Statistik & Graphiken in R

Forschungsorientierte Vertiefung – Methodenlehre

Dipl.-Psych. Norman Rose

29.11.2007

Agenda

- T-Test
- Lineare Regressionen
- Logistische Regression

t-Test

- Datenerzeugung & Deskriptive Statistiken

```
# Datenerzeugung

x <- c(rep(0,50), rep(1,40))              # Gruppenvariable
y <- 10 + 5*x + rnorm(n=90,mean=0,sd=2)   # Regressand

# Deskritive Statistiken

summary(y)   # Median, Mittelerte & Quantile
mean(y)      # Mittelwert
var(y)       # Varianz
sd(y)        # Standardabweichung
```

t-Test

- t-Test in R

```
# allgemeine t-Test Formel

t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)
```

- Optionen:
  - t-Test bei einer Stichprobe
  - t-Test bei unabhängigen Stichproben
  - t-Test bei abhängigen Stichproben

t-Test

- t-Test bei einer Stichprobe

```
# allgemeine t-Test Formel
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

# t-Test bei einer Stichprobe

t.test(y, mu=20)                                # 2-seitiger Test
t.test(y, alternativ = "less", mu=20)     # einseitiger Test
t.test(y, alternativ = "greater", mu=20)  # einseitiger Test
```

- Argumente:
  - „alternativ“ = Alternativhypothese!

t-Test

- t-Test bei unabhängigen Stichproben

```
# allgemeine t-Test Formel
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

# t-Test bei einer Stichprobe

t.test(y~x)                                  # 2-seitiger Test
t.test(y~x, alternativ = "less")             # einseitiger Test
t.test(y~x, alternativ = "greater")          # einseitiger Test
```

- Argumente:
  - „alternativ“ = Alternativhypothese!
  - „var.equal“ = Varianzhomogenität?

t-Test

- t-Test bei unabhängigen Stichproben

```
# allgemeine t-Test Formel
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = FALSE, var.equal = FALSE,
       conf.level = 0.95, ...)

# t-Test bei einer Stichprobe

t.test(y~x)                                    # 2-seitiger Test
t.test(y~x, alternativ = "less“, mu = -2)  # einseitiger Test
t.test(y~x, alternativ = "greater “, mu = -2)# einseitiger T.
```

... Test der Mittelwertsdifferenz gegen einer bestimmten Wert!

t-Test

- t-Test bei **abhängigen** Stichproben

```
# allgemeine t-Test Formel
t.test(x, y = NULL,
       alternative = c("two.sided", "less", "greater"),
       mu = 0, paired = TRUE, var.equal = FALSE,
       conf.level = 0.95, ...)

# t-Test bei einer Stichprobe

t.test(y0,y1,paired=T)                              # 2-seitiger Test
t.test(y0,y1,paired=T, alternativ = "less“)   # einseitiger Test
t.test(y0,y1,paired=T, alternativ = "greater“) # einseitiger T.
```

Lineare Regression

- Exkurs: multivariate Datenerzeugung

```
# Datenerzeugung (1) via Modellgleichungen

x <- rnorm(n=50,mean=10,sd=2)
y <- 1*x + rnorm(n=50,mean=5,sd=2)

# Datenerzeugung 2 (multivariat)

library(MASS) # Package „MASS“ laden

CovMat<-matrix(c(1,0.5,
                 0.5,1)
                 ,nrow=2,ncol=2,byrow=T)    #Var.-Kov.-Matrix
means<-c(0.2,0.1)                           # Mittelwertsvektor

dat<-mvrnorm(n = 50, mu=means, Sigma=CovMat, empirical = F)
```

Lineare Regression

- Einfache Lineare Regression

```
# Lineares Modell: lm

lm(y~x)              # Einfache lineare Regression  E(Y | X) = α0 + α1X

# zugehörige Funktionen


LReg <- lm(y~x)
coef(LReg)               # Regressionskoeffizienten & Intercept
summary(LReg)            # Modellzusammenfassung
fitted(LReg)
residuals(LReg)
rstandard(LReg)
plot(LReg)
```

Lineare Regression

- Graphiken zum Linearen Regressionsmodell

```
# Lineares Modell: Graphiken

# 4 Graphiken nacheinander!

plot(lm(y~x))

# 4 Graphiken gleichzeitig

LReg<-lm(y~x)
layout(matrix(c(seq(1:4)),nrow=2,byrow=T))
plot(LReg)

# Graphik mit Regressionsgerade

plot(x,y)
abline(LReg)        # Fkt. Zeichnet in den bestehenden Plot!
```

Multiple Lineare Regression

- Funktionen

```
# Lineares Modell: lm

lm(y ~ x + z)              # zweifache lineare Regression
lm(y ~ x + z + z*x)        # bedingte lineare Regression


# zugehörige Funktionen

add1(); drop1(); update(); # zur Modellanpassung/-modifikation
                           # ➔ Terme hinzufügen oder aus dem
                           # Modell nehmen
```

Multiple Lineare Regression - Übung

1. Erzeugen Sie ein Datenbeispiel für ein multiples lineares Regressionsmodell mit einer abhängigen Variable *Y* und drei korrelierten Prädiktoren *X*, *Z* und *W*. Dabei soll der Regressand *Y* linear sein in *X*, *Z*, *W* und *X*W* (Interaktion)!

2. Speichern sie den erzeugten Datensatz als Textfile ab, so das es in anderen Programmen zu Berechnungen zur Verfügung steht!

3. Berechnen sie die multiple Regression in R.

Multiple Lineare Regression - Übung

- Zu Aufgabe 1:

```
# korrelierte Prädiktoren erzeugen
    CovMat<-matrix(c(1, 0.5,0.4,
                     0.5, 1,0.2,
                     0.4,0.2, 1)  ,nrow=3,ncol=3,byrow=T)
    means<-c(0.2,0.1)
    dat<-mvrnorm(n = 100, mu=means, Sigma=CovMat, empirical = F)

# Datentabelle mit Variablennamen erzeugen:
    dat<-data.frame(dat)
    colnames(dat)=c("x","z")

# Regressand erzeugen

      attach(dat)      # „öffnet“ Objekte für R-Suchpfad!
      y<- 4 + 0.5*x + 1*z - 2*w + 0.5*x*w + rnorm(100,0,1)
      detach(dat)      # „schließt“ Objekte für R-Suchpfad!

      daten<-data.frame(dat,y) # Gesamtdaten
```

Multiple Lineare Regression - Übung

- Datensatz speichern

```
# Datensatz in einem Verzeichnis/Ordner speichern

 setwd("C:\\Rechnen")
 write.table(daten, file="tutorial.dat", dec=".", sep="\t",
             col.names=T,row.names=F,quote=F)
```

- Multiple Regression rechnen & Graphiken

```
# Multiples Regressionsmodell
     attach(dat)
     MLReg<-lm(y ~ x + z + w + w*x)
     detach(dat)
     summary(MLReg)

# Graphiken
     layout(matrix(c(seq(1:4)),nrow=2,byrow=T))
     plot(MLReg)
```

Multiple Lineare Regression

- R^2-Differenzentest

```
# R^2-Differenzentest

    attach(dat)
    MLReg1<-lm(y ~ x + z)            # Regressionsmodell 1
    MLReg2<-lm(y ~ x + z + w + w*x) # Regressionsmodell 1
    detach(dat)

    anova(MLReg1,MLReg2)  # Varianzanalyse über lm- oder glm-
                            Objekte!
```

Multiple Lineare Regression

- Berechnung des Modells erfolgt im Allgemeinen Linearen Modell `glm()` in R:

```
# Logistische Regression

    attach(dat)
    MLReg1<-lm(y ~ x + z)            # Regressionsmodell 1
    MLReg2<-lm(y ~ x + z + w + w*x) # Regressionsmodell 1
    detach(dat)

    anova(MLReg1,MLReg2)  # Varianzanalyse über lm- oder glm-
                            Objekte!
```

Logistische Regression

- Daten mit einem dichotomen Regressanden generieren

```
# Datenerzeugung

    CovMat<-matrix(c(1, 0.5,
                     0.5, 1)  ,nrow=2,ncol=2,byrow=T)
    means<-c(0.5,0.1)
    dat<-mvrnorm(n = 100, mu=means, Sigma=CovMat, empirical = F)
    dat<-data.frame(dat)
    colnames(dat)=c("x","z")
```

- Stochastische Responsen erzeugen

```
# Antwortwahrscheinlichkeiten für Y = 1 berechnen!

      attach(dat)
      Py<-((exp(0 + 2*x - 1*z))/(1 + (exp(0 + 2*x - 1*z))))
      detach(dat)
```

Logistische Regression

- Mit den individuellen Antwortwahrscheinlichkeiten „würfeln“

```
# Datenerzeugung

      wuerfel<-runif(length(Py))
      cutter<- wuerfel - Py

      y<-rep(0,length(cutter))
      for(i in 1 : length(cutter))
          {
            y[i] <- if (cutter[i] > 0) 0 else 1
          }
```

Logistische Regression

- Über das Argument `family` wird die Linkfunktion im `glm` spezifiziert!

```
attach(dat)
LogReg<-glm(y ~ x + z, family=binomial)
summary(LogReg)
```

- `binomial` ➔ Logistische Funktion wird als Linkfunktion verwendet ➔ somit logistische Regression!