Übungszettel Logistische Regression, Exercise Sheet Logistic Regression

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

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

06 Mai, 2021 23:22

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 (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

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 8	Buchkapitel, das Schritt für Schritt erklärt, worum es geht, und wie man logistische Regressionen in R durchführt. **Große Empfehlung!**

Tipp der Woche

Wenn Sie Zeilenumbrüche in Ihrem Code verwenden, ist er für Sie und Andere deutlich besser lesbar. Sie können z.B. nach jeder Pipe (`%>%`) oder nach dem `+` in ggplot2-Befehlen sinnvolle Zeilenumbrüche einbauen. Zögern Sie auch nicht, längere einzelne Befehle in mehrere Zeilen umzubrechen: Nach jedem Komma kann eine neue Zeile beginnen, wenn Sie es möchten.

Es kann sein, dass die Einrückung bei mehrzeiligen Befehlen irgendwann unordentlich wird. Wenn das passiert, markieren Sie den Code und drücken `strg` + `i` (windows), bzw. `cmd` + `i` (mac). Ihr Code wird dadurch autamtisch eingerückt.

1 Daten einlesen

1. Laden Sie das `tidyverse` und fügen Sie die entsprechende Code-Zeile an den Anfang Ihrer .Rmd-Datei ein.
2. Setzen Sie ein sinnvolles Arbeitsverzeichnis und fügen Sie die entsprechende Code-Zeile an den Anfang Ihrer .Rmd-Datei ein.
3. Laden Sie den Datensatz `stats_lectures` unter diesem Link herunter und speichern ihn in Ihrem Arbeitsverzeichnis, dort z.B. in einem Unterordner `data`.
4. Lesen Sie den Datensatz in R ein.

Erklärung der Variablen

Jede Zeile enthält Daten von einem/r StudentIn. Beobachtet wurde, ob die Studierenden an Statistik-Lehrveranstaltungen teilnehmen. Erhoben wurden dazu die Temperatur, wie sehr die Studis Sonne genießen und wie sehr sie Statistik genießen. Letzteres wurde mit einem Fragebogen gemessen, der aus 5 Items besteht.

Hinweis: Die Daten sind vollständig fiktiv und spiegeln natürlich auch nicht die Meinung der Lehrkräfte wieder. Dem Tutor ist nur gerade kein besseres Beispiel eingefallen.

Name	Bedeutung
attend_or_not	Hat zwei Ausprägungen: “course not attended” und “course attended”. Die Ausprägungen geben an, ob der/die Studi zur Lehrveranstaltung gegangen ist.
temperature	Die Temperatur. Höher ist heißer.
sun_joy	Wie sehr genießen die Studis Sonne. Höher ist stärkerer Genuss
stats_joy1 bis stats_joy5	Wie sehr genießen die Studis Statistik. Höher ist stärkerer Genuss.

Lösung

```
library(tidyverse)
```

Unteraufgabe 1

Unteraufgabe 2 Beachten Sie hier, dass in R alle Ordner mit `/` getrennt werden. Deshalb können Sie Windows-Dateipfade nicht einfach kopieren, diese sind mit \ getrennt. Ersetzen Sie dort \ durch `/`.

Der Code ist:

```
setwd()
```

Hier ein Beispiel mit meinem Dateipfad.

```
# Beispiel
setwd("~/ownCloud/_Arbeit/Hiwi Peter/gitlab_sheets/public")
```

Unteraufgabe 3 Bitte folgen Sie den Anweisungen in der Aufgabenstellung.

```
#stats_lectures <- read_csv("public/mv/data/stats_lectures.csv")
stats_lectures <- read_csv("http://md.psych.bio.uni-goettingen.de/mv/data/div/stats_lectures.csv")
```

Unteraufgabe 4

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   attend_or_not = col_character(),
##   temperature = col_double(),
##   sun_joy = col_double(),
##   stats_joy1 = col_double(),
##   stats_joy2 = col_double(),
##   stats_joy3 = col_double(),
##   stats_joy4 = col_double(),
##   stats_joy5 = col_double()
## )
```

2 Datenaufbereitung

1. Bilden Sie für jede Person im Datensatz den Mittelwert aus den 5 Items stats_joy1 bis stats_joy5 und fügen Sie diesen als neue Variable dem Datensatz hinzu. *Hinweis: Eine Google-Suche kann hier sehr hilfreich sein. Achten Sie auch auf Kleinigkeiten in den Befehlen, die Sie finden.* Falls Sie diese Aufgabe nicht lösen können, aber dennoch weiter machen möchten, finden Sie hier funktionierende Syntax, die Sie in Ihr Script kopieren können. Link
2. Nutzen Sie den Befehl `factor()`, um die Variable `attend_or_not` in einen Faktor umzuformen. Verwenden Sie das Argument `levels = c()`, um die Faktorstufen anzugeben. So stellen Sie sicher, dass Sie wissen, welche Kategorie die Baseline darstellt. Falls Sie diese Aufgabe nicht lösen können, aber dennoch weiter machen möchten, finden Sie hier funktionierende Syntax, die Sie in Ihr Script kopieren können. Link

Lösung

Unteraufgabe 1 Der Befehl `mean()` muss hier mit `mean(c())` verwendet werden, damit die Zeilenweisen Mittelwerte richtig gebildet werden. Wir verwenden mutate(), um eine neue Varibale zu erstellen.

```r
stats_lectures <- stats_lectures %>%
  rowwise() %>% # wir wollen zeilenweise vorgehen
  mutate(stats_joy = mean(c(stats_joy1, stats_joy2, # mittelwert bilden
                            stats_joy3, stats_joy4,
                            stats_joy5)))
```

```r
stats_lectures <- stats_lectures %>% mutate(attend_or_not = factor(attend_or_not, levels = c("course no
```

Unteraufgabe 2

3 Erste logistische Regression

1. Nutzen Sie den Befehl `glm()` mit dem Argument `family = binomial()`, um die Kursbesuche der Studierenden mit deren Genuss von Statistik vorherzusagen. Benutzen Sie dafür den Mittelwert, den Sie in der vorherigen Aufgabe gebildet haben, als Prädiktor.
2. Lassen Sie sich den Output mit `summary()` anzeigen. "Null deviance" gibt die Deviance-Statistik für das Null-Modell an, "Residual Deviance" für das Alternativmodell (user spezifiziertes). Was können Sie aus der Deviance ableiten? (Siehe Field, Kapitel 8.3.2: Assessing the model: the deviance statistic)
3. Bilden Sie mit `exp()` das Exponential zur Basis e für die Regressionskoeffizienten. Dies ist das Odds-Ratio. *Tipp: Mit `$coefficients` können Sie direkt auf die Koeffizienten zugreifen, wenn Sie den Namen, den Sie dem Modell gegeben haben, vor das $-Zeichen schreiben.*
4. Interpretieren Sie das Odds Ratio für den vorliegenden Fall. (Siehe Field, Kapitel 8.3.6: The odds ratio)
5. Wenden Sie `confint()` auf das Modell an, um die Konfidenzintervalle für die Prädiktoren zu erhalten. Bilden Sie auch hiervon das Exponential mit `exp()`, um die Konfidenzinternalle für die Odds-Ratios zu erhalten.
6. Erstellen Sie einen Plot, um Ihr Modell zu visualisieren.
    a) Wenden Sie den Befehl `fitted()` auf das Modell an, um die durch das Modell vorhergesagten Wahrscheinlichkeiten für Ihre Studierenden zu erhalten.
    b) Fügen Sie diese Werte Ihrem Datensatz hinzu.
    c) Nutzen Sie `gpplot`, um einen Plot zu erstellen. Verwenden Sie auf der X-Achse stats_joy und auf der y-Achse die in a) und b) erstellten vorhergesagten Wahrscheinlichkeiten.

Lösung

```r
model1 <- glm(attend_or_not ~ stats_joy,
              data = stats_lectures, family = binomial())
```

Unteraufgabe 1

```r
summary(model1)
```

Unteraufgabe 2

```
##
## Call:
## glm(formula = attend_or_not ~ stats_joy, family = binomial(),
##     data = stats_lectures)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4889   0.1323   0.3730   0.6735   1.8789
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.6812     0.5830  -6.315 2.71e-10 ***
## stats_joy     1.0519     0.1301   8.085 6.21e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 433.82  on 399  degrees of freedom
## Residual deviance: 337.01  on 398  degrees of freedom
## AIC: 341.01
##
## Number of Fisher Scoring iterations: 5
```

Die Deviance ist ein Indikator dafür, wie viel Information nicht durch das Modell aufgeklärt werden kann. Eine größere Deviance bedeutet, dass weniger Information erklärt werden kann. Mehr dazu können Sie hier nachlesen: Field, Kapitel 8.3.2: Assessing the model: the deviance statistic.

```
exp(model1$coefficients)
```

Unteraufgabe 3

```
## (Intercept)   stats_joy
##  0.02519251  2.86315905
```

Unteraufgabe 4 Wenn hier bei Studi A die Freude an Statistik um einen Punkt höher ist als bei Studi B, dann heißt das, dass die Chance, dass Studi A die Statistik-Vorlesung besucht, ca. 2.86-mal größer ist, als dass Studi B es tut.

D.h. eine Steigerung der Freude an Statistik um einen Punkt führt zu einer 2.86-mal höheren Chance, die Statistik-Vorlesung zu besuchen.

```
exp(confint(model1))
```

Unteraufgabe 5

```
## Waiting for profiling to be done...
```

```
##                   2.5 %   97.5 %
## (Intercept) 0.007655385 0.075732
## stats_joy   2.245264202 3.744471
```

Unteraufgabe 6

a) Ich nutze hier head(), um nur die ersten 6 Werte anzeigen zu lassen. Andernfalls würde das Dokument sehr voll werden.

```
head(fitted(model1))
```

```
##         1         2         3         4         5         6
## 0.9630842 0.7609053 0.5783756 0.6286649 0.9327892 0.9011060
```

b)

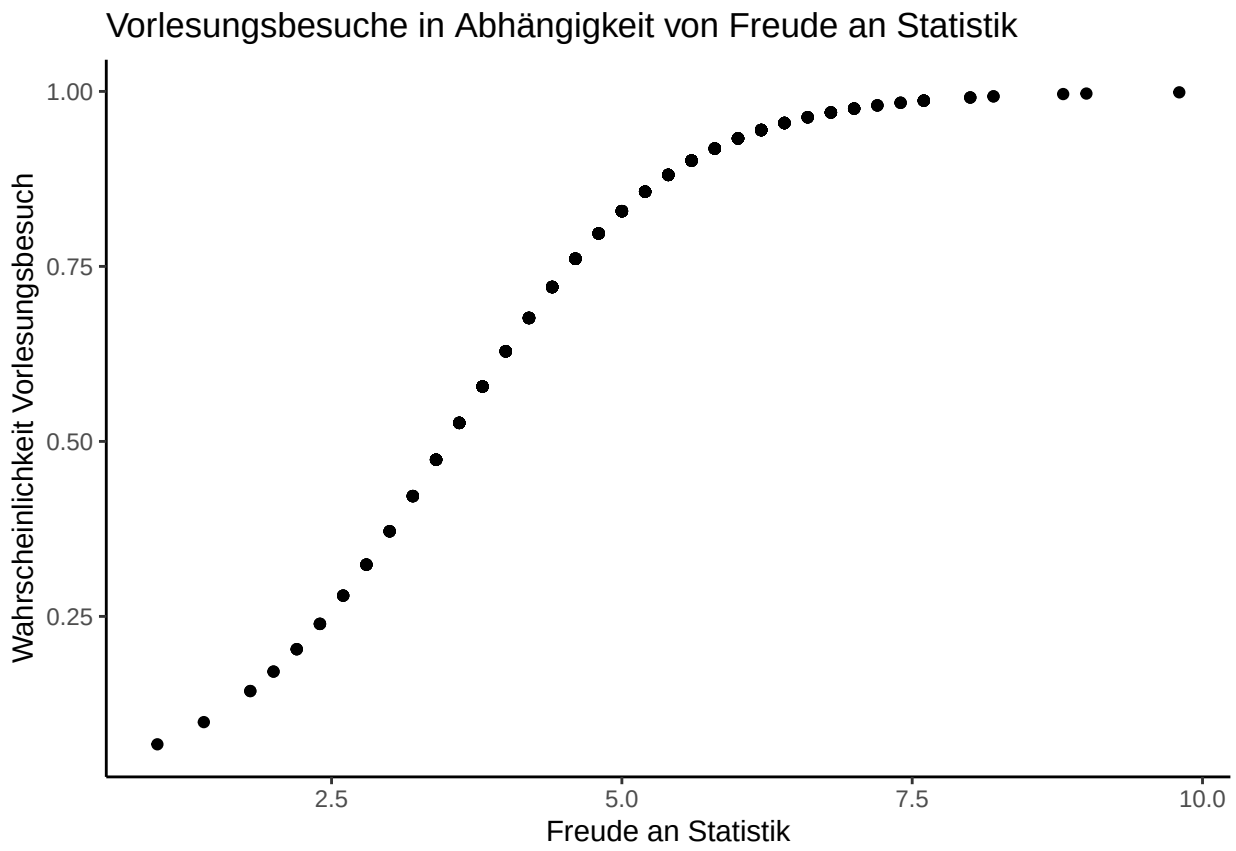
```
stats_lectures$pred_prob <- fitted(model1)
```

c)

```
# Objekt erstellen
plot1 <- ggplot(stats_lectures, aes(x = stats_joy, y = pred_prob))

# Plot befüllen
plot1 <- plot1 + geom_point() +
  labs(x = "Freude an Statistik",
       y = "Wahrscheinlichkeit Vorlesungsbesuch",
       title = "Vorlesungsbesuche in Abhängigkeit von Freude an Statistik") +
  theme_classic()

# plot anzeigen
plot1
```



4 Modellvergleich logistische Regression

1. Erstellen Sie drei weitere Modell mit dem `glm()`-Befehl:
   a) Fügen Sie den ersten Modell den Prädiktor temperature hinzu
   b) Fügen Sie dem Modell aus a) den Prädiktor sun_joy hinzu
   c) Fügen Sie dem Modell aus b) die Interaktion aus temperature und sun_joy als Prädiktor hinzu.
2. Testen Sie mit dem Befehl `anova()`, welches der Modelle am besten auf die Daten passt. Nutzen Sie die Option `test = "Chisq"`, um einen p-Wert zu erhalten.
   a) Sehen Sie sich mit `summary()` den Output für das beste Modell an.
   b) Welche Hypothesen würden Sie aus dieser Analyse ableiten?
3. Berechnen Sie für das beste Modell die Odds Ratios der Prädiktoren und deren Konfidenzintervalle.
4. Berechnen Sie für das beste Modell die drei Pseudo-R^2 Indices (Siehe Field, Kapitel 8.3.3: Assessing the model: R and R^2). Sie können dafür die unten stehende Funktion nutzen. *Mehr dazu in Field, R's Souls' Tip 8.2.*
   a) Kopieren Sie den Code und führen Sie in aus.
   b) Sie können nun die Funktion `logisticPseudoR2s()` auf ihr Modell anweden.

```
logisticPseudoR2s = function(LogModel) {
  dev = LogModel$deviance
  nullDev = LogModel$null.deviance
  modelN = length(LogModel$fitted.values)
  R.l = 1 - dev / nullDev
```

```
  R.cs = 1 - exp(-(nullDev - dev) / modelN)
  R.n = R.cs / (1 - (exp(-(nullDev / modelN))))

  cat("Pseudo R^2 for logistic regression\n")
  cat("Hosmer and Lemeshow R^2 ", round(R.l, 3), "\n")
  cat("Cox and Snell R^2 ", round(R.cs, 3),"\n")
  cat("Nagelkerke R^2 ", round(R.n, 3), "\n")
}
```

Lösung

```
# das erste Modell
model1 <- glm(attend_or_not ~ stats_joy,
              data = stats_lectures, family = binomial())

# a)
model2 <- glm(attend_or_not ~ stats_joy + temperature,
              data = stats_lectures, family = binomial())

# b)
model3 <- glm(attend_or_not ~ stats_joy + temperature + sun_joy,
              data = stats_lectures, family = binomial())

# c) (kann auf zwei Arten geschrieben werden)
model4 <- glm(attend_or_not ~ stats_joy + temperature*sun_joy,
              data = stats_lectures, family = binomial())

model4 <- glm(attend_or_not ~ stats_joy + temperature +
                sun_joy + temperature:sun_joy,
              data = stats_lectures, family = binomial())
```

Unteraufgabe 1

```
anova(model1, model2, model3, model4, test = "Chisq")
```

Unteraufgabe 2

```
## Analysis of Deviance Table
##
## Model 1: attend_or_not ~ stats_joy
## Model 2: attend_or_not ~ stats_joy + temperature
## Model 3: attend_or_not ~ stats_joy + temperature + sun_joy
## Model 4: attend_or_not ~ stats_joy + temperature + sun_joy + temperature:sun_joy
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       398     337.01
## 2       397     336.47  1    0.545    0.4602
## 3       396     336.24  1    0.225    0.6356
```

```
## 4       395     287.45  1   48.789 2.851e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Modell 4 passt mit deutlichem Abstand am besten auf die Daten.

a)

```
summary(model4)
```

```
##
## Call:
## glm(formula = attend_or_not ~ stats_joy + temperature + sun_joy +
##     temperature:sun_joy, family = binomial(), data = stats_lectures)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.64691   0.06321   0.29704   0.56997   1.87235
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.85863    0.63877  -6.041 1.53e-09 ***
## stats_joy            1.15949    0.14579   7.953 1.82e-15 ***
## temperature          0.05099    0.16847   0.303    0.762
## sun_joy             -0.10237    0.16675  -0.614    0.539
## temperature:sun_joy -1.23205    0.21454  -5.743 9.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 433.82  on 399  degrees of freedom
## Residual deviance: 287.45  on 395  degrees of freedom
## AIC: 297.45
##
## Number of Fisher Scoring iterations: 6
```

b) Man kann folgende Hypothesen ableiten:

1. Je höher die Freude an Statistik ist, desto höher ist die Wahrscheinlichkeit, dass Studierende Statistik-Vorlesungen besuchen.
2. Es gibt eine Interaktion zwischen Temperatur und Freude an der Sonne. Möglicherweise ist es so, dass eine hohe Temperatur bei SonnenliebhaberInnen dazu führt, dass die Statistik-Veranstaltungen weniger häufig besucht werden.

```
# Odds Ratios
exp(model4$coefficients)
```

Unteraufgabe 3

```
##         (Intercept)           stats_joy         temperature             sun_joy temperature:sun_joy
##          0.02109698          3.18829769          1.05231642          0.90269675          0.29169494
```

```
# Konfidenzintervalle. exp() nicht vergessen!
exp(confint(model4))
```

```
## Waiting for profiling to be done...
```

```
##                            2.5 %     97.5 %
## (Intercept)          0.005684811 0.07011893
## stats_joy            2.432316639 4.31611607
## temperature          0.755915622 1.46723210
## sun_joy              0.648570517 1.25053448
## temperature:sun_joy 0.186819991 0.43393613
```

Unteraufgabe 4

a) Siehe Aufgabenstellung.

b)

```
logisticPseudoR2s(model4)
```

```
## Pseudo R^2 for logistic regression
## Hosmer and Lemeshow R^2  0.337
## Cox and Snell R^2  0.306
## Nagelkerke R^2  0.463
```

5 Rendern (knit)

Lassen Sie die Datei mit `Strg` + `Shift` + `K` (Windows) oder `Cmd` + `Shift` + `K` (Mac) rendern. Sie sollten nun im "Viewer" unten rechts eine "schön aufpolierte" Version ihrer Datei sehen. Falls das klappt: Herzlichen Glückwunsch! Ihr Code kann vollständig ohne Fehlermeldung gerendert werden. Falls nicht: Nur mut, das wird schon noch! Gehen Sie auf Fehlersuche! Ansonsten schaffen wir es ja in der Übung vielleicht gemeinsam.

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 as PDF

Exercise sheet with solutions included

Exercise sheet with solutions included as PD

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

Some hints

1. Please try to solve this sheet in an .Rmd file. You can create one from scratch using `File > New file > R Markdown...`. You can delete the text beneath *Setup Chunk* (starting from line 11). Alternatively, you can download our template file unter this link (right click > save as...).
2. You'll find a lot of the important information on the website of this course
3. Please don't hesitate to search the web for help with this sheet. In fact, being able to effectively search the web for problem solutions is a very useful skill, even R pros work this way all the time! The best starting point for this is the R section on the programming site Stackoverflow
4. On the R Studio website, you'll find highly helpful cheat sheets for many of R topics. The base R cheat sheet might be a good starting point.

Ressources

Since this is a hands-on seminar, we won't be able to present each and every new command to you explicitly. Instead, you'll find here references to helpful ressources that you can use for completing this sheets.

Ressource	Description
Field, chapter 8	Book chapter explaining step by step the why and how of logistic regression in R. **Highly recommended!**

Hint of the week

You should use new lines in your code to make it better readable for yourself and for others. You could f. e. continue in a new line after every pipe (`%>%`) or after a `+` in a sequence of ggplot2 commands. Don't hesitate to divide a complex and long command into several lines. After every comma (`,`) you can continue in a new line, if you like.

If your code looks scattered because of unregular indentation, you may mark a section of code and press `strg` + `i` (Windows), or `cmd` + `i`(Mac) to automatically indent your source code.

1 Read data

1. Load package `tidyverse` and insert the apropriate code at the beginning of your .Rmd document.
2. Change to an adequate working directory. Take care: The render process assumes, that your working directory is set to the location of your .Rmd document.
3. Load the data `stats_lectures` from this link and store it in your working directory or in a subfolder there. You might want to call it `data`.
4. Read the data in R. Alternatively read the data directly from the URL above.

Explanation of the variables

Each line has the data of one student. You find, whether the student took part in a statistic lession, the current temperature, to which extend the person enjoys sun and, on the other hand, statistics. This was measured using a 5 items questionnaire.

Hint: These are completely fictive data. They don't reflect the docents opinion. This is just an spontaneous idea of a student helper.

name	meaning
attend_or_not	dichotomous variable: “course not attended” and “course attended”. The values code, whether the student attendet the course or not.
temperature	high values code high temperatures
sun_joy	the higher the value the higher the joy
stats_joy1 to stats_joy5	the higher the values the more the student enjoys statistics

Solution

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

Subtask 1

Subtask 2 Take care: R uses / to specify pathes. So you cannot simply copy and paste Windows pathes into R. You have to replace all \ with /. Again: Remember the special behavior of the rendering with regard to working directories.

The code would be:

```
setwd()
```

Here an example with a local path. Adapt that to your needs.

```
# example
setwd("P:/R/mv")
```

Subtask 3 Please follow the instructions in subtask 3.

```
# stats_lectures <- read_csv("data/stats_lectures.csv") # relative path, working directory has to be se
# alternatively read it from the URL
stats_lectures <- read_csv("http://md.psych.bio.uni-goettingen.de/mv/data/div/stats_lectures.csv")
```

Subtask 4

```
##
## -- Column specification --------------------------------------------------------------------------
## cols(
##   attend_or_not = col_character(),
##   temperature = col_double(),
##   sun_joy = col_double(),
##   stats_joy1 = col_double(),
##   stats_joy2 = col_double(),
##   stats_joy3 = col_double(),
##   stats_joy4 = col_double(),
##   stats_joy5 = col_double()
## )
```

2 Data preparation

1. Calculate the mean of items stats_joy1 to stats_joy5 and store it in the dataframe resp. tibble. *Hint: A google search might help. Take also care of the details of the commands, you find.* If you aren't able to solve this part of the task and want to continue, you find a working syntax for this part under https://pzezula.pages.gwdg.de/data/06_sheet_task2-1.R that you might integrate in your script.
2. Use the command `factor()` to transform variable `attend_or_not` into a factor. Use the argument `levels = c()` to specify factor levels. This is how you set the correct reference group or baseline. If you want to keep on and don't know how to do tis, you find a working syntax under https://pzezula.pages.gwdg.de/data/06_sheet_task2-2.R

Solution

Subtask 1 The command `mean()` has to be done like `mean(c())` to have the means calculated rowwise. We use `mutate()` to create a new variable.

```
stats_lectures <- stats_lectures %>%
  rowwise() %>% # wir want rowwise calculation
  mutate(stats_joy = mean(c(stats_joy1, stats_joy2, # calculate means
                            stats_joy3, stats_joy4,
                            stats_joy5)))
# we could alternatively use base systems rowMeans()
# {}
```

```
stats_lectures <- stats_lectures %>% mutate(attend_or_not = factor(attend_or_not, levels = c("course no
```

Subtask 2

3 First logistic regression

1. Use the command `glm()` with argument `family = binomial()` to predict, which students attend the statistics course and whether they enjoy it. The predictor is the mean, that you calculated recently.
2. Check the output using `summary()`. "Null deviance" is the deviance statistic for the null model, "residual deviance" is the same for the user defined alternative model. What can we learn from the deviance? C. f. Field (2012), chapter 8.3.2: Assessing the model: the deviance statistic.
3. Use `exp()` to calculate the exponet to base e for the regression coefficient. The result is the odds ratio. *Tip: `$coefficients` gives you direct access to the coefficients. As always you have to put the name of your result model and a $ before.*
4. Interprete the odds ratio for your example. (See Field (2012), chapter 8.3.6: The odds ratio)
5. Apply `confint()` to your result model to get confidence intervals for your predictors. Also use `exp()` to get the confidence intervals for the odds ratios.
6. Make a plot to visualize your model.
    a) Apply `fitted()` to your model to get the predicted probailities for your student subjects.
    b) Add these values to your data set.
    c) Use `ggplot()` to make a plot. Put `stats_joy` to the x-axes and on the y-axes the predicted probilities, you calculated in a) and b).

Solution

```
model1 <- glm(attend_or_not ~ stats_joy,
              data = stats_lectures, family = binomial())
```

Subtask 1

```
summary(model1)
```

Subtask 2

```
##
## Call:
## glm(formula = attend_or_not ~ stats_joy, family = binomial(),
##     data = stats_lectures)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4889   0.1323   0.3730   0.6735   1.8789
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -3.6812     0.5830  -6.315 2.71e-10 ***
## stats_joy     1.0519     0.1301   8.085 6.21e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 433.82  on 399  degrees of freedom
## Residual deviance: 337.01  on 398  degrees of freedom
## AIC: 341.01
##
## Number of Fisher Scoring iterations: 5
```

The deviance is an indicator of how much information can *not* be explained by our model. The bigger the deviance, the less information can be explained. Read more about this in Field (2012), chapter 8.3.2 Assessing the model: The deviance statistic.

```
exp(model1$coefficients)
```

Subtask 3

```
## (Intercept)   stats_joy
##  0.02519251  2.86315905
```

Subtask 4 If student A has a stats_joy of 1 point higher than student B, this means, that the chance, that student A visits the statistics class is 2.86 times higher than for student B.

In other words: An increase of joy in statistics of 1 point results in a 2.86 times higher probability to visit the statistics class.

```
exp(confint(model1))
```

Subtask 5

```
## Waiting for profiling to be done...
```

```
##                   2.5 %   97.5 %
## (Intercept) 0.007655385 0.075732
## stats_joy   2.245264202 3.744471
```

Subtask 6

a) We use `head()`, to see the first 6 values only. Thus we avoid an extremely large document.

```
head(fitted(model1))
```

```
##         1         2         3         4         5         6
## 0.9630842 0.7609053 0.5783756 0.6286649 0.9327892 0.9011060
```

b)

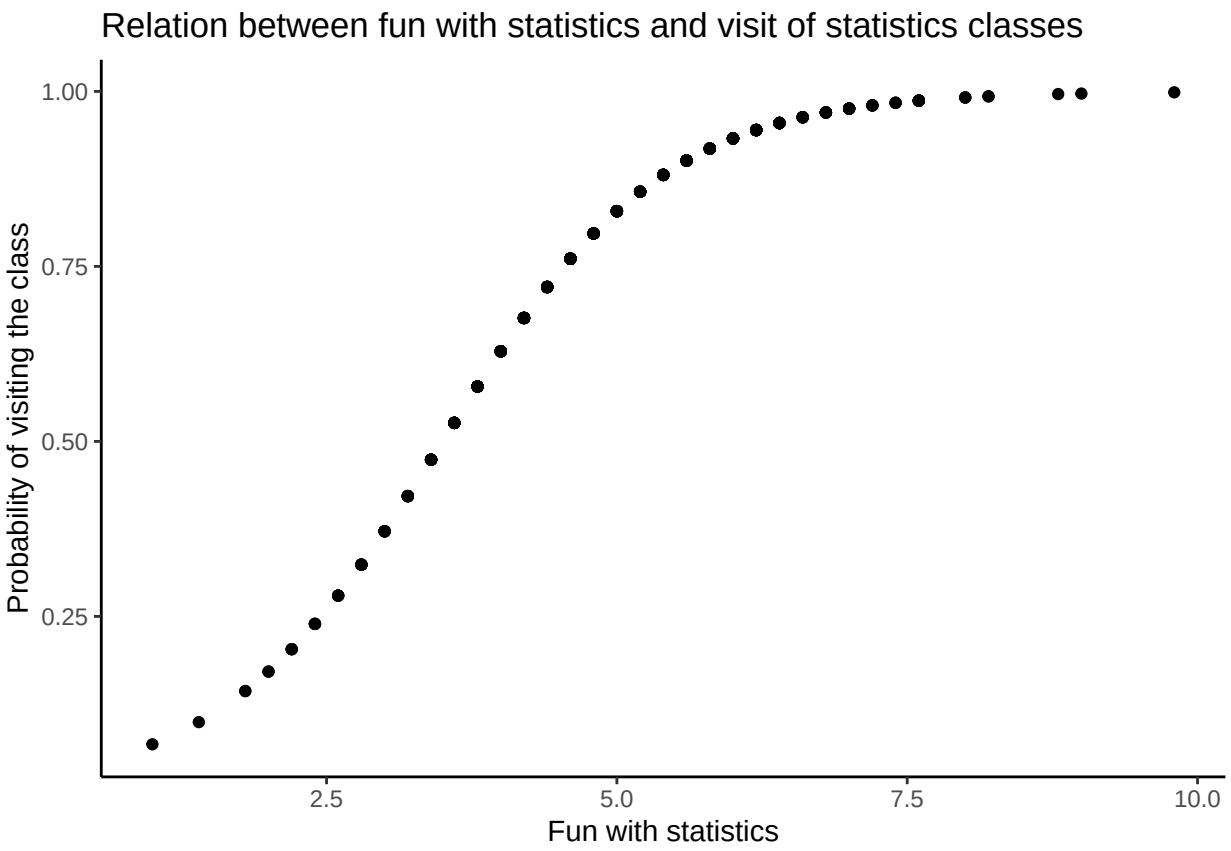
```
stats_lectures$pred_prob <- fitted(model1)
```

c)

```
# Create a result object
plot1 <- ggplot(stats_lectures, aes(x = stats_joy, y = pred_prob))

# Fill the plot
plot1 <- plot1 + geom_point() +
         labs(x = "Fun with statistics",
              y = "Probability of visiting the class",
         title = "Relation between fun with statistics and visit of statistics classes") +
         theme_classic()

# Show plot
plot1
```



4 Model comparison in logistic regression

1. Adapt three more models using the command `glm()`:
   a) Add predictor `temperature`
   b) Add predictor `sun_joy`
   c) Add to the model in b) the interaction of `temperature` and `sun_joy`.
2. Compare the models adapted using the command `anova()`. Which of the models fit the data best, considering parsimonia. Use the parameter `test = "Chisq"` to get a p-value.
   a) Get the `summary()` of the best model.
   b) Which hypothesis would you deduct from this analysis.
3. Calculate the odds ratios and the confidence intervales for the parameters of the best model.
4. Calculate the pseudo R^2 for the best model (c. f. Field (2012), chapter 8.3.3: Assessing the model: R and R^2). You may use the function below for this. *Read more on that in Field (2012), R's Souls' Tip 8.2.*
   a) Copy the code and execute it
   b) You can apply `logisticPseudoR2s()` to your model.

```
logisticPseudoR2s = function(LogModel) {
  dev = LogModel$deviance
  nullDev = LogModel$null.deviance
  modelN = length(LogModel$fitted.values)
  R.l = 1 - dev / nullDev
```

```
  R.cs = 1 - exp(-(nullDev - dev) / modelN)
  R.n = R.cs / (1 - (exp(-(nullDev / modelN))))

  cat("Pseudo R^2 for logistic regression\n")
  cat("Hosmer and Lemeshow R^2 ", round(R.l, 3), "\n")
  cat("Cox and Snell R^2 ", round(R.cs, 3),"\n")
  cat("Nagelkerke R^2 ", round(R.n, 3), "\n")
}
```

Solution

```
# the first model
model1 <- glm(attend_or_not ~ stats_joy,
              data = stats_lectures, family = binomial())

# a)
model2 <- glm(attend_or_not ~ stats_joy + temperature,
              data = stats_lectures, family = binomial())

# b)
model3 <- glm(attend_or_not ~ stats_joy + temperature + sun_joy,
              data = stats_lectures, family = binomial())

# c) (we could write that in two manners)
model4 <- glm(attend_or_not ~ stats_joy + temperature*sun_joy,
              data = stats_lectures, family = binomial())

model4 <- glm(attend_or_not ~ stats_joy + temperature +
                sun_joy + temperature:sun_joy,
              data = stats_lectures, family = binomial())
```

Subtask 1

```
anova(model1, model2, model3, model4, test = "Chisq")
```

Subtask 2

```
## Analysis of Deviance Table
##
## Model 1: attend_or_not ~ stats_joy
## Model 2: attend_or_not ~ stats_joy + temperature
## Model 3: attend_or_not ~ stats_joy + temperature + sun_joy
## Model 4: attend_or_not ~ stats_joy + temperature + sun_joy + temperature:sun_joy
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1       398     337.01
## 2       397     336.47  1    0.545    0.4602
## 3       396     336.24  1    0.225    0.6356
```

```
## 4       395     287.45  1   48.789 2.851e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

It is model 4 that fits the data best.

a)

```
summary(model4)
```

```
##
## Call:
## glm(formula = attend_or_not ~ stats_joy + temperature + sun_joy +
##     temperature:sun_joy, family = binomial(), data = stats_lectures)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.64691   0.06321   0.29704   0.56997   1.87235
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.85863    0.63877  -6.041 1.53e-09 ***
## stats_joy            1.15949    0.14579   7.953 1.82e-15 ***
## temperature          0.05099    0.16847   0.303    0.762
## sun_joy             -0.10237    0.16675  -0.614    0.539
## temperature:sun_joy -1.23205    0.21454  -5.743 9.32e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 433.82  on 399  degrees of freedom
## Residual deviance: 287.45  on 395  degrees of freedom
## AIC: 297.45
##
## Number of Fisher Scoring iterations: 6
```

b) We could deduct the hypothesis:

1. The more students enjoy statistics, the higher is the probability, that they visit statistics classes.
2. Is there an interaction between temperature and enjoyment of sun. Might it be, that higher temperatures lead to less probability to visit classes for students, that enjoy sun?

```
# Odds Ratios
exp(model4$coefficients)
```

Subtask 3

```
##         (Intercept)           stats_joy         temperature             sun_joy temperature:sun_joy
##          0.02109698          3.18829769          1.05231642          0.90269675          0.29169494
```

```r
# Confidence intervals, don't forget exp()
exp(confint(model4))
```

```
## Waiting for profiling to be done...
```

```
##                           2.5 %     97.5 %
## (Intercept)         0.005684811 0.07011893
## stats_joy           2.432316639 4.31611607
## temperature         0.755915622 1.46723210
## sun_joy             0.648570517 1.25053448
## temperature:sun_joy 0.186819991 0.43393613
```

Subtask 4

a) See the text above

b)

```r
logisticPseudoR2s(model4)
```

```
## Pseudo R^2 for logistic regression
## Hosmer and Lemeshow R^2  0.337
## Cox and Snell R^2  0.306
## Nagelkerke R^2  0.463
```

5 Render (knit)

Render your document using `Strg` + `Shift` + `K` (Windows) oder `Cmd` + `Shift` + `K` (Mac). After that, you shoud see a nice looking version of your document. I that works out, congratulations! Your code could be rendered without any errors. If not, don't be frustrated. Fix your errors. We will help you in our exercise class.

Literature

Note: These sheets are based partially on exercises from the book *Dicovering Statistics Using R* (Field, Miles & Field, 2012). They've been modified for the porpuses of this seminar, and the R code was updated.

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

Version: 06 Mai, 2021 23:22