R Cheat Sheet Statistik III - HS 2023

Dr. Boris Mayer, Institut für Psychologie, Universität Bern

Partial- und Semipartialkorrelation

```
# Partialkorrelation als Korrelation von Regressionsresiduen:
lm_X <- lm(data$X ~ data$Z)
lm_Y <- lm(data$Y ~ data$Z)
cor(lm_X$residuals, lm_Y$residuals)

# Semipartialkorrelation als Korrelation einer Residualvariablen E(X|Z) und Y:
cor(lm_X$residuals, data$Y)
```

Einfache und Multiple Regressionsanalyse

```
# Einfache Regression von Y auf X1
model1 <- lm(Y ~ X1, data = data)

# Multiple Regression von Y auf X1, X2 und X3
model2 <- lm(Y ~ X1 + X2 + X3, data = data) # ODER
model2 <- update(model1, . ~ . + X2 + X3)

confint(model) # Konfidenzintervalle der Regressionskoeffizienten

lm.beta(model) # Standardisierte Regressionskoeffizienten mit 'Quantpsych'

anova(model1, model2) # Hierarchische Regression/Modellvergleichstest für das Prädiktorenset (X2, X3)

# Prädizierte Werte mit Konfidenzintervallen für die bedingten Erwartungswerte
# (hier für eine Einfache Regression - model1)
predict(model1, interval = "confidence", level = 0.95)

# Prädizierte Werte mit Konfidenzintervallen für individuelle Kriteriumswerte
# (für zukünftige Daten von Personen mit den X1-Werten a, b und c)
predict(model1, newdata = data.frame(X1 = c(a, b, c)),
        interval = "prediction", level = 0.95)
```

Moderierte Regressionsanalyse

```
# Grundlegende Modellgleichungen
model <- lm(Y ~ X1*X2, data = data) # ODER
model <- lm(Y ~ X1 + X2 + X1:X2, data = data)

# Modelle mit zentrierten/transformierten Variablen zur Analyse von Simple Slopes
# an bestimmten Moderator-Stellen:
```

```
# Mean (Zentrierung)
data_a <- data |> mutate(X1_a = X1 - mean(X1),
                         X2_a = X2 - mean(X2))
model_a <- lm(Y ~ X1_a*X2_a, data = data_a)


# Mean + 1 SD (_sdo-Transformation)
data_sdo <- data |> mutate(X1_sdo = X1 - (mean(X1) + sd(X1)),
                           X2_sdo = X2 - (mean(X2) + sd(X2)))
model_sdo <- lm(Y ~ X1_sdo*X2_sdo, data = data_sdo)


# Mean - 1 SD (_sdu-Transformation)
data_sdu <- data |> mutate(X1_sdu = X1 - (mean(X1) - sd(X1)),
                           X2_sdu = X2 - (mean(X2) - sd(X2)))
model_sdu <- lm(Y ~ X1_sdu*X2_sdu, data = data_sdu)

# Alternativ können die _sdo- und _sdu-Variablen auch ausgehend von den zentrierten Variablen
# berechnet werden (hier beispielhaft nur für den Moderator X2):
X2_sdo = X2_a - sd(X2)
X2_sdu = X2_a + sd(X2)


# Simple Slopes und Plots mit 'interactions' (mit X2 als Moderator)
sim_slopes(model, pred = "X1", modx = "X2") # Default = Mean +/- 1 SD
sim_slopes(model, pred = "X1", modx = "X2", modx.values = c(1, 4)) # Bestimmte Werte ("Pick-a-point")

# Regions of Significance (JNI)
johnson_neyman(model, pred = "X1", modx = "X2", alpha = 0.05)
```

Analyse nichtlinearer Zusammenhänge (Beispiel: Quadratischer Zusammenhang)

```
# Berechnung einer quadrierten Prädiktorvariable X_square
data <- data |>  mutate(X_square = X^2)

# Berechnung als hierarchische Regression und Modellvergleich
model1 <- lm(Y ~ X, data = data) # Lineares Modell
model2 <- update(model1, . ~ . + X_square, data = data) # Quadratisches Modell
anova(model1, model2) # Modellvergleich nicht unbedingt nötig, Signifikanz des
# quadratischen Effekts auch direkt aus model2 (Effekt von X_square) ablesbar

# Alternative direkte Berechnung (Berechnung der Variable X_square nicht nötig)
model2 <- lm(Y ~ X + I(X^2), data = data)
```

Regression mit kategorialen unabhängigen Variablen

```
# Regression mit Dummy-Codierung
# Dummy-Codierung mit 'psych' (UV ist FACTOR mit 4 Kategorien)
dummies <- dummy.code(data$FACTOR) # Bildung der Dummy-Variablen
data <- data.frame(data, dummies) # Hinzufügen der Dummy-Variablen zum Datensatz
model1 <- lm(Y ~ dummy1 + dummy2 + dummy3, data = data)
```

```
# Referenzkategorie ist diejenige, deren Dummyvariable nicht als UV verwendet wird

# Dummy-Codierung in lm() als Default bei einem Faktor als UV (erste Faktorstufe = Referenzkategorie)
model2 <- lm(Y ~ FACTOR, data = data)

# Für die Regressionen mit Effekt-Codierungen muss nur die contrast()-Definition
# geändert werden, die Berechnung des Models bleibt gleich:

# Definition einer ungewichteten Effekt-Codierung mit 'memisc'
contrasts(data$FACTOR) <- contr.sum(levels(data$FACTOR), base = "Referenzkategorie")

# Definition einer gewichteten Effekt-Codierung mit 'wec'
contrasts(data$FACTOR) <- contr.wec(data$FACTOR, omitted = "Referenzkategorie")
```

Logistische Regressionsanalyse

```
# Einfache logistische Regression von Y (binär) auf X1
model1 <- glm(Y ~ X1, family = binomial(link = "logit"), data = data)

# Multiple logistische Regression von Y (binär) auf X1, X2 und X3
model2 <- glm(Y ~ X1 + X2 + X3, family = binomial(link = "logit"), data = data) # ODER
model2 <- update(model1, . ~ . + X2 + X3)

exp(coef(model)) # Odds-Ratios

confint(model) # Konfidenzintervalle der Regressionskoeffizienten
exp(confint(model)) # Konfidenzintervalle der Odds-Ratios

# Likelihood-Ratio-Test für das Prädiktorenset (X2, X3) (Modellvergleich)
anova(model1, model2, test = "Chi")

# Likelihood-Ratio-Test für das Gesamtmodell (hier: model2)
model0 <- glm(Y ~ 1, family = binomial(link = "logit"), data = data) # Nullmodell schätzen
anova(model0, model2, test = "Chi") # Modellvergleich

# Effektgrössen für das Gesamtmodell mit 'DescTools'
PseudoR2(model2, which = c("McFadden", "CoxSnell", "Nagelkerke"))

# Prädizierte Werte (Logits) für das Gesamtmodell (für alle Fälle im Datensatz)
predict(model2, type = "link")


# Prädizierte Werte für das Gesamtmodell (zur Klassifikation neuer Personen
# mit den UV-Werten X1 = a, X2 = b, X3 = c):

# in Form bedingter Wettquotienten (Odds)
exp(predict(model2, type = "link", newdata = data.frame(X1 = a, X2 = b, X3 = c))
# Klassifikation zu (Y = 1) wenn >= 1, sonst zu (Y = 0)

# in Form bedingter Wahrscheinlichkeiten
predict(model2, type = "response", newdata = data.frame(X1 = a, X2 = b, X3 = c))
# Klassifikation zu (Y = 1) wenn >= 0.5, sonst zu (Y = 0)
```