd (Right click and "store as" to download ...)

Some hints

1. Please give your answers in a .Rmd file. You may generate one from scratch using the file menu: 'File > new file > R Markdown ...' Delete the text below *Setup Chunk* (starting from line 11). Alternatively you may use this sample Rmd by donloading it.
2. You may find the informations useful that you can find on the start page of this course.
3. Don't hesitate to google for solutions. Effective web searches to find solutions for R-problems is a very useful ability, professionals to that too ... A really good starting point might be the R area of the programmers platform Stackoverflow
4. You can find very useful cheat sheets for various R-related topics. A good starting point is the Base R Cheat Sheet.

Ressources

This is a hands on course. We cannot present you all the useful commands in detail. Instead we give you links to useful ressources, where you might find hints to help you with the exercises.

Ressource	Description
Field, Chapter 7 (7.1 - 7.5, 7.9)	Book chapter with a step for step introduction to simple regression and how to do it in R. **Recommendation!**
R for Data Science	Textbook with an introduction to R
Peters Simple Regression Pages	Peters unit on simple regression. A resource to find running examples.
R Tutorial	A step by step introduction to working with R. authored by Christian Treffenstädt. Useful as a reference for basic stuff.

Tip of the week

You can run directly the code chunk, where your cursor is currently in by using shortcut: `strg` + `alt` + `c` (Windows) or `cmd` + `alt` + `c` (Mac). You can run the following chunk with: `strg` + `alt` + `n` (Windows) or `cmd` + `alt` + `n` (Mac).

1) Read data

1. Define an appropriate working directory for this exercise sheet. This should usually be the folder, where your Rmd-file is located. But be careful: The render process always assumes that your working directory is the directory, your Rmd-file is in. This is expecially important if you work with relative links.
2. Load the data starwars.csv and store it in your working directory. You might still have the folder you used for your last sheet - then store the data in a data subdirectory.
3. Assure, that the packages of `tidyverse` are loaded. Insert a code line for that in the beginning of your Rmd-file.
4. Read datafile `starwars.csv` and store it as a data object named `sw_data`.

Solutions

Subtask 1 and 2 Please follow the recommendation of the exercise text.

```
# library(tidyverse)
# or better
require(tidyverse)
```

Subtask 3 Annotation: `library()` and `require()` are both commands to load packages. Require is used often inside functions, as it outputs a warning and continues if the package is not found, whereas library will throw an error. See `?library` or `?require` for details.

```
# syntax would be
# sw_data <- read_csv("data/starwars.csv")
# alternative reading from URL, independent from local situation
sw_data <- readr::read_csv("https://md.psych.bio.uni-goettingen.de/mv/data/div/starwars.csv")
```

Subtask 4

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   name = col_character(),
##   height = col_double(),
##   mass = col_double(),
##   hair_color = col_character(),
##   skin_color = col_character(),
##   eye_color = col_character(),
```

```
##   birth_year = col_double(),
##   gender = col_character(),
##   homeworld = col_character(),
##   species = col_character()
## )
```

2) Regression model

0. See chapter 7.4.2 of *Discovering Statistics Using R* (Field, 2012) for the usage of `lm()` and chapter 7.5 for the interpretation of the results.
1. Create a regression model named `m_height` where you predict **mass** by **height**. Use function `lm()` for that.
2. Take a look at the results using `summary()`.
3. Write down a regression equation using the results.
4. Interprete the results of your model:
    a. How good are your data fitted by the model?
    b. Is heigth a significant predictor of weight?

Solutions

```
m_height <- lm(mass ~ height, data = sw_data)
```

Subtask 1

```
summary(m_height)
```

Subtask 2

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -61.43  -30.03  -21.13  -17.73 1260.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.8103   111.1545  -0.124    0.902
## height        0.6386     0.6261   1.020    0.312
##
## Residual standard error: 169.4 on 57 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.01792,    Adjusted R-squared:  0.0006956
## F-statistic:  1.04 on 1 and 57 DF,  p-value: 0.312
```

Subtask 3 We can write a more general form of a regression equation like this:

$$y_i = \beta_0 + \beta_1 \cdot x_i + \epsilon_i$$

We cannot calculate the *true* values of β_0 and β_1 but we can estimate them using our data. Therefore we have a separate equation for the estimators:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot x_i$$

The hat ˆ above the letters means, that the values are estimated. In the equation where we estimate $\hat{y}_i$ (put left) we do not need ϵ_i because the residuum ϵ_i is the difference between y_i und $\hat{y}_i$ and thus the difference between estimated outcome and real outcome.

We can now insert the values of our `summary()` into the equation above. $\hat{\beta}_0$ is the intercept, $\hat{\beta}_1$ is the coefficient for predictor height (`height`).

$$\widehat{mass}_i = -13.81 + 0.64 \cdot height_i$$

If we don't put the predicted weight $\widehat{mass}_i$ at the left side of the equation but the real weight $mass_i$ we have to add the residuals also:

$$mass_i = -13.81 + 0.64 \cdot height_i + \epsilon_i$$

Both posibilities are o.k.

Subtask 4

Part a) The F-Test at the end of the `summary()` is relevant. In our case the test with $F(1, 57) = 1.04$ and $p = 0.312$ doesn't reach the usual significance level of 5%. This means that our model cannot predict the data better than a model that only uses the mean as a predictor. This is the so called null model and is always use as a comparison for the fitted model.

Part b) Here the relevant part is the t-test for the coefficient of predictor `height` in the output of the `summary()`. This test is not significant with $t(57) = 1.02$ and $p = 0.312$. This means, the regression coefficient fo `height` is not significantly different from 0.

3) Scatterplot

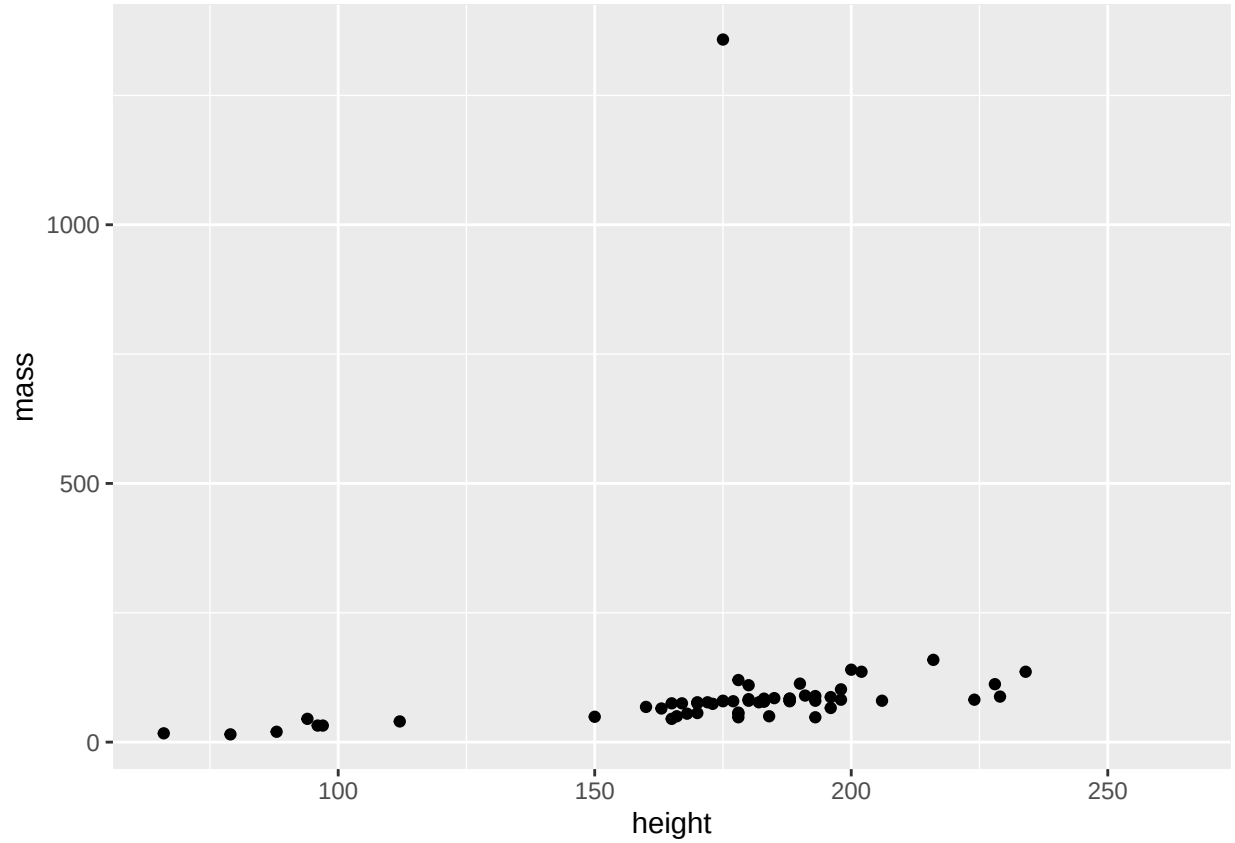
1. Create a scatterplot using the ggplot commands you already know. Show height on the x-axis and weight on the y-axis. Hint: the values in `mass` are kg.
2. Add a regression line.
3. Add meaningful titles and axis labels.
4. Give your plot a *theme* to make it compliant with publication rules.
5. Store your plot in format .png in your working directory.

Solutions

```r
# create objekt
height_plot <- ggplot(sw_data, aes(x = height, y = mass))

# add a layer that is storede with the plot object
height_plot <- height_plot + geom_point()

# show plot
height_plot
```

Subtask 1

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



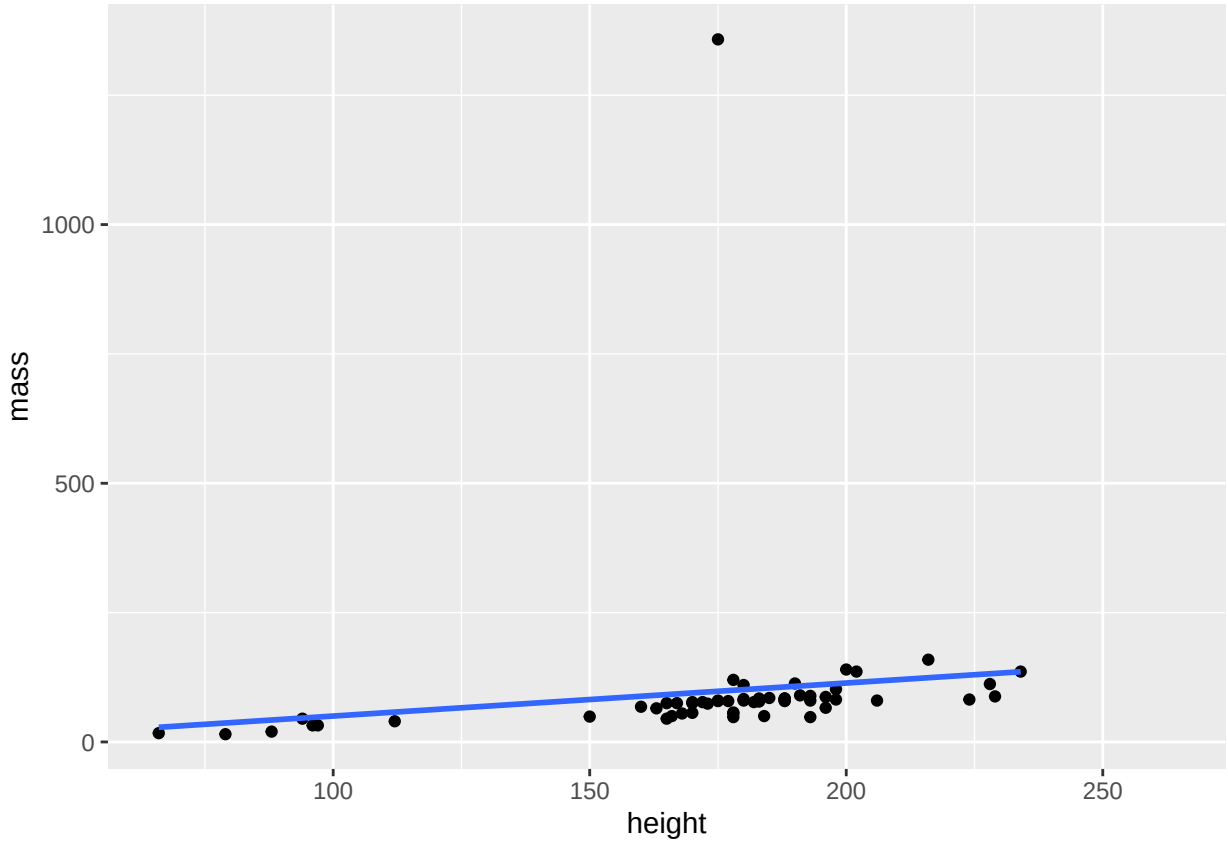

```r
# add layer and store it directly using the same object name
height_plot <- height_plot + geom_smooth(method = "lm", se = F)

# show the stored plot object
height_plot
```

Subtask 2

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 28 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



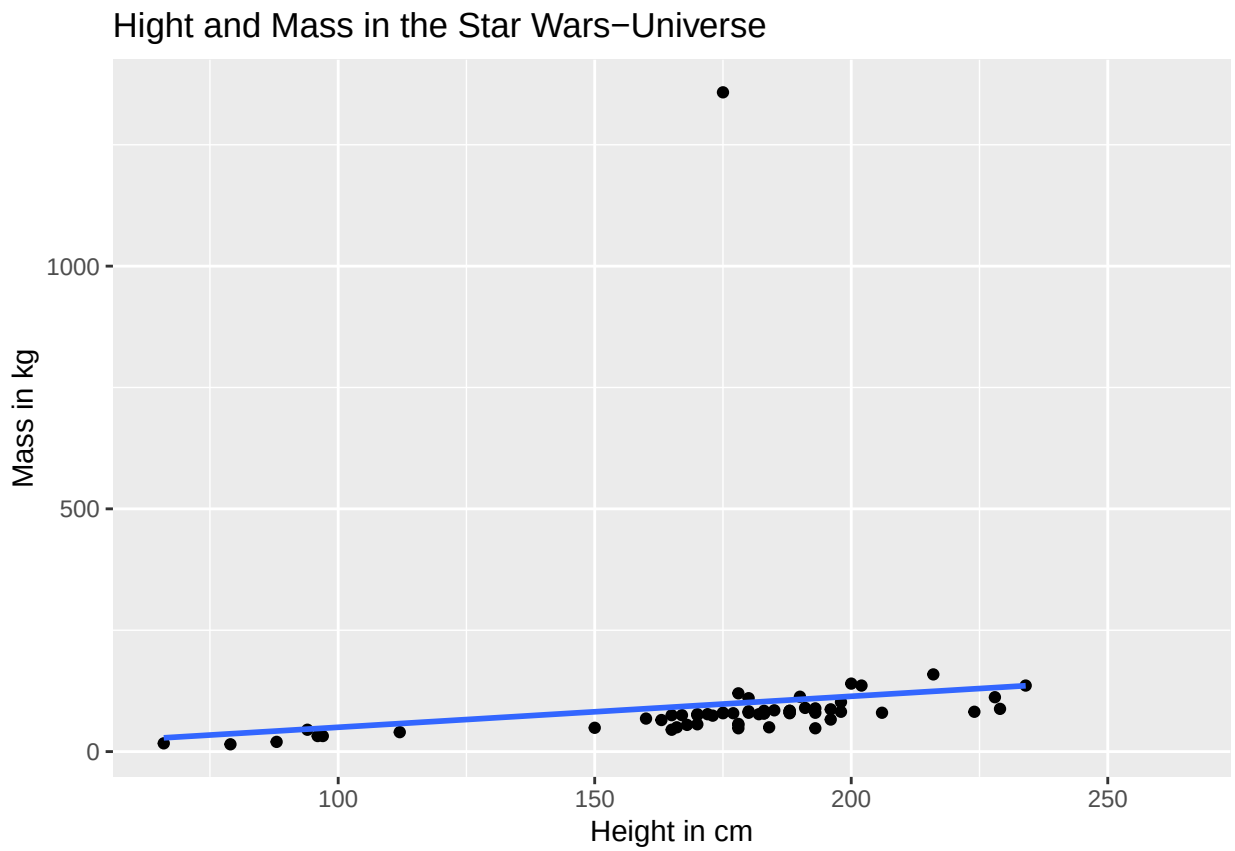

```
# add layer and store it directly using the same object name
height_plot <- height_plot +
  labs(title = "Hight and Mass in the Star Wars-Universe",
       x = "Height in cm",
       y = "Mass in kg")

# show plot
height_plot
```

Subtask 3

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 28 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



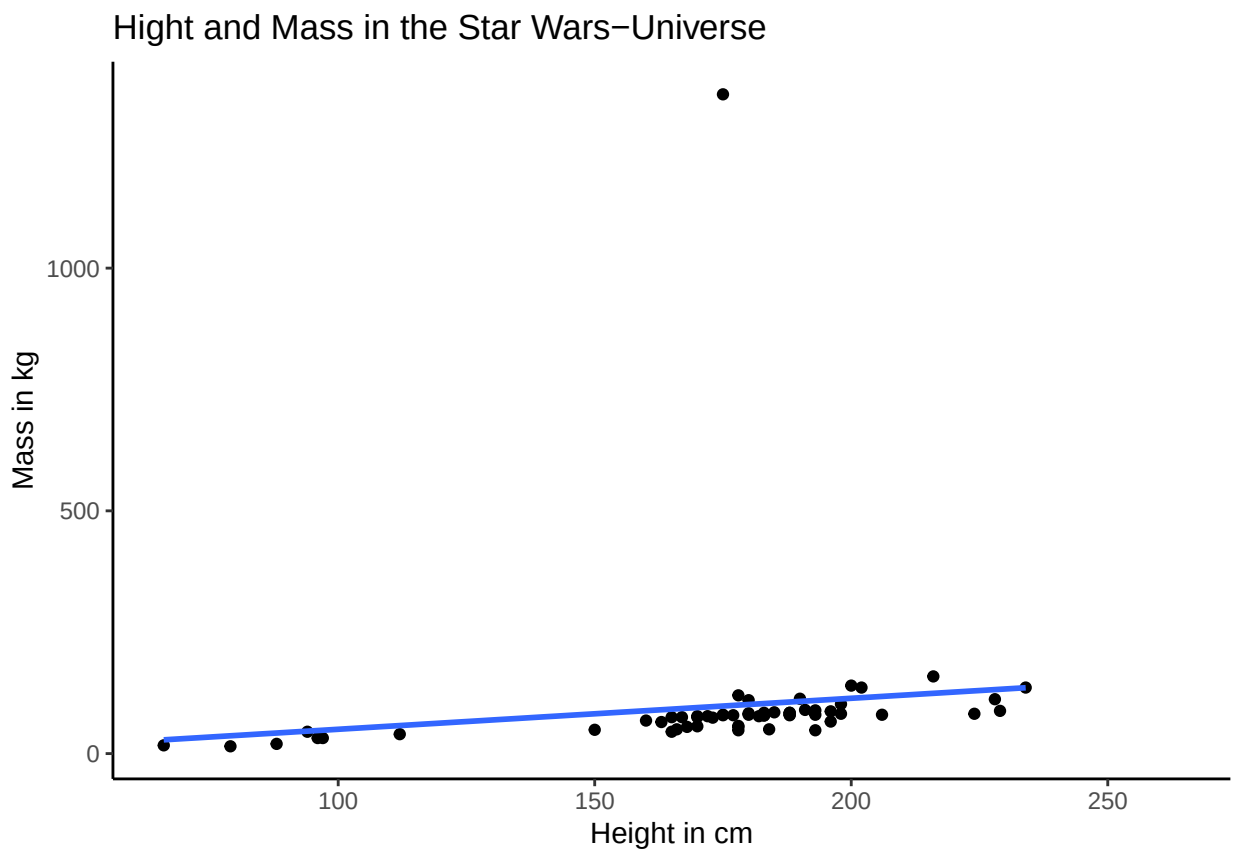
Subtask 4 Hint: There might be more *themes* to be used. To our opinion `theme_classic()` is the one that most fits the APA guidelines.

```
# add layer and store it directly using the same object name
height_plot <- height_plot + theme_classic()

# show plot
height_plot
```

```
## `geom_smooth()` using formula 'y ~ x'
```

```
## Warning: Removed 28 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 28 rows containing missing values (geom_point).
```



Subtask 5 Take care: In the above parts the plot was overwritten at each step. Therefore we can store simply the plot object `height_plot`.

```
ggsave("height_plot.png", height_plot)
```

Here again the whole syntax

```
# create plot object
height_plot <- ggplot(sw_data, aes(x = height, y = mass))

# add layers and overwrite object
height_plot <- height_plot +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(title = "Hight and Mass in the Star Wars-Universe",
       x = "Heigth in cm",
       y = "Mass in kg") +
  theme_classic()
```

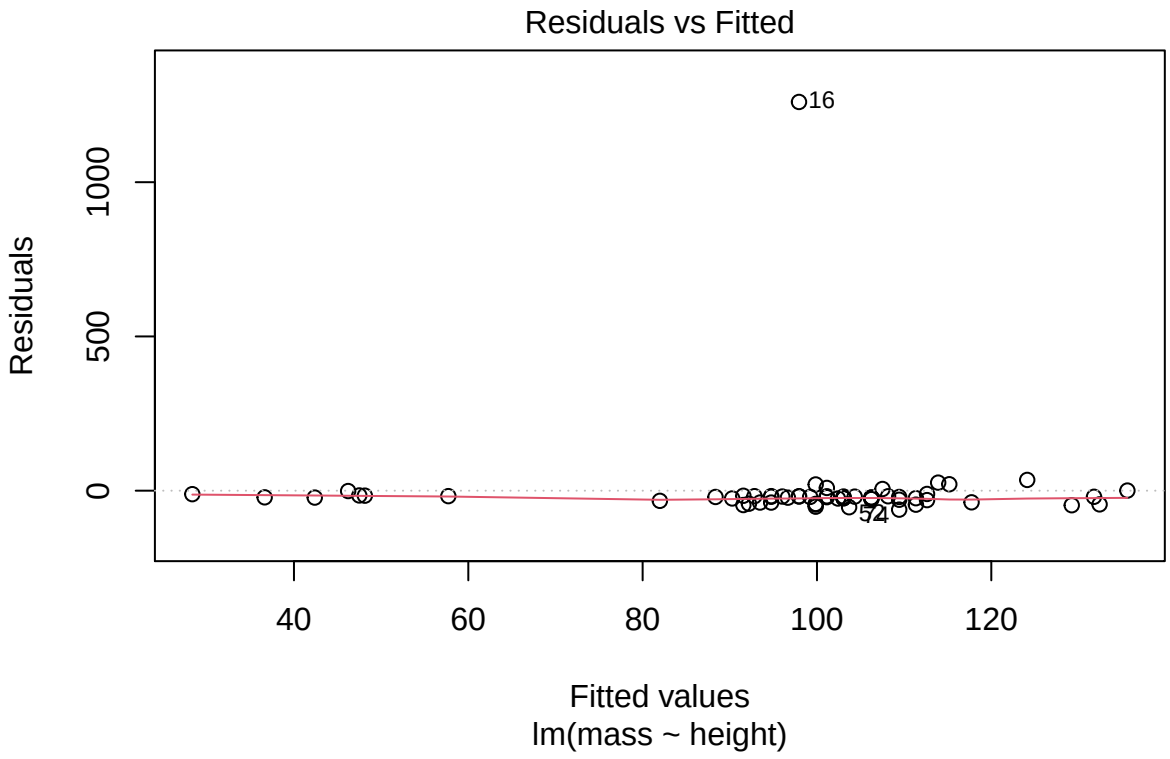
4) Check regression quality

From the plot we suspect, that only one single outlier might have had too much influence on our results. We want to clarify that.

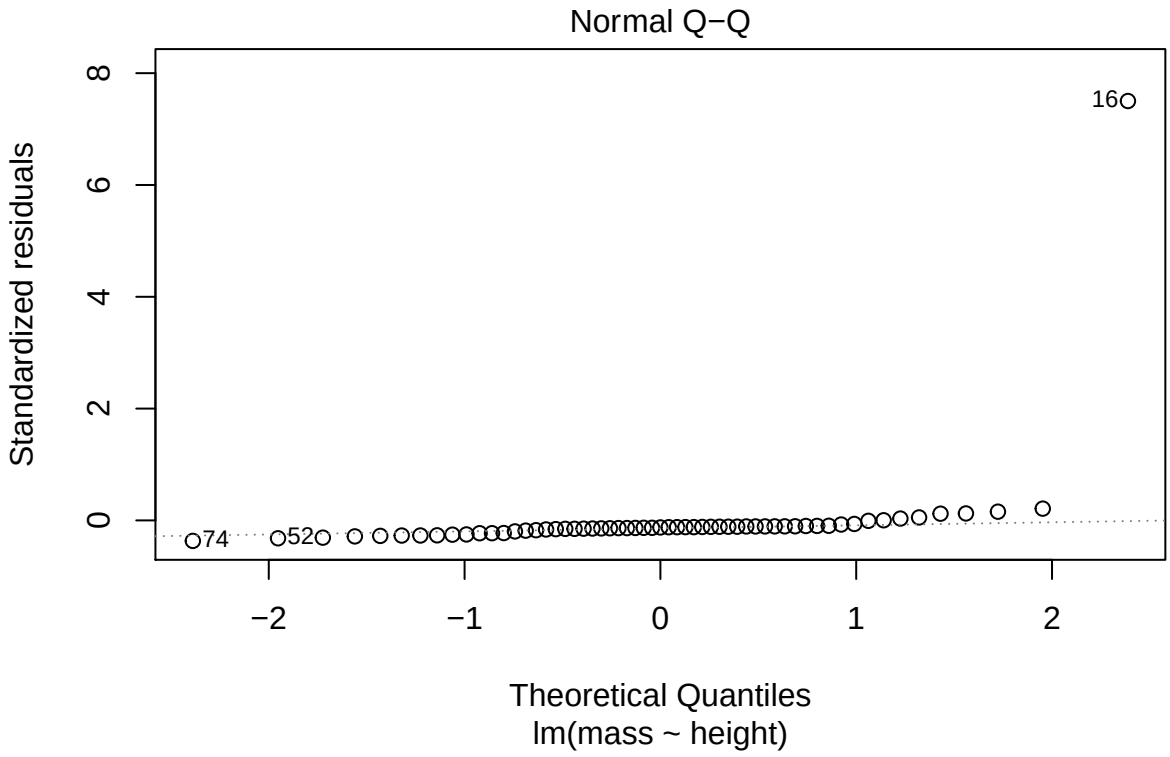
1. Apply `plot()` to the above generated regression model. Follow the hints on the console. There you should find "press enter for the next plot". You will see four plots in total, one by one.
2. Read this short article. Find out:
    a. Which preconditions of regression can be checked with these diagnostic plots? (see Field, 2012; chap. 7.7.2.1)
    b. What should each plot look like if everything looks good?
    c. What patterns indicate potential problems in the fitted model? If you want to learn more or if you need additional information please refer to chapter 7.7.1 of *Field (2012): Discovering Statistics Using R* hilfreich.
3. Take another look to the four plots. What is special?
4. Identify the outlier (hint: You may find the corresponding line in the data in the diagnostic plots)

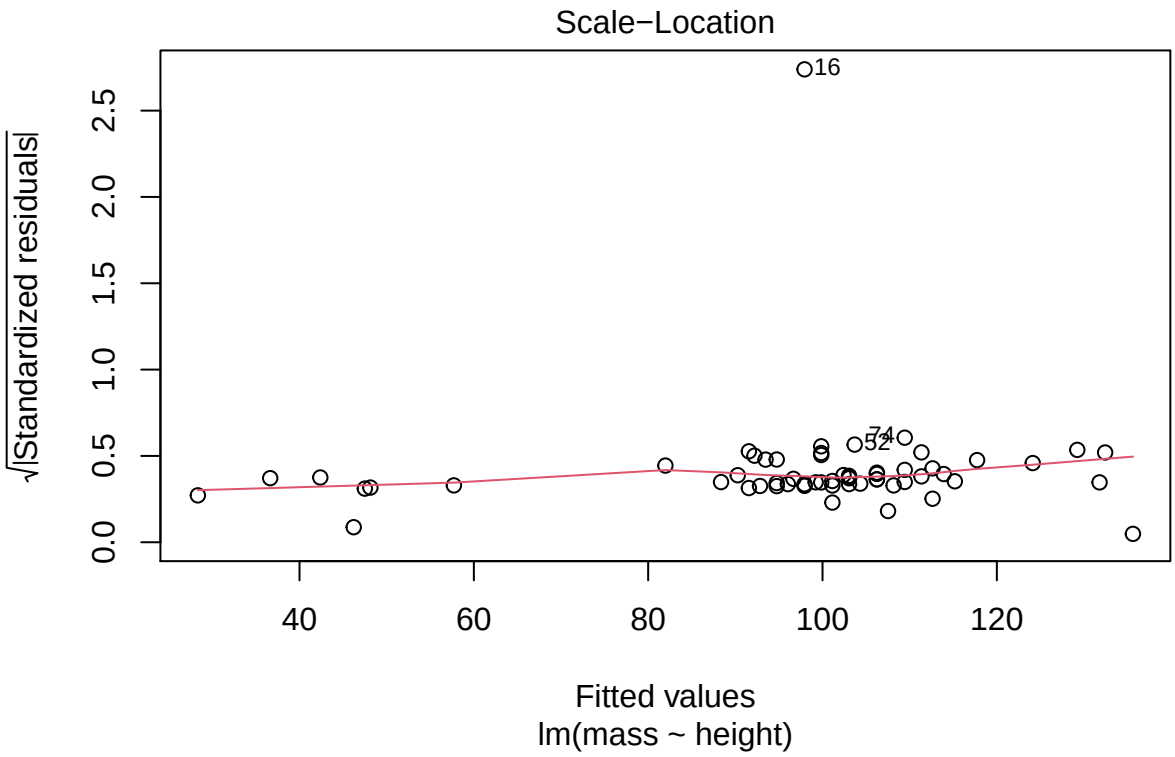
Solutions

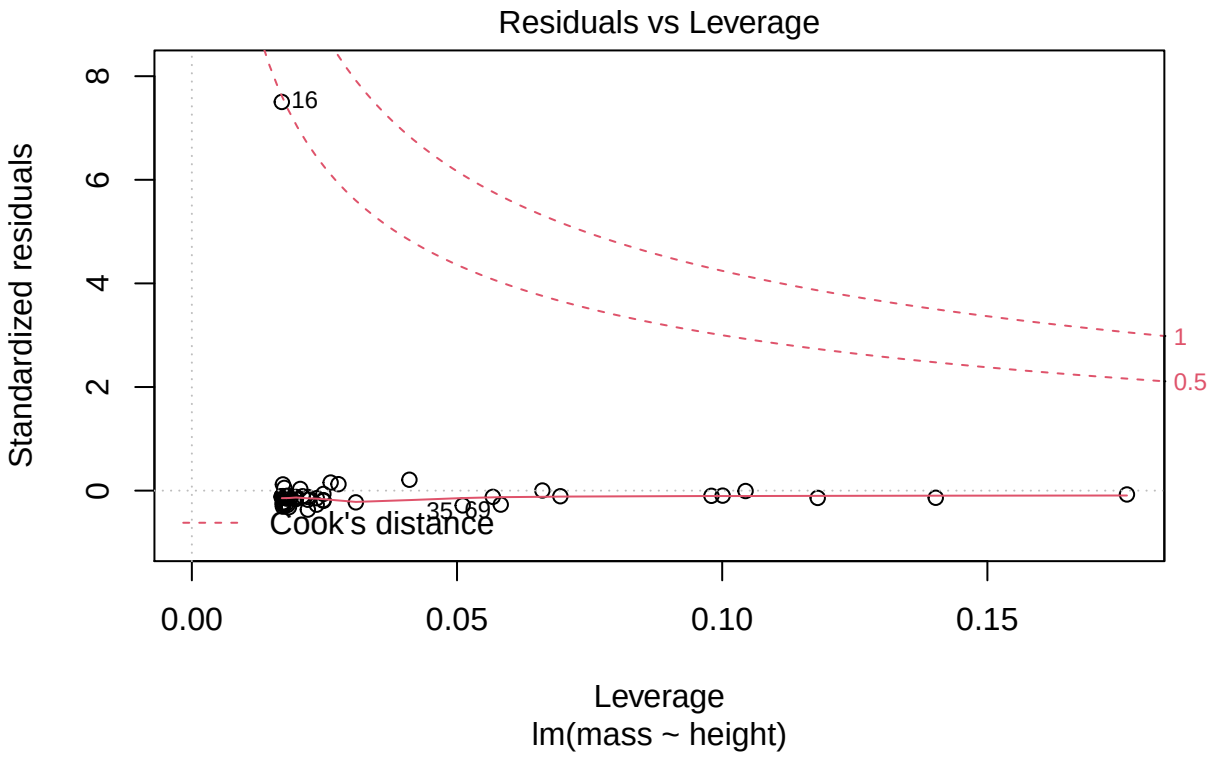
```
plot(m_height)
```


Residuals vs Fitted
16
Residuals
1000
500
0
40
60
80
100
120
Fitted values
lm(mass ~ height)

Subtask 1

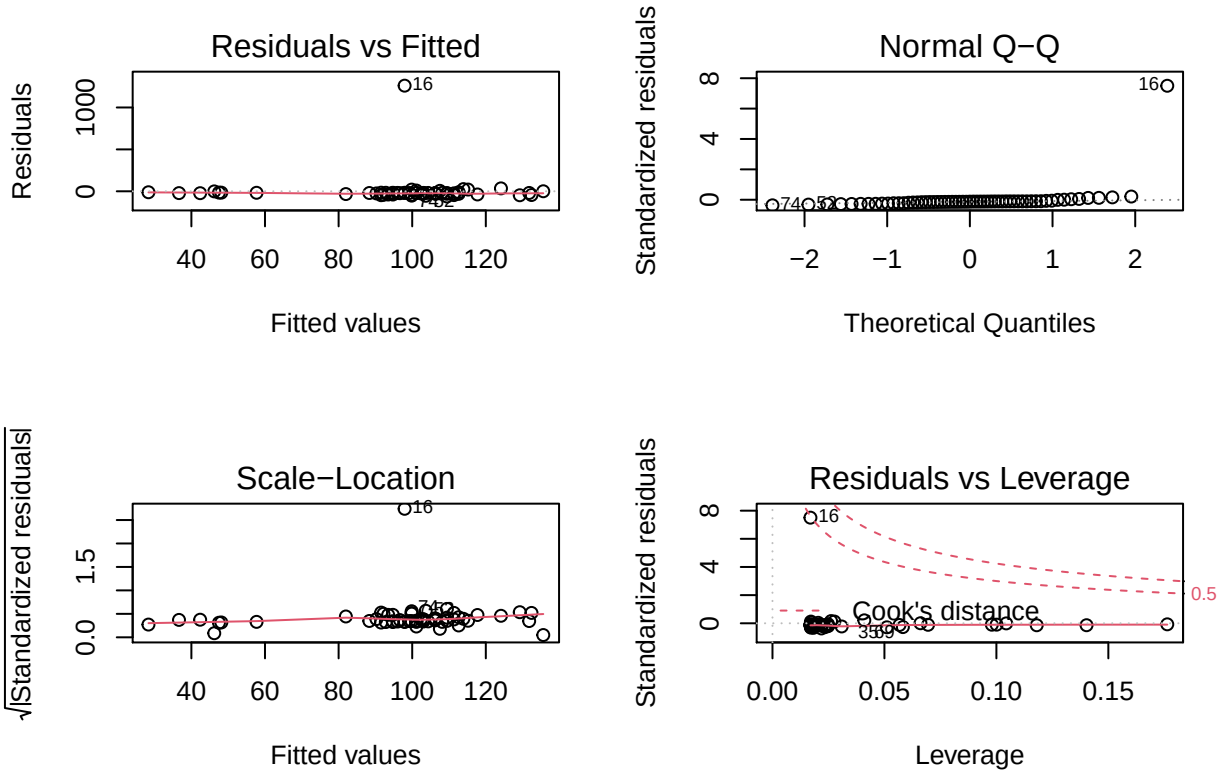

Normal Q−Q
16
74
52
Standardized residuals
8
6
4
2
0
−2
−1
0
1
2
Theoretical Quantiles
lm(mass ~ height)


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
16
74
52
40
60
80
100
120
Fitted values
lm(mass ~ height)



Alternatively we can see all plots at once:

```
par(mfrow = c(2,2))
plot(m_height)
```



Subtask 2

a. Precoditions

1. With the first plot we can check the linear correlation of predictor and outcome.
2. With the second plot we can check the assuption of normally distributed errors.
3. With the third plot we can check the assumption of homoskedacity.
4. The fourth plot doesn't check assumptions, it helps to identify influencial outliers instead.

b. Everything ok is ...

1. when we have a random distribution of the points
2. when we have a almost straight line
3. when we have a horizontal line and randomly scattered points around
4. when we don't have observations that show up in the upper right or lower right area outside the dotted red lines.

c. We might suspect problems if ...

1. we see an apparent pattern that indicates a non linear relation between predictor and outcome.
2. we see an apparent diversion from a straiht line, that might indicate that errors are not normally distributed.
3. we see a pattern that might indicate heteroskedacity
4. values very much to the upper right or lower right outside the dotted red line might indiccate a too influencial impact of an observation on the regression model.

Subtask 3 In all of the plots observation no 16 is special. All the other values look good.

Subtask 4 We know, that observation 16 is problematic. We can check this observation using the slicing method (square brackets) to check what we have in this line.

Take care: Although we want to see a single line only we need to use the method `[<lines>, <columns>]`. Therefore we need to put the comma. We may leave the value for columns empty. So the default shows us all columns that fit to the console.

```
sw_data[16,]
```

```
## # A tibble: 1 x 10
##   name          height  mass hair_color skin_color    eye_color birth_year gender    homeworld speci
##   <chr>          <dbl> <dbl> <chr>      <chr>         <chr>          <dbl> <chr>     <chr>     <chr>
## 1 Jabba Desili~    175  1358 <NA>       green-tan, b~ orange           600 hermaphr~ Nal Hutta Hutt
```

Jabba seems to be responsable for the problems. Looking at his data we see, that this makes sense: With a height of 1.75 m his weight is 1.358 tons.

5) Modified regression

1. Make a copy of `sw_data` named `sw_data_ex` where you exclude the problematic observation.
2. Repeat your regression analysis and check the output of `summary()`.
3. Make a scatterplot, like above, based on the new data. Hint: you may reuse a lot of the code above.
4. Compare the first and the second regressin analysis with respect to
    a. percentage of explained variance (multiple R^2)
    b. F-test for model fit
    c. t-test for the coefficient $\hat{\beta}_1$
5. Interprete the results of your new analysis.

Solutions

Subtask 1 We may use the command `filter()` for this purpose. We would then need the exact name of the observation in question. We could choose the data using square brackets (slicing) via `<tibble>[<rows>, <columns>]`

```
sw_data[16,1]
```

```
## # A tibble: 1 x 1
##   name
##   <chr>
## 1 Jabba Desilijic Tiure
```

Now, that we know the full name, we could use it in the command `filter()`. We use `name != "Jabba Desilijic Tiure"` to get all the observations (rows) where the name is *not* equal *Jabba Desilijic Tiure* (this is, what `!=` stands for).

```
sw_data_ex <- sw_data %>% filter(name != "Jabba Desilijic Tiure")
```

Subtask 2 Fit model

```
m_height_ex <- lm(mass ~ height, data = sw_data_ex)
```

Show output

```
summary(m_height_ex)
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data_ex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.382  -8.212   0.211   3.846  57.327
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -32.54076   12.56053  -2.591   0.0122 *
## height        0.62136    0.07073   8.785 4.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.14 on 56 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.5795, Adjusted R-squared:  0.572
## F-statistic: 77.18 on 1 and 56 DF,  p-value: 4.018e-12
```

Subtask 3 The only changes we need compared to the code above is ... 1. ... the name of the data object is changed 2. ... the data in use (`sw_data_ex`) 3. ... the subtitle, that we didn't use before. Here we indicate, that this plot excludes the data of Jabba.

```
# create object
height_plot_ex <- ggplot(sw_data_ex, aes(x = height, y = mass))

# add layers and overwrite object
height_plot <- height_plot +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(title = "Hight and Mass in the Star Wars-Universe",
       subtitle = "Jabba excluded",
       x = "Heigth in cm",
       y = "Mass in kg") +
  theme_classic()
```

Unteraufgabe 4 Best if we view the two outputs in parallel:

```
summary(m_height)    # Model with Jabba
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -61.43  -30.03  -21.13  -17.73 1260.06
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -13.8103   111.1545  -0.124    0.902
## height        0.6386     0.6261   1.020    0.312
##
## Residual standard error: 169.4 on 57 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.01792,    Adjusted R-squared:  0.0006956
## F-statistic:  1.04 on 1 and 57 DF,  p-value: 0.312
```

```
summary(m_height_ex) # Model without Jabba
```

```
##
## Call:
## lm(formula = mass ~ height, data = sw_data_ex)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -39.382  -8.212   0.211   3.846  57.327
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -32.54076   12.56053  -2.591   0.0122 *
## height        0.62136    0.07073   8.785 4.02e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.14 on 56 degrees of freedom
##   (28 observations deleted due to missingness)
## Multiple R-squared:  0.5795, Adjusted R-squared:  0.572
## F-statistic: 77.18 on 1 and 56 DF,  p-value: 4.018e-12
```

Comparison	with Jabba	without Jabba
R^2	0.018	0.589
F-Test	$F(1, 57) = 1.04$, $p = 0.312$	$F(1, 56) = 77.18$, $p < .001$
t-Test for $\hat{\beta}_1$	$t(57) = 1.02$, $p = 0.312$	$t(56) = 8.79$, $p < .001$

a. With Jabba included the model explains only 1.8 % of the variance, without him the model explains 58.9 %.
b. With Jabba included the model is not better than the null model, without him the fit is significantly better.
c. Height is no significant predictor for weight, if Jabba is included, whereas it is highly significiant without Jabba.

Subtask 5 It is important to distinguish between **exploratory** results and **confirmatory** ones. As we did not have hypotheses beforehead, we have exploratory results here. We can use them, to generate new

hypotheses, that we can check afterwards in a new series of experiments. If we find then the **predicted** effects in such a **confirmatory** study, this is good evidence **in favor** of a hypothesis or theory.

It is obvious, that Jabba is a special case. For him there is no such a relation of height and weight as for the other observations in the dataset.

So we could derive the hypothesis, that species is also an important factor for the weight of an individual. This hypothesis could be checked confirmatively with newly collected data.

From the regression without Jabba, we can derive the hypothesis, that there is a close relation between weight and height for a lot of characters in the Star Wars Universe. So we could assume, that we could predict weight from height in such a population. Now we can collect new data to check this explorative hypothesis confirmatorily.

Literature

Annotation: This exercise sheet bases in part on exercises, that you can find in the textbook *Dicovering Statistics Using R* (Field, Miles & Field, 2012). They were modified for the purpose of this sheet and the R-code was actualized.

Field, A., Miles, J., & Field, Z. (2012). *Discovering Statistics Using R*. London: SAGE Publications Ltd.