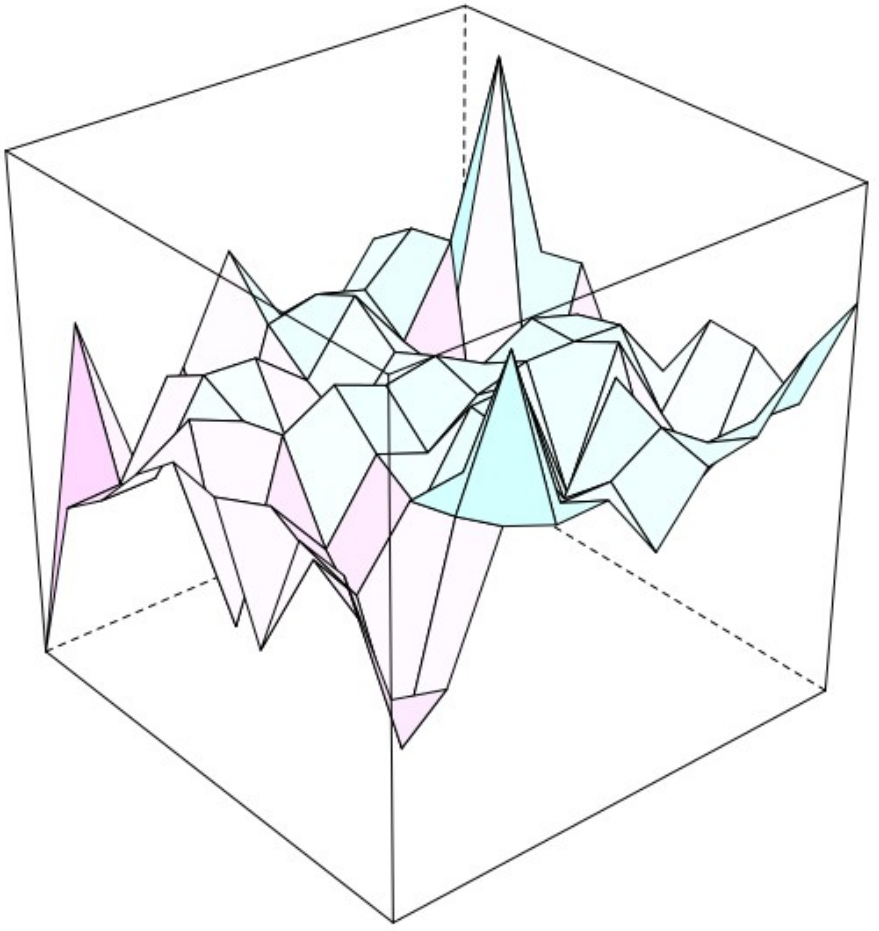
Datenanalyse mit R

Prof. Dr. Dennis Klinkhammer



(I) GRUNDLAGEN

Wissenschaftliche Gütekriterien

Wissenschaftliche Gütekriterien

- Grundlage für seriöse Forschung
- Beziehen sich auf mögliche Schwachstellen
  - des Datensatzes
  - der Erhebung des Datensatzes
  - der Forscherinnen und Forscher selbst

Wissenschaftliche Gütekriterien

- Ermöglichen Berücksichtigung von Schwachstellen in der Ergebnisinterpretation
- Unterscheidung in drei Hauptarten
  - Objektivität
  - Reliabilität
  - Validität

Wissenschaftliche Gütekriterien

- Objektivität bedeutet Unabhängigkeit
- Der Forschungsprozess wird nicht für eigene oder „Wunschergebnisse“ Dritter beeinflusst
- Insbesondere die Finanzierung von Forscherinnen und Forschern durch Dritte bietet Anlass zur Frage nach deren Objektivität

Wissenschaftliche Gütekriterien

- Reliabilität bedeutet Reproduzierbarkeit unter Einsatz geeigneter Erhebungsinstrumente
- Zentrale Aspekte
  - Stabilität und Genauigkeit einer Messung
  - Berücksichtigung und Ausweisung der Rahmenbedingungen einer Messung

Wissenschaftliche Gütekriterien

- Validität bezieht sich auf die Präzision des Erhebungsinstruments
- Drei Unterarten werden unterschieden
  - Konstruktvalidität (Womit wird gemessen?)
  - Kriteriumsvalidität (Was wird gemessen?)
  - prognostische Validität (Sind Schlüsse möglich?)

Wissenschaftliche Gütekriterien

- Validität bezieht sich auch auf die Rahmenbedingungen einer Erhebung
- Ergebnisinterpretationen erfordern eine Ausweisung der zugrundeliegenden Validität
  - Interne Validität (kontrollierte Bedingungen)
  - Externe Validität (natürliche Bedingungen)

Wissenschaftliche Gütekriterien

- Validität setzt Objektivität und Reliabilität voraus
- Forschung ohne entsprechende Ausweisung der wissenschaftlichen Gütekriterien ist keine seriöse Forschung

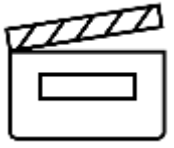
Obligatorischer Exkurs

- Simpson-Paradoxon
  - Die Bewertung verschiedener Gruppen fällt unterschiedlich aus, je nachdem ob man die Ergebnisse der Gruppen kombiniert oder nicht
  - Zusatzaufgabe: Recherche eines Beispiels

Klausurrelevant (!)

(I) GRUNDLAGEN

R und RStudio



R und RStudio

- R ist eine Programmiersprache für statistische Berechnungen und Grafiken
- RStudio ist eine grafische Benutzeroberfläche für die Programmiersprache R
- Die in der Vorlesung genannten Beispiele und Übungsaufgaben erfordern R und RStudio

R und RStudio

- R ist eine objektbasierte Programmiersprache
- Alles was existiert ist ein Objekt
  - Datensätze
  - Variablen
  - Merkmalsausprägungen

R und RStudio

- Alles was passiert ist eine Funktion
  - Funktionen bearbeiten Objekte
  - Funktionen führen zu neuen Objekten
  - Funktionen berücksichtigen Argumente
- Die Syntax bildet Objekte, Funktionen und Argumente ab und ermöglicht Dritten die Reproduktion

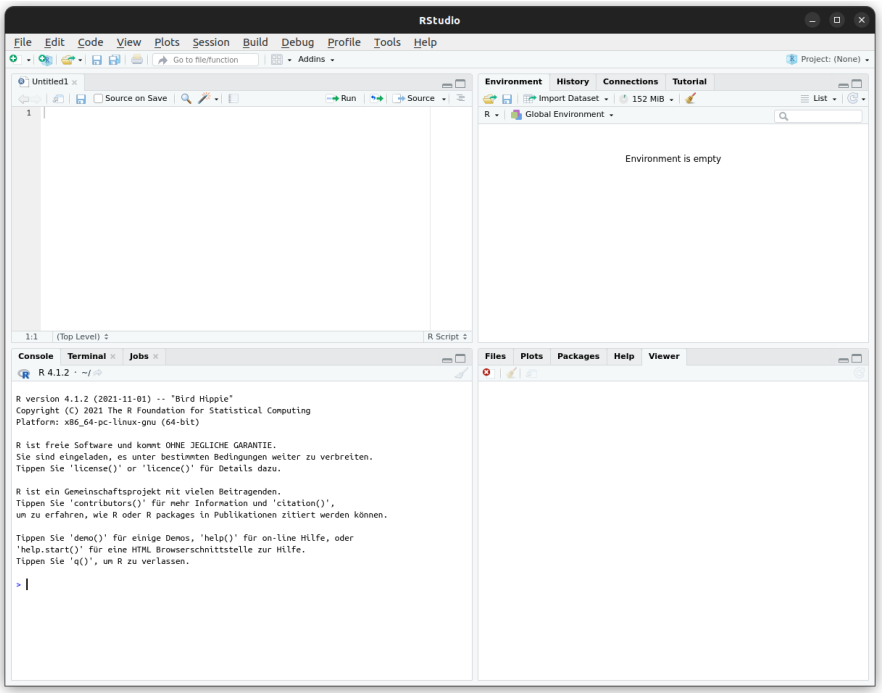
R und RStudio

- Erläuterungen werden über einen # eingegeben und dadurch von R nicht interpretiert
- Der Zuweisungspfeil legt neue Objekte an:

```
# Erläuterung mit Syntaxbeispiel
Kuchen <- backen(Milch, Mehl, Eier, Zucker)
```

R und RStudio

- RStudio: 4 Fenster
  - Syntax (links oben)
  - Konsole (links unten)
  - Historie (rechts oben)
  - Grafiken (rechts unten)



R und RStudio

- Die Funktion plot(…) kann auf den Datensatz swiss direkt angewendet oder als neues Objekt zwischengespeichert werden:

```
# Erste Beispiele
plot(swiss)
neues_objekt <- plot(swiss)
```

R und RStudio

- Packages erweitern den Funktionsumfang
- Eine Korrelation kann ohne erweiterten Funktionsumfang über die Funktion cor(…) aufgerufen werden:

```
# Ohne Package CORRPLOT
cor(swiss)
```

R und RStudio

- Die Funktionen install.packages(…) und library(…) installieren Packages und aktivieren diese in RStudio:

```
# Package CORRPLOT installieren und aktivieren
install.packages(corrplot)
library("corrplot")
```

R und RStudio

- Das neu installierte und aktivierte Package bietet zusätzlich die Funktion corrplot(…) und Abwandlungen derselben:

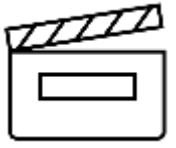
```
# Mit Package CORRPLOT
noch_ein_neues_objekt <- cor(swiss)
corrplot.mixed(noch_ein_neues_objekt)
```

R und RStudio

- Für viele Objekte und Funktionen stehen über die Funktion help(…) weiterführende Informationen bereit:

```
# Hilfe zum Package CORRPLOT
help(corrplot)
```

TREES Datensatz



TREES Datensatz

- Der TREES Datensatz ermöglicht einen ersten Einblick in die Datenanalyse mit R
- Ziele einer Datenanalyse
  - Beschreiben
  - Erklären
  - Vorhersagen

TREES Datensatz

- Die verschiedenen Ziele der Datenanalyse werden mit unterschiedlichen Teilbereichen der Statistik umgesetzt
  - Univariate Statistik
  - Bivariate Statistik
  - Multivariate Statistik

TREES Datensatz

- Ziel: Ein allgemeines Modell zur Vorhersage des Volumens von Bäumen aufstellen
- Variablen des TREES Datensatzes
  - (X_1) Girth
  - (x_2) Height
  - (Y) Volume

TREES Datensatz

- Die Funktionen summary(…) und boxplot(…) sind die gängigsten Funktionen der univariaten Statistik und beschreiben jeweils eine Variable:

```
# Univariate Statistik
summary(trees)
boxplot(trees)
```

TREES Datensatz

- Die Korrelation mit der Funktion cor(…) gehört zum Teilbereich der bivariaten Statistik
- Hier werden die Zusammenhänge zwischen zwei Variablen erklärt:

```
# Bivariate Statistik
cor(trees)
```

TREES Datensatz

- Die Zusammenhänge zwischen zwei Variablen können zusätzlich über die Funktionen plot(…) und abline(…) visualisiert werden:

```
# Bivariate Statistik
plot(trees$Volume~trees$Girth)
abline(lm(trees$Volume~trees$Girth))
```

TREES Datensatz

- In der multivariaten Statistik können mehrere unabhängige Variablen zur Vorhersage einer abhängigen Variable herangezogen werden
- Dabei kann zusätzlich die Stärke des Einflusses der unabhängigen Variablen auf die abhängige Variable analysiert werden

TREES Datensatz

- Der lineare Zusammenhang ermöglicht die Funktion lm(…) in der multivariaten Statistik:

```
# Multivariate Statistik
regression_model <- lm(Volume~., data=trees)
regression_model
summary(regression_model)
```

(I) GRUNDLAGEN

1. Übungsaufgabe

1. Übungsaufgabe

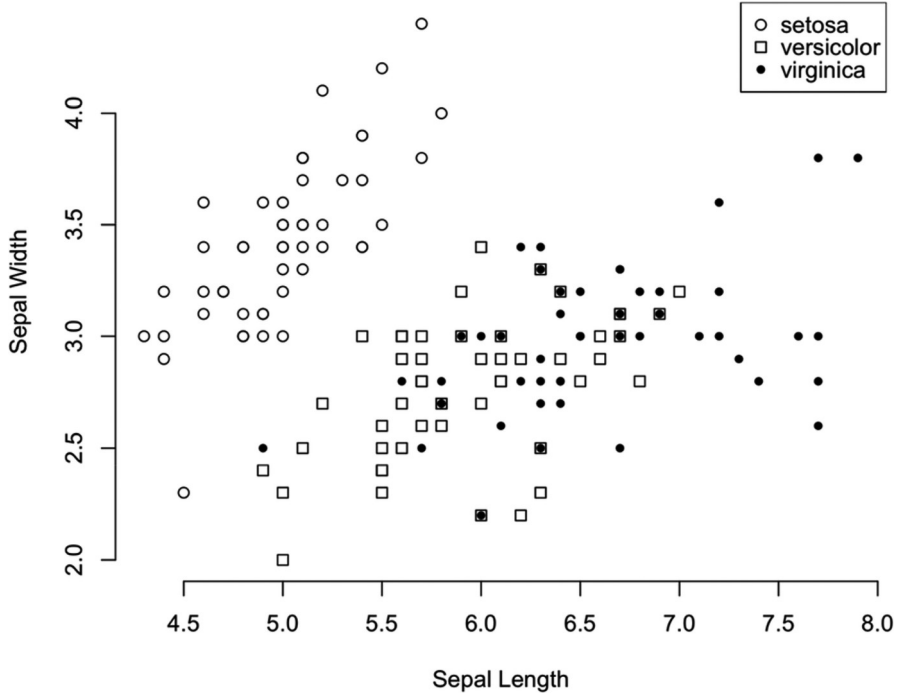
- Der IRIS Datensatz beinhaltet Variablen zu den Schwertlilienarten setosa, versicolor und virginica
- Ziel: Eine manuelle Identifikation der Schwert-lilienarten über die unabhängigen Variablen und die Funktion subset(...)

1. Übungsaufgabe

- Variablen im IRIS Datensatz
  - (x_1) Sepal.Length
  - (x_2) Sepal.Width
  - (X_3) Petal.Length
  - (X_4) Petal.Width
  - (Y) Species

1. Übungsaufgabe

- Herausforderung: Schwertlilienarten können in Länge und Breite ihrer Blätter übereinstimmen



1. Übungsaufgabe

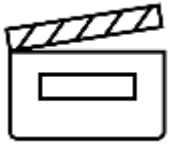
- Die Schwertlilienart setosa kann bspw. wie folgt identifiziert werden:

```
identification <- subset(iris, Petal.Length<="2" & Petal.Width<="1", select=c(Species))
summary(identification)
```

1. Übungsaufgabe

- Alle erforderlichen Funktionen, Objekte und Argumente sind in der Übungsaufgabe hervorgehoben und können entsprechend in die eigene Syntax überführt werden
- Entsprechend sind noch versicolor und virginica zu identifizieren

Analysemodelle



Analysemodelle

- Analysemodelle visualisieren die theoretisch fundierten Zusammenhänge zwischen allen unabhängigen und abhängigen Variablen
- Dadurch werden Haupt- und Nebenhypothesen abgebildet und mögliche Interdependenzen offengelegt

Analysemodelle

- Mit dem Package DiagrammeR können Analysemodelle in R visualisiert werden
- Dazu muss DiagrammeR zunächst über install.packages(…) und library(…) installiert und aktiviert werden
- Diese Installation erfordert Dependencies

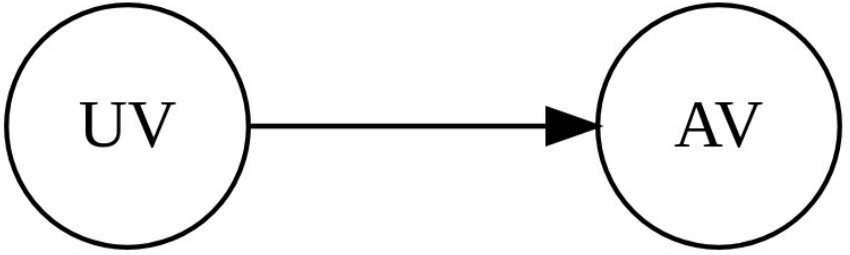
Analysemodelle

- Funktionen aus Packages lassen sich auch über :: aufrufen
- Es können label vergeben und mit -> verbunden werden

```
DiagrammeR::grViz("
digraph {graph [layout = circo]
node [shape = circle]
A [label = 'UV']
B [label = 'AV']
edge []
A -> B} ")
```

Analysemodelle

- Hypothese: Je mehr (weniger) UV, desto mehr (weniger) AV
- Dies ist ein Beispiel für eine spezifische Hypothese



Analysemodelle

- Rückblick: Der TREES Datensatz beinhaltet die Variablen Girth, Height und Volume
- In der vorherigen Analyse konnte festgestellt werden, dass alle drei Variablen voneinander abhängen

Analysemodelle

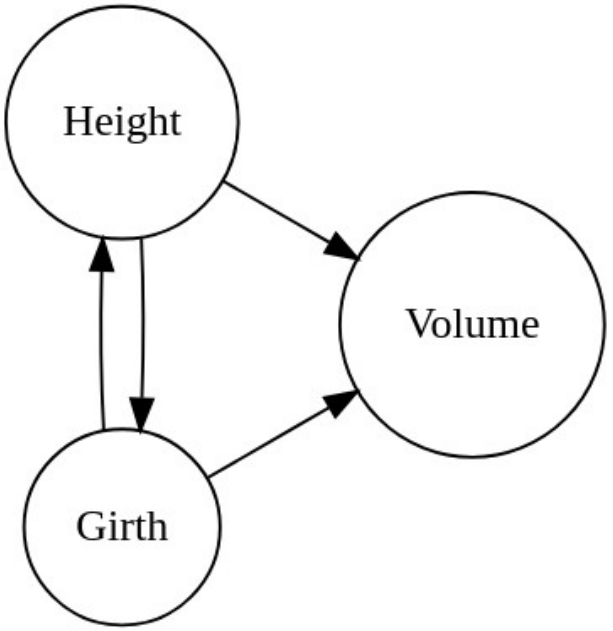
- Entsprechende Hypothesen könnten lauten
  - Je mehr Girth [Height], desto mehr Height [Girth]
  - Je mehr Height, desto mehr Volume
  - Je mehr Girth, desto mehr Volume
- Für Girth [Height] und Height [Girth] liegt eine unspezifische Hypothese vor

Analysemodelle

- Demnach beeinflussen sich die unabhängigen Variablen gegenseitig; Dies nennt man Interdependenzen
- Über die multivariate Statistik konnte ermittelt werden, dass Girth einen stärkeren Einfluss auf Volume nimmt

Analysemodelle

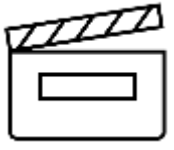
```
DiagrammeR::grViz("
digraph {graph [layout = circo]
node [shape = circle]
A [label = 'Girth']
B [label = 'Height']
C [label = 'Volume']
edge []
A -> C
A -> B
B -> A
B -> C} ")
```



Analysemodelle

- Es empfiehlt sich, zu jedem neuen Datensatz zunächst ein Analysemodell zu skizzieren
- Analysemodelle vereinfachen die Interpretation der Befunde aus der bivariaten und multivariaten Statistik

SWISS Datensatz



SWISS Datensatz

- Während der TREES Datensatz einen Einblick in die Teilbereiche der Statistik ermöglicht, bietet der SWISS Datensatz darüber hinaus einen Einblick in die Moderationseffekte der unabhängigen Variablen

SWISS Datensatz

- Die Funktionen head(…) und help(…) zeigen erste Details zum SWISS Datensatz:

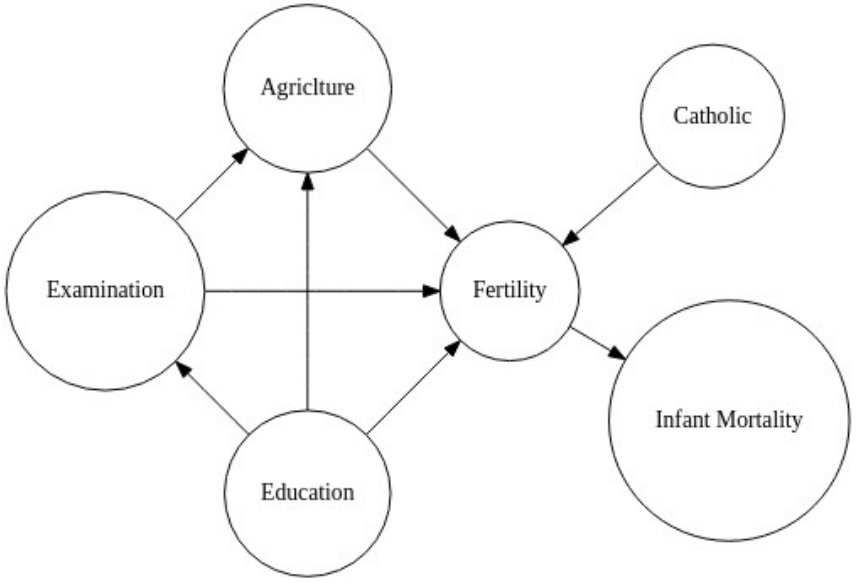
```
# Datensatz einsehen
head(swiss)
help(swiss)
```

SWISS Datensatz

- Unabhängige Variablen
  - (X_1) Agriculture
  - (X_2) Examination
  - (X_3) Education
  - (X_4) Catholic
  - (X_5) Infant.Mortality
- Abhängige Variable
  - (Y) Fertility
- Die Variablen ergeben ein komplexes Analysemodell

SWISS Datensatz

- Variablenmoderation
  - Agriculture
  - Examination
  - Education
- Auswirkungen auf die multivariate Statistik



SWISS Datensatz

- Für die univariate Statstik stehen erneut die Funktionen summary(…) und boxplot(…) zur Verfügung:

```
# Univariate Statistik
summary(swiss)
boxplot(swiss)
```

SWISS Datensatz

- Moderationseffekte lassen sich über einen Vergleich zwischen bivariater und multivariater Statistik erkennen

```
# Bivariate Statistik
cor(swiss)

# Multivariate Statistik
lm(Fertility~., data=swiss)
```

SWISS Datensatz

- Mögliche Folgen der Moderationseffekte
  - Veränderung bei den Effektstärken
  - Veränderung bei den Vorzeichen
- Die korrekte Interpretation von Moderations-effekten erfordert eine theoretische Fundierung

2. Übungsaufgabe

2. Übungsaufgabe

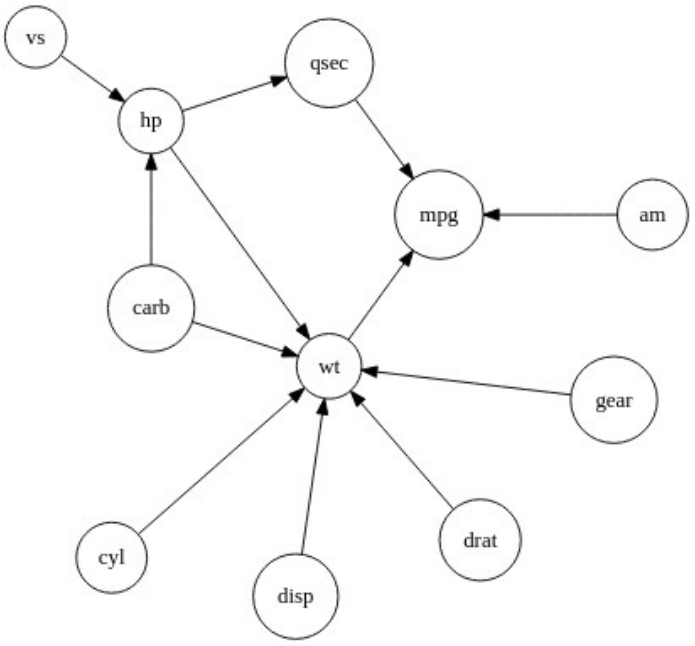
- Für die 2. Übungsaufgabe steht der MTCARS Datensatz zur Verfügung
- Dieser beinhaltet Variablen zu den technischen Eigenschaften von 32 verschiedenen Automobilen aus dem Jahr 1974

2. Übungsaufgabe

- Unabhängige Variablen
  - (X_1) cyl
  - (X_2) disp
  - (X_3) hp
  - (X_4) drat
  - (X_5) wt
  - (X_6) qsec
  - (x_7) vs
  - (x_8) am
  - (x_9) gear
  - (x_{10}) carb
- Abhängige Variablen
  - (Y_1) mpg
  - (X_6) → (Y_2) qsec
- Auch im MTCARS Datensatz ergeben die Variablen ein komplexes Analysemodell

2. Übungsaufgabe

- Variablenmoderation
  - vs
  - hp
  - carb
  - wt
  - qsec
  - cyl
  - disp
  - drat
  - gear



2. Übungsaufgabe

- Zur Analyse von naturwissenschaftlichen Datensätzen kann im Rahmen der multivariaten Statistik auf die Funktion step(…) zurückgegriffen werden:

```
# Multivariate Statistik
step(lm(data=mtcars, mpg~.), trace=0, steps=11)
```

2. Übungsaufgabe

- Alle erforderlichen Funktionen, Objekte und Argumente sind wieder in der Übungsaufgabe hervorgehoben und können entsprechend in die eigene Syntax überführt werden
- Anstatt alle unabhängigen Variablen manuell einzugeben, kann auch ein . gesetzt werden

Datengewinnung

Datengewinnung

- Häufig stellen Befragungen die Grundlage für zu analysierende Datensätze dar
- Tools zur Erstellung von Onlinebefragungen und zur Berechnung der Stichprobengröße
  - https://www.soscisurvey.de/
  - https://www.surveymonkey.de/mp/sample-size-calculator/

Datengewinnung

- Wenn Daten mittels einer Befragung und / oder Onlinebefragung gewonnen werden sollen, so sind insgesamt acht Herausforderungen in der Interpretation dieser Daten zu berücksichtigen

Datengewinnung

- Reaktivität
  - Wenn Teilnehmerinnen und Teilnehmer den Forschungsgegenstand kennen, dann kann sich daraufhin ihr (Antwort-)verhalten verändern
  - Beispiel: Gesundheitsbezogenes Verhalten

Datengewinnung

- Soziale Erwünschtheit
  - Eine mögliche Verzerrung des Antwortverhaltens aufgrund der Annahme der Teilnehmerinnen und Teilnehmer über mit der Antwort in Verbindung stehende Normen und / oder Erwartungen
  - Beispiel: Deviantes Verhalten

Datengewinnung

- Schweigeverzerrung
  - Teilnehmerinnen und Teilnehmer können ein anderes Antwortverhalten aufweisen als nicht-Teilnehmerinnen und nicht-Teilnehmer
  - Beispiel: Produktbewertungen im Internet

Datengewinnung

- Selektive Aufmerksamkeit
  - Menschliche Wahrnehmung ist ein selektiver Prozess, orientiert an vertrauten Mustern und Strukturen und somit hinsichtlich der zu verarbeitenden Informationsmenge begrenzt
  - Beispiel: Eltern sehen mehr Gefahrenquellen

Datengewinnung

- Tendenz zur Mitte
  - Bei mehrstufigen Antwortskalen bewirkt die Tendenz zur Mitte ein Antwortverhalten in der Mitte
  - Beispiel: Unreflektierte Befragungsteilnahme

Datengewinnung

- Tendenz zur Milde / Härte
  - Bei mehrstufigen Antwortskalen erfolgt an Stelle eines objektiven Antwortverhaltens eine subjektive Unter- / Überbewertung eines realen Phänomens
  - Beispiel: Sympathie bei Lehrevaluationen

Datengewinnung

- Retrospektionseffekt
  - Erlebnisse und Ereignisse können im Rückblick, bspw. am nächsten Tag, positiver oder negativer bewertet werden, als in der Situation selbst
  - Beispiel: Verkehrsunfall

Datengewinnung

- Rückschaufehler
  - Unzutreffende Erinnerungen, wenn Menschen, nachdem Sie den tatsächlichen Ausgang eines Ereignisses erfahren haben, sich falsch an ihre frühere Vorhersage des Ausgangs erinnern
  - Beispiel: Ausgang von politischen Wahlen

(II) FORMELSAMMLUNG

Formeln und Verteilungstabellen

Formeln und Verteilungstabellen

- Zusammenfassung der grundlegenden Formeln und Verteilungstabellen für die gängigsten statistischen Analyseverfahren
- Dabei bauen die Formeln der univariaten, bivariaten und multivariaten Statistik aufeinander auf

Formeln und Verteilungstabellen

- Die korrigierte Stichprobenvarianz wird bspw. für die Standardabweichung benötigt
- Tipp: Die Standardabweichung lässt sich mit einer Taste auf dem Taschenrechner aufrufen

Formeln und Verteilungstabellen

- Gleichermaßen erfordert der Korrelations-koeffizient die Standardabweichung
- Tipp: Vorherige Ergebnisse können einfach in die nächste Formel überführt werden

Formeln und Verteilungstabellen

- Schließlich führen die Standardabweichung und der Korrelationskoeffizient zu den linearen Regressionsgewichten
- Tipp: Dabei können auch die Ergebnisse aus mehreren Formeln kombiniert werden

Formeln und Verteilungstabellen

- Für manche Formeln sind darüber hinaus Verteilungstabellen mit statistischen Referenzwerten erforderlich
  - Chi-Quadrat-Test
  - t-Test

(III) DATENANALYSE

Skalenniveaus

Skalenniveaus

- Skalenniveaus werden häufig auch als Messniveaus bezeichnet
- Sie beziehen sich auf den Detailgrad der Merkmalsausprägungen einer Variable

Skalenniveaus

- Geeignete Skalenniveaus sind in der Lage, die unterschiedlichen Facetten eines realen Phänomens ausreichend abzubilden
- Für eine zielführende Datenanalyse mit R ist die Differenzierung von vier unterschiedlichen Skalenniveaus erforderlich

Skalenniveaus

- Von der Nominalskala zur Verhältnisskala nimmt der Detailgrad der Merkmalsausprägungen zu
  - Nominalskala
  - Ordinalskala
  - Intervallskala
  - Verhältnisskala

Skalenniveaus

- Die Nominalskala differenziert zwischen Merkmals-ausprägungen ohne vorgegebene Rangfolge
  - Geschlecht (männlich, weiblich, divers) → 1 ≠ 2 ≠ 3
  - Postleitzahl (50939, 53225) → 1 ≠ 2
- Ein Spezialfall sind die dichotomen Variablen
  - Postleitzahl (50939 für Köln, 53225 für Bonn) → 0 ≠ 1

Skalenniveaus

- Mit der Ordinalskala kann eine Rangfolge unter den Merkmalsausprägungen abgebildet werden, ohne diese Rangfolge im Detail interpretieren zu können
  - Einkommen (niedrig, hoch) → niedrig < hoch
  - Schulnoten (sehr gut, gut) → sehr gut > gut

Skalenniveaus

- Bei der Intervallskala lassen sich die unterschiedlichen Merkmalsausprägungen mittels Zahlen abbilden und interpretieren
  - Intelligenz (110, 100) → 110 - 10 = 100
  - Temperatur (30°C, 35°C) → 30°C + 5°C = 35°C
- Bei der Intervallskala gibt es keinen Nullpunkt

Skalenniveaus

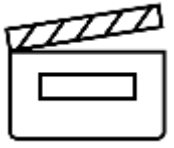
- Die Verhältnisskala erlaubt die Berechnung der exakten Beziehung zwischen den Merkmalsausprägungen
  - Einkommen (3000, 1500) → 3000 : 2 = 1500
  - Temperatur (20°K, 40°K) → 20°K * 2 = 40°K
- Die Verhältnisskala besitzt einen Nullpunkt

Skalenniveaus

- Höhere Skalenniveaus können in niedrigere Skaleniveaus überführt werden
- Umgekehrt ist dies jedoch nicht möglich
  - Rauchverhalten (ja, nein) → Anzahl an Zigaretten?
  - Einkommen (hoch) → Betrag in Euro?

(III) DATENANALYSE

Univariate Statistik



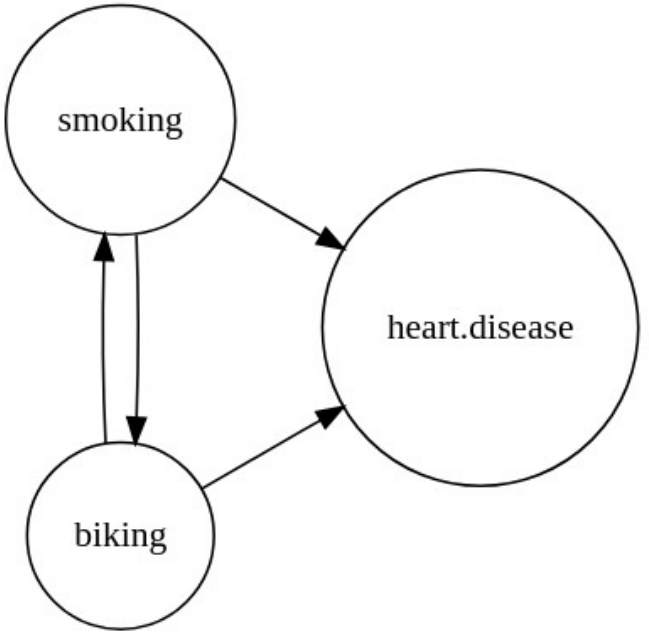
Univariate Statistik

- Für die univariate Statistik wird auf den externen HEART Datensatz zurückgegriffen, der über die Funktion read.csv(…) eingelesen werden kann:

```
# Datensatz einlesen
heart.data <- read.csv("https://raw.githubusercontent.com/statistical-thinking/datasets/main/heart.data.csv")
```

Univariate Statistik

- Dieser beinhaltet zwei unabhängige und eine abhängige Variable
  - (X_1) biking
  - (X_2) smoking
  - (Y) heart.disease



Univariate Statistik

- Wie bisher geben die Funktionen dim(…), summary(…) und boxplot(…) einen ersten Überblick über die Variablen:

```
# Univariate Statistik
dim(heart.data)
summary(heart.data)
boxplot(heart.data)
```

Univariate Statistik

- Neben dem Minimum und Maximum der Merkmalsausprägungen werden zusätzlich Quartile und Lagemaße (hier: der Median und das arithmetische Mittel) ausgewiesen

Univariate Statistik

- Quartile geben über Schwellenwerte (1. Quartil, Median, 3. Quartil) Auskunft zu vier Gruppen
  - ≤ 25% der Fälle
  - ≤ 50% der Fälle
  - ≤ 75% der Fälle
  - ≤ 100% der Fälle

Univariate Statistik

- Lagemaße sollen die zentrale Tendenz eines Datensatzes zum Ausdruck bringen
  - Arithmetisches Mittel
  - Median
  - Modus

Univariate Statistik

- Arithmetisches Mittel
  - Auch als Durchschnittswert oder Mean bekannt
  - Summe der Merkmalsausprägungen dividiert durch die Anzahl der Fälle

$$\bar{x} = \frac{1}{n}(x_1 + x_2 + x_3 + \cdots + x_n)$$

Univariate Statistik

- Median
  - Auch als Zentralwert bekannt
  - Die Merkmalsausprägung in der Mitte einer aufsteigend sortierten Liste aller Merkmalsausprägungen

$$\tilde{x}_{ungerade} = x_{\frac{n+1}{2}} \quad \text{bzw.} \quad \tilde{x}_{gerade} = \frac{1}{2}\left(x_{\frac{n}{2}} + x_{\frac{n}{2}+1}\right)$$

Univariate Statistik

- Modus
  - Auch als Modalwert bekannt
  - Bezeichnet die häufigste Merkmalsausprägung innerhalb einer Variable

$$\bar{x}_d = Häufigster\ Beobachtungswert$$

Univariate Statistik

- Sowohl die Quartile als auch die Lagemaße geben Auskunft über die Verteilung der Merkmalsausprägungen einer Variable
- Allerdings lässt sich aus diesen Funktionen nicht ableiten, wie häufig einzelne Merkmalsaus-prägungen in einer Variable vertreten sind

Univariate Statistik

- Auskunft über die Häufigkeit einzelner Merkmalsausprägungen einer Variable liefert ein Histogramm in Verbindung mit der dazugehörigen Dichtefunktion

Univariate Statistik

- Ein Histogramm kann zunächst über die Funktionen hist(…) aufgerufen werden:

```
# Verteilungsfunktion und Lagemaße
hist(heart.data$heart.disease, freq=FALSE, breaks=40,
ylim=c(0,0.10), xlim=c(-5,26))
```

Univariate Statistik

- Die dazugehörige Dichtefunktion ergibt sich daraufhin aus der Funktion curve(…):

```
curve(dnorm(x,mean=mean(heart.data$heart.disease), sd=sd(heart.data$heart.disease)), add=TRUE, lwd=5)
```

Univariate Statistik

- Anschließend lassen sich über die Funktion abline(…) die Lagemaße ergänzen:

```
abline(v=10.17, col="red")   # Mean
abline(v=10.38, col="green") # Median
abline(v=6.75, col="blue")   # Modus
```

Univariate Statistik

- Für eine passende Legende wird auf die Funktion legend(…) zurückgegriffen:

```
legend(19, 0.1, legend=c("Mean", "Median", "Modus"), col=c("red", "green", "blue"), lty=1)
```

Univariate Statistik

- Die Dichtefunktion der Variable heart.disease entspricht der sogenannten Normalverteilung
  - glockenförmig
  - asymptotisch
  - symmetrisch
  - unimodal

Univariate Statistik

- Dabei gilt: Das Integral der Dichtefunktion f(x) ist die sogenannte Verteilungsfunktion F(x)
- Die Verteilungsfunktion gibt an, wie groß die Wahrscheinlichkeit ist, dass die Merkmals-ausprägung ≤ x ist

Univariate Statistik

- Neben den Lagemaßen lassen sich auch die Streuungsmaße in den Plot überführen
- Zunächst die Funktionen var(…) und sd(…):

```
# Varianz und Standardabweichung (Teil 1)
var(heart.data$heart.disease)
sd(heart.data$heart.disease)
```

Univariate Statistik

- Streuungsmaße beziehen sich auf die Streubreite einzelner Merkmalsausprägungen um ausgewiesene Lagemaße (häufig das arithmetische Mittel)
  - Korrigierte Stichprobenvarianz
  - Standardabweichung

Univariate Statistik

- Korrigierte Stichprobenvarianz
  - In vielen Fällen einfach Varianz genannt
  - Mittlere quadratische Abweichung vom arithmetischen Mittel

$$s_x{}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Univariate Statistik

- Standardabweichung
  - Wurzel aus der korrigierten Stichprobenvarianz
  - Durchschnittliche Entfernung aller Merkmals-ausprägungen vom arithmetischen Mittel

$$s_x = \sqrt{{s_x}^2}$$

Univariate Statistik

- Mittels der Standardabweichung kann auf die 68-95-Regel zurückgegriffen werden
  - 68% der Fälle sind ± eine Standardabweichung vom arithmetischen Mittel entfernt
  - 95% der Fälle sind ± zwei Standardabweichungen vom arithmetischen Mittel entfernt

Univariate Statistik

- Zunächst wird für den Plot erneut auf die Funktion hist(…) zurückgegriffen:

```
# Varianz und Standardabweichung (Teil 2)
hist(heart.data$heart.disease, freq=FALSE, breaks=40,
ylim=c(0,0.10), xlim=c(-5,26))
```

Univariate Statistik

- Zur besseren Veranschaulichung wird auch die Dichtefunktion über die Funktion curve(…) wieder ergänzt:

```
curve(dnorm(x,mean=mean(heart.data$heart.disease), sd=sd(heart.data$heart.disease)), add=TRUE, lwd=5)
```

Univariate Statistik

- Die Funktion abline(…) ermöglicht daraufhin die Anwendung der 68-95-Regel:

```
abline(v=5.6, col="red")   # Untere Grenze (68 %)
abline(v=14.74, col="red") # Obere Grenze (68 %)
abline(v=1.03, col="green")   # Untere Grenze (95 %)
abline(v=19.31, col="green") # Obere Grenze (95 %)
```

Univariate Statistik

- Schließlich verdeutlicht die Funktion legend(…) die Bereiche, in dem sich 68% bzw. 95% der Merkmalsausprägungen befinden:

```
legend(20, 0.1, legend=c("68 %", "95 %"), col=c("red", "green"), lty=1)
```

(III) DATENANALYSE

z-Werte

z-Werte

- Der z-Wert ist die Differenz eines Rohwertes vom arithmetischen Mittel dividiert durch die Standardabweichung:

$$z = \frac{x - \bar{x}}{s_x}$$

z-Werte

- Mit der zugrundeliegenden z-Transformation werden Rohwerte in Normwerte überführt
- Normwerte ermöglichen gegenüber Rohwerten einen standardisierten Vergleich

z-Werte

- Zwei Schüler haben in unterschiedlichen Fächern an der PISA-Studie teilgenommen
  - Moritz in Mathematik
  - Fritz in Deutsch
- Beide haben 620 Punkte erzielt
  - Wer hat besser abgeschnitten?

z-Werte

- Für die z-Transformation benötigte Angaben
  - Durchschnittliche Punktezahl in Mathematik (490)
  - Standardabweichung in Mathematik (99,4)

  - Durchschnittliche Punktezahl in Deutsch (484)
  - Standardabweichung in Deutsch (110,9)

z-Werte

- Moritz in Mathematik
  - $z_M = (620 - 490)/99{,}4$
  - $z_M = 1{,}31$
- Fritz in Deutsch
  - $z_F = (620 - 484)/110{,}9$
  - $z_F = 1{,}22$

z-Werte

- Fritz hat in der Differenz zur durchschnittlichen Punktezahl in Deutsch mehr Punkte als Moritz in seiner Vergleichsgruppe erzielt
- Die unterschiedlichen Standardabweichungen in den beiden Vergleichsgruppen bedingen aber einen höheren z-Wert zugunsten von Moritz

z-Werte

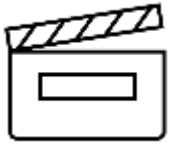
- Demnach hat Moritz ($z_M = 1{,}31$) besser abgeschnitten als Fritz ($z_F = 1{,}22$)

z-Werte

- Die z-Transformation einer Variable führt zur Standardisierung derselben, so dass aus einer Normalverteilung eine Standardnormalverteilung mit $\mu = 0$ und $\sigma = 1$ wird
- Tipp: Die Funktion scale(…) führt zu Normwerten

(III) DATENANALYSE

Bivariate Statistik



Bivariate Statistik

- Die bivariate Statistik ermöglicht das Erklären der Zusammenhänge zwischen zwei Variablen
- Gängige Verfahren der bivariaten Statistik
  - Korrelation (und Kovarianz)
  - Chi-Quadrat-Test
  - t-Test

Bivariate Statistik

- Die Korrelation misst die Stärke des (linearen) Zusammenhangs von zwei Variablen
- Der Korrelationskoeffizient ist ungerichtet, d.h. die Richtung der Stärke des Zusammenhangs muss theoretisch begründet werden
- Korrelationen sind kein Beweis für Kausalität

Bivariate Statistik

- Formal setzt sich der Korrelationskoeffizient aus der Kovarianz und den dazugehörigen Standardabweichungen zusammen

$$r_{xy} = \frac{\hat{\sigma}_{xy}}{s_x * s_y} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} * \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Bivariate Statistik

- Die zugrundeliegende Kovarianz ist eine Erweiterung der empirischen Stichproben-varianz um eine zweite Variable

$$\hat{\sigma}_{xy} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

Bivariate Statistik

- Demnach ist die Kovarianz primär dazu geeignet, einen positiven oder negativen Zusammenhang zu identifizieren
- Eine standardisierte Interpretation dieses Zusammenhangs ist allerdings nur mit der Korrelation möglich

Bivariate Statistik

- Die Standardisierung bedingt die Ausprägungen des Korrelationskoeffizienten
  - perfekt negativer Zusammenhang (- 1,00)
  - perfekt positiver Zusammenhang (+ 1,00)
  - kein Zusammenhang (± 0,00)

Bivariate Statistik

- Der Blick für die Stärke des Zusammenhangs einer Korrelation zwischen zwei Variablen lässt sich natürlich auch ganz anschaulich trainieren
  - http://www.guessthecorrelation.com/

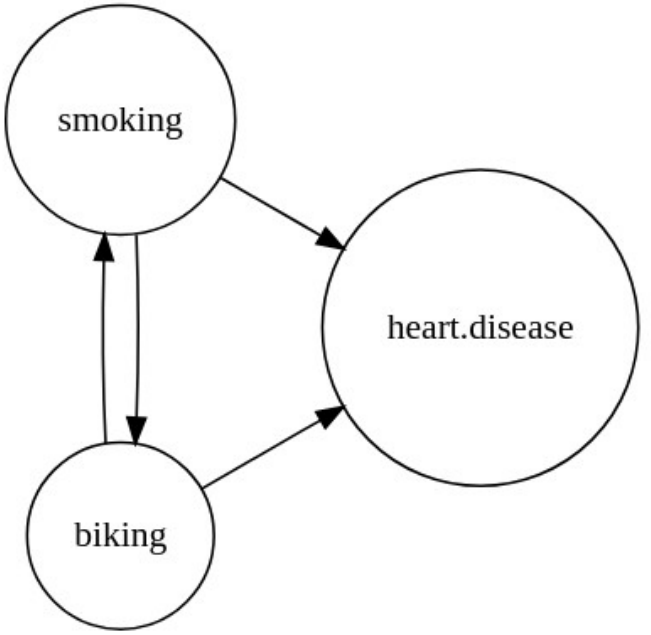
Bivariate Statistik

- Für die Umsetzung in R kann wieder auf den externen HEART Datensatz über die Funktion read.csv(…) zurückgegriffen werden:

```
# Datensatz einlesen
heart.data <- read.csv("https://raw.githubusercontent.com/statistical-thinking/datasets/main/heart.data.csv")
```

Bivariate Statistik

- Diesmal geht es um den Zusammenhang der Variablen
  - (X_1) biking
  - (X_2) smoking
  - (Y) heart.disease



Bivariate Statistik

- Bei überschaubaren Datensätzen lassen sich die Korrelationskoeffizienten in der Korrelations-matrix über die Funktion cor(…) aufrufen:

```
# Korrelation (Teil 1)
cor(heart.data)
```

Bivariate Statistik

- Der Zusammenhang zwischen zwei Variablen lässt sich darüber hinaus mit den Funktionen plot(…) und abline(…) visualisieren:

```
# Korrelation (Teil 2)
plot(heart.data$heart.disease~heart.data$biking)
abline(lm(heart.data$heart.disease~heart.data$biking), col="green")
```

Bivariate Statistik

- Demnach setzt die (lineare) Korrelation mindestens intervall- oder verhältnisskalierte Variablen voraus
- Liegen hingegen zwei nominalskalierte Variablen vor, kann auf den Chi-Quadrat-Test zurückgegriffen werden

Bivariate Statistik

- Weil der HEART Datensatz keine nominal-skalierten Variablen beinhaltet, werden diese zunächst über die Funktion ifelse(…) generiert:

```
# Chi-Quadrat-Test
nominal_y <- ifelse(heart.data$heart.disease > 10.17, 1, 0)
nominal_x <- ifelse(heart.data$smoking > 15.43, 1, 0)
```

Bivariate Statistik

- Nominalskalierte Variablen lassen sich über den table(…) Befehl als Tabelle aufrufen:

  table(nominal_y, nominal_x)

- Schließlich wird der Chi-Quadrat-Test mittels der Funktion chisq.test(…) durchgeführt:

  chisq.test(nominal_y, nominal_x)

Bivariate Statistik

- Die zugrundeliegende Formel des Chi-Quadrat-Tests vergleicht beobachtete Werte (n_j) mit erwarteten Werten (n_{j0})

$$\chi^2 = \sum_{j=1}^{m} \frac{(n_j - n_{j0})^2}{n_{j0}}$$

Bivariate Statistik

- Die beobachteten Werte (n_j) befinden sich in der zuvor generierten Tabelle mit den nominal-skalierten Variablen
- Ausgehend von den beobachteten Werten (nj) soll die Hypothese (H) untersucht werden, dass ein Zusammenhang zwischen x und y vorliegt

Bivariate Statistik

- Beim Chi-Quadrat-Test wird die Hypothese (H) allerdings nicht direkt überprüft, sondern indirekt über die Nullhypothese (H_0)
- Die Nullhypothese (H_0) geht von keinem Zusammenhang aus und bedingt dadurch die erwarteten Werte (n_{j0})

Bivariate Statistik

- Beobachtete Werte (n_j)

	x_0	x_1
y_0	140	101
y_1	105	152

- H: Rauchen (x) führt zu Herzerkrankungen (y)

- Erwartete Werte (n_{j0})

?

- H_0: Rauchen (x) führt nicht zu Herzerkrankungen (y)

Bivariate Statistik

- Wahrscheinlichkeit für eine Herzerkrankung
  - $P(y_1) = (105 + 152) / 498$
  - $P(y_1) = 51{,}6\%$
- Wahrscheinlichkeit für keine Herzerkrankung
  - $P(y_0) = 100\% - 51{,}6\%$
  - $P(y_0) = 48{,}4\%$

Bivariate Statistik

- Wenn die Nullhypothese (H_0) gilt, dann müssten diese Wahrscheinlichkeiten unabhängig vom Rauchverhalten (x_0 und x_1) sein
  - 51,6% mit und 48,4% ohne Herzerkrankung bei x_0
  - 51,6% mit und 48,4% ohne Herzerkrankung bei x_1

Bivariate Statistik

- Die Wahrscheinlichkeiten sind also gleich verteilt und ergeben die erwarteten Werte (n_{j0})
  - Für x_0: 245 * 51,6% = 126
  - Für x_0: 245 * 48,4% = 119
  - Für x_1: 253 * 51,6% = 131
  - Für x_1: 253 * 48,4% = 122

Bivariate Statistik

- Beobachtete Werte (n_j)

	x_0	x_1
y_0	140	101
y_1	105	152

- H: Rauchen (x) führt zu Herzerkrankungen (y)

- Erwartete Werte (n_{j0})

	x_0	x_1
y_0	119	122
y_1	126	131

- H_0: Rauchen (x) führt nicht zu Herzerkrankungen (y)

Bivariate Statistik

- Der berechnete Chi-Quadrat-Wert beträgt
  - $\chi^2 \approx (140 - 119)^2 / 119 + (101 - 122)^2 / 122 + (105 - 126)^2 / 126 + (152 - 131)^2 / 131$
  - $\chi^2 \approx 15$
- Der Chi-Quadrat-Wert wird mit den Referenz-werten der Chi-Quadrat-Wert-Verteilungstabelle verglichen

Bivariate Statistik

- Referenzwerte in Abhängigkeit von Freiheits-graden (dF) und Irrtumswahrscheinlichkeiten (α)

			1 - α			
Freiheitsgrade	**00,85**	**00,90**	**00,95**	**00,975**	**00,99**	**00,995**
1	02,07	02,71	03,84	05,02	06,63	07,88
2	03,79	04,61	05,99	07,38	09,21	10,60
3	05,32	06,25	07,81	09,35	11,34	12,84
4	06,74	07,78	09,49	11,14	13,28	14,86
5	08,12	09,24	11,07	12,83	15,09	16,75
(…)	(…)	(…)	(…)	(…)	(…)	(…)

Bivariate Statistik

- Berechnung der Freiheitsgrade (dF)
  - dF = (Zeilen – 1) * (Spalten – 1)
  - dF = (2 – 1) * (2 – 1)
  - dF = 1
- Die Irrtumswahrscheinlichkeit steht für die Wahrscheinlichkeit, dass H_0 irrtümlich abgelehnt wird

Bivariate Statistik

- Ist der Chi-Quadrat-Wert größer als der Referenz-wert, wird die Nullhypothese (H_0) verworfen
- Dies bedeutet im vorliegenden Beispiel
  - 15 > 7,88 (Referenzwert)
  - 99,5% Sicherheit in der Ablehnung von H_0
  - H: Rauchen (x) führt zu Herzerkrankungen (y)

Bivariate Statistik

- Ein ähnliches Verfahren unter Rückgriff auf Referenzwerte zeigt sich beim t-Test
- Dabei geht es um den Zusammenhang zwischen einer mindestens intervallskalierten abhängigen Variable und einer nominal-skalierten unabhängigen Variable

Bivariate Statistik

- Die nominalskalierte unabhängige Variable ermöglicht dem t-Test u.a. Gruppenvergleiche
  - Arithmetisches Mittel der Gruppe x_1
  - Arithmetisches Mittel der Gruppe x_2
- Tipp: Die Gruppen können auch nach den Merkmalsausprägungen 0 oder 1 benannt sein

Bivariate Statistik

- Formal ist dies durch den standardisierten Vergleich beider arithmetischer Mittel in Abhängigkeit von der Anzahl der Fälle (N) möglich

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}}}$$

Bivariate Statistik

- Beim t-Test beinhaltet die t-Wert-Verteilungs-tabelle die entsprechen Referenzwerte

			1 - α			
Freiheitsgrade	**00,85**	**00,90**	**00,95**	**00,975**	**00,99**	**00,995**
20	01,06	01,32	01,72	02,09	02,53	02,86
40	01,05	01,30	01,68	02,02	02,42	02,70
60	01,04	01,29	01,67	02,00	02,39	02,66
80	01,04	01,29	01,66	01,99	02,37	02,63
100	01,04	01,29	01,66	01,98	02,36	02,62
(…)	(…)	(…)	(…)	(…)	(…)	(…)

Bivariate Statistik

- Anwendung und Interpretation erfolgen analog dem für den Chi-Quadrat-Test vorgestellten Beispiel
- Tipp: Der hier vorgestellte t-Test ist eigentlich der Welch-Test, bei dem die intervallskalierte abhängige Variable nicht normalverteilt sein muss

Bivariate Statistik

- Zur Visualisierung eines Gruppenvergleichs werden im HEART Datensatz Gruppen über die Funktion subset(…) angelegt:

```
# t-Test (Teil 1)
low_smoking_areas <- subset(heart.data,
heart.data$smoking < 15.43)
high_smoking_areas <- subset(heart.data,
heart.data$smoking > 15.43)
```

Bivariate Statistik

- Über die Funktion par(…) werden die beiden Funktionen boxplot(...) zusammengefasst:

```
par(mfrow=c(1,2))
boxplot(low_smoking_areas[c(3)], ylim=c(0, 22),
main="Low Smoking Areas (0)")
boxplot(high_smoking_areas[c(3)], ylim=c(0, 22),
main="High Smoking Areas (1)")
```

Bivariate Statistik

- Schließlich lassen sich die arithmetischen Mittel der beiden Gruppen auch direkt über die Funktion t.test(…) vergleichen:

```
# t-Test (Teil 2)
t.test(heart.data$heart.disease~nominal_x)
```

Obligatorischer Exkurs

- Exponentialfunktion mit negativem Exponenten:

```
x <- seq(0,50,1)
y <- runif(1,5,5)*exp(-runif(1,0.1,0.1)*x)+rnorm(51,0,0.5)
a_start <- 10
b_start <- 2*log(2)/a_start
m <- nls(y~a*exp(-b*x), start=list(a=a_start, b=b_start))
plot(x,y)
lines(x, predict(m), col="blue", lty=2,lwd=3)
```

Klausurrelevant (!)

Obligatorischer Exkurs

- Michaelis-Menten-Gleichung:

```
x <- seq(0,50)
y <- (runif(1,10,20)*x)/(runif(1,0,10)+x)+rnorm(51,0,1)
a_start <- 10
b_start <- 2*log(2)/a_start
m <- nls(y~a*x/(b+x), start=list(a=a_start, b=b_start))
plot(x,y)
lines(x, predict(m), lty=2, col="blue", lwd=3)
```

Klausurrelevant (!)

(III) DATENANALYSE

3. Übungsaufgabe

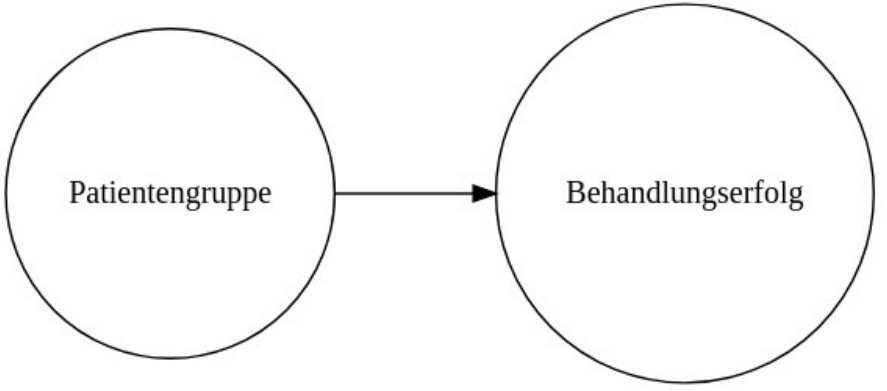
3. Übungsaufgabe

- In dieser Übungsaufgaben stehen zwei nominalskalierte Variablen zur Verfügung
  - Patientinnen und Patienten in Gruppe A oder B
  - Behandlung erfolgreich bzw. nicht erfolgreich
- Diesmal ist der Datensatz nicht in R enthalten

3. Übungsaufgabe

- Nominalskalierte Variablen
  - Patientengruppe (X)
  - Behandlungserfolg (Y)

- Hypothese: Behandlungs-erfolg in Gruppe A größer als in Gruppe B



3. Übungsaufgabe

- Über die Funktionen cbind(…) und rbind(…) kann der Datensatz aus der Übungsaufgabe in R überführt werden:

```
# Zusätzliche Syntax für R (Teil 1)
erfolgreich <- cbind(70, 55)
nicht_erfolgreich <- cbind(30, 45)
matrix <- rbind(erfolgreich, nicht_erfolgreich)
```

3. Übungsaufgabe

- Die Funktion chisq.test(…) ermittelt in R das exakte Signifikanzniveau:

```
# Zusätzliche Syntax für R (Teil 2)
chisq.test(matrix)
```

3. Übungsaufgabe

- Ziele dieser Übungsaufgabe (ohne R)
  - Nullhypothese aufsetzen
  - Erwartete Werte bestimmen
  - Chi-Quadrat-Wert bestimmen
  - Nullhypothese mit Referenzwert prüfen

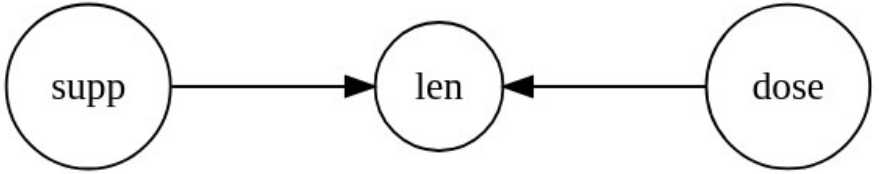
4. Übungsaufgabe

4. Übungsaufgabe

- Die Übungsaufgabe zum TOOTHGROWTH Datensatz kann mit dem t-Test gelöst werden
- Hierbei geht es um den Einfluss von Orangen-saft oder purem Vitamin C als Supplement (Variable: supp) auf das Wachstum der Zähne (Variable: len) von Meerschweinchen

4. Übungsaufgabe

- Insgesamt stehen drei Variablen zur Verfügung
  - supp (X_1)
  - dose (X_2)
  - len (Y)



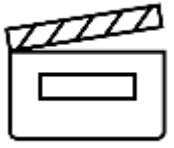
4. Übungsaufgabe

- Dabei sollen die Unterschiede zwischen den Gruppen der Meerschweinchen zunächst über die Funktionen subset(…) und boxplot(…) visualisiert werden:

```
# Vitamin C Meerschweinchen
vc <- subset(ToothGrowth, supp=="VC")
boxplot(vc[c(1,3)], main="VC")
```

(III) DATENANALYSE

Multivariate Statistik



Multivariate Statistik

- Das Prinzip der multivariaten Statistik, also der Vorhersage einer abhängigen Variable über mehrere unabhängige Variablen, kann über zwei Verfahren veranschaulicht werden
  - Lineare Regression
  - Logistische Regression

Multivariate Statistik

- Bei der (einfachen) linearen Regression wird über den Korrelationskoeffizienten und die dazugehörigen Standardabweichungen das Regressionsgewicht (b_1) der unabhängigen Variable bestimmt

$$b_1 = r_{xy} * \frac{s_y}{s_x}$$

Multivariate Statistik

- Mit dem Regressionsgewicht (b_1) und dem arithmetischen Mittel für y und x_1 lässt sich daraufhin der Schnittpunkt der Regressions-geraden mit der y-Achse (a) bestimmen

$$a = \bar{y} - b_1 \bar{x}_1$$

Multivariate Statistik

- Regressionsanalysen geben zusätzlich Auskunft über den Grad an Varianzaufklärung
- Der Determinationskoeffizient (R^2) basiert bei der (einfachen) linearen Regression auf dem Korrelationskoeffizienten

$$R^2 = (r_{xy})^2$$

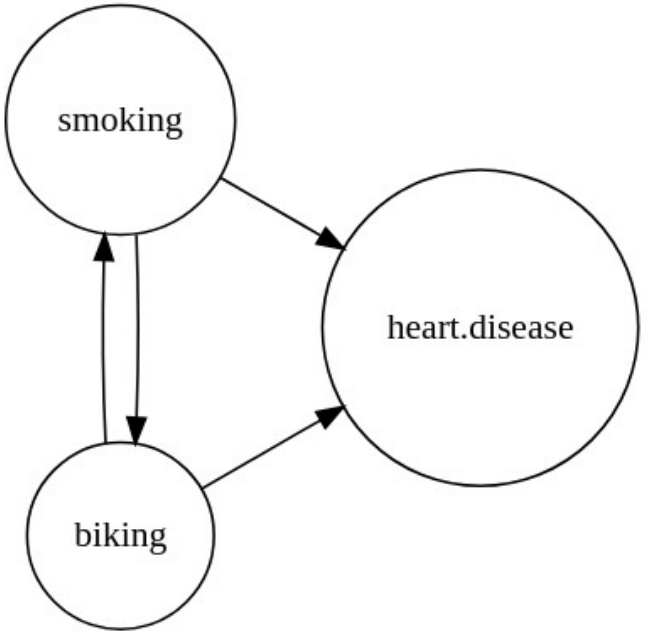
Multivariate Statistik

- Um dies zu veranschaulichen wird ein letztes Mal auf den externen HEART Datensatz über die Funktion read.csv(…) zurückgegriffen:

```
# Datensatz einlesen
heart.data <- read.csv("https://raw.githubusercontent.com/statistical-thinking/datasets/main/heart.data.csv")
```

Multivariate Statistik

- Diesmal geht es um die Vorhersage von heart.disease
  - (X_1) biking
  - (X_2) smoking
  - (Y) heart.disease



Multivariate Statistik

- Eine (einfache) lineare Regression kann über die Funktion lm(…) aufgerufen werden:

```
# Multivariate Statistik (Teil 1)
(lm(heart.disease~biking, data=heart.data))
```

- Tipp: Über die äußere (…) lässt sich das Regressionsmodell bei Bedarf mit anderen Funktionen kombinieren, bspw. mit summary(...)

Multivariate Statistik

- Die Kennwerte dieses Regressionsmodells lassen sich mit den Funktionen plot(…) und abline(…) visualisieren:

```
plot(heart.data$heart.disease~heart.data$biking)
abline(lm(heart.data$heart.disease~heart.data$biking), col="green")
```

Multivariate Statistik

- Das Regressionsgewicht (b_1) entspricht dabei der Steigung der linearen Funktion
- Der Schnittpunkt mit der y-Achse (a) kann als theoretisch begründbarer Ausgangspunkt der linearen Funktion betrachtet werden

Multivariate Statistik

- Die Kennwerte der (einfachen) linearen Regression für die unabhängige Variable smoking lassen sich ebenfalls über die Funktion lm(…) aufrufen:

```
(lm(heart.disease~biking+smoking, data=heart.data))
```

Multivariate Statistik

- Die Funktion lm(…) ermöglicht auch die lineare Regression mit beiden unabhängigen Variablen:

```
# Multivariate Statistik (Teil 2)
reg1 <- (lm(heart.disease~., data=heart.data))
summary(reg1)
```

- Die Funktion summary(…) gibt dabei eine zusätzliche Modellzusammenfassung aus

Multivariate Statistik

- In der Modellzusammenfassung sind insbesondere die Residuen von Interesse
- Dabei geht es um die “Fehler“ in der Vorhersagegenauigkeit des Regressionsmodells

Multivariate Statistik

- Die Residuen sollten bei der linearen Regression normalverteilt sein
- Dies lässt sich über die Modellzusammen-fassung mit der Funktion summary(…) überprüfen

Multivariate Statistik

- Darüber hinaus sollten die Residuen keine Muster beinhalten, die auf eine nicht-lineare Vorhersagbarkeit schließen lassen
- Dies ist bspw. bei Heteroskedastizität der Fall: Die Varianz der Residuen variiert mit zunehmenden bzw. abnehmenden Werten des Prädiktors

Multivariate Statistik

- Die Residuen eines Regressionsmodells können über die Funkion lm(…) auf solche Muster hin überprüft werden:

  plot(reg1$residuals)

Multivariate Statistik

- Wenn die abhängige Variable lediglich als nominalskalierte Variable vorliegt, so kann auf die logistische Regression zurückgegriffen werden

Multivariate Statistik

- Zur Veranschaulichung wird über die Funktion ifelse(…) eine nominalskalierte abhängige Variable generiert:

```
# Multivariate Statistik (Teil 3)
nominal_y <- ifelse(heart.data$heart.disease > 10.17, 1, 0)
```

Multivariate Statistik

- Die Herausforderung in der Interpretation des Zusammenhangs zeigt sich in der Visualisierung über die Funktion plot(…):

```
plot(nominal_y~heart.data$biking)
```

Multivariate Statistik

- Einer nominalskalierten abhängigen Variablen fehlen kleinschrittige Merkmalsausprägungen, so dass eine lineare Funktion nicht zur Vorhersage geeignet ist
- In diesem Fall wird auf eine logistische Funktion mit S-förmigen Verlauf zurückgegriffen

Multivariate Statistik

- Dafür wird in der Funktion glm(…) das Argument family=binomial spezifiziert:

  reg2 <- glm(nominal_y~heart.data$biking, family=binomial)
  summary(reg2)

- Achtung: Die Estimates sind nicht wie lineare Regressionsgewichte zu interpretieren!

Multivariate Statistik

- Die nominalskalierte abhängige Variable wird in der logistischen Regression mit logarithmierten Odds Ratios als Regressionsgewichte vorhergesagt
- Dazu ein Beispiel: Die logarithmierte Odds Ratio ein nerdiger Star Wars Fan zu sein

Multivariate Statistik

- 7 von 10 Nerds sind Star Wars Fans
- 4 von 10 Normalos sind Star Wars Fans
  - 7 / 10 = 70% als Wahrscheinlichkeit für Nerds
  - 4 / 10 = 40% als Wahrscheinlichkeit für Normalos
  - 70% / (1 – 70%) = 2,33 als Odds für Nerds
  - 40% / (1 – 40%) = 0,66 als Odds für Normalos

Multivariate Statistik

- Die Odds von 2,33 für einen Nerd und 0,66 für KEINEN Nerd ergeben demnach…
  - 2,33 / 0,66 = 3,5 als Odds Ratio
  - ln(3,5) = 1,25 als logarithmierte Odds Ratio
- Dieser Wert wird auch Logit genannt
  - Logit = ln(Odds Ratio)

Obligatorischer Exkurs

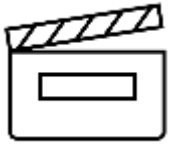
- Logarithmierte Odds Ratio eines nerdigen Star Wars Fans:

```
starwars <- data.frame(
Nerd=c(1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0),
Fan=c(1,1,1,1,1,1,1,1,0,0,0,1,1,1,1,0,0,0,0,0,0))
glm(data=starwars, Fan~Nerd, family=binomial)
```

Klausurrelevant (!)

(III) DATENANALYSE

Komplexitätsreduktion



Komplexitätsreduktion

- Datensätze können viele Variablen und Fälle beinhalten, die sich dennoch (theoriegeleitet) zusammenfassen lassen
  - Variablen der Bildung vs. des Einkommens
  - Fälle im Schwimmverein vs. Basketballverein

Komplexitätsreduktion

- Bekannte Verfahren der Komplexitätsreduktion
  - Faktorenanalyse (bzgl. Variablen)
  - Clusteranalyse (bzgl. Fälle)
- Zunächst wird eine Einführung in die Faktorenanalyse gegeben

Komplexitätsreduktion

- Die Packages PSYCH und CORRPLOT beinhalten Funktionen für die Faktorenanalyse und können nach erfolgreicher Installation über die Funktion library(…) aufgerufen werden:

```
# Funktionsumfang von R erweitern
library(psych)
library(corrplot)
```

Komplexitätsreduktion

- Als Beispiel wird der BFI Datensatz des Packages PSYCH herangezogen
- Erste Details zum BFI Datensatz verraten die Funktionen dim(…) und summary(…):

```
# Deskriptive Statistik
dim(bfi)
summary(bfi)
```

Komplexitätsreduktion

- Fehlwerte (NA) werden über die Funktion na.omit(…) ausgeschlossen und relevante Variablen in den Spalten 11 bis 20 fokussiert:

```
data <- na.omit(bfi[c(11:20)])
dim(data)
summary(data)
```

Komplexitätsreduktion

- Variablen der <u>Extraversion</u>
  - (X_1) E1
  - (X_2) E2
  - (X_3) E3
  - (X_4) E4
  - (X_5) E5
- Variablen des <u>Neurotizismus</u>
  - (X_6) N1
  - (X_7) N2
  - (X_8) N3
  - (X_9) N4
  - (X_{10}) N5

Komplexitätsreduktion

- Fasst man die Variablen von Extraversion zusammen, lässt sich das arithmetische Mittel des Summenscores über die Funktion mean(…) aufrufen:

```
# Summenscores vergleichen
extraversion_sum <-
(data$E1+data$E2+data$E3+data$E4+data$E5)/5
mean(extraversion_sum)
```

Komplexitätsreduktion

- Analog ist dies für Neurotizismus über die Funktion mean(…) möglich:

  ```
  neuroticism_sum <-
  (data$N1+data$N2+data$N3+data$N4+data$N5)/5
  mean(neuroticism_sum)
  ```

- Die arithmetischen Mittel unterscheiden sich

Komplexitätsreduktion

- Zusätzlich bestätigt die Funktion cor(...), dass die Variablen stärker innerhalb ihrer theoretischen Konstrukte korrelieren:

```
# Bivariate Statistik
cor_matrix <- cor(data)
corrplot(cor_matrix)
```

Komplexitätsreduktion

- Schließlich führen die Funktionen fa.parallel(…) und fa(…) zur Faktorenanalyse:

```
# Faktorenanalyse
fa.parallel(data, fa="both")
factors <- fa(data, nfactors=2, rotate="varimax")
factors
```

Komplexitätsreduktion

- Die Funktion plot(…) visualisiert die Befunde:

  plot(factors)

- Faktorenanzahl und Rotationsverfahren lassen sich über die Funktion fa(…) variieren, sollten aber theoretisch begründet werden können:

  fa(data, nfactors=3, rotate="varimax")

Komplexitätsreduktion

- Auswahl an Rotationsverfahren
  - Varimax (Reduktion der Varianz)
  - Promax (wie Varimax mit Standardisierung)
- Rotationsverfahren drehen das zugrunde-liegende Koordinatensystem, bis ein Kriterium erfüllt wird (bspw. Anzahl an Faktoren)

Komplexitätsreduktion

- Tipp: Ein Abgleich der Faktorenanalyse mit den Befunden der ursprünglichen Studie ist über die Funktion help(...) möglich:

  help(bfi)

Komplexitätsreduktion

- Für die Clusteranalyse steht das Package CLUSTER zur Verfügung, welches nach erfolgreicher Installation über die Funktion library(cluster) aufgerufen wird:

```
# Funktionsumfang von R erweitern
library(cluster)
```

Komplexitätsreduktion

- Hierbei wird auf den bereits bekannten IRIS Datensatz und die Variablen zu den Schwertlilienarten setosa, versicolor und virginica zurückgegriffen

Komplexitätsreduktion

- Variablen im IRIS Datensatz
  - (x_1) Sepal.Length
  - (x_2) Sepal.Width
  - (X_3) Petal.Length
  - (X_4) Petal.Width
  - (Y) Species < - - - Cluster (!)

Komplexitätsreduktion

- Die Funktionen dim(…) und summary(…) beschreiben den IRIS Datensatz:

```
# Deskriptive Statistik
dim(iris)
summary(iris)
```

Komplexitätsreduktion

- Mittels z-Transformation und der Funktion scale(…) werden die unabhängigen Variablen standardisiert:

```
# Relevante Variablen z-transformieren
cluster_data <- scale(iris[c(1:4)])
summary(cluster_data)
```

Komplexitätsreduktion

- Danach werden über die Funktion sum(...) die Within Cluster Sum of Squares (WSS) herangezogen, um die Anzahl an Clustern zu ermitteln:

```
# Within Cluster Sum of Squares (WSS) ermitteln
wss <- (nrow(cluster_data)-1)*sum(apply(cluster_data,2,var))
for (i in 2:10) wss[i] <- sum(kmeans(cluster_data, centers=i)
$withinss)
```

Komplexitätsreduktion

- Die WSS lassen sich über die Funktion plot(…) wieder entsprechend visualisieren:

```
# Grafische Darstellung der Anzahl an Clustern (WSS)
plot(1:10, wss, type="c", xlab="Cluster", ylab="WSS",
main="Clusteranzahl in Abhängigkeit von WSS")
```

Komplexitätsreduktion

- Mit der entsprechenden Anzahl an Clustern wird die Clusteranalyse bspw. über die Funktion kmeans(…) aufgerufen:

```
# K-Means Algorithmus anwenden
k_means_cluster <- kmeans(iris[,-5], 3, nstart=30)
clusplot(iris, k_means_cluster$cluster, color=TRUE,
shade=TRUE, lines=0)
```

Komplexitätsreduktion

- Die Präzision der Clusteranalyse lässt sich über die Funktion table(…) überprüfen:

```
# Präzision des K-Means Algorithmus
table(iris$Species, k_means_cluster$cluster)
```

5. Übungsaufgabe

5. Übungsaufgabe

- Diese Übungsaufgabe orientiert sich am Beispiel zum BFI Datensatz
- Diesmal sind alle fünf Faktoren mittels Faktorenanalyse zu identifizieren

5. Übungsaufgabe

- Alle benötigten Informationen sind bereits in der Übungsaufgabe enthalten und können ggfls. über die Funktion help(…) spezifiziert werden:

  help(bfi)

- Tipp: Die Packages PSYCH und CORRPLOT setzen Dependencies voraus (!)

(IV) MACHINE LEARNING

Theoretische Einführung

Theoretische Einführung

- Die in einem Datensatz enthaltenen Informationen werden für automatisierte Verallgemeinerungen herangezogen
- Dazu werden die im Datensatz enthaltenen Informationen in erlernbare Beispiele überführt

Theoretische Einführung

- Algorithmen sollen dabei Muster und Gesetz-mäßigkeiten über die erlernbaren Beispiele hinaus identifizieren können
- Algorithmen sind Handlungsvorschriften, oftmals unter Rückgriff auf statistische Formeln

Theoretische Einführung

- Beim Machine Learning sind zwei Vorgehens-weisen voneinander zu unterscheiden
  - Supervised Machine Learning
  - Unsupervised Machine Learning

Theoretische Einführung

- Beim Supervised Machine Learning werden Input und Output definiert und mittels Algorithmen in ein Modell überführt
- Dieses Modell soll den Output vorhersagen
  - Prediction (bspw. Lineare Regression)
  - Classification (bspw. Logistische Regression)

Theoretische Einführung

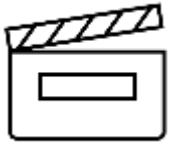
- Beim Unsupervised Machine Learning versuchen die Algorithmen zunächst Muster und Gesetzesmäßigkeiten zu erfassen und daraufhin in ein Modell zu überführen
  - Dimensionality Reduction (bspw. Faktorenanalyse)
  - Clustering (bspw. Clusteranalyse)

Theoretische Einführung

- Mögliche Anwendungsgebiete
  - Bilderkennung (bspw. auf dem Smartphone)
  - Sentiment Analysis (bspw. auf Social Media)
  - eMail Klassifizierung (bspw. SPAM identifizieren)

(IV) MACHINE LEARNING

Praktische Einführung



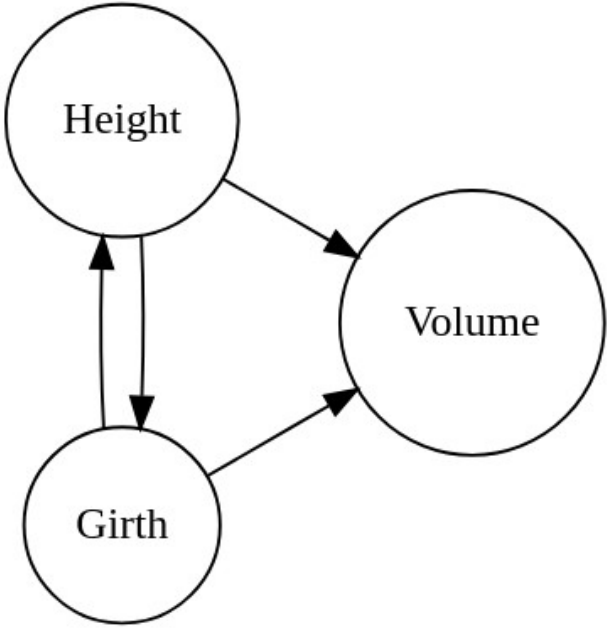
Praktische Einführung

- Für die praktische Einführung wird erneut auf den bereits bekannten TREES Datensatz zurückgegriffen
- Die Funktion dim(…) gibt dabei Auskunft über die Anzahl der enthaltenen Fälle:

```
dim(trees)
```

Praktische Einführung

- Ein Algorithmus soll ausgehend von ausgewählten Fällen ein Modell zur Vorhersage des Volumens ermöglichen



Praktische Einführung

- Über die Funktion subset(…) werden die Fälle 1 bis 5 sowie 26 bis 30 ausgewählt:

```
# Trainingsdatensatz anlegen
subset1 <- subset(trees[1:5,])
subset2 <- subset(trees[26:30,])
```

Praktische Einführung

- Die beiden Subsets werden über die Funktion rbind(…) zu einem Trainingsdatensatz zusammengefasst:

  training_data <- rbind(subset1, subset2)
  training_data

- Der Algorithmus lernt am Trainingsdatensatz

Praktische Einführung

- Die verbleibenden Fälle 6 bis 25 sowie 31 werden über die Funktion subset(…) für den Validierungsdatensatz ausgewählt:

```
# Validierungsdatensatz anlegen
subset3 <- subset(trees[6:25,])
subset4 <- subset(trees[31,])
```

Praktische Einführung

- Der über die Funktion rbind(…) zusammengefasste Validierungsdatensatz ist somit kleiner als der Trainingsdatensatz:

```
validation_data <- rbind(subset3, subset4)
validation_data
```

Praktische Einführung

- Der Trainingsdatensatz ist in der Regel größer als der Validierungsdatensatz
  - 70 % Trainingsdatensatz
  - 30 % Validierungsdatensatz
- Oftmals wird zusätzlich ein unabhängiger Testdatensatz zur Überprüfung eingesetzt

Praktische Einführung

- Anhand der Trainingsdaten und unter Rückgriff auf Algorithmen wird ein Modell aufgesetzt
  - Algorithmus (bspw. eine lineare Regression)
  - Modell (bspw. Befunde einer linearen Regression)
- Anhand des Validierungsdatensatzes lässt sich das Modell überprüfen

Praktische Einführung

- Für den TREES Datensatz eignet sich eine lineare Regression über die Funktion lm(…) zur Vorhersage der Variable Volume:

```
# Lineares Regressionsmodell als Algorithmus
algorithm <- lm(data=training_data, Volume~Girth+Height)
```

Praktische Einführung

- Eine Überprüfung ermöglicht die Funktion predict(…), welche die Befunde der linearen Regression auf die Merkmalsausprägungen des Validierungsdatensatzes anwendet:

```
# Algorithmus auf Validierungsdatensatz anwenden
validation <- predict(algorithm, validation_data)
```

Praktische Einführung

- Die Unterschiede zwischen den tatsächlichen Merkmalsausprägungen und den vorhergesagten Werten verdeutlicht die durchschnittliche Differenz über die Funktion mean(…):

```
difference <- validation_data-validation
mean(difference$Volume)
```

Praktische Einführung

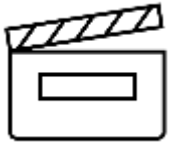
- Die Präzision lässt sich über einen Vergleich der durchschnittlichen Differenz mit der Standardabweichung über die Funktion sd(…) abschätzen:

  sd(trees$Volume)

- Weniger Abweichungen als bei der Standardabweichung sind ein präzises Modell

(IV) MACHINE LEARNING

CARET Package



CARET Package

- Das Package CARET stellt Algorithmen zum Classification and Regression Training in RStudio bereit
- Die Installation erfordert Dependencies und nimmt etwas mehr Zeit in Anspruch (!)

CARET Package

- Die Installation und Aktivierung erfolgt über die Funktionen install.packages(…) und library(…):

```
# Zusatzprogramm CARET installieren und aktivieren
install.packages("caret", dependencies=TRUE)
library(caret)
```

CARET Package

- Im nächsten Schritt werden über den Zuweisungspfeil die unabhängigen (x) und abhängigen Variablen (y) als Features definiert:

```
# Features anlegen
x <- iris[,1:4]
y <- iris[,5]
```

CARET Package

- Die Funktion featurePlot(…) visualisiert die Merkmalsausprägungen der unabhängigen Variablen in Bezug auf die Schwertlilienarten:

```
# Grafische Darstellung der Features
featurePlot(x=x, y=y, plot="box")
```

CARET Package

- Über die Funktion shapes(…) kann den Schwertlilienarten ein Symbol zur besseren Unterscheidung zugewiesen werden:

```
# Grafische Darstellung: SEPAL.WIDTH und SEPAL.LENGTH
shapes=c(1,0,20)
shapes <- shapes[as.numeric(iris$Species)]
```

CARET Package

- Die Funktionen plot(…) und legend(…) visualisieren das Zusammenspiel der unabhängigen Variablen:

```
plot(x=iris$Sepal.Length, y=iris$Sepal.Width, frame=FALSE,
xlab="Sepal Length", ylab="Sepal Width", pch=shapes)
legend("topright", legend=levels(iris$Species), pch=c(1,0,20))
```

CARET Package

- Das Package CARET ermöglicht über die Funktion createDataPartition(…) die Erstellung eines Trainings- und Validierungsdatensatzes:

```
# Training Data und Validation Data anlegen
validation_index <- createDataPartition(iris$Species, p=0.80, list=FALSE)
validation <- iris[-validation_index,]
training <- iris[validation_index,]
```

CARET Package

- Trainings- und Validierungsdatensatz können über die Funktion summary(…) inspiziert werden:

```
# Training Data und Validation Data einsehen
summary(validation)
summary(training)
```

CARET Package

- Im nächsten Schritt erfordert das Package CARET die Spezifizierung einer Validierungsmethode
- Bei der Kreuzvalidierung wird die Vorhersage-genauigkeit eines Modells über Teilmengen des Datensatzes in mehreren Schritten überprüft

CARET Package

- Eine Kreuzvalidierung über zehn Schritte wird mit der Funktion trainControl(…) spezifiziert:

```
# Validierung festlegen
control <- trainControl(method="cv", number=10)
metric <- "Accuracy"
```

CARET Package

- Schließlich werden verschiedene Machine Learning Algorithmen aufgerufen
  - Linear Discriminant Analysis (LDA)
  - K-Nearest-Neighbor (KNN)
  - Random Forest (RF)

CARET Package

- Der LDA-Algorithmus versucht die aus den Merkmalsausprägungen zwei unabhängiger Variablen bestehenden Datenpunkte mittels einer linearen Funktion voneinander zu trennen

CARET Package

- Über die Funktion train(…) wird der LDA-Algorithmus aufgerufen:

```
# Linear Discriminant Analysis trainieren
fit.lda <- train(Species~., data=training, method="lda", metric=metric, trControl=control)
```

CARET Package

- Der KNN-Algorithms verbindet naheliegende Datenpunkte ausgehend von ihrem Schwerpunkt der Verteilung und trennt diese von weiter entfernt liegenden Datenpunkten

CARET Package

- Über die Funktion train(…) wird der KNN-Algorithmus aufgerufen:

```
# K-Nearest Neighbor trainieren
fit.knn <- train(Species~., data=training, method="knn", metric=metric, trControl=control)
```

CARET Package

- Der RF-Algorithmus kombiniert die Ergebnisse verschiedener Entscheidungsbäume durch Variation einzelner Bedingungen unter den Merkmalsausprägungen, um bestmögliche Entscheidungen zu treffen

CARET Package

- Schließlich wird auch der RF-Algorithmus über die Funktion train(…) aufgerufen:

```
# Random Forest trainieren
fit.rf <- train(Species~., data=training, method="rf", metric=metric, trControl=control)
```

CARET Package

- Das Abschneiden der Algorithmen kann zunächst über die Funktion resample(…) aufgerufen und zwischengespeichert werden:

```
# Vergleich der Machine Learning Algorithmen
results <- resamples(list(lda=fit.lda, knn=fit.knn, rf=fit.rf))
```

CARET Package

- Über die Funktionen summary(…) und dotplot(…) ist ein Vergleich der Algorithmen möglich:

  summary(results)
  dotplot(results)

CARET Package

- Ist die Entscheidung für einen Algorithmus gefallen, kann dieser über die Funktion print(…) im Detail analysiert werden:

```
# Fokus auf besten Algorithmus
print(fit.lda)
```

CARET Package

- Die Validitätsüberprüfung erfolgt schließlich über die Funktionen predict(…) und confusionMatrix(…):

```
# Algorithmus auf Validation Data anwenden
predictions <- predict(fit.lda, validation)
confusionMatrix(predictions, validation$Species)
```

Obligatorischer Exkurs

- Weitere Datensätze, viele davon für Machine Learning Algorthmen geeignet, sind online zur kostenlosen Nutzung hinterlegt
  - https://www.kaggle.com/datasets/
  - https://openpsychometrics.org/_rawdata/

Klausurrelevant (!)

6. Übungsaufgabe

6. Übungsaufgabe

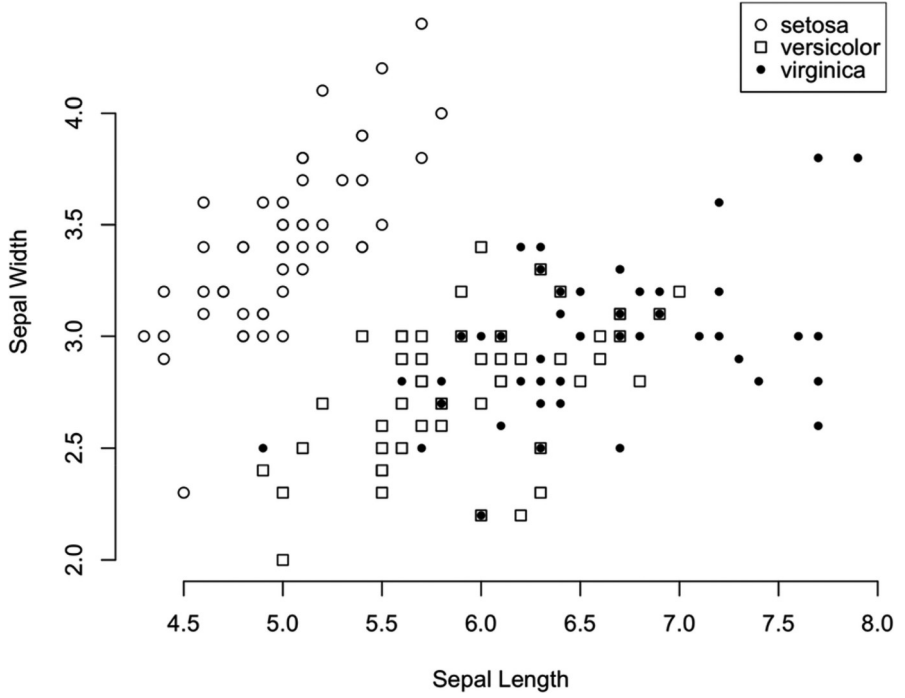
- Diese Übungsaufgabe reproduziert die zuvor vorgestellten Schritte zur Anwendung von Machine Learning Algorithmen am IRIS Datensatz
- Ziel: Der beste Machine Learning Algorithmus ist zu bestimmen

6. Übungsaufgabe

- Variablen im IRIS Datensatz
  - (x_1) Sepal.Length < - - - Feature (!)
  - (x_2) Sepal.Width < - - - Feature (!)
  - (X_3) Petal.Length < - - - Feature (!)
  - (X_4) Petal.Width < - - - Feature (!)
  - (Y) Species

6. Übungsaufgabe

- Erinnerung an die Herausforderung: Schwertlilienarten können in Länge und Breite ihrer Blätter übereinstimmen

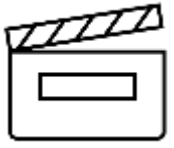


6. Übungsaufgabe

- Zusatzaufgabe: Wie schneiden die Machine Learning Algorithmen im Vergleich zu der Lösung aus der 1. Übungsaufgabe ab?

(IV) MACHINE LEARNING

Resampling



Resampling

- Machine Learning Algorithmen können nur im Rahmen der vorgegebenen Datensätze (oftmals Stichproben) trainiert werden
- Ein Bias in den Daten führt demnach zu einem Bias des Modells

Resampling

- Mit Resampling werden Verfahren bezeichnet, die auf Basis einer Stichprobe wiederholt kleine Stichproben generieren
- Demnach werden statistische Kennwerte nicht einmalig berechnet, sondern für die Anzahl der wiederholten Stichproben gemittelt

Resampling

- Bewährte Resamplingverfahren sind die Permutationstests und das Bootstrapping
  - Permutationstest: Variation der Reihenfolge aller Fälle einer Stichprobe zur Abschätzung der p-Werte
  - Bootstrapping: Wiederholte Ziehung einzelner Fälle einer Stichprobe mit Zurücklegen zur Abschätzung der weiteren statistischen Kennwerte

Resampling

- Für Machine Learning Algorithmen sind insbesondere die weiteren statistischen Kennwerte von Bedeutung
- Bootstrapping: Beispiel zum Regressions-gewicht bei der linearen Regression

Resampling

- Bestimmung des Regressionsgewichtes im MTCARS Datensatz über die Funktion lm(…):

  ```
  # Lineare Regression
  lm(mtcars$mpg~mtcars$wt)
  ```

- Regressionsgewicht ($b_{Regression}$) einmalig ermittelt

Resampling

- Zur Vorbereitung des Resamplingverfahrens und zur Reproduzierbarkeit der Befunde wird auf die Funktion set.seed(...) zurückgegriffen:

```
# Resampling
set.seed(1701)
```

Resampling

- Das Resampling (R) soll 1.000-fach wiederholt werden und als Regressionsgewicht ($b_{Bootstrap}$) angelegt werden:

```
R <- 1000
b <- vector(length=R)
```

Resampling

- Eine entsprechende Bootstrapping-Funktion:

```
for(i in 1:R){
boot.data <-
mtcars[sample(1:nrow(mtcars),size=100,replace=T),]
boot.result  <-
summary(lm(boot.data$mpg~boot.data$wt))
b[i] <- boot.result$coefficients[2,1]
}
```

Resampling

- Die Summe aller Regressionsgewichte ($b_{Bootstrap}$) kann daraufhin über die Funktion summary(…) aufgerufen werden:

  summary(b)

Resampling

- Die Funktionen plot(…) und abline(…) stellen b_{Reg} und $b_{Bootstrap}$ gegenüber:

```
# Verteilung der Regressionsgewichte
plot(density(b), main="Stichprobenwiederholung der Regressionsgewichte", ylim=c(0.05,1.25), xlim=c(-7,-3.5))
abline(0,0,0,-5.344, col="red")
```

Resampling

- Mit der Funktion legend(…) wird eine entsprechende Legende hinzugefügt:

  legend("topright", inset=.01, legend=c("b-coefficient (original) = -5.344"), col=c("red"), box.lty=0, lty=1:2, cex=1.0)

Resampling

- Tipp: Resamplingverfahren tragen nicht nur im Rahmen des Machine Learnings zur besseren Interpretation statistischer Kennwerte bei

Obligatorischer Exkurs

- Tipp: Die Bootstrapping-Funktion lässt sich entsprechend adaptieren:
  - 1. Zusatzaufgabe: p-Werte für
    t.test(ToothGrowth$len~ToothGrowth$supp)
  - 2. Zusatzaufgabe: Chi-Quadrat-Werte für
    mtcars$binary_mpg <- ifelse(mtcars$mpg<=20,1,0)
    chisq.test(mtcars$binary_mpg,mtcars$vs)

Klausurrelevant (!)

Herausforderungen

Herausforderungen

- Die von den Machine Learning Algorithmen verwendeten Muster und Gesetzmäßigkeiten entsprechen nicht immer einer theoretischen Fundierung oder basieren auf einem realen Phänomen

Herausforderungen

- Schließlich gilt: Korrelation ≠ Kausalität
- Herausforderungen beim Machine Learning
  - Unzureichende Datenqualität
  - Übermäßige Optimierung
  - Limitationen bei der Durchführung
  - Fehlende Mathematik- und Statistikkenntnisse

Herausforderungen

- Unzureichende Datenqualität
  - Statistische Kennwerte und Machine Learning Algorithmen sind nur unter Berücksichtigung der Datenqualität angemessen interpretierbar
  - Fokus auf die univariate Statistik in Bezug auf die relevanten Variablen (Verteilung, Fehlwerte, etc.)

Herausforderungen

- Übermäßige Optimierung
  - Weniger Präzision der Machine Learning Algorithmen in Bezug auf die Trainings- und Validierungsdatensätze kann von Vorteil sein
  - Es gilt: Ein flexibler Dietrich ist manchmal besser als ein starrer Schlüssel

Herausforderungen

- Limitationen auf einen Algorithmus / Trainings- und Validierungsdatensatz
  - Der Vergleich verschiedener Algorithmen beinhaltet wertvolle Informationen zur Interpretation
  - Resampling vermag die Varianz eines realen Phänomens besser abzubilden als ein einmaliger Datensatz / eine einmalige Stichprobe

Herausforderungen

- Fehlende Mathematik- / Statistikkenntnisse
  - Die zugrundeliegenden Analyseverfahren, bspw. die Faktorenanalyse, sind primär der mathematischen Statistik zuzuordnen
  - Bei Herausforderungen in der Anwendung empfiehlt sich ein Verständnis der zugrundeliegenden Funktionsweise, bspw. dem Rechnen mit Vektoren

Herausforderungen

- Tipp: In vielen Fällen führen die klassischen Analyseverfahren aus der Statistik, bspw. die lineare Regression, zu verlässlichen Befunden
- Machine Learning Algorithmen sind demnach nicht immer erforderlich

Herausforderungen

- Schließlich lauten zwei der wichtigsten Regeln für eine zielführende Datenanalyse mit R
  - Prediction is only as good as the sum of its parts
  - KISS: Keep It Short & Simple (!)

ANHANG

Fakultativer Exkurs

Fakultativer Exkurs

- Tipp: Weiterführende Kursempfehlungen
  - https://www.coursera.org/
    - Mathematics for Machine Learning (Imperial College London)
    - Data Science Specialization (Johns Hopkins University)
    - Machine Learning (Stanford Online)

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Simpson-Paradoxon:

```
x <- c(3, 4, 5, 7, 8, 9)
y <- c(8, 9, 10, 2, 3, 4)
simpson <- data.frame(x, y)
plot(simpson, ylim=c(1, 11), xlim=c(1, 11))
abline(lm(data=simpson, y~x))
abline(lm(data=simpson[c(1, 2, 3),], y~x), col="red")
abline(lm(data=simpson[c(4, 5, 6),], y~x), col="green")
legend(8, 10.5, legend=c("Gruppe A", "Gruppe B"),
col=c("red", "green"), lty=1)
```

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Parallelen zwischen t-Test und linearer Regression:

  t.test(ToothGrowth$len~ToothGrowth$supp)
  lm(ToothGrowth$len~ToothGrowth$supp)

- Mittelwert der Gruppe 0 entspricht dem Schnittpunkt mit der y-Achse (a)
- Differenz zwischen Gruppe 0 und Gruppe 1 entspricht der Steigung der Regressionsgeraden (b_1)

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Moderationseffekt (Teil 1):

```
set.seed(1701)
data <- data.frame(
fan = rep(c("Star Wars", "Star Trek"), each = 30),
nerdindex = rep(c("1", "2", "3"), each = 10, times = 2),
tvindex = c(runif(10, -3, 3), runif(10, 0, 5), runif(10, 4, 6),
runif(10, -4, 2), runif(10, 0, 3), runif(10, 5, 8)))
```

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Moderationseffekt (Teil 2):

```
interaction.plot(x.factor = data$nerdindex,
trace.factor = data$fan, response = data$tvindex,
fun = median, main = "Moderationseffekt (Schnittstelle)",
ylab = "TV-Index (Stunden pro Tag)",
xlab = "Nerd-Index (1=klein, 2=mittel, 3=groß)",
col = c("red", "blue"), lty = 2, lwd = 2,
trace.label = "Franchise")
```

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Moderationseffekt (Teil 3):
  - Schnittpunkte in der Interaktionsgrafik deuten an, dass zwischen den Variablen nerdindex und fan ein Moderationseffekt vorliegt
  - Dies wird in der Funktion lm(…) berücksichtigt:

    *summary(lm(data=data, tvindex~nerdindex*fan))*

Nicht klausurrelevant (!)

Fakultativer Exkurs

- Plot der ersten Seite (Cover):

```
library(mgcv)
library(lattice)
x <- rnorm(100)
y <- rnorm(100)
z <- rnorm(100)
tab <- data.frame(x,y,z)
mod <- gam(z~te(x,y), data=tab)
z <- matrix(fitted(mod), ncol=10)
wireframe(z, drape=TRUE, colorkey=TRUE)
```

Nicht klausurrelevant (!)


The End