MULTIPLE REGRESSION

EINFACHE REGRESSION:

$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$ → $\varepsilon_i \sim N(0,\sigma^2)$ i.i.d.

x_i: erklärende Variable
y_i: dazugehörige Messung i (Zielvariable)
β_0, β_1: y-Achsenabschnitt, Steigung
ε_i: Residuum = Fehler oder Abweichung der Messung
σ: Residual Standard Error (Abschätzung Standardabweichung der Fehler)

β_0= Achsenabschnitt → (Intercept, 1.Zeile)
β_1= Steigung von x1 → (x1 in summary, 2. Zeile)

→ Y~X, wenn sich X um eine Einheit erhöht, **erhöht sich Y um B_1**
fitE <- lm(y ~ x1, data = dat) → mit Intercept, mit Referenzlevel
fito<-lm(y~g-1, data=dat) → ohne Intercept, ohne Referenzlevel

MULTIPLE LINEARE REGRESSION

$Y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \ldots + \beta_{p-1} x_{i,p-1} + \varepsilon_i$ → $\varepsilon_i \sim N(0,\sigma^2)$ i.i.d.
→ Falls x_1 um 1 Einheit erhöht & alle anderen x_i gleich bleiben, erhöht sich y um β_1
Vorteile: Andere Einflüsse werden ausgeschaltet

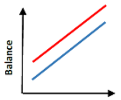
Lineare Regression in R-Studio plotten:

plotten und entscheiden, welche Transformation die Beste ist
Linear: $y = \beta 0 + \beta 1 x + \epsilon$ (keine Transformation nötig)
fit <- lm(y ~ x1 + x2 + x3, data = dat)
plot(datx,daty) →(ergibt eine Gerade)
exponentiell: $\log y = \beta 0 + \beta 1 x + \epsilon \rightarrow y = \exp(\beta 0) \cdot \exp(\beta 1 x) \cdot \exp(\epsilon)$
fit <- lm(log(y) ~ x1 + x2 + x3, data = dat) x:y ist nur interaktion interaktion
plot(dat$x, log(dat$y)), exp<-lm(log(y)~x, data=dat) x*y ist interaktion und summe
Polynomiell: $\log y = \beta 0 + \beta 1 \cdot \log x + \epsilon \rightarrow y = \exp(\beta 0 + \beta 1 \cdot \log x + \epsilon)$
fit <- lm(log(y) ~ log(x1) + log(x2) + log(x3), data = tab)
plot(log(dat$x), log(dat$y))
summary(fit) → output der linearen Regression -> Koeffizienten und Signifikanz
Tukey-Anscombe Plot: Modellwert vs. Residuen -> Soll konstant gestreut sein
QQ-Plot: Empirische Quantile vs. theoretische Quantile-> Muss Gerade ergeben

FAKTOREN ALS ERKLÄRENDE VARIABELN

str(datei) → zeigt Zusammenfassung von jeder Spalte an
dim(datei) → Zeigt Anzahl Zeilen und Spalten an
levels(datei$Gender) → "Male", "Female"; gibt alle Level dieser Spalte

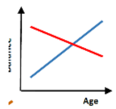
Modell OHNE Interaktion:



Modell ohne Interaktion:
$Balance_i = (\beta_0 + \beta_2 * x_i) + \beta_1 * Age_i$
→ Geraden sind parallel

1.Intercept = Referenzlevel
2.β1=Steigung (für beide gleich)
3.Zeile=Veränderung des Achsenabschnitts von Referenzgruppe zu anderer Gruppe
Achsenabschnitt der anderen Gruppe= Intercept+3.Zeile
Oww <- lm(y ~ g + x, data = dat) → lin. Reg ohne WW

Modell MIT Interaktion:



Modell mit Interaktion:
$Balance_i = (\beta_0 + \beta_2 * x_i) + (\beta_1 + \beta_3 * x_i) * Age_i$
→ Geraden sind nicht parallel, Steigungen in den zwei Gruppen unterschiedlich

2.Steigung des Referenzlevels
3.Änderung Achsenabschnitt
4. Änderung Steigung 2er Gruppe
→ wenn nicht Signifikant dann Model ohne WW nehmen!
Mww <- lm(y ~ g * x, data = dat) → lin. Reg mit WW

X ∈ {R,Z} = {REFERENZLEVEL, ABHÄNGIGE VARIABLE}
Für das Referenzlevel: $y = \beta_0 + \beta_1 * x$ → β_0=Achsenabschnitt
Für den zweiten: $y = (\beta_0 + \beta_2) + (\beta_1 + \beta_3) * x$ → $\beta_0 + \beta_2$=Achsenabschnitt

addieren wenn effekt plus ww

Beispiel in R: Wechselwirkung

```
Call:
lm(formula = Balance ~ Age * Gender, data = dat2)

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       478.6139   113.8643   4.203 3.25e-05 ***
Age                 0.5610     1.9590   0.286    0.775
GenderFemale       73.4491   156.3361   0.470    0.639
Age:GenderFemale   -0.9652     2.6835  -0.360    0.719

Residual standard error: 461.3 on 396 degrees of freedom
Multiple R-squared:  0.0007906, Adjusted R-squared:  -0.006779
F-statistic: 0.1044 on 3 and 396 DF,  p-value: 0.9575
```

Achsenabschnitt: Männer
Steigung: Männer
Änderung Achsenabschnitt: Frauen
Änderung Steigung: Frauen

$Balance_i = (478.6 + 73.4 * Gender_i) + (0.56 - 0.97 * Gender_i) * Age_i + \varepsilon_i$
$\varepsilon_i \sim N(0, 461.3^2)$

Männer: $Balance_i = 478.6 + 0.56 * Age_i + \varepsilon_i,\ \varepsilon_i \sim N(0, 461.3^2)$
Frauen: $Balance_i = 552.0 - 0.41 * Age_i + \varepsilon_i,\ \varepsilon_i \sim N(0, 461.3^2)$

BSP
intercept : -7.14 A:y=ma*x2+ca
xlb: 5.25 B:y=mb*x2+cb
x2:1.08
xlb:x2:-3.8 b zb anderes geschelcht, nicht kontinuirliche var
ma=x2=1.08, ca=intercept=-7.14
mb=x2+xlb:x2=-2.1, cb= intercept +xb=1.9

Interpretation des Outputs:

- # Freiheitsgrade = # Messungen - # β's
- t-value = estimate/std.error
- Zweiseitiger Verwerfungsbereich von βi = (-∞, -tn-1, 1-α/2] u [tn-1, 1-α/2, ∞)
- 95%-V.I. von βi = βi •} 2 * std.error
- **Signifikant Wechselwirkung falls P-Wert kleiner als das Signifikanzlevel α ist**

MULTIPLE R-SQUARED:

$R^2 = 1 - \frac{SS_{err}}{SS_{tot}}$
R^2= % Anteil der beobachteten Varianz, die durch das Modell erklärt wird.
→Je näher R^2 an 1, desto besser passen die Werte→Je grösser R^2 desto relevanter
→grosses R^2= grosse Streuung
→Ergebnis kann signifikant (kleiner P-Wert) aber nicht relevant (kleiner R-Wert)!

VERTRAUENSINTERVALLE (VI):

confint(fit) → Gibt 95% VI von fit aus
confint(fit, level = 0.99) → 99%-VI

Vorhersage für konkrete Werte

fit <- lm(y ~ age + g, data = dat) → Lineare Regression

newdat <- data.frame(age = 25, g = "Female") →Vorhersage für neue Daten
predict.lm(fit, newdata = newdat) → Erwarteter Wert für Werte aus newdat

predict.lm(fit, newdata = newdat, interval = „confidence", level = 0.90)
→ 90% Vertrauensintervall für vorhergesagte Werte aus newdat
predict.lm(fit, newdata = newdat, interval = „prediction", level = 0.95)
→ Gibt ein 95% Vorhersageintervall für erwarteten Wert aus newdat

F-TEST: mit was kann man rechnen für bestimme person

Gibt es mind. einen signifikanten Einfluss?
H_0: $\beta_1 = \beta_2 = \ldots = \beta_p = 0$ HA: mind. ein βi ≠ 0
Wenn F-Test signifikant ist→ suche sign. Variablen bei t-Tests.

KOLLINEARITÄT:

Wenn zwei β's stark korrelieren: F-Test ist signifikant, aber keiner der Parameter ist signifikant **→ Einen der korrelierenden Parameter weglassen**

Kollinearität erkennen:

plot(x, y) → Korrelation stark, wenn Punkte auf Geraden liegen
cor(x, y) → Je näher 1 desto stärkere Korrelation

coefficients

	Estimate	**Std. Error**	**t value**	**Pr(>|t|)**
(Intercept)	β_0	$\sigma(\beta_0)$	t-Werte	p-Werte
x_1	β_1	$\sigma(\beta_1)$	für	für
x_2	β_2	$\sigma(\beta_2)$	$\beta_0, \beta_1, \beta_2, \beta_3$	$\beta_0, \beta_1, \beta_2, \beta_3$
x_3	β_3	$\sigma(\beta_3)$		

Residual standard error: *5,914* on *270* degrees of freedom *[kursiv = erfundene Beispielzahlen]*
Multiple R-squared: *0.8115* **Adjusted R-squared:** *0.8108* *[Adjusted R-squared benutzen]*
F-statistic: *1162* on *1* and *270* DF, p-value < *2,2 e-16* *[p-value = p-Wert vom F-Test]*

QUIZAUFGABEN

LINEARISIEREN:

Y=A+B*X+e => lineares Modell
A= β_0= Intercept, B=x, 2. Zeile

EXPONENTIELLER ZUSAMMENHANG
$y = a \cdot b^{\,x} \cdot \exp(\epsilon)$ → log(y)=log(a) + log(b)*x + ϵ
$\log(y) = \beta 0 + \beta 1 x + \epsilon \rightarrow y = \exp(\beta 0) \cdot \exp(\beta 1 x) \cdot \exp(\epsilon)$
→ *mit* X=x *und* $a = \exp(\beta_0)$ *und* $b = \exp(\beta_1)$
fit<-lm(log(y)~ x, dat=data) → A = Estimate

POLYNOMIELLER ZUSAMMENHANG
$y = a \cdot x^b \cdot \exp(\epsilon)$ → log(y) = a + b*log(x) + ϵ
$\log(y) = \beta 0 + \beta 1 \cdot \log(x) + \epsilon$ → $y = \exp(\beta 0 + \beta 1 \cdot \log x + \epsilon)$
→ *mit* $a = \exp(\beta_0)$=exp(A) *und* $b = \beta_1$=B

y = a * exp(b*x) * exp(e) → log(y)=log(a) + b*x + e

y = exp(sqrt(a) + log(b)*log(x) + e) → log(y)=sqrt(a) + log(b)*log(x) + e **(log(b)=B → b=exp(B))**

NICHT LINEARISIERBARER TERM:
$y = a \cdot b^{\,x}$ *+ ϵ (müsste multiplikativ sein!)*
*y = a+b*log(x)+exp(e)*

Anzahl Datenpunkte = nrow(dat) (=Anzahl Zeilen)
Anzahl Spalten: ncol(dat) oder auch head(dat)

Signifikanz: siehe summary(fit): Pr(> t) < α → signifikant
(Signifikant falls der p-Wert kleiner ist als der α Wert (10%-Niveau → α = 0.1)

Standardabweichung: in summary(fit): Residual standard error
Varianz: (Standardabweichung)^2 → (Residual standard error)^2
Wie gross ist die geschätzte Varianz der Residuen? → Varianz!

Vertrauenintervall: confint(fit, level = 0.99)

Bestimmte Werte finden:

Fitten Sie eine lineare Regression mit einer Wechselwirkung (Mww) zwischen Spannung x und Geschlecht g. Welchen erwarteten (skalierten) Kraftzuwachs sagt dieses Modell für eine Frau W mit einer Spannung von 1.313 vorher

newP <- data.frame(x = 1.313, g = "W")
predict.lm(Mww, newdata = newP)

90% Vertraueninterval bei gegebenen Werten :
predict.lm(Mww, newdata = newP, interval = "confidence", level = 0.90)
95% Vorhersageintervall bei gegebene Wert
predict.lm(Mww, newdata = newP, interval = "prediction", level = 0.95)

MSE-MEAN SQUARED ERROR

$MSE = \frac{1}{n} * \sum_{i=1}^{n} R_i^2 = \frac{1}{n} * RSS$

→ Root mean squared error= $\sqrt{MSE}$

Ziel: Modell finden, bei dem Vorhersage möglichst genau ist.

TRAININGS-MSE VS. TEST-MSE

Training MSE: Fehler auf bisher gesehenen Daten -> Kann beliebig klein gemacht werden, wenn wir nur genügend Parameter verwenden

Test MSE: Fehler auf zukünftigen Daten -> Selbst bei perfektem f(x) ist TESTMSE>0, da in der Realität immer auch zufällige Fehler (ε) in Daten ist

→Paradox: Modell das die bisherigen Daten am besten beschreibt (minimale Training MSE) ist nicht unbedingt das beste für zukünftige Daten (minimaler Test MSE)!

Fazit: Training MSE ≠ Test MSE

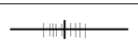
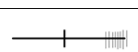
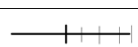
→ Um gute Vorhersagen zu machen, müssen wir Test MSE minimieren

→Um Test MSE zu minimieren reicht es NICHT den Training MSE zu minimieren

BIAS UND VARIANZ

Bias: Liegt Mittelwert der Schätzer nahe beim wahren Parameter? → Systematischer Fehler $\boldsymbol{Bias}\ (\boldsymbol{B})\ \boldsymbol{von}\ \boldsymbol{Y}: E(Y-y)$

Varianz eines Schätzers: Ist Streuung gross?→Im Mittel richtig, aber wenn Daten abweichen = falsche Ergebnisse $\boldsymbol{Varianz}\ (\boldsymbol{V})\ \boldsymbol{von}\ \boldsymbol{Y}: E((Y-y)2)$

	Varianz klein	Varianz gross
Bias klein		
Bias gross		

Test-MSE Schätzen

Direkte Methoden:

1. Test Datensatz
2. Cross-validation (CV)

Indirekte Methoden: Cp, AIC, $\boldsymbol{BIC}$, $Adjusted\ R\ 2$

- Korrektur vom Training MSE
- Approximation von direkter Methode:

→Je mehr Beobachtungen, desto genauer die Approx.

- Schnell: Gut, wenn viele oder aufwändige Modelle zu schätzen sind

dat=Trainingsdatensatz

mean(fit$residuals^2) → Trainings-MSE

dat1= Testdatensatz

mean((predict(fit, dat1) - dat1$y)^2) → Test-MSE

Anzahl **Datenpunkte**: nrow(dat)

Anzahl **Variablen:** variables-1 *(zahl oben rechts -1)*

Anzahl **mögliche Modelle:** choose(#Variablen$_{gesamt}$, #Variablen$_{verwendet}$)

Grösse eines Residuums: fit$residuals

0. Grades → fit0 <- lm(y ~ 1, dat)

1. Grades → fit1 <- lm(y ~ x1 , dat)

4. Grades → fit3 <- lm(y ~ x1+x2+x3+x4 , dat)

Alle Variablen → fit<-lm(y, dat)

(TEST-MSE SCHÄTZEN: 1. TEST DATENSATZ)

Prinzip: Daten aufteilen in einen Test- und einen Trainingssatz. Das Modell f(x) auf den Trainingssatz schätzen und auf den Testsatz evaluieren

Vorteil: Einfach, schnell

Nachteil: - Je nach Wahl vom Testdatensatz: Unterschiedlicher MSE & Trainingsdatensatz kleiner als Originaldatensatz → Test MSE wird überschätzt

R-Studio: Trainings-MSE: vorher auf dat fitten

fit0 <- lm(y ~ 1, dat) #Grad 0 -> Polynom nur mit Intercept

fit3 <- lm(y ~ x1+x2+x3, dat) #Polynom 3. Grades

summary(fit..) → Infos zur Regression fit

sum(fit$residuals^2) → RSS

mean(fit$residuals^2) → Trainings-MSE

Anzahl mögliche Modelle

choose(30,20) # Wie viele mögliche Modelle der 30 Trainings-Daten gibt es mit exakt 20 Variablen? (Falls 30 von 30-> # Modelle=1)

R-Studio: Test-MSE: vorher auch auf dat fitten

mean((predict(fit, dat1) - dat1$y)^2) → Test-MSE

pred <- predict(fit, dat1) → Macht Vorhersage für alle Datenpunkte aus dat1

quadratResid <- (dat$y, - pred)^2 → Residuenquadrat für alle Datenpunkte

quadratResidTest <- quadratResid[-train] → Alle Datenpunkte aus Test- Set

TestMSE <- mean(quadratResidTest) → Test-MSE

TestRMSE <- sqrt(TestMSE) → Root Squared Error vom Test-MSE

TEST-MSE SCHÄTZEN: 2. CROSS-VALIDATION (CV)

Leave-one-out cross-validation (LOOCV):

Jede Zeile ist einmal Testset; Rest ist Trainingsset

Nachteil: Langsam, weil ein Fit pro Zeile

R-Studio: Single-Line Methode

fitVoll <- glm(y ~ . , data = dat) → glm() ≈ lm()

cv.err <- cv.glm(data = dat, glmfit = fitVoll) (package:'boot')

sqrt(cv.err$delta) →Erste Zahl ist der ~~Test-MSE~~ ~~test-RSME~~

K-FOLD PRINZIP:

Teile Daten in 10 Blöcke; jeder Block ist einmal Testset

Nachteil: Je nach Unterteilung in K Blöcke unterschiedliches Test MSE

R-Studio: 10-fold

fitVoll <- glm(y ~ . , data = dat) → glm() ≈ lm()

cv.err10 <- cv.glm(data = dat, glmfit = fitVoll, K = 10) sqrt(cv.err10$delta) → Erste Zahl ist der

TEST-MSE SCHÄTZEN: INDIREKTE VARIANTE

Kriterium K verbindet RSS (Güte) und Anzahl Parameter d

→ K = RSS + f(d)

Modell mit bestem K ist optimal

Praxis: Verwenden, falls viele oder komplizierte Modelle geschätzt werden müssen

Vorteil: Schnell zu rechnen und einfacher!

Nachteil: Approximativ; macht Annahmen; Test MSE nicht berechnet

MODELLWAHL MIT BIC

K kann verschiedene Formen annehmen, im Paket „leaps" ist

BIC = (RSS + log(n) * d * sd^2)/n

RSS → sum(fit$residuals^2)

n = Anzahl Datenpunkte **→** nrow(dat)

d = Anzahl geschätzter Koeffizienten im Modell **→ 0.Grades = 1,**

(ACHTUNG: beta0 (Intercept) ist auch geschätzt!) → 8.Grades=9

sd = residual standard error **→ summary(fit)**

Ziel: BIC soll möglichst klein sein

→ Finde ein Modell mit möglichst kleinem Vorhersagefehler

→ Finde diejenigen Variablen, die für eine gute Vorhersage nötig sind

Faustregel: Mache Test-MSE mit Cross Validation und die Modellwahl mit dem BIC

BEST SUBSET

Prinzip: Berechne eine lin. Regr. für alle mögl. Komb. von erklärenden Variablen, speichere BIC

- findet das beste Subset
- Rechenaufwand riesig: p Variablen → 2^p Subsets

SUBSETS STEPWISE FORWARD

Prinzip: Starte mit leerem Mod.; füge eine Variable nach der anderen hinzu; STOP, sobald BIC nicht mehr kl. wird

STEPWISE BACKWARD

Prinzip: Starte mit vollem Modell, lasse eine Variable nach der anderen weg; STOP, sobald BIC nicht mehr kl. Wird

R-Studio: Modellwahl

library(leaps) → Package leaps wird benötigt

beste Teilmenge der Variablen finden:

m<-regsubsets(y ~. , data = dat, method = "exhaustive", nvmax = 20)

→wenn nicht explizit nach „forward" oder „backward" verlangt, immer mit exhaustive. nvmax gibt maximale Setgrösse = maximale #Variablen an

summary(m): Ausgabe: Spalten=Variablen, Zeilen =#Variablen, *=beste Kombination von Variablen für diese #Variablen

s <- summary(m) → s speichert beste Modelle für vorgegebene #Variablen

s$bic → Gibt die BIC-Werte für die Modelle mit 1,2,3,... Variablen aus

ncoef <- which.min(s$bic)

→ Zeigt welches Modell mit welcher #Variablen die beste Vorhersage hat

coef(m, ncoef) → Zeigt Variablen inkl. Werte für bestes Modell am

fitBest <- glm(y ~ x3 + x5, data = dat) → Bestes Modell;

wähle bestbeschreibende Variablen (x) aus coef(m, ncoef)

cv.errBest <- cv.glm(data = dat, glmfit = fitBest) sqrt(cv.errBest$delta)

→ erste Zahl = Test-MSE

mean((predict(fitBest, dat1) – dat1$y)^2) → Test-MSE;

wobei fitBest mit Trainings- Daten und dat1 mit Test Daten

GLM – GENERALISIERTES LINEARES MODELL
LOGISTISCHE REGRESSION

Parameter einer Verteilung hängt von erklärenden Variablen ab.
Bsp.: lineare Regression, logistische Regression, ANOVA,...

einfache logistische Regression: y ~ x
y = A + B * x
fit.x <- glm(y ~ x, data = dat0, family = "binomial")

multiple logistische Regression ohne Wechselwirkung: y ~ x + g
y = (A + C * g) + B * x
fit.ohne <- glm(y ~ x + g, data = dat0, family = "binomial")
→ family = binomial wenn Zielgrösse y true = 1/false = 0

multiple logistische Regression mit Wechselwirkung: y ~ x * g
y = (A + C * g) + (B + D * g) * x
fit.mit <- glm(y ~ x * g, data = dat0, family = "binomial")

LOGISTISCHE REGRESSION

$\log\left(\frac{P(x)}{1-P(x)}\right) = y = \beta_0 + \beta_1 * x$ (log odds) $y \sim \text{Bin}(1, P(X))$ odds∈[0,inf) log-odds∈(-inf,inf)

$\frac{P(A)}{1-P(A)} = e^{\beta_0+\beta_1*x}$ odds

→ **y ist nicht die W'keit, sondern die log-odds der W'keit!**
→Einfache Aussage über Änderung der Wahrscheinlichkeit ist nicht möglich !

x+1: die **log-odds** erhöhen sich um $+\beta_1$ (add)
x+1: die odds erhöhen sich um $*e^{\beta_1}$ (mult.)
95%-VI der log-odds= $[\beta_1 \pm 2*\text{std.err}(\beta_1)]$
95%-VI der odds=$e^{[\beta_1 \pm 2*\text{std.err}(\beta_1)]}$
log-odds(gesund)=-log-odds(krank)
Werden die log-odds (krank) kleiner, sowird die Wkeit krank zu sein auch kleiner

ODDS UND W'KEITEN BERECHNEN

Angegeben werden die **log-odds!**

$$log - odds = \log\left(\frac{P(A)}{1-P(A)}\right)$$

Log-odds in odds umrechnen :
odds= exp(log-odds) aus output
$\text{odds}=e^{\text{log-odds}}$

odds in Wahrscheinlichkeiten umrechen :
P(x) = odds(x)/(1+odds(x))

$$odds(A) = \frac{P(A)}{1-P(A)}$$

$P(A)=\frac{\text{odds}(A)}{1+odds(A)}$ → $P(A) = \frac{e^{\beta_0+\beta_1*x}}{1+e^{\beta_0+\beta_1*x}}$

Log-odds(1) = - log-odds(0)

log-odds und odds wachsen monoton mit der Wahrscheinlichkeit:
P(A) grösser → odds(A) grösser → log-odds(A) grösser

R-STUDIO LOGISTISCHE REGRESSION:

Vorhersagen (→log-oods(y)) für neue Daten:
predict.glm(fit, newdata=dat0, type="link")
p<- predict.glm(fit, newdata=dat0, type="link")
p$fit/(1-p$fit) → gibt die odds(y) aus. (p=W'keit P(y))

Lieben mit eigener Var

Vorhersagen (→Wahrscheinlichkeiten P(y)) für neue Daten:
predict.glm(fit, newdata=dat0, type = "response")
predict.glm(fit, newdata = dat0[122,], type = "response")
→ WK für den 122 Patienten

Vorhersage (Klasse) für neue Daten:
predict.glm(fit, newdata=dat0, type="response")>=0.5

Wahrscheinlichkeiten für das ganze Test-Set:
pp <- predict.glm(fit, newdata = dat0, type = "response")
Klassen bestimmen (dat0 ist prediciton für 0 patient)
class <- (pp >= 0.5) TRUE wenn pp ≥ 0.5
Anzahl korrekt klassifizierten Patienten= # Messw. (≙ Zeilen) bei denen Voraussage durch Regression richtiges Resultat geliefert hat
sum(class == dat0$y)
Anzahl falsch klassifizierte: → **sum(class != dat0$y)**
Wie gross ist Fehler der falsch klassifizierten Patienten → **class != dat0$y/nrow(dat0)**
classn <- as.numeric(class) → Vektorclass n bei dem TRUE = 1 & FALSE = 0
sum(classn == dat0$y)

SIMPSON-PARADOX

Wenn man eine zweite Variable einfügt, ändert sich die Steigung β_1 für die erste Variable. Falls sich die Steigung extrem stark ändert (vielleicht sogar von negativer Steigung zu positiver), kann das sehr paradox wirken.
Beispiel: Studenten und Schulden: Studenten haben oft hohe Schulden und deshalb ein hohes Risiko für Pleite. Aber wenn zwei Leute hohe Schulden haben ist der nicht- Student schneller Pleite.

QUIZAUFGABEN

y ∈ {0,1} = {männlich, weiblich} = {M, W} →
M(0) falls log-odds < 0 → odds < 1 → p(x) <0.5
F(1) falls log-odds ≥ 0 → odds(x) ≥ 1 → p(x) ≥ 0.5

Wechsel bei :
x=-A/B (Referenzlevel) oder
x = -(A + C)/(B + D) (anhängige Variable)

Temperatur= x

	Estimate	Name
Intercept	0.6	A
x	-1.2	B
nR	5.9	C
x:nR	0.0	D

n ∈ {0,1} = {trocken, Regen} = {T, R}
T=Referenzlevel
log-odds T(x) = A + B*x → log-odds T(x) = 0.6 + -1.2*x
R=abhängige Variable
log-odds R(x) = (A + C) + (B+D)*x → log-odds R(x) = 6.5 + -1.2*x

EXPERIMENTES DESIGN – GUTE PRINZIPIEN

HGYPOTHESES FORMULIEREN:

Experimente sorgfältig planen! Vorgehen:
1. **Testbare Hypothese formulieren** & testbare Vorhersage machen
2. Entscheiden, **welche statistische Auswertung** gemacht wird und wie **gross die Stichproben** sein müssen.
3. **Pilotstudie** machen
4. **Daten erheben und auswerten**

1. Hypothese Formulieren
Die Hypothese muss präzise formuliert, testbar und objektiv sein
Aus der Hypothese muss man eine testbare Vorhersage machen.
Manchmal Kompromiss eingehen: Präzision ↔ Pragmatik

2. Statistische Auswertung
Muss vor der Datenerhebung definiert werden!
- Wie viele und welche Daten werden benötigt?
- Wie gross sollen erhobene Stichproben sein (Poweranalyse mit Daten aus Pilotstudie, kann erst nach Pilotstudie definitiv bestimmt werden)?
- Welche Auswertungsmethoden/ statistischen Tests werden verwendet (z.B. t-Test)?

3. Pilotstudie
Wenige Daten erheben. Ziel: Erfahrungen sammeln in welchen Datenerhebungen sowie Schwierigkeiten erkannt und behoben werden.
Testen, ob man die Daten wie geplant auswerten kann.
→ Wie viele Daten braucht es? (Stichwort $\sqrt{n}$ – Gesetz: Wie viele Daten brauche ich für einen genügend kleinen Standartfehler?)
→ Fehlen noch Daten, die für die Auswertung benötigt werden? Wie kann man diese fehlenden Daten erheben?
→ letzte Korrekturen machen

$\sqrt{n}$ – Gesetz: Einzelschwankungen wären ~120 Sekunden. Bei 16 Messungen wäre die Schwankung = Standartabweichung $\sigma = 120s * \frac{1}{\sqrt{16}} = 120s * \frac{1}{4} = 30s$

PRINZIPIEN FÜR GUTES EXPERIMENTELLES DESIGN

Ursache/Wirkung → Problem **«Spurious Correlation»:** zwischen zwei Variablen gibt es eine Korrelation, aber keinen Kausalzusammenhang. Mit Korrelation alleine ist es kaum möglich Kausalzusammenhänge zu beweisen → **Experimente machen!**
→ **Experimente** sind am besten dafür geeignet, einen **Ursache-Wirkungs Zusammenhang** zu finden
ACHTUNG: Korrelation ≠ Kausalzusammenhang → Finde also den Kausaleffekt!
→ mittels KERZ
1. Kontrolle → Kontrollgruppe, die zufällig zugeteilt ist.
2. Experiment → gut aufbauen

Optimal: Randomisiertes, kontrolliertes Experiment mit Replikaten.
Diese sind aber **nicht immer machbar** (zu teuer, preisaufwändig, nicht machbar oder Ethik) →Falls nicht machbar, mache eine **Beobachtungsstudie:** Vergleiche zwei Gruppen, die in möglichst vielen Punkten übereinstimmen! Aber wir können nie sicher sein, dass nicht noch irgendwelche relevanten Unterschiede vorhanden sind. → skeptisch sein! Es könnten viele Confounder vorhanden sein (Confounder = Faktor, der das Resultat mitbeeinflusst)

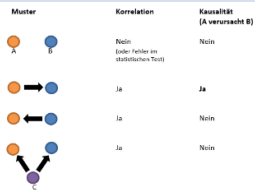
3. Replikation
Experiment muss wiederholbar sein (z.B an versch. Orten durchführen)
4. Zufall
Zufall ist wichtig, z.B für Kontrollgruppenzuteilung. ABER: Gut durchdenken, damit keine Condounder entstehen.

DIE 7 TODSÜNDEN

PRÜFE STUDIEN IMMER AUF DIESE 7 PUNKTE!

1. KORRELATION UND KAUSALITÄT

→**Korrelation beweist keine Kausalität**
→Mit Beobachtungsstudien kann man Korrelation zeigen, keine Kausalität!
→Um Kausalität zu beweisen, muss man Experimente machen (hohe Beweislast)
→Kausalität durch Korrelation zu beweisen braucht extremer Zeitaufwand
→Denkmuster: Korrelation <-> Kausalität



2. PSEUDOREPLIKATE

→Replikate müssen **unabhängige** Versuchseinheiten sein
(sonst sind es Pseudoreplikate) (z.B. wiederholte Messungen an einem Baum)
→Ziel ist es, die Versuche so unabhängig wie möglich zu machen
→Schwierig, keine möglichen Abhängigkeiten zu finden
→bei unkontrollierten Umgebungen können auch Fehler passieren *(vielleicht hatte 1. Gruppe eine andere Umgebung als 2.* →*fälschliche Abhängigkeit)*
→Mixed Effects Models oder Blockfaktoren können weiterhelfen

3. BEHANDLUNGEN HABEN CONFOUNDER

→Confounder sind Faktoren, die das Ergebnis beeinflussen können
→Die Variable, die wir wirklich untersuchen wollen darf nicht mit einem Confounder korrelieren (von etwas anderem beeinflusst werden)
→Versuche sollte sich nur in einer einzigen Variabel unterschieden! *(oder z.B. zufällige Reihenfolgen machen, um Lerneffekte usw.zu vermeiden)*

4. BEOBACHTER HAT BIAS

→Erwartungen beeinflussen die Wahrnehmung
→ Trifft sowohl auf die Wahrnehmung des Studienleiters als auch auf diejenige der Studienteilnehmer zu (→ Deshalb wirken Placebos)
→**möglichst objektive Messmethoden** wählen→Optimal wären doppelblinde Studien *(weder der Studienleiter noch Teilnehmer von Erwartungen beeinflusst)*

5. VERHALTENSÄNDERUNG WEGEN EXPERIMENT-SETTING

→Sowohl Tiere als auch Menschen **verhalten sich je nach Umgebung** anders
→Experiment-Aufbau kann das Verhalten der Probanden beeinflussen
→ **Störungen minimieren!** (z.B. mittels Tarnung)

6. SCHLECHTE/KEINE KONTROLLEN

→Kontrollen müssen **sehr sorgfältig gewählt** werden: sollten möglichst ähnlich zur Behandlungsgruppe sein → keine Confounder verzerren das Ergebnis
→ Muss immer vor der Studie überlegen, wie die Kontrollgruppe aussehen soll

7. NULLHYPOTHESE „BEWEISEN“

→Wenn die Nullhypothese H_0 nicht verworfen wird, heisst das **nicht**, dass sie automatisch stimmt
→Wenn die Nullhypothese H_0 verworfen wird, heisst das **nicht**, dass die Alternativhypothese bewiesen ist
→Wenn wir H_0 nicht verwerfen: Entweder ist H_0 tatsächlich falsch oder wir haben zu wenig Macht, um eine Abweichung festzustellen
→Alternative (Bsp: Münzwurf): **Vergleiche Vertrauensintervall und irrelevanten Bereich**

vermischung erklärender var: placebo nur frauen

MIXED EFFECTS MODELS

Fixed Effects ohne Wechselwirkung: zb. wachstumskurven
yi = (A + Aj) + B * x + e — erlauben aussagen über individuen
fitFoww <- lm(y ~ x + R, data = dat)

Fixed Effects mit Wechselwirkung: anzahl koeffizienten aus summary
yi = (A + Aj) + (B + Bj) * x + e — geschatzte parameter= koeff +!
fitFmww <- lm(y ~ x * R, data = dat)

library(lmerTest)
Mixed Effects (RI):
yi = (A + ai) + B * x + e
fitM1 <- lmer(y ~ x + (1 | R), data = dat)

Mixed Effects (RIRS):
yi = (A + ai) + (B + bi) * x + e
fitM2 <- lmer(y ~ x + (x | R), data = dat)
A, B, Ai, Bi sind fixe Koeffizienten → ai und bi sind zufällig

BLOCK-EFFECTS

$$y_{ij} = (\beta_0 + \beta_{0,i}) + \beta_1 x_j + \varepsilon_{ij}, \quad \varepsilon_{ij} \sim N(0, \sigma^2)\ i.i.d$$

"fixe" Effekte

Schätze: $\beta_0, \beta_{0,i}, \beta_1, \sigma$

Block Effekte sind
eigentlich lineare Regression mit oder ohne WW
• β's sind fixe Effekte=konkrete Werte
→ **Anzahl fixe Koeffizienten**= Anzahl R + 1 (β_0 ist auch ein fixer Koeffizient!)
→Dieses Model erlaubt Aussagen über **Indiviiduen** (nicht Population)

MIXED EFFECTS

• Fixed effects +random effects = mixed effects
→ Dieses Model erlaubt Aussagen über die **ganze Population**
→ nicht über Individuen!

Random Intercept RI: individueller Achsenabschnitt
(1|subject) = Zufällige Schwankung aber keine Schwankung in der Steigung

$$y_{ij} = (\beta_0 + u_i) + \beta_1 x_j + \varepsilon_{ij},$$
$$\varepsilon_{ij} \sim N(0, \sigma^2),\ u_i \sim N(0, \sigma_u^2)\ \ i.i.d$$

"zufälliger" Effekt — "fixe" Effekte

Schätze: $\beta_0, \beta_1, \sigma, \sigma_u$

• β's sind fixe Effekte=konkrete Werte
• u_i=zufälliger Effekt=kein konkreter Wert

```
Random effects:
 Groups   Name        Variance Std.Dev.
 Subject  (Intercept) 1378.2   37.12
 Residual              960.5   30.99
Number of obs: 180, groups:  Subject, 18

Fixed effects:
            Estimate Std. Error t value
(Intercept) 251.4051     9.7467   25.79
Days         10.4673     0.8042   13.02
```

$$y_{ij} = (251.4 + u_{1,i}) + 10.5 \cdot x_j + \varepsilon_{ij},$$
$$\varepsilon_{ij} \sim N(0, 30.1^2)\ i.i.d$$
$$u_{1,i} \sim N(0, 37.1^2)$$

Random Slope Random Intercept RIRS: individueller A.Abschnitt und Steigung
(x|subject)= Zufällige Schwankung Achsenabschnitt &Steigung pro Person

"fixe" Effekte — "zufällige" Effekte

Möglichkeit 2: Mixed Effects Model

$$y_{ij} = (\beta_0 + u_{1,i}) + (\beta_1 + u_{2,i}) x_j + \varepsilon_{ij},$$
$$\varepsilon_{ij} \sim N(0, \sigma^2)\ i.i.d$$
$$u_{1,i} \sim N(0, \sigma_1^2), u_{2,i} \sim N(0, \sigma_2^2), cor(u_1, u_2) = \rho$$

Schätze: $\beta_0, \beta_1, \sigma, \sigma_1, \sigma_2, \rho$

```
Random effects:
 Groups   Name        Variance Std.Dev. Corr
 Subject  (Intercept) 612.09   24.740
          Days         35.07    5.922   0.07
 Residual             654.94   25.592
Number of obs: 180, groups:  Subject, 18

Fixed effects:
            Estimate Std. Error      df t value
(Intercept) 251.405      6.825  16.998  36.838
Days         10.467      1.346  16.995   6.771
```

SCHÄTZEN VON MIXED EFFE

$$y_{ij} = (251.4 + u_{1,i}) + (10.5 + u_{2,i}) x_j + \varepsilon_{ij},$$
$$\varepsilon_{ij} \sim N(0, 25.6^2)\ i.i.d$$
$$u_{1,i} \sim N(0, 24.7^2), u_{2,i} \sim N(0, 5.9^2), cor(u_1, u_2) = 0.07$$

MAXIMUM LIKELIHOOD ML

- Varianzschätzungen („ui“ im Modell oben) haben Bias
+ Tests zwischen Modellen mit verschiedenen fixen Effekten sind möglich

RESTRICTED MAXIMUM LIKLIHOOD REML-

- Kann nur Modelle mit gleichen fixen Effekten vergleichen
+ Varianzschätzungen haben keine Bias
→ Ist die default-Einstellung in R

typische schwankung mit random effekts

IN R-STUDIO:

library(lme4), library(lmerTest), library(lattice), library(Matrix), library(ggplot2)

RIRS und RI : 1 für RT

fit <- lmer(y ~ x + (x|subject) , data = dat)
summary(fit) → Output (Auszug aus dem wichtigen Teil des Outputs):
x=Achsenabschnitt & Steigung für Gesamtbevölkerung

RESIDUENANALYSE

summary(dat) → Mittelwert
plot(fit) → Tukey-Anscombe Plot
qqnorm(residuals(fit)) → QQ-Plot der Residuen
qq <- ranef(fit) →Zufällige Achsenabschnitt und Steigung p.P. (Tabelle)
qqnorm(qq$Subject[,1]) → QQ-Plot von zufälligen Achsenabschnitten p.P.
qqnorm(qq$Subject[,2]) → QQ-Plot von zufälligen Steigungen pro Person

Geradengleichung für Mixed Effects Models :
ranef(fitM1)→ Zeigt u's für Aussagen über Individuen addieren auf beta

```
> ranef(fm1)
$Subject
    (Intercept)       Days
308   2.2585654  9.1989719
309 -40.3985769 -8.6197032
310 -38.9602458 -5.4488799
330  23.6904985 -4.8143313
331  22.2602027 -3.0698946
332   9.0395259 -0.2721707
333  16.8404311 -0.2236244
334  -7.2325792  1.0745761
335  -0.3336958 -10.7521591
337  34.8903508  8.6282840
```

Z.B. Geradengleichung für Person 308:

$$y_{ij} = (251.4 + 2.3) + (10.5 + 9.2) x_j + \varepsilon_{ij}$$
(andere Parameter wie bisher)

RI ODER RIRS? Vergleich von Mixed Effects Models:

anova(fitM1, fitM2) → Vergleiche beim Output, welches Modell tieferes = besseres AIC und **BIC** hat → Wähle dieses Modell

tiefer = besser AIC mit AIC(fit)

VERTRAUENSINTERVALLE

confint(fit) → 95% Vertrauensintervall für Parameter
confint (fit, level = 0.99) → 99% Vertrauensintervall für Parameter
Was sagen diese Intervalle aus? Wenn ein Wert ausserhalb des V.I. liegt, ist der Parameter signifikant. (P-Wert kleiner als α)
confint von RIRS – Modell : $y_{ij}=(\beta_0+ u_{1,i})+ =(\beta_0+ u_{2,i})*x_j+\varepsilon_{ij}$

```
> confint(fitRIRS)
Computing profile confidence intervals ...
                  2.5 %    97.5 %
.sig01        3.918095 15.358961   → σ1 (von u1,i) = Schwankung von Achsenabschnitt
.sig02       -1.000000  1.000000   → ρ = cor(u1,i , u2,i)
.sig03        0.000000  4.204986   → σ2 (von u2,i) = Schwankung von Steigung
.sigma        2.587218  5.010960   → σ (von ε)
(Intercept) 37.269447 51.997214   → β0
x           18.533068 21.600264   → β1 steigung
```

ANOVA – ANALYSIS OF VARIANCE - VARIANZANALYSE

- Macht Aussagen über Mittelwert (H0 = alle sind gleich!) anhand von Varianzen
- Ist ein Spezialfall einer linearen Regression
- Grundfrage: Sind die Unterschiede der Mittelwerte signifikant?

EINWEG – ANOVA MODELL

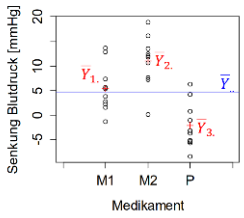
Modell: Y ~ X + ε i → Y kontinuierlich, X Faktor

ANOVA: Idee

p = # Beobachtung pro Gruppe (in jeder Gruppe gleich)
g = # Gruppen
$\bar{Y}_i$= Mittelwert der einzelnen Gruppen
$\bar{Y}$= Mittelwert aller Werte zusammen

Streuung zwischen Gruppen:
"Between-Sum-of-Squares" (SS_B)
RSS der Gruppenmittelwerte (rote Kreuze) um den totalen Mittelwert (blaue Linie)

$$SS_B = p * \sum_{i=1}^{g} (\bar{Y}_{i.} - \bar{Y}_{..})^2$$



Streuung innerhalb Gruppen:
"Within-Sum-of-Squares" (SS_W)
RSS der Einzelbeobachtungen (schwarze Kreise) um die einzelnen Mittelwerte (rote Kreuze)

$$SS_W = \sum_{i=1}^{g} \sum_{j=1}^{p} (Y_{ij} - \bar{Y}_{i.})^2$$

Teststatistik $\approx \frac{SS_B}{SS_W}$

SSB=Between-sum-of-Squares = Streuung zw. den Gruppen
SS_B ist grösser, je unterschiedlicher die Mittelwerte

SS_W=Within-sum-of-Squares = Streuung innerhalb der Gruppen
Schaut Streuung innerhalb der Gruppen an, also Streuung aller Werte
SS_W ist grösser, je grösser die Streuung innerhalb der Gruppen.

DETAILLIERTES MODELL

$Y_{ij} = \mu + \alpha_i + \varepsilon_{ij}$ $\quad \varepsilon_{ij} \sim N(0, \sigma^2)$ iid

- Y_{ij}= Zielgrösse. Y hat beliebige (unbekannte) Verteilung.
- μ = Mittelwert über alle Gruppen, ε_{ij}= zufälliger Fehler
- α = Verschiebungen des Mittelwertes μ_i pro Gruppe

$M_i = Y_i$
u2= y2-y1

TESTSTATISTIK

Teststatistik ist grösser, je grösser der Unterschied der Mittelwerte zwischen den Gruppen und je kleiner die Streuung innerhalb der Gruppen ist.
(Teststatistik T ist proportional zu $\frac{SS_B}{SS_W}$). Exakte Teststatistik T = F-Wert

$T = F = \frac{MS_B}{MS_W}$

$MS_B = \frac{SS_B}{df_B}$

$MS_W = \frac{SS_W}{df_W}$

df_B = g – 1
g = # Levels im beschr. Fakt.

df_W = g * (p-1)
p = # Beob. /Messw. pro Gruppe (in allen Gr. gleich)

$$SS_B = p * \sum_{i=1}^{g} (Y_i - Y)^2$$

p = Anzahl Beobachtungen pro Gruppe
g= Anzahl Gruppen

$$SS_w = \sum_{i=1}^{g} \sum_{I=1}^{p} (Y_{ij} - Y_i)^2$$

Analyse der Teststatistik:

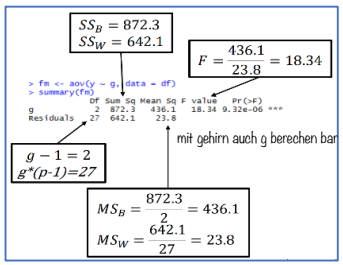
H_0: $\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = \ldots = 0$ → Faktoren haben keinen Einfluss auf Y

Verteilung der Teststatistik T unter H_0

$T \sim F_{g-1, g*(p-1)}$ →g-1 und g*(p-1) geben die Freiheitsgrade an
Nun F-Wert mit Standardwerten aus Tabelle vergleichen: Falls F-Wert höher als kritischer Wert in Tabelle → Test signifikant

R-STUDIO

fit <- aov(y ~ x, data = dat) → Fitte ANOVA mit beschreibendem Faktor x



summary(fit) → Output:
hier g=3, p=10

Df = Anzahl Freiheitsgrade = N-1
Sum Sq = Sum of Squares = S
Mean Square= MS
F value = Teststatistik
Pr(>F)= -> Wenn p-Wert ≤ α, dann ist der Einfluss von M auf Y signifikant

median(dat[,1]) → gibt den Median aller Daten der 1. Spalte aus
mean(dat$y[dat$M=="M2"]) →Mittelwert der Gruppe mit Medikament M2

Ist mind. Eine Gruppe auf 5%-Level sign?

TukeyHSD(aov(y ~ M, dat), conf.level = 0.05)

Ohne Korrektur

t.test(dat$y[dat$M=='Mj'], dat$y[dat$M=='Mi'], conf.level = alpha)

Bonferori Korrektur

t.test(dat$y[dat$M=='Mj'], dat$y[dat$M=='Mi'], conf.level = 1-(1-alpha)/K)

FALLS ANOVA SIGNIFIKANT:

Zw. welchen Gruppen sind Unterschiede signifikant?

Methode 1: t-Test für alle Gruppenpaare durchführen. → Problem: Gibt $\binom{n}{2}$ Tests bei n Gruppen. Da es so viele Tests gibt, gibt es auch viele falsch positive Tests.
→Lösung**: Bonferroni-Korrektur:**
→ neues Signifikanzniveau $\alpha_{neu} = \frac{\alpha}{n}$,
→ Konfidenzlevel = 1- α_{neu} → 1 - (1-Konfidenz)/(Anzahl Tests)

Methode 2: TukeyHSD = Tukey Honestly Significant Difference

+ Gibt V.I. für die Differenzen der Gruppenmittelwerte an
+ Die W'keit, dass alle wahren Differenzen im V.I. liegen ist = α
(liefert kürzere Vertrauensintervalle als BSD)

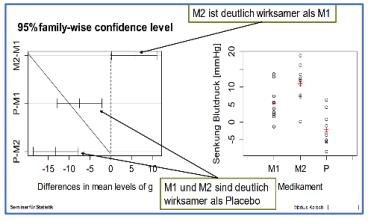
TukeyHSD(fit)

	diff	lwr	upr	p adj
M2-M1	0.2042983	-3.4580073	3.866604	0.9999981
N1-M1	3.4631251	-0.1991805	7.125431	0.0766331
N2-M1	6.1961208	2.5338153	9.858426	0.0000309

- diff = Geschätzte Unterschiede zw. Gruppen M2 zu M1, N1 zu M1 ,....
- lwr / upr = Grenzen des 95% V.I. für die geschätzten Unterschiede
- p adj = für multiples Testen korrigierter p-Wert

plot(TukeyHSD(fit))

- Striche= 95%-V.I. für die Unterschiede zw. den Levels
- Strich in der Mitte= geschätzter Unterschied



•Intervall kreuzt die Nulllinie: Unterschied ist nicht signifikant
• Intervall komplett auf der rechten (+) Seite:
1. Level ist signifikant wirksamer als das 2.
• linke (-) Seite: 2. Level ist signifikanter als 1.

KONTRASTE

Statt Differenz von 2 Gruppen schauen wir Linearkombinationen von beliebigen Gruppen an.
Sind die beiden Medikamente im Mittel besser als das Placebo?
Was wir dafür definieren müssen:
- **Vektor mit Gruppenmittelwerten μ:** Vektor μ = (μ_{Rot}, $\mu_{Grün}$, μ_{Gelb})
- **Kontrast-Matrix K:** Definieren, welche Gruppen wir vergleichen
- **Parameter-Vektor m:** Kann man selber definieren, meist Nullvektor
Idee: Nullhypothese **H_0: K * μ = m** → Computer berechnet die p-Werte für alle in K definierten Varianten und korrigiert für multiples Testen

KONTRASTMATRIX K

Die Zeilen der Kontrastmatrix geben immer eine Linearkombination von Gruppen an, die getestet werden soll.
-> eine Zeile = eine Hypothese = ein Vergleich = ein Kontrast.
•Je weniger Kontraste (= Zeilen) umso mehr Macht hat der Test.
• R-Studio korrigiert die p-Werte für multiples Testen
• Korrektur für multiples Testen wird nur pro Funktionsaufruf gemacht. Deshalb definiert man am Anfang einen einzigen Satz von Kontrasten (= eine einzige Kontrastmatrix). Man untersucht danach keinen neuen Satz mehr.
Kontrastmatrix → muss in der **Summe immer +1 bzw. -1** geben
Kontrastmatrix * μ = m

KONTRASTMATRIX IN R-STUDIO:

Summary(dat)
Anzahl Patiente: 13-1=12
Anzahl Medikamente : M1+M2+N1+N2+N3=5

```
       y          M
 Min.   :-13.850   M1:12
 1st Qu.:-10.945   M2:12
 Median : -9.156   N1:12
 Mean   : -8.462   N2:12
 3rd Qu.: -6.509   N3:12
 Max.   :  2.356   P :12
```

fit <- aov(y ~ M, data = dat)
summary(fit)

```
Fit: aov(formula = y ~ g, data = df)

Linear Hypotheses:
           Estimate Std. Error t value Pr(>|t|)
M-P == 0     10.331      1.889    5.47 1.73e-05 ***
M2-M1 == 0    5.670      2.181    2.60   0.0294 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

Die Medikamente sind deutlich wirksamer als Placebo

M2 ist deutlich wirksamer als M1

Kontrastmatrix erstellen :

```
K <- rbind("M1 - P" = c(1,0,0,0,0,0,-1),
           "M2 - P" = c(0,1,0,0,0,0,-1),
           "N1 - P" = c(0,0,1,0,0,0,-1),
           "N2 - P" = c(0,0,0,1,0,0,-1),
           "N3 - P" = c(0,0,0,0,1,0,-1),
           "N4 - P" = c(0,0,0,0,0,1,-1))
```

```
Linear Hypotheses:
           Estimate Std. Error t value Pr(>|t|)
M1 - P == 0  -6.3886     1.2197  -5.238  <0.001 ***
M2 - P == 0  -6.1843     1.2197  -5.070  <0.001 ***
N1 - P == 0  -2.9255     1.2197  -2.399   0.0834 .
N2 - P == 0  -0.1925     1.2197  -0.158   1.0000
N3 - P == 0  -2.0478     1.2197  -1.679   0.3555
N4 - P == 0  -1.5157     1.2197  -1.243   0.6542
```

library(multcomp)
summary(glht(fit, linfct = mcp(M= K)))

• **Anzahl Zeilen einer Kontrastmatrix** mit allen paarweisen Vergleichen (inkl. Placebo) → bei m = 7 Behandlungen gilt m*(m-1)/2
• Der Unterschied von M1 zu P ist geschätzt -6.3886
• 95% - Vertrauensintervall vom Unterschied von M1 zu P:
[estimate ± 2 * std. error] = [-6.3886 ± 1.2197]
• Pr(>|t|)= p-Wert des Unterschiedes zwischen den verglichenen Faktoren.
Falls P ≤ α → Unterschied signifikant

Namen der Matrixelemente:

colnames(K) <- levels(dat$M)

ZWEIWEG-ANOVA

$Y \sim X_1 + X_2 + \varepsilon_i$ Y ist kontinuierlich (belibige unbk. Verteilung), X_i sind Faktoren

DETAILLIERTES MODELL

immer $\varepsilon ij \sim N(0, \sigma 2)$ i.i.d.

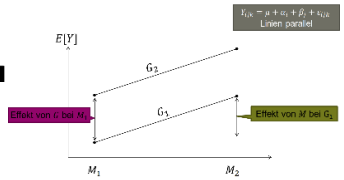
Modell ohne WW:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \varepsilon_{ijk}$$

Messung von Person k mit Medikament i und Geschlecht j | Effekt von Geschlecht j | Fehlerterm | Effekt von Medikament i

effekt geschlecht

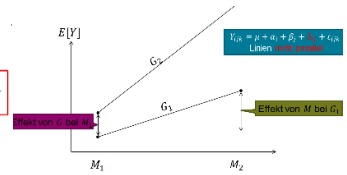
effekt medi



Modell mit WW:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \delta_{ij} + \varepsilon_{ijk},$$

Effekt von Medikament | Effekt von Geschlecht | Evtl. Interaktion: Geschlechtsspezifischer Effekt von Medikament



Hier 3 Nullhypothesen: Beispielhaft

$H_{0,1}$: $\alpha i = 0$ *für alle i* → Kein Medikamenten-Effekt

$H_{0,2}$: $\beta j = 0$ *für alle j* → Kein Geschlechter-Effekt

$H_{0,3}$: $\delta ij = 0$ *für alle i,j* → Kein Geschlechtsspezifischer Effekt von Medikament (keine Interaktion)

paarweise Vergleiche mit versch. Medis und versch. Geschlecht = $\frac{m \cdot g \cdot (m \cdot g - 1)}{2}$

μ entspricht dem Mittelwert μ_{ij} pro Gruppe

α, β = beschr. Faktoren, von denen Zielgrösse Y abh.

α_i, β_j sind die Versch. zum Mittelwert μ_{ij} pro Gruppe

δ_{ij} beschr. Interaktion zw. den Faktoren α und β

ε_{ijk} ist der zufällige Fehler

SS=SUM-OF SQUARES= STREUUNG ZW. DEN GRUPPEN

Jede der Faktorstufen ist eine eigene Gruppe.

JE GRÖSSER DER SS-WERT EINER VARIABLE, desto untersch. die Mittelwerte und UMSO SIGNIFIKANTER DER EFFEKT!

p = # Beobachtung pro Gruppe (in jeder Gr. gleich)

m = # Medikamente (BSP: 3; M1, M2, M3)

g = # Gruppen (BSP: 2 Gr.; männl & weibl.)

Y_{ijk} = Jeder einzelne Wert separat

$\overline{Y}_i$ = Mittelwert der einzelnen Gr. vom Faktor *M*

$\overline{Y}_j$ = Mittelwert der einzelnen Gr. vom Faktor *G*

$\overline{Y}_{ij}$ = Mittelwert jeder Medi-Gruppe-Kombination

$\overline{Y}$ = Mittelwert aller Werte

$SS_M = p * g * \sum_{i=1}^{m} (\overline{Y}_i - \overline{Y})^2$

→ Wie weit sind Mittelw. der Gruppen vom Faktor Medikament M gestreut?

$SS_G = p * m * \sum_{i=1}^{g} (\overline{Y}_i - \overline{Y})^2$ →Wie weit sind die Mittelw. Der Gruppen vom Faktor Gruppe (G) gestreut?

$SS_{MG} = p * \sum_{i=1}^{m} \sum_{j=1}^{g} (\overline{Y_{ij}} - Y_i - Y_j - \overline{Y})^2$ →Woe unterschiedlich sind die Mitttelw. Der Medikamente je nach Faktor G

$SS_W = SS_{Res} = \sum_{i=1}^{m} \sum_{j=1}^{g} \sum_{k=1}^{p} (Y_{ijk} - \overline{Y_{ij}})^2$ →Wie weit sind einzelne Werte der einzelnen Gruppenkomb. gestreut? BSP: Streuung von M1 bei Gruppe männlich

DEGREES OF FREEDOM (DF):

df_M : m -1 = # Medis – 1

df_G : g – 1 = # Gruppen – 1

df_{MG} : (m – 1) * (g – 1)

df_{Res}: m * g * (p – 1) = #Medis* #Gruppen *(#Messungen pro Gr. – 1)

N = m*g*p → N = totale #Messungen (∑ df = N – 1)

p=N/(m*g) = totale#Messungen/(#Medis*#Gruppen)

MEAN SQUARES (MS):

$$MS_M = \frac{SS_M}{df_M} \qquad MS_G = \frac{SS_G}{df_G}$$

$$MS_{MG} = \frac{SS_{MG}}{df_{MG}} \qquad MS_{Res} = \frac{SS_{Res}}{df_{Res}}$$

TESTSTATISTIK UND VERTEILUNG UNTER $H_{0,1}$,$H_{0,2}$,$H_{0,3}$

Falls $H_{0,1}$ stimmt:

$$T_{1(M)} = \frac{MS_M}{MS_{Res}} \sim F_{df_M;\, df_{Res}}$$

Falls $H_{0,2}$ stimmt:

$$T_{2(G)} = \frac{MS_G}{MS_{Res}} \sim F_{df_G;\, df_{Res}}$$

Falls $H_{0,3}$ stimmt:

$$T_{3(MG)} = \frac{MS_{MG}}{MS_{Res}} \sim F_{df_{MG};\, df_{Res}}$$

→ mit Tabellenwerten vergleichen: wenn Teststatistik höher als kritischer Wert, dann ist der Test signifikant;

R-Studio berechnet dies jedoch autom. damit ANOVA signifikant ist, muss nur eine Gruppe einen sign. Unterschied zeigen!

→Falls **P Wert signifikant,** unterscheidet sich min. 1 Medikament!

EINWEG-ANOVA IN R-STUDIO

fit <- aov(Y ~ F, dat) oder fit <- aov(Y ~ H, dat)

ZWEIWEG-ANOVA IN R-STUDIO

fit <- aov(y ~ F * H, data = dat) → ANOVA mit WW mit Faktoren F und H

fit <- aov(y ~ F + H, data = dat) → ANOVA ohne WW mit Faktoren F und H

summary(fit) → Output:

```
> summary(fit)
             Df Sum Sq Mean Sq F value  Pr(>F)
F             2  329.8  164.90  56.945   5e-14 ***
H             1   32.1   32.10  11.084 0.00157 **
F:H           2    0.4    0.21   0.072 0.93101
Residuals    54  156.4    2.90
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '
```

Df = Degrees of Freedom

Sum Sq = Sum of Squares = SS

Mean Sq=Mean of Squares= MS

F value = F-Wert = Teststatistik

p-Werte:

$p(F_i)$ in R-Studio: pf(F_i, df_i, dfR, lower.tail=FALSE)

→Falls p-Wert signifikant (<α), gibt es signifikanten Medikamenten- & Gruppeneffekt

TukeyHSD : so kann man das Problem vom Multiplen Testen umgehen

TukeyHSD(fit) → Output:

TukeyHSD(fit, conf.level = 0.99) → 99%-V.I. (95% ist default)

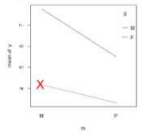
ANOVA& TURKEYHSD VS. LIN. REGRESSION

Methoden technisch gesehen gleichwertig

ABER: In der Praxis völlig unterschiedliche Interpretation

ANOVA & TukeyHSD: "Totale Effekte"

Lineare Regression: Effekte bzgl. Referenzlevel



Referenzlevel: Medikamentengruppe, Frauen

geben totale Effekte an, es gibt kein Referenzlevel Lineare Regression gibt Effekte bezüglich Referenzlevel an

RANDOMIZED BLOCK DESIGN

= Verallgemeinerung des gepaarten t-Test. Statt ein Medikament & ein Placebo pro Person wird neu pro Person mehrere Medikamente und ein Placebo verabreicht (Reihenfolge zufällig). Auswertung via 2-weg Anova:

Y ~ Medi + Person

Blockfaktor (hier "Person"): Nicht von Interesse; nur um Streuung zu reduzieren

Konvention: Keine Interaktion mit Blockfaktor

UNBALANCIERTES DESIGN:

Balanciertes Design: Alle Gruppen haben gleiche Anzahl Stichproben

Reihenfolge ist egal → bevorzugt!

Unbalanciertes Design: Verschieden grosse Stichproben pro Gruppe

- Parameter können nicht nacheinander, sondern müssen gleichzeitig geschätzt werden.
- Quadratsumme (SS) kann nicht mehr einfach den einzelnen erklärenden Variablen zugewiesen werden

ANOVA hat eine "Schwäche", falls Daten unbalanciert sind:

→ p-Werte in aov, summary() hängen von Reihenfolge ab (balanciert ok)

→ p-Werte in drop1() nicht → bevorzugen (für unbalanciert verwenden!)

lm() hat keine vergleichbare Schwäche → output von summary() ist bzgl. Reihenfolge stabil

KATEGORIELLE DATEN

FISCHER'S EXACT TEST

→2 x 2 – Tabellen, Verteilung der Teststatistik exakt

Nullhypothese H_0 = Spalten und Zeilen sind unabhängig

Hypergeometrische Verteilung (Repetition)

Urne mit *N* Kugeln, *m* sind markiert, *n* Kugeln ohne zurücklegen ziehen → wie viele davon sind markiert? = ZV *X*

X ~ Hyper(*N,m,n*) $P(X = x) = \frac{\binom{m}{x}\binom{N-m}{n-x}}{\binom{N}{n}}$

$E(X) = \frac{n*m}{N}$ $Var(X) =$ kompliziert

Ist das Ergebnis plausibel unter der Nullhypothese H_0?

BSP:	Medi	Plac.	Total
Geheilt	15 = X	9	24 = m
Nicht Gh.	10	11	21
Total	25 = n	20	45 = N

- ZV X = # geheilter Patienten in der Medi-Gruppe
- **Nullhypothese:** Behandlung und Medikament sind **unabhängig**
  H_0 = Medikament hat keinen Einfluss auf die Heilung: X ~ Hyper(N,m,n)

P-Wert = P(X ≥ x) → Wenn P-Wert ≤ α, dann Nullhypothese verwerfen.

Ist es hier plausibel, in der Medikamenten-Gruppe 15 oder mehr geheilte Patienten zu beobachten? X ~ Hyper(45, 25, 24) → 1 – P(X ≤ 14) = 0.76 = 0.24 (ja!)

Odds(Geheilt) = (24/45) / (21/45) = 24/21 = 1.14
Odds(Geheilt mit Medi) = (15/25) / (10/25) = 15/10 = 1.5
Odds(Geheilt ohne Medi) = (9/20) / (11/20) = 9/11 = 0.82
Odds-ratio: Odds(Geheilt mit Medi) / Odds(Geheilt ohne Medi) = 1.5 / 0.82 = 1.83

FISCHER-TEST IN R-STUDIO:

Tabelle Erstellen : 2x2-Tabelle !

tab <- xtabs(Freq~A2+B2, data=dat) → A und B haben jeweils nur 2 Klassen

phyper(x , m , N-m , n) ("weniger als")
1-phyper(x-1 , m , N-m , n) („mehr oder gleich")
→ 1 – phyper(14, 24, 21, 25) = 0.24 = P-Wert (Wk 15 oder mehr geheilte Patienten)
m <- matrix(c(15,10,9,11) , 2 , 2) → 2,2 = # Zeilen und Spalten
→ 15 = geheilt mit Medi, 10 = nicht geheilt mit Medi, 9 = geheilt ohne Medi, 11 = nicht geheilt ohne Medi

Fisher's Ecact Test :

fisher.test(tab, alternative = "greater") → *"odds-ratio grösser als 1"*
fisher.test(tab, alternative = "two.sided") → "H_A *: ungleich*"
fisher.test(tab, alternative = "less") niedriger
→*signifikanter Effekt falls P<α*

fit<-fisher.test(m , alternative = "greater")→Resultat unter fit speichern
ft$conf.int → 95% - Vertrauensintervall für Odds

imtext nach hinweis schauen ob grösser oder kleiner oder beides

wenn 2x3 mat gegeben und ! spalte weg lassen, mit m[,2:3] nur zweite bis dritte spalte

CHI QUADRAT TEST

→m x n – Tabellen, Verteilung der Teststatistik asymptotisch bekannt

Ziel: Visualisierung oder Abhängigkeiten finden

Nullhypothese H_0

Spalten und Zeilen sind unabhängig → H0=wahr falls p>α

W'keit, dass eine Messung genau in der Spalte i und der Zeile j landet, ist unter

$P(A = i \cap B = j) = P(A = i) * P(B = j) \approx \hat{P}(A = i) * \hat{P}(B = j) = \frac{N_i}{N} * \frac{N_j}{N}$

H_0 = [A=Spalten,B = Zeilen, i = Nummerierung der Spalten, j = Nummerierung der Zeilen]

Wobei N_i die #Beobachtungen in der i-ten Spalte und N_j die #Beobachtungen in der j-ten Zeile sind (ALLE SUMMIEREN! → Tabelle mit Totalen ergänzen!)

Falls H_0 stimmt ist der Erwartungswert jeder Zelle Eij: ERWARTETE ANZAHL!!

$E_{ij} = N * \frac{N_i}{N} * \frac{N_j}{N} = \frac{N_i * N_j}{N}$ (Erwartungswert falls H_0 stimmt dass Ni und Nj eintrifft)

Wen 1 ϵ VI des odds-ratio → Nullhypothese verworfen →signifikanter Zus.hang

PEARSON CHI-QUADRAT STATISTIK:

Fragestellung: Wie verschieden sind beobachtete und erwartete Werte?

$X^2 = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{(o_{ij}-E_{ij})^2}{E_{ij}} = \sum_{i=1}^{n}\sum_{j=1}^{m} R_{ij}^2$

Falls H_0 stimmt, folgt Teststatistik X^2 einer Chi-Quadrat-Verteilung mit (n-1)*(m-1) Freiheitsgraden. erwarteter wert mit chisq.test(dat)$expected

PEARSON RESIDUEN

$R_{ij} = \frac{o_{ij}-E_{ij}}{\sqrt{E_{ij}}}$ = Beitrag jeder Zelle zur Abweichung vom Modell

[Oij = beobachteter Wert in der Spalte i und Zeile j]

Faustregel: Wann ist die Approximation gut?

$E_{ij} < 1$ für mind. ein Tabellenfeld → ungenügend
$E_{ij} > 1$ für alle Tabellenfelder i,j → gerade noch OK
$E_{ij} > 5$ für alle Tabellenfelder i,j → sehr gut

Falls Faustregel nicht erfüllt: Kategorien zusammenfassen oder anderer Test verwenden

CHI-QUADRAT IN R-STUDIO:

Tabelle Erstellen: n x m-Tabelle

tab <- xtabs(Freq~A+B, data=dat)

→Alle Spalten werden mitbezogen. Falls man nur einzelne will: A1+B1 (Spaltennamen)

Chi-Quadrat Test :

chisq.test(tab)→ Output:

X-Squared: Pearson-Chi-Quadrat-Test-Statistik
df : (m-1)*(n-1)
p-value: 1-pchisq(q=X^2, df=*df*) → **signifikante Abhängigkeit falls p<α**

Residuen abfragen :

1. test<- chisq.test(tab)
2. test$residuals → Gib die Pearson Residuen aus.

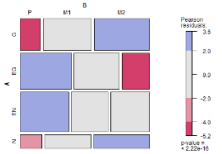
test$expectet

Visualisierung - Mosaic Plot=Flächen proportional zu Tabellen Einträgen

Library (vcd)

Mosaic(Y~X+Z, data=df, schade=TRUE)

→ Einfärben vom Mosaik, Farbe falls Pearson Residuen aus [-2,2] liegen
Rot= sehr kleiner Tabelleneintrag,
Blau=sehr grosser Tabelleneintrag
→Unten rechts: p-Wert des Chi-Quadrat Tests



dicke ist häufigkeit

für gruppen wechsel exp(faktor)

LOGISTISCHE REGRESSION

→2 x m – Tabellen, Verteilung der Teststatistik asymptotisch bekannt

Mix aus mehreren kontinuierlichen & kategoriellen erklärenden Variablen

Repetition:

Die logistische Regression kann man benutzen, wenn die Zielvariable binär ist (0/1, richtig/falsch, krank/gesund, …)

Tabelle Erstellen:

tab <- xtabs(Freq~A2+B, data=dat) → 2 x m Tabelle

Logistische Regression:

fit <- glm(A2 ~ B, weights = Freq, family = binomial, data = dat)
fit <- glm(A2 ~ B * K, weights = Freq, family = binomial, data = dat)
summary(fit)

log-odds: predict.glm(fit,newdata=data.frame(x=0,y="W"),se.fit= TRUE

EINFACHE ODER MULTIPLE REGRESSION

Einfache Regression: Y~X; Wenn sich X um eine Einheit erhöht, erhöht sich Y um β_1

Multiple Regression: Y~X+Z; Wenn sich X um eine Einheit erhöht und Z gleichbleibt, erhöht sich Y um β_1

QUIZAUFGABEN

BEOBACHTETE WERTE

	M1	M2	P	TOTAL
Geheilt	65	39	18	122
Eher geheilt	29	25	15	69
Eher nicht geheilt	17	22	36	75
Nicht geheilt	30	20	42	92
TOTAL	141	106	111	358

ERWARTUNGSWERTE →Falls Unabhängig

	M1	M2	P
Geheilt	122*141/358	122*106/358	122*111/358
Eher geheilt	69*141/358	69*106/358	69*111/358
Eher nicht geheilt	75*141/358	75*106/358	75*111/358
Nicht geheilt	92*141/358	92*106/358	92*111/358

PEARSON RESIDUAL: (beobachtet - erwartet)/sqrt(erwartet)

Wahrscheinlichkeiten:

dass eine Person ein Placebo erhalten hat **P(P)= 111/358**

Odds(A)=P(A)/(1-P(A))
→Odds(geheilt)= (122/358)/(1-(122/358))
→Odds(geheilt mit M1)= (65/141)/(1-65/141))

Log-Odds(A)=log(Odds(A) **P(A)=** Odds(A)/(1+Odds(A)

→ Wirksamkeit eines Medikamenten überprüfen via Odds-ratio:
Odds-ratio: Odds(geheilt mit Medi)/Odds(geheilt ohne Medi)

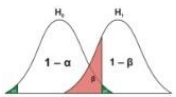
POWERANALYSE

Wie viele Stichproben braucht es, um eine bestimmte Alternative mit einer bestimmten Macht erkennen zu können?
Zu viele Stichproben: unnötiger Aufwand, Zeitverschwendung
Zu wenige: machen die Studie nutzlos
Idee: **Vor dem Experiment** eine **Power-Analyse** durchführen, um die Stichprobengrösse zu bestimmen. **Nach dem Experiment** ein **Vertrauensintervall** für die gesuchten Parameter bestimmen → um das Resultat besser interpretieren zu können

MACHT

Fehler 1. Art	Fehler 2. Art
H_0 stimmt wird aber verworfen W'keit für **Fehler 1. Art ≤ α**	**H_A stimmt**, H_0 stimmt nicht H_0 wird aber nicht verworfen Wk für **Fehler 2. Art = (1 – Macht)**

Macht= H_0 verwerfen falls H_A stimmt →Wahrscheinlichkeit eine richtige Alternativhypothese H_A zu erkennen. **Macht = 1 – P(Fehler 2. Art)**
Fehler 2. Art wie auch Macht können nur mit einer konkreten Alternativhypothese berechnet werden.
(*Im Bild: weiss)*



Berechnung der Macht :

Theorie: (+) Genau, schnelle Berechnung, **(-)** kompliziert
Simulation: (+) Fast immer möglich, **(-)** evtl. langsam, ungenau(er), Programmieraufwand gross.
Problem bei beiden Varianten: Was soll die konkrete Alternative H_A sein?

Simulation

Prinzip: Simuliere 1'000 Binomialtests für eine Alternativhypothese und schaue, wie oft Test verworfen wird. Die Macht ist dann = $\frac{\#\ \text{Verworfene Tests}}{1000}$

Binomialverteilung: Bin(n,∏) -binom(x, n, ∏)
X ~ Binom(13, 0.1), Ges: P(X = x) für x = 1 → **dbinom**(1, 13, 0.1)
X ~ Binom(20, 0.5) Ges: P(X > x) für x = 4 →
pbinom(4, 20, 0.5, FALSE) oder 1 - **pbinom**(4, 20, 0.5)

Quantile: qt()
Sei X ~ t(3), Ges: x für welches gilt P(X <= x) = 0.6 → qt(0.6, 3)

Standard-Schätzer für p :
Sei X ~ Binom(11, 0.4). Ziehe 9 Realisationen von dieser Verteilung
mean(rbinom(9, 11, 0.4)/11)

Realisationen ziehen : rnorm(13, 7.3, 0.4)
N(7.3, 0.16), ziehe 13 Relationen → **rnorm**(13, 7.3, 0.4)

Normalverteilung: N(μ,σ²) -norm(x, μ, √σ²) ≤
X ~ N(-8.1, 0.25), Ges: (X <= x) für x = -10.1 → **pnorm**(-10.1, -8.1, 0.5)

z-Test *(mu0>muA)*
1. c <- qnorm (alpha, mu0(=μ), sqrt(σ²)/sqrt(n), FALSE)
2. macht<-pnorm(c, muA, sqrt(σ²)/sqrt(n), FALSE)

Macht des Binomialtests in R

set.seed() 9 ACHTUNG : nur einmal am Anfang aufrufen !!!

Macht beim Binomialtest:

```
machtBinom <- function(n = 50 , reps = 1000 , alpha = 0.05 , pA = 0.7 , p0 = 0.5 , alt = "two-sided") {
    res <- vector("numeric" , reps)
    for (i in 1:reps) {
        x <- rbinom(n = 1 , size = n , prob = pA)
        tmp <- binom.test(x , n = n , p = p0 , alternative = alt)
        res[i] <- (tmp$p.value < alpha)
    }
    list (m = mean(res), s = sd(res)/sqrt(reps))
}
```
→ *m = Macht, s = Standardfehler*

machtBinom() → Berechnen mit Default-Werten
machtBinom(n = 120) → Berechnen mit n = 120, restliche Variablen mit Default

macht(n, β0, βA, α, reps)
!!Differenz der Genauigkeit der Macht =Standardabweichung

Suche der richtigen Stichprobengrösse - von Hand

(Funktioniert auch bei t-Test und Anova, dort muss man den Befehl machtBinom ersetzen)
Test gibt Macht für eine bestimmte Stichprobengrösse an → Um Stichprobengrösse für eine bestimmte Macht zu berechnen, kann man von Hand annähern.
Beispiel: Wir wollen eine Macht ≥90% haben. Wie gross soll Stichprobe sein?
machtBinom(n = 68) → Macht zu klein,
machtBinom(n = 69) → Macht zu gross →also **n=69** da wir Macht **≥**90% wollen

SUCHE DER STICHPROBENGRÖSSE - MIT R

```
nall <- seq(10 , 100 , by = 10)  [by = 1 wenn es exakt sein muss]
macht <- vector("numeric" , length(nall))
for (j in 1:length(nall)) {
    n <- nall[j]
    macht[j] <- machtBinom(n = n, pA
     = 0.75 , alt = "greater")
}    [--> befehl machtBinom muss zuerst definert werden!]
which(macht > 0.9)[1] → erste Stichprobengrösse, bei der Macht > 0.9 [1] bedeutet dass man 1. Wert nimmt bei dem Macht ≥ 0.9 ist (also TRUE) ist.
```

MACHT IN ZWEISEITIGEM T-TEST

MACHT ZWEISEITIGER T-TEST - IN R

```
machtTtest <- function(n1 = 20 , n2 = 20 , m1 = 0 , m2 = 1 , s1 = 1 , s2 = 1 , reps = 1000 , alpha = 0.05) {
    res <- vector("numeric" , reps)                    oder mit power.t.test()
    for (i in 1:reps) {
        x <- rnorm(n = n1 , mean = m1 , sd = s1)
        y <- rnorm(n = n2 , mean = m2 , sd = s2)
        tmp <- t.test(x , y , paired = FALSE)
        res[i] <- (tmp$p.value < alpha)
    }
    list(m = mean(res) , s = sd(res)/sqrt(reps))}
```
→list(…) : m = Macht, s = Standardfehler

Suche der Stichprobengrösse

- seq(…)→ Start- und Endpunkt der Stichprobengrösse sowie Schrittgrösse
- n1, n2 müssen definiert sein
- s1=1, s2=5 sind Beispiele, es müssen dann die richtigen Werte eingesetzt werden

macht ist eine Matrix in denen die Macht je nach Gruppengrösse eingetragen ist. Beim ges. Wert geben die Spalten und die Zeilen die Gruppengrössen an.
- Spalte 1= erster Wert für die Gr. n1=Stichprobengrösse 10
- Zeile 3= dritter Wert für Gr. n2= Stichprobengrösse 30

```
nall <- seq(10 , 100 , by = 10)  [by = 1 wenn es exakt sein muss]
nn <- length(nAll)
macht <- matrix(0 , nn , nn)
for (j1 in 1:nn) {
    cat("Schleife" , j1 , " von " , nn , "\n")
    for (j2 in 1:nn) {
        n1 <- nall[j1]
        n2 <- nAll[j2]
        macht[j1 , j2] <- machtTtestt(n1 =n1 , n2 =n2, s1 =1 , s2 =5)$m
    }
}
```

MACHT BEI DER 1-WEG ANOVA power.anova.test

```
machtAnova <- function(n , mu , s = 1 , reps = 1000, alpha = 0.05) {
    res <- vector("numeric" , reps)
    for (i in 1:reps) {
        x <- rep(LETTERS[1:length(n)] , times = n)
        y <- vector("numeric", 0)
        for (j in 1:length(n)) {
            y <- c(y , rnorm(n[j] , mean = mu[j] , sd = s))
        }
        df <- data.frame (x = x , y = y)
        sm <- summary(aov(y ~ x , data = df))
        pval <- sm[ [ 1 ] ] [ [ 5 ] ] [ 1 ]
        res[i] <- ( pval < alpha )
    }
    list(m = mean(res) , s = sd(res)/sqrt(reps))
}
```
→ list(…) : m = Macht, s = Standardfehler

Macht T-Test:
machtTtest(alt=**"greater")** → $H_A > H_0$
machtTtest(alt=**"two.sided")** → $H_A = H_0$
machtTtest(alt=**"less")** → $H_A < H_0$
machtTtest1(n, m0, mA, s, reps, alpha, alt="two sided")
machtTtest2(n1, n2, m1, m2, s1, s2, reps, alpha, alt="two sided")

Macht eines 2-seitigen 1Stichproben T-test
(n, β0, βA, s, alpha = 0.05)

Macht einer linearen Regression
machtLM(s) → S= Standardabweichung

Macht eines ANOVA-Tests
machtAnova1(n=c(10,10), mu=c(1,rep(0, length(n)-1)), s, reps, alpha)
machtAnova1(n = c(11,11,11)) → *3 Levels mit default Werte (mit n=11)*

Macht eines Fisher Tests
machtFisher(n1, n2, p1, p2, reps, alpha)

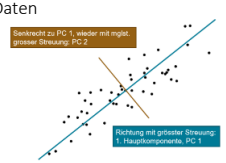
Macht eines Binomialtests
machtBinom(n, p0, pA, reps, alpha, alt="two sided")

Macht einer linearen Regression y=b0+b1*x:
machtLM(n, b0, b1, s, reps, alpha)

PCA-PRINCIPAL COMPONENT ANALYSIS

= **Hauptkomponentenanalyse**. Gut für grosse Menge Daten

Prinzip: Man legt Koordinatenachsen so hin, dass eine möglichst hohe Streuung der Daten um diese Achsen herum sind und alle Achsen im rechten Winkel zueinanderstehen *(ist in Theorie möglich)*
 # PCs ≙ # Variablen (wenn nicht kompr.)



PCA= „Gute Projektion in wenigen Dimensionen (**Gut=** möglichst grosse Streuung der Daten = Möglichst viel Varianz)

Ziele :
Visualisieren von Hoch-domensionalen Datensätzen (<3D)
Komprimieren: viele → wenige Variablen
1-Dimensionalen Index erstellen, der Subjekte möglichst gut unterscheidet

EINHEITEN

Wenn die Einheiten der Messung verändert werden, verändert sich die Varianz → Messung in Metern hat 1'000x grössere Varianz als in Kilometern

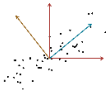
Faustregel:
- Daten **immer zentrieren**
- Falls alle Variablen in der gleichen Einheit sind: **Nicht skalieren**
- Falls Variablen in unterschiedlichen Einheiten sind: **Skalieren**

VORGEHEN

1. Konvention: Zentrieren
Lege den Ursprung der **Koordinatenachse** ins Zentrum der Punktwolke
→Mittelwerte=Achsen

2. Setze die 1. Hauptkomponente
Linearkombination mit grösster empirischer Varianz:
Lege Gerade in Richtung der grössten Streuung der Daten *(Egal, in welche Richtung)*
→ Gerade = **„1. Hauptkomponente" = PC1**. **Normiere** diesen Vektor auf die **Länge 1** $\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right) =: (\phi_{11}, \phi_{21})$ *(Ursprung auf dem Ursprung des Koordinatensystems)* → normaler Vektor PC1 = (ϕ_{11} , ϕ_{21})

3. Setze die 2. Hauptkomponente
Lege eine **Gerade senkrecht zur 1. Hauptkomponente** *(durch den Ursprung des Koordinatensystems)* Gerade soll **in Richtung der grössten Streuung schauen**.
Normiere dann **2. Hauptkomponente PC2** auf die Länge 1
PC2=(φ_{12} , φ_{22}) (Vektor geht durch Ursprung und ist senkrecht zu PC1)



4. Setze die 3. Hauptkomponente
.. und immer so weiter. Es gibt immer genau so viele Hauptkomponenten wie der Datensatz Dimensionen hat. (Die Dimensionalität ändert sich also nicht)

LOADINGS

=Richtung der Hauptkomponenten bezüglich Standardbasis.
PC1=(ϕ_{11} , ϕ_{21}) → φ_{11} , φ_{21} sind die loadings der Hauptkomponente 1
PC2=(ϕ_{12} , ϕ_{22}) → φ_{12} , φ_{22} sind sie loadings der Hauptkomponente 2
Vorzeichen der Werte ϕ_{11} , ϕ_{21}, ... ändern je nach Orientierung der PC's.

$PC1 = \left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right) =: (\phi_{11}, \phi_{21})$

$PC2 = \left(-\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right) =: (\phi_{12}, \phi_{22})$

Datenpunkte: x = [x1, x2]
zentrierte Datenpunkte xm = [x1 - m1, x2 - m2] = [xm1, xm2]
rotierte Daten: (jeweils PC hinunter rechnen!)
$z_1 = PC_1 * xm = xm1*PC1,1+xm2*PC1.2$
$z_2 = PC_2 * xm = xm1*PC2.1+xm2*PC2.2$

	x1	x2
1	-4.9	3.4
2	-2.1	2.2
3	0.9	-3.0

	z1	z2
1	-5.893	-1.065
2	-3.053	0.071
3	2.769	-1.491

Daten mit weniger Dimensionen repräsentieren → man nehme die erste Dimension der rotierten Daten (z1)
*Xm = Matrix mit den zentrierten Daten → Z = Xm * M = rotierte Datenmatrix (M= die Rotationsmatrix mit den PC's in den Spalten)*
Merke: **Varianz= σ^2** → *erklärte Varianz*

PCA-BASISWECHSEL MIT ROTATIONSMATRIX

Standardbasis = ursprüngliches Koordinatensystem
Punkte am Anfang werden durch Koordinaten der Standardbasis beschrieben
Koordinaten bzgl. Standardbasis $xi = (xi1, xi2, \dots, xip)$
→neue Basis ist ein Koordinatensystem, das durch die Hauptkomponenten PC1, PC2, ... beschrieben wird.
Koordinaten bzgl. PC-Basis $zi = (zi1, zi2, \dots, zip)$

Rotationsmatrix ϕ → zum rechnen
Die **Spalten der Rotationsmatrix sind die loadings:**
(erste Spalte = loadings von PC1, zweite Spalte = loadings von PC2, etc.)

$$\phi = \begin{matrix} PC1 & PC2 \\ \end{matrix} \begin{pmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{pmatrix}$$

Basiswechsel von der Standardbasis zur PC-Basis für den Punkt P1 = (Z1 , Z2):

$$\phi\text{-}1 = \begin{pmatrix} \phi_{11} & \phi_{21} \\ \phi_{12} & \phi_{22} \end{pmatrix} * \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}$$ [ϕ wird an Hauptdiagonalen gespiegelt = ϕ-1]

Basiswechsel von der PC-Basis zurück zur Standardbasis:

$$\varphi = \begin{pmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{pmatrix} * \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$$

Beispiel: *Wir machen einen Basiswechsel vom Basissystem zu einem System mit den Hauptkomponenten PC1 = (φ_{11} , φ_{21}) = ($\frac{1}{\sqrt{2}}$, $\frac{1}{\sqrt{2}}$) und PC2 = (φ_{12} , φ_{22}) = ($-\frac{1}{\sqrt{2}}$, $\frac{1}{\sqrt{2}}$) .*
Wir wollen nun die neuen Koordinaten für den Punkt P1 = (X1, X2) = (2, 5) berechnen:

Z_1 = P_1 * PC1 = (X_1, X_2) * (φ_{11} , φ_{21}) = **X_1 * φ_{11} + X_2 * φ_{21}** = $2*\frac{1}{\sqrt{2}} + 5*\frac{1}{\sqrt{2}} \approx 4.9$

Z_2 = P_1 * PC2 = (X_1, X_2) * (φ_{12} , φ_{22}) = **X_2 * φ_{12} + X_2 * φ_{22}** = $2*(-\frac{1}{\sqrt{2}}) + 5*\frac{1}{\sqrt{2}} \approx 2.1$

→ P_1 = (Z_1 , Z_2) = (4.9, 2.1)

Mit R: Matrizen bilden und Z_1=Skalarprodukt mit PC1, Z_2= Skalarprodukt mit PC2
%*% → Skalarprodukt auf R

Scores =Koordinaten bezüglich der Hauptkomponenten (=neue Z Werte)
(4.9, 2.1) sind also die Scores des Punktes P1. Die Scores ändern die Vorzeichen je nach dem, in welche Richtung (positiv/neg.) die Vektoren PC1, PC2... schauen

PC'S FINDEN MITHILFE VON NUMERIK

R-Studio:
prcomp() → Singulärwertzerlegung der Kovarianzmatrix.
[princomp() macht Eigenwertzerlegung, ist eine schlechtere Alternative zu prcomp()]
pr.out <- prcomp(dat, center = TRUE) →Daten werden zentriert, nicht skaliert
pr.out <- prcomp(dat, scale = TRUE) → Daten skalieren
pr.out <- prcomp(dat, center = FALSE) → Daten nicht zentrieren
str(pr.out) → output zeigt Struktur von pr.out (≈ Spaltennamen)
pr.out$center → Koordinaten vom ehemaligen Zentrum
pr.out$rotation
→ gibt die loadings aus = Richtung der PC's bzgl. Standardbasis

Interpretation: Bsp: [1,] beschreibt die **Kraft** und [2,] die **Geschwindigkeit** → **PC1** beschreibt vor allem die **Kraft**, **PC2** vor allem die **Geschwindigkeit.**
Grund: loadings von PC1 sind bezüglich [1,] und von PC2 bezüglich [2,] viel grösser als die anderen loadings.

	PC1	PC2
[1,]	-0.9563643	0.2921768
[2,]	-0.2921768	-0.9563643

→ Wenn man bei Person P die Kraft=2 und die Geschwindigkeit=4 misst, dann sind die Koordinaten dieser Person bezüglich PC1 = 2*-0.956 + 4*-0.292 = -3.08
Die Koordinaten bezüglich PC2 = 2*0.292 + 4*-0.956 = 0.20.

Z <- pr.out$x → gibt die Scores aus = Koordinaten der Punkte bezgl. PC-Basis
→ Output: Zeilen = Punkte P1, P2, P3, ...; Spalten: PC1, PC2, PC3, ...
Z['Datenpunkt','PC'] → Koordinate des rotierten Datenpunkts
Z[2,] → gibt die Koordinate von der 2. Zeile aus
Z[,2] → gibt die scores der Datenpunkte 1,2,3,... bzgl. PC2 aus

summary(pr.out) → output:

```
> summary(pr.out)
Importance of components:
                          PC1     PC2
Standard deviation     3.2269 0.82619
Proportion of Variance 0.9385 0.06152
Cumulative Proportion  0.9385 1.00000
```

- Standard Derivation= absolute Varianz (PC1 > PC2 >...)
  - Proportion of Variance = %Varianz, die mit PC1/PC2 /... erklärt wird
  - Cumulative Proportion = relative Varianz (%) [letzter PC immer = 100%]

plot(pr.out) → nachschauen wo ist Knick
biplot(pr.out, scale = 0) → Projektion vom x-Dimensionalen Raum in 2D

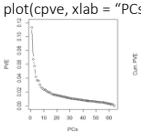
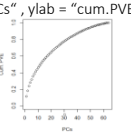
SCREE-PLOT: DIMENSIONEN VERRINGERN

Kompromiss:
0Pcs= perfekt komprimiert aber Varianz nicht erfasst
Alle PCS : nicht komprimiert aber Varianz in Daten perfekt erfasst
Maximale Anzahl PCs: min(Anzahl X-Variablen, Anzahl Samples-1)
Ziel: möglichst viel Varianz in den Daten erfassen
→Varianz nimmt entlang der PC's immer weiter ab → **Abflachung der Varianz**
Faustregel: Meistens behält man so viele PC's, dass **80% der Varianz** erklärt wird.
Vorgehen: Kumulative Varianzen anschauen: Wo ist zum ersten Mal ≥0.8? → Bis und mit dieser PC behalten.

1. Variante:
summary(pr.out) → Schaue bei ***Cumulative Proportion***, wo es ≥ 0.8 wird

2. Variante:
sum <- summary(pr.out)
sum$importance → Gleicher Output wie bei summary(pr.out)
sum$importance[2,] → gibt 2. Zeile aus = unkumulative Varianzen der PC's
sum$importance[2,1] → gibt die unkumulative Varianz von PC1 aus
sum$importance[3,] → gibt die 3. Zeile aus = kumulative Varianzen der PC's
sum$importance[3,1] → gibt die kumulative Varianz von PC1 aus
which(sum$importance[3,] > 0.95)[1] →#PC's die für 95% Varianz gebraucht werden

Erstelle einen Scree-Plot:
```
pve <- sum$importance[2,]
cpve <- sum$importance[3,]
par(mfrow = c(1,2))
plot(pve, xlab = "PCs" , ylab = "PVE" , ylim = c(0,1), type = "b")
plot(cpve, xlab = "PCs" , ylab = "cum.PVE" , ylim = c(0,1), type = "b")
```

links: Varianz erklärt mit PC1/PC2/PC3/...
rechts: kumulative Varianz erklärt mit
PC1 / PC1+2 / PC1+2+3 / PC1+2+3+4 / ...

Weniger Dimensionen:
datNeu <- pr.out$x[1:10] → nimmt PC1 bis PC10 in die Datei datNeu
Wichtig: obwohl Dimensionalität kleiner, braucht immer noch alle Messungen!

PCA durchführen
dim(dat) # n x p
pca <- prcomp(dat, center = TRUE)
summary(pca)
2. Zeile = proportionaler Anteil der gesamten Varianz
3. Zeile= kummulative Varianz = erklärte Varianzen

Rotationsmatrix erstellen : *mit PC1,PC2,.. in den Spalten*
dim(M) # p x p
M <- pca$rotation
«Was ist der 28. Eintrag im PC143-Vektor?» → *M[28, 143]*

rotieren (und ev. zentrieren) der Daten :
dim(Z) # n x p
Z <- pca$x
Z[122,77] → *122.* ***Zeile= Datenpunkt****122, 77.* ***Spalte= PC****77*

Matrix erstellen und bearbeiten

m <- rbind(vektorA, c(5,8,3)) → Matrix mit Zeile 1 = vektorA und Zeile 2 = Vektor (5,8,3) !alle Vektoren müssen gleich lang sein!

df <- as.data.frame(*tab*) → Wandelt die Tabelle *tab* in ein Dataframe um
tab <- xtabs(*y ~ . , data = df*) → Wandelt Dataframe *df* in eine Tabelle um
tab <- xtabs(*y~a+b*, data=*df*) →Wandelt Teil Teil des Dataframe *df* in Tabelle um
counter <- (personen$Jahrgang >= 1990) → Zählt wie viele Pers.einen Jahrgang grösser oder gleich 1990 haben
stud <- (personen[,"*Fach*"] **%in% c**("*Bio*","*Chemie*")) → Spalte *Fach* des Dataframes *personen* wird mit *Bio* und *Chemie* verglichen → *stud* ist ein logischer Vektor mit TRUE überall dort wo das Fach *Bio* oder *Chemie* ist
personen[stud,] → Es werden diejenigen Zeilen vom Dataframe *personen* angezeigt, bei welchen der Vektor *stud* TRUE ist (= alle diejenigen, die Bio oder Chemie studieren
jgstud <- ((personen[,"Fach"] **%in% c**("Bio","Chemie") **&** (personen[,"Jahrgang"] **< 1990**)) → *jgstud* ist ein logischer Vektor der TRUE hat wo das Fach *Bio* oder *Chemie* ist und der *Jahrgang* <1990
personen[jgstud,] → Es werden diejenigen Zeilen vom Dataframe *personen* angezeigt, bei welchen der Vektor *jgstud* TRUE ist (= Alle die Bio oder Chemie studieren und älter als 1990 sind)
plot(x,y) → Zeigt Graphik von x und y an.
plot(x, y, **main** ="*Titel*", **xlab** = "*X-Achse*", **ylab** = "*Y-Achse*", **type** = "l") → Beschriftungen eingestellt. type = "l" = linie, type = "b" = Punkt+Linie
lines (x = c(*2,8*), y = c(*15,5*)) → Linie einfügen von x=2 bis x=8 und von y=15 bis y=5
boxplot(*x~y*, **data=***dat*) → Zeigt Boxplot an
par(mfrow=c(1,3)) → Die *3* nächsten Plots werden nebeneinander dargestellt

Vertrauensintervall: predict.lm(fit, newdata = newP, interval = "confidence", level = 0.95) → Für den Erwartungswert
Vorhersageintervall: predict.lm(fit, newdata = newP, interval = "prediction", level = 0.95) → Einzelne neue Beob. Z.B einzelne Frau

Faktoren als erklärende Variabeln

Faktoren: Haarfarbe, Alter, Geschlecht, Hobbies,...
Level: Werte, die der Faktor annehmen kann

Zwei Levels

Beispiel: $x_i = 0$ für männlich, $x_i = 1$ für weiblich
notion Körpergrösse = $\beta_0 + \beta_1 * x_i + \beta_2 * \text{Alter}_i + \epsilon_i$
wechsel von mann zu frau

Z.B
Intercept: M
gW: change from M to W

Modell

fit <- lm(income ~ Age * Gender, data = *datei*) → Lineare Regression mit WW
fit <- lm(income ~ Age + Gender + Age:Gender, data = *datei*) → = wie oben

output: summary(fit) → Expected Income for Male
Intercept a = Achsenabschnitt für *Male* (Referenzlevel) => Anzahl (m)
Age b = Steigung für *Male* (Referenzlevel) => Schwankung (m), increase per year
GenderFemale c = **Änderung** Achsenabs. für *Female* (Achsenabs.= a + c)
Age:GenderFemale d = **Änderung** Steigung für *Female* (Steigung= b + d)
Modell: Income = (a + c*Gender$_i$) + (b + d*Gender$_i$) * Age + ϵ_i → Change that needs to be added cause of gender eff.
$\epsilon_i \sim N(0,(\text{residual std. error})^2)$
Gender$_i$=0 für Referenzlevel (*Male*), Gender$_i$ = 1 für *Female*
plot(*fit*1, which = c(1,2)) → Residuenanalyse Use yi = ...

R-Studio: Trainings-MSE:

MSE = $\frac{RSS}{n}$

n <- **nrow(***dat***)**
train <- **sample(***n*,*90***)** → aus Zahlenwerten *n* werden *90* Zufallswerte ausgewählt
fit <- **lm(***y ~ . ,* **data =** *dat*, **subset =** *train***)** → **Lineare Regression** mit y abhängig von allen anderen Variablen mit denjenigen Werten aus der Datei *dat*, die durch die 90 Zufallswerte von *train* bestimmt werden.
fit0 <- **lm(***y ~ 1,* **data =** *dat*, **subset =** *train***)** → Bsp. für Modell mit Grad 0
fit3 <- **lm(***y ~ x1 + x2 + x3,* **data =** *dat,* **subset =** *train***)** → Bsp. Grad 3
summary(*fit***)** → Infos zur Regression *fit*
sum(*fit***$residuals^2)** → RSS
mean(*fit***$residuals^2)** → Trainings-MSE Test Data

R-Studio: Test-MSE:

Based on trainings Data
yHut <- **predict(***fit, dat***)** → Vorhersage für alle Datenpunkte aus *dat*
quadratResid <- **(***dat***$***y* **- *yHut*)^2** → Residuenquadrat für alle Datenpunkte aus *dat*
quadratResidTest <- *quadratResid*[-*train*] → Alle Datenpunkte aus Test-Set
TestMSE <- **mean(***quadratResidTest***)** → Test-MSE
TestRMSE <- **sqrt**(TestMSE) → Root Squared Error vom Test-MSE

→ Neue, zukünftige Daten, Fehler auf zukünftige Daten, wenn minimal zukünftige Daten gut erklärt. Gute Vorhersagen, Overfitting vermeiden => Modell mit kl. Test MSE

erklärende Variable => ncol(dat) - 1
0 Je Grösser Residuensumme -> umso schlechter passt das Modell

R-Studio: Modellwahl mit BIC

library(leaps) → Package *leaps* wird benötigt
m <- **regsubsets(***y ~ . ,* **data =** *dat*, **method =** "*exhaustive*", **nvmax =** *20***)** → oder method = "*forward*" oder "*backward*". nvmax gibt maximale Setgrösse = maximale #Variablen an (für weniger Rechenaufwand)
ms <- **summary(**m**)** → *ms* speichert beste Modelle für vorgegebene #Variablen
*ms***$bic** → Gibt die BIC-Werte für die Modelle mit 1,2,3,... Variablen aus
ncoef <- **which.min(***ms***$bic)** → Zeigt welches Modell mit welcher #Variablen die beste Vorhersage hat

coef(m,ncoef**)** → Zeigt die Variablen inkl. Werte für das beste Modell an
Bsp: *bei coef kommen die Variablen x3 und x5 heraus. Dann macht man so weiter:*
fitBest <- **glm(y ~** x3 + x5**, data =** dat**)** → Das ist das Beste Modell!

Falls man noch den Test-MSE bestimmen will:
cv.errBest <- **cv.glm(data =** *dat*, **glmfit =** *fitBest***)**
sqrt(*cv.errBest***$delta)** → erste Zahl = Test-MSE

Neues Dataframe erstellen um **logodds** zu erhalten:
new <- **data.frame(***x = 5, g = „w"***)** →Erstelle neues Df mit allen Variablen
logodds <- **predict.glm(***fit*, **newdata=***new*, **type ="link", se.fit=TRUE)**
Zeilen aus bestehendem Datensatz (beispiel):
logodds <- **predict.glm(***fit*, **newdata = dat[42,], type = "link", se.fit = TRUE)**
=> Werte der Zeile 42 aus *dat*

Wahrscheinlichkeit mit „response" statt mit „link":
p <- **predict.glm(***fit*, **newdata = dat[42,] , type = "response", se.fit = TRUE)**
*p***$fit/(1-***p***$fit)** → odds(y)

Geschätzte Steigung vom 3. Resti./Ansteig vom RIRS für best. Person: (fit! -> RIRS)
summary(fit!) Steig.
ranef(fit!) Steigung für 3. Resti } Add

Individuelle Steigung mit FEM
summary(Fit!/Fit2) Steigung
Add to xR3 Steigung } Add

summary(glht(*fit*, linfct = mcp(*x* = K))) → *x* = beschreibender Faktor aus *fit*

Output

```
> summary(glht(fit,linfct=mcp(M=K)))

         Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: User-defined Contrasts


Fit: aov(formula = y ~ M, data = dat)

Linear Hypotheses:
           Estimate Std. Error t value Pr(>|t|)
M1-P == 0  -6.3886     1.2197  -5.238   <0.001 ***
M2-P == 0  -6.1843     1.2197  -5.070   <0.001 ***
N1-P == 0  -2.9255     1.2197  -2.399   0.0834 .
N2-P == 0  -0.1925     1.2197  -0.158   1.0000
N3-P == 0  -2.0478     1.2197  -1.679   0.3553
N4-P == 0  -1.5157     1.2197  -1.243   0.6542
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

- Der Unterschied von *M1* zu *P* ist geschätzt *-6.3886*
- 95% - Vertrauensintervall vom Unterschied von *M1* zu *P* ist *[estimate ± 2 * std. error] = [-6.3886 ± 2 * 1.2197]*
- Pr(>|t|) ist der p-Wert des Unterschiedes zwischen den verglichenen Faktoren (z.B. zwischen M1 und P). Ist der *p-Wert* ≤ α dann ist der Unterschied *signifikant*

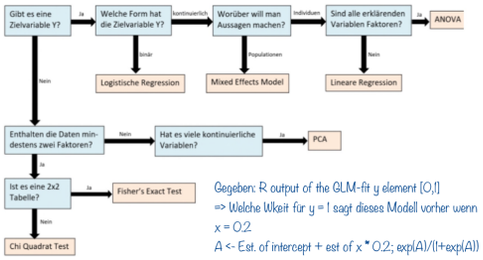
TukeyHSD(*fit*) → Output:
TukeyHSD(*fit*, conf.level = *0.99***)** → 99%-V.I. (95% ist default)

```
> TukeyHSD(fit)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Y ~ F * H, data = Peperoni)

$F
                diff       lwr      upr     p adj
Gruen-Gelb 1.633479 0.3366259 2.930332 0.0101554
Rot-Gelb   5.584670 4.2878167 6.881523 0.0000000
Rot-Gruen  3.951191 2.6543379 5.248044 0.0000000

$H
                diff       lwr      upr     p adj
Spanien-Mexiko 1.462805 0.5819206 2.34369 0.0015736

$`F:H`
                          diff        lwr        upr     p adj
Gruen:Mexiko-Gelb:Mexiko     1.5960868 -0.6523081  3.8444817 0.3041327
Rot:Mexiko-Gelb:Mexiko       5.7392972  3.4909023  7.9876921 0.0000000
Gelb:Spanien-Gelb:Mexiko     1.5409625 -0.7074324  3.7893574 0.3421850
Gruen:Spanien-Gelb:Mexiko    3.2118334  0.9634385  5.4602283 0.0012607
Rot:Spanien-Gelb:Mexiko      6.9710046  4.7226097  9.2193995 0.0000000
Rot:Mexiko-Gruen:Mexiko      4.1432104  1.8948155  6.3916053 0.0000187
Gelb:Spanien-Gruen:Mexiko   -0.0551243 -2.3035192  2.1932706 0.9999997
Gruen:Spanien-Gruen:Mexiko   1.6157466 -0.6326483  3.8641415 0.2911892
Rot:Spanien-Gruen:Mexiko     5.3749178  3.1265229  7.6233127 0.0000000
Gelb:Spanien-Rot:Mexiko     -4.1383347 -6.4467296 -1.9499398 0.0000144
Gruen:Spanien-Rot:Mexiko    -2.5272638 -4.7758587 -0.2786688 0.0190459
```

- *diff* = Geschätzte Unterschiede zwischen den Gruppen
- *lwr / upr* = Grenzen des 95% V.I. für die geschätzten Unterschiede
- *p adj* = für multiples Testen korrigierter p-Wert
- **$F** : Zeigt Unterschied vom Mittelwert an, wenn man Farbe wechselt
  Wenn man von Grünen zu Gelben Peperoni wechselt, ist der Unterschied in der Grösse im Schnitt 1.633479. *[Grüne sind im Schnitt um so viel grösser]*
- **$H** : Unterschied vom Mittelwert, wenn man Herkunftsland wechselt
  Wenn man von Spanien zu Mexiko wechselt, ist der Unterschied in der Grösse der Peperoni im Schnitt 1.462805
- **$'F:H'** : Zeigt Unterschiede, je nachdem wie man Gruppen kombiniert
  - Wenn man von grünen Peperoni aus Mexiko zu Gelben Peperoni aus Mexiko wechselt, sind die grünen Peperoni im Schnitt 1.5960868 [cm] grösser → Dieser Unterschied ist aber nicht signifikant, da der p-Wert = 0.30
  - Wenn man von gelben Peperoni aus Spanien zu roten Peperoni aus Mexiko wechselt, sind die gelben Peperoni aus Spanien im Schnitt -4.1983347 [cm] grösser → Unterschied ist signifikant, da p-Wert ≈ 0.00

plot(*fit*, which = 1:2) → **Residuenanalyse und QQ-Plot**
Annahmen: Daten sind in jeder Gruppe normalverteilt, gleiche Varianz in den Gruppen, Fehler ϵ_{ij} ist unabhängig.

Test, ob 2 Wkeiten gleich sind: power.prop.test()
Gruppengrösse berechnen use: p1, p2, sig.level, power, alternative
p1 = W keit 1. Gruppe; p2 = W keit 2. Gruppe



Gegeben: R output of the GLM-fit y element [0,1]
=> Welche Wkeit für y = 1 sagt dieses Modell vorher wenn x = 0.2
A <- Est. of intercept + est of x * 0.2; exp(A)/(1+exp(A))

RIRS Modell: Interpretation

```
Random effects:
 Groups   Name        Variance Std.Dev. Corr
 Subject  (Intercept) 612.09   24.740
          Days         35.07    5.922   0.07
 Residual             654.94   25.592
Number of obs: 180, groups:  Subject, 18

Fixed effects:
            Estimate Std. Error     df t value Pr(>|t|)
(Intercept)  251.405      6.825 16.998  36.838  < 2e-16
Days          10.467      1.546 16.995   6.771 3.27e-06
```

```
> confint(fm1, oldNames = FALSE)
Computing profile confidence intervals ...
                              2.5 %     97.5 %
sd_(Intercept)|Subject       14.3815822 37.715996
cor_Days.(Intercept)|Subject -0.4815007  0.684966
sd_Days|Subject               3.8011641  8.733383
sigma                        22.8982669 28.857997
(Intercept)                 237.6806955 265.129515
Days                          7.3586533 13.575919
```

- Die mittlere Reaktionszeit (zu Beginn des Experiments) ist 251 ms (95%-VI: [238 ms, 265 ms] – Genauigkeit der Schätzung)
- Eine typische Schwankung der (anfänglichen) Reaktionszeit in der Bevölkerung ist ca. 25 ms (95%-VI: [14 ms, 38 ms] – Streuung in der Bevölkerung)
- Pro Nacht mit Schlafentzug wird die Reaktionszeit im Mittel um 10 ms schlechter (95%-VI: [7 ms/Tag, 14 ms/Tag] – Genauigkeit der Schätzung)
- Eine typische Schwankung der Reaktion auf Schlafentzug ist ca. 6 ms/Tag (95%-VI: [3.8 ms/Tag, 8.8 ms/Tag] – Streuung in der Bevölkerung)
- Es gibt keinen signifikanten Zshg zwischen anfänglicher Reaktionszeit und Wirkung des Schlafentzugs (95%-VI für ρ: [-0.48; 0.68])

Significant wenn nicht 0 im Intervall

3 Methoden, um Signifikanz von k paarweisen Vergleichen auf einem α Konfidenzlevel zu testen:

1) ohne Korrektur: two-sample t-Test von Medikament M_i und M_j
→ **t.test**(dat$y[dat$M == M_j], dat$y[dat$M == M_i], conf.level = α)

2) Bonferroni Korrektur: two sample t-Test von Medi M_i und M_j
→ **t.test**(dat$y[dat$M == M_j], dat$y[dat$M == M_i], conf.level = 1 - (1-α)/K)

3) TukeyHSD-Adjustierung: fit <- aov(y ~ M, dat)
TukeyHSD(fit, conf.level = α)

Anzahl paarweise Vergleiche: m(m-1)/2 (m: Anz. Behandlungen)
= k ⇐ von allen

PCA Beispiele

Beispiele:

1) Daten mit möglichst wenig Informationsverlust nur mit einer Zahl beschreiben: nur erste Dimension → z1 Koordinate

2) Führe PCA durch, 107. Eintrag im PC2 Vektor? → M[107, 2]

3) Führe PCA durch, 70. Datenpunkt, wie gross ist Koordinate vom rotierten Datenpunkt bezüglich PC1?
→ Z[70, 131] ("rotierte Daten": Z)

4) Führe PCA durch, wie gross ist Koordinate der 1. PC vom Datenpunkt in Zeile 306? Z <- pca$x → Z[306,1]

Logistische Regression: Um welchen Faktor ändert sich die odds für Lungenkrebs, wenn man von der Gruppe Männer in Gruppe Frauen wechselt?
=> exp(gW)
Wie gross ist gemäss unserem Modell? => prediction
Falls der Koeffizient signifikatn grösser als der pWert ist ist die strecke gefährlicher