Musterloesung Aufgabe 4.2S: multiple logistische Regression

Lese-Empfehlung Kapitel 6 von Manny Gimond

Lese-Empfehlung Kapitel 4 von Gareth (2016)

Download R-Skript

Download PDF

---

kommentierter Lösungsweg

```
# Genereiert eine Dummyvariable: Fleisch 1, kein Fleisch 0
df <- nova_survey %>%  # kopiert originaler Datensatz
  rename(umwelteinstellung = tho_2) %>% # änderung name der variable
  mutate(umwelteinstellung = case_when(umwelteinstellung == 4 ~ 1,
                                       umwelteinstellung == 3 ~ 1,
                                       umwelteinstellung == 2 ~ 0,
                                       umwelteinstellung == 1 ~ 0)) %>%
  # lasse die kleine Gruppe mit x weg
  dplyr::filter(!str_detect(string = "x", gender)) %>%
  # lasse die kleine Gruppe "andere" weg
  dplyr::filter(!str_detect(string = "Andere", member)) %>%
  # wähle nur die relevanten variablen aus
  dplyr::select(mensa, age_groups, gender, member, umwelteinstellung, meat)

# Schaut euch die Missings an in der Kriteriumsvariable "mensa"
sum(is.na(df$mensa))
```

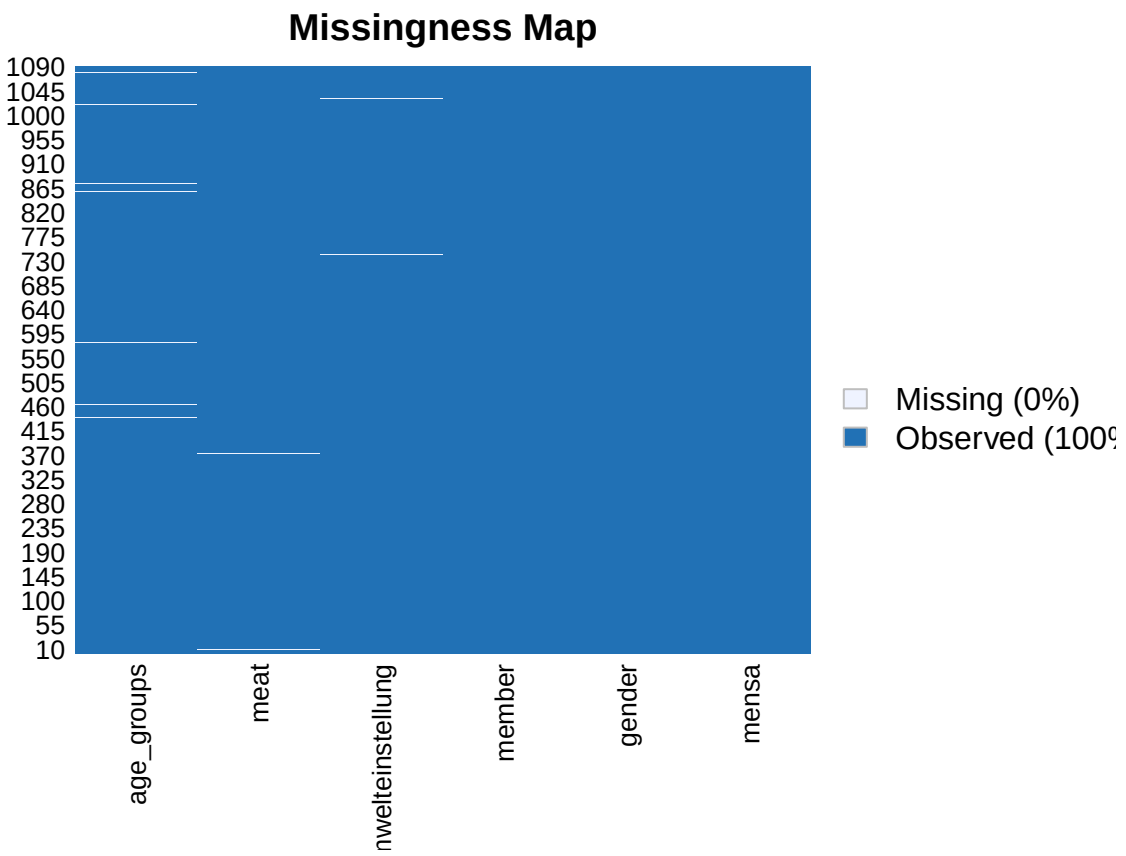
```
## [1] 0
```

```
# schaut euch die Missings an in den Prädiktorvariablen "Alter", "Geschlecht", "Hochschulzugehörigkeit"
Amelia::missmap(df)
```

```
## Warning: Unknown or uninitialised column: `arguments`.

## Warning: Unknown or uninitialised column: `arguments`.

## Warning: Unknown or uninitialised column: `imputations`.
```




```
# vieles deutet darauf hin, dass die missings (fehlende Werte)
# zufällig zustande gekommen sind (sog. MCAR); für mehr Informationen: https://uvastatlab.github.io/201

# bester Weg wäre, die wenigen fehlenden Werte zu imputieren;
# einfachheitshalber löschen wir sie aber :)
df %<>%
  drop_na()

#  sieht euch die Verteilung zwischen Mensagänger und Selbstverpfleger an
# sind nicht gleichmässig verteilt, bei der Vorhersage müssen wir das berücksichtigen
table(df$mensa)
```

```
##
##   0   1
## 282 786
```

```
df %>% count(mensa) # alternativ
```

```
## # A tibble: 2 x 2
##   mensa     n
##   <dbl> <int>
## 1     0   282
## 2     1   786
```

```r
# definiert das logistische Modell und wendet es auf den Datensatz an

mod0 <-glm(mensa ~ gender + member + age_groups + meat + umwelteinstellung,
           data = df, binomial("logit"))
summary.lm(mod0) # Umwelteinstellung scheint keinen Einfluss auf die
```

```
##
## Call:
## glm(formula = mensa ~ gender + member + age_groups + meat + umwelteinstellung,
##     family = binomial("logit"), data = df)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -5.6740 -0.8078  0.3712  0.5867  1.2379
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -0.18889    0.40225  -0.470 0.638750
## genderMann                    0.71017    0.16018   4.434 1.02e-05 ***
## memberStudent/in             -0.63072    0.29442  -2.142 0.032404 *
## age_groups26- bis 34-jaehrig  1.09429    0.19574   5.591 2.88e-08 ***
## age_groups35- bis 49-jaehrig  1.75379    0.45968   3.815 0.000144 ***
## age_groups50- bis 64-jaehrig  2.43530    0.78923   3.086 0.002083 **
## meat                          0.19945    0.05055   3.945 8.49e-05 ***
## umwelteinstellung             0.19334    0.18688   1.035 0.301107
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.009 on 1060 degrees of freedom
## Multiple R-squared:  0.004042,   Adjusted R-squared:  -0.002536
## F-statistic: 0.6145 on 7 and 1060 DF,  p-value: 0.7443
```

```r
# Verpflegung zu haben, gegeben die Daten

# neues Modell ohne Umwelteinstellung
mod1 <- update(mod0, ~. -umwelteinstellung)
summary.lm(mod1)
```

```
##
## Call:
## glm(formula = mensa ~ gender + member + age_groups + meat, family = binomial("logit"),
##     data = df)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -6.0117 -0.8060  0.3584  0.6100  1.2407
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.03212    0.34053   0.094 0.924860
## genderMann                    0.69697    0.15951   4.369 1.37e-05 ***
## memberStudent/in             -0.64418    0.29426  -2.189 0.028806 *
## age_groups26- bis 34-jaehrig  1.11651    0.19458   5.738 1.25e-08 ***
```

```
## age_groups35- bis 49-jaehrig  1.77409    0.45947   3.861 0.000120 ***
## age_groups50- bis 64-jaehrig  2.44683    0.78953   3.099 0.001992 **
## meat                          0.18070    0.04709   3.837 0.000132 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.01 on 1061 degrees of freedom
## Multiple R-squared:  0.003998,   Adjusted R-squared:  -0.001635
## F-statistic: 0.7098 on 6 and 1061 DF,  p-value: 0.6418
```

```
# Modeldiagnostik (wenn nicht signifikant, dann OK)
1 - pchisq(mod1$deviance, mod1$df.resid) # Ok
```

```
## [1] 0.4509591
```

```
#Modellgüte (pseudo-R²)
1 - (mod1$dev / mod1$null) # eher kleines pseudo-R2
```

```
## [1] 0.1354244
```

```
# Konfusionsmatrix vom  Datensatz
# Model Vorhersage
# hier ein anderes Beispiel:
predicted <- predict(mod1, df, type = "response")

# erzeugt eine Tabelle mit den beobachteten
# Mensagänger/Selbstverpfleger und den Vorhersagen des Modells
km <- table(predicted > 0.5, df$mensa)
# alles was höher ist als 50% ist
# kommt in die Kategorie Mensagänger

# anpassung der namen
dimnames(km) <- list(
  c("Modell Selbst", "Modell Mensa"),
  c("Daten Selbst", "Daten Mensa"))
km
```

```
##               Daten Selbst Daten Mensa
## Modell Selbst           87          59
## Modell Mensa           195         727
```

```
#############
### reminder: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62
#############

#TP = true positive: you predicted positive and it's true; hier vorhersage
# mensagänger stimmt also (727)

#TN = true negative: you predicted negative and it's true, hier vorhersage der
# selbstverpfleger stimmt (87)

#FP = false positive (fehler 1. art, auch spezifizität genannt) you predicted
```

```r
# and it's false. hier modell sagt mensagänger vorher
# (obwohl in realität selbstverpfleger) (195)

#FN = false negative (fehler 2. art, auch sensitivität genannt),
# you predicted negative and it's false. hier modell sagt selbtverpfleger vorher
# (obwohl in realität mensagänger) (59)


# es scheint, dass das Modell häufig einen alpha Fehler zu machen, d.h. es
# das Modell weist keine hohe Spezifizität auf: konkret werden viele Mensagänger als
# Selbstverpfleger vorhergesagt resp. klassifiziert. Dafür gibt es mehere Gründe:

#1) die Kriteriumsvariable ist sehr ungleich verteilt, d.h. es gibt weniger
# Selbstverpfleger als Mensgänger im Datensatz

#2) nicht adäquates Modell z.B. link mit probit zeigt besserer fit

#3) Overfitting: wurde hier nicht berücksichtigt, in einem Paper/Arbeit
# müsste noch einen Validierungstest gemacht werden z.B. test-train
# Cross-Validation oder k fold Cross-Validation

# kalkuliert die Missklassifizierungsrate
mf <- 1-sum(diag(km)/sum(km)) # ist mit knapp 23 %  eher hoch
mf
```

```
## [1] 0.2378277
```

```r
# kleiner exkurs: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2636062/
# col wise proportion, da diese die "realität" ist
km_prop <- prop.table(km,2)


# specificity = a / (a+c) => ability of a test to correctly
# classify an individual as disease-free is called the test s specificity
spec = km_prop[1] / (km_prop[1] + km_prop[2])
spec
```

```
## [1] 0.3085106
```

```r
# sensitivity = d / (b+d) => Sensitivity is the ability of a
# test to correctly classify an individual as  diseased
sens = km_prop[4] / (km_prop[3] + km_prop[4])
sens
```

```
## [1] 0.9249364
```

Methode

In der Aufgabe war es das Ziel zu schauen, ob wir einen potenziellen Besuch eines Mensagasts vorhersagen können und zwar in Abhängigkeit von den sozioökonimischen Variablen, wahrgenommene Fleischkonsum

und der Umwelteinstellung. Die Kriteriumsvariable “Mensa” weist eine binäre Verteilung auf: Deshalb rechnen wir eine multiple logistische Regression mit den Prädiktoren “Alter”, “Geschlecht”, “Hochschulzugehörigkeit”, “Fleischkonsum” und ”Umwelteinstellung. Mehr Informatinen zu den logistischen Regressinen findet ihr im Buch von Crawley (2015) oder auch im Buch von Gareth (2016), Kapitel 4.3; passendes Video hier.

Ergebnisse

	Daten Selbst	Daten Mensa
Modell Selbst	87	59
Modell Mensa	195	727

Der Output des logistischen Models mit der Linkfunktin “logit” sagt und, dass das Modell nicht gut zu den Daten passt, d.h. mit dem Modell (gegeben die Daten) können wir nur schlecht vorhersagen, ob eine Person zukünftig sich in der Mensa verpflegt oder ihr Mittagessen selber mitnimmt. Hinweise dafür geben das kleine pseudo-R2 (14%) als auch die hohe Missklassifizierungsrate (24%): bei genauerer Betrachtung fällt auf, dass das Modell häufig einen alpha-Fehler begeht, d.h. unser Modell sagt zu viele Mensagänger vorher, obwohl diese in Realität Selbstverpfleger sind. Es gibt verschiedene Gründe für diesen schlechten Modelfit:

- die Kriteriumsvariable ist sehr ungleich verteilt, d.h. es gibt weniger Selbstverpfleger als Mensgänger im Datensatz (26% vs. 74%)
- die Prädiktorvariablen sind alle entweder kategorial oder ordinal: dies kann dazu führen, dass das Model keinen guten fit zu den Daten erzielt

Fazit: Es sollte nach einem weiteren adäquateren Modell gesucht werden: insbesondere ein Modell, welches einen mit ordinalen Prädiktorvariablen umgehen kann:

- eine bessere Link-Funktion für das GLM suchen z.B. probit
- polynomiale Kontraste
- Smooth Splines hier