Programmieren in Anwendungen

Annette Bieniusa

Technische Universität Kaiserslautern

bieniusa@cs.uni-kl.de

10.07.2014

Überblick

Regressionsdiagnostik

Generalisierte Lineare Modelle

Funktionen in R

Literatur und Quellen

- Beispiele dieser Vorlesung sind entnommen:
  - *Rob Kabacoff. R in Action - Data Analysis and Graphics with R. Manning, 2010*
  - *Drew Conway, John Myles White. Machine Learning for Hackers. O'Reilly 2012*
  - *Slava Rokicki, R For Public Health*
    `http://rforpublichealth.blogspot.de/2014/07/3-ways-that-functions-can-improve-your.html`

Regressionsdiagnostik

Multiple lineare Regression

- Beispiel: Morde in US Staaten
- Datensatz `state.x77` aus dem Basis-Packet liefert Basisdaten zu US-Staaten (aus dem Zeitraum 1970-1975)
- Größe der Bevölkerung `Population`
- Pro-Kopf-Einkommen `Income`
- Analphabetenrate `Illiteracy`
- Tage mit Temperaturen unter 0 `Frost`
- Mord und Totschlag pro 100.000 Einwohner `Murder`

```
#Umwandlung von Matrix in Data frame
> states <- as.data.frame(state.x77
   [,c("Murder","Population","Illiteracy","Income","Frost")])
> head(states)
           Murder Population Illiteracy Income Frost
Alabama      15.1       3615        2.1   3624    20
Alaska       11.3        365        1.5   6315   152
Arizona       7.8       2212        1.8   4530    15
Arkansas     10.1       2110        1.9   3378    65
California   10.3      21198        1.1   5114    20
Colorado      6.8       2541        0.7   4884   166
```

Korrelation der Faktoren

- Mordrate steigt mit Populationsgröße und Analphabetenrate, sie fällt mit Einkommen und Frosttagen
- Andererseits haben frostige Staaten weniger Bevölkerung, niedrigere Analphabetenraten und höhere Einkommen

```
> cor(states)
               Murder Population Illiteracy     Income      Frost
Murder      1.0000000  0.3436428  0.7029752 -0.2300776 -0.5388834
Population  0.3436428  1.0000000  0.1076224  0.2082276 -0.3321525
Illiteracy  0.7029752  0.1076224  1.0000000 -0.4370752 -0.6719470
Income     -0.2300776  0.2082276 -0.4370752  1.0000000  0.2262822
Frost      -0.5388834 -0.3321525 -0.6719470  0.2262822  1.0000000
```

Fitten des Models

```
> fit <- lm(Murder ~ ., data = states)
> summary(fit)

Call:
lm(formula = Murder ~ ., data = states)

Residuals:
    Min      1Q  Median      3Q     Max
-4.7960 -1.6495 -0.0811  1.4815  7.6210

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.235e+00  3.866e+00   0.319   0.7510
Population  2.237e-04  9.052e-05   2.471   0.0173 *
Illiteracy  4.143e+00  8.744e-01   4.738 2.19e-05 ***
Income      6.442e-05  6.837e-04   0.094   0.9253
Frost       5.813e-04  1.005e-02   0.058   0.9541
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.535 on 45 degrees of freedom
Multiple R-squared:  0.567, Adjusted R-squared:  0.5285
F-statistic: 14.73 on 4 and 45 DF,  p-value: 9.133e-08
```

Erklärungen zum Modell

- Statistisch signifikant sind der Einfluss von Analphabetismus und Populationsgröße auf die Mordrate
- Die Mordrate ist nicht linear abhängig von Einkommen oder Temperatur
- Insgesamt erklären die Variablen etwa einen Anteil von 56%

Welches ist das beste Regressionsmodell?

- Es gibt verschiedene Möglichkeiten das Modell zu verändern
  - Enfernen von Messpunkten
  - Hinzufügen von polynomiellen Termen
  - Hinzufügen von Termen bestehend aus Interaktionen von Variablen
  - Hinzunehmen oder Entfernen von unabhängigen Variablen
- Was liefert nun das beste Modell?

ANOVA in für lineare Regressionsmodelle

- Varianzanalyse (engl. *ANOVA* - analysis of variance)
- Nur anwendbar auf genestete lineare Regressionsmodellem, die auf dem gleichen Datensatz beruhen
- Betrachtet die Varianz der Residuen für die gefitteten Modelle
- Frage: Wird die Varianz der Residuen signifikant reduziert durch das erweiterte Modell?
- Hypothese bei ANOVA: Die beiden Modelle sind gleich
- Ist der p-Wert nun < 0.05, kann diese Hypothese verworfen werden.

Vergleich der beiden Modelle mittels ANOVA

- Wird durch Hinzunahme der Parameter Income und Frost das Modell 1 signifikant verbessert?
- Hier: Modell 1 ist in Modell 2 genestet, da es Modell 2 verfeinert und erweitert
- Hypothese: Modell 1 und 2 sind gleich
- Da der Test definitiv nicht signifikant ist ($Pr(> F) = 0.9939$), kann diese Hypothese nicht verworfen werden!

```
> fit1 <- lm(Murder ~ Population + Illiteracy + Income + Frost,
             data=states )
> fit2 <- lm(Murder ~ Population + Illiteracy,
             data=states )
> anova(fit2,fit1)
Analysis of Variance Table

Model 1: Murder ~ Population + Illiteracy
Model 2: Murder ~ Population + Illiteracy + Income + Frost
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     47 289.25
2     45 289.17  2  0.078505 0.0061 0.9939
```

Variablenauswahl

- Regression durch schrittweises Hinzufügen von Variablen (vorwärts), durch schrittweises Entfernen von Variablen (rückwärts), oder bidirektionale, iterative Kombination (siehe `stepAIC()` im `MASS` Package)
- Regression durch alle möglichen Untermengen von Variablen (siehe `regsubsets()` im `leaps` Package)

Generalisierte Lineare Modelle

Generalisierte Lineare Modelle

- Bisher: Lineare Modelle für normalverteilte Zufallsvariablen
- Generalisierte Lineare Modelle sind eine Verallgemeinerung für Zufallsvariablen aus der Exponentialfamilie
- Dazu gehören u.a. die Normalverteilung, Binomialverteilung, Poisson-Verteilung
- Dies erlaubt Regression für kategorielle Daten und auch zählbare Ereignisse

Logistische Regression

- Anwendungsfall: Abhängige Variable ist dichotom (d.h. sie nimmt nur zwei mögliche Werte an)

```
# Modell
m <- glm(y ~ x1 + x2 + x3, family=binomial)
# Vorhersage
dfrm <- data.frame(x1=value,x2=value,x3=value)
predict(m, type="response", newdata=dfrm)
```

Drum prüfe wer sich ewig bindet,...

- Beispiel: Datensatz zur Untreue in Paket `AER`
- Erfasste Daten: Anzahl der Affären, Geschlecht, Alter, Ehedauer, Kinder, Religiosität (1=anti...5=sehr), Ausbildung (9=Volksschule..20=Dr.), Beschäftigung (Hollingshead-Klassifikation), Bewertung der Ehe (1=very unhappy...5=very happy)
- Schritt 1: Installieren des Packages und Laden des Datensatzes

```
install.packages("AER")
data(Affairs)
```

Übersicht des Datensatzes

```
> summary(Affairs)
   affairs         gender         age
 Min.   : 0.000  female:315   Min.   :17.50
 1st Qu. : 0.000  male  :286   1st Qu.:27.00
 Median  : 0.000               Median :32.00
 Mean    : 1.456               Mean   :32.49
 3rd Q . : 0.000               3rd Qu.:37.00
 Max.    :12.000               Max.   :57.00

   yearsmarried   children     religiousness
 Min.   : 0.125    no :171     Min.   :1.000
 1st Qu.: 4.000    yes:430     1st Qu.:2.000
 Median : 7.000                Median :3.000
 Mean   : 8.178                Mean   :3.116
 3rd Qu.:15.000                3rd Qu.:4.000
 Max.   :15.000                Max.   :5.000

   education       occupation      rating
 Min.   : 9.00   Min.   :1.000   Min.   :1.000
 1st Qu.:14.00   1st Qu.:3.000   1st Qu.:3.000
 Median :16.00   Median :5.000   Median :4.000
 Mean   :16.17   Mean   :4.195   Mean   :3.932
 3rd Qu.:18.00   3rd Qu.:6.000   3rd Qu.:5.000
 Max.   :20.00   Max.   :7.000   Max.   :5.000
```

Wer hatte eine Affäre?

- Wie oft gingen die Befragten fremd?

```
> table(Affairs$affairs)
  0   1   2   3   7  12
451  34  17  19  42  38
```

- Schritt 2: Hinzufügen eines Factors, ob ein Affäre stattfand

```
> Affairs$hadAffair[Affairs$affairs > 0] <- 1
> Affairs$hadAffair[Affairs$affairs == 0] <- 0
> Affairs$hadAffair <- factor(Affairs$hadAffair,levels=c(0,1),
                              labels=c("No","Yes"))
> table(Affairs$hadAffair)

 No Yes
451 150
```

Fitten eines Models

- Gesucht: Modell, das vorhersagt, ob eine Person eine Affäre hat
- Schritt 3: Fitten mittels logistischer Regression

```
fit.full <- glm(hadAffair ~ gender + age + yearsmarried
                + children + religiousness + education
                + occupation + rating, data=Affairs, family=binomial)
```

Übersicht des Modells

```
> summary(fit.full)

Call:
glm(formula = hadAffair ~ gender + age + yearsmarried + children +
    religiousness + education + occupation + rating, family = binomial,
    data = Affairs)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5713  -0.7499  -0.5690  -0.2539   2.5191

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.37726    0.88776   1.551 0.120807
gendermale    0.28029    0.23909   1.172 0.241083
age          -0.04426    0.01825  -2.425 0.015301 *
yearsmarried  0.09477    0.03221   2.942 0.003262 **
childrenyes   0.39767    0.29151   1.364 0.172508
religiousness -0.32472    0.08975  -3.618 0.000297 ***
education     0.02105    0.05051   0.417 0.676851
occupation    0.03092    0.07178   0.431 0.666630
rating       -0.46845    0.09091  -5.153 2.56e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 675.38  on 600  degrees of freedom
Residual deviance: 609.51  on 592  degrees of freedom
AIC: 627.51

Number of Fisher Scoring iterations: 4
```

Einfacheres Modell durch Reduzieren der Faktoren

```
> fit.reduced <- glm(hadAffair ~ age + yearsmarried
+ + religiousness + rating, data=Affairs, family=binomial)
> summary(fit.reduced)

Call:
glm(formula = hadAffair ~ age + yearsmarried + religiousness +
    rating, family = binomial, data = Affairs)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6278  -0.7550  -0.5701  -0.2624   2.3998

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.93083    0.61032   3.164 0.001558 **
age          -0.03527    0.01736  -2.032 0.042127 *
yearsmarried  0.10062    0.02921   3.445 0.000571 ***
religiousness -0.32902    0.08945  -3.678 0.000235 ***
rating       -0.46136    0.08884  -5.193 2.06e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 675.38  on 600  degrees of freedom
Residual deviance: 615.36  on 596  degrees of freedom
AIC: 625.36

Number of Fisher Scoring iterations: 4
```

Interpretation des Modells

- Regressionskeoffizienten beschreiben die Veränderung im *Logarithmus* der Odds, dass die Vorhersagevariable den Wert 1 annimmt.
- Einfacher: Wende die Exponentialfunktion an!

```
> exp(coef(fit.reduced))
(Intercept)           age  yearsmarried religiousness        rating
  6.8952321     0.9653437     1.1058594     0.7196258     0.6304248
```

- Alter, Religiosität und Zufriedenheit in der Ehe verringern die Chance, Dauer der Ehe erhöht die Chance auf eine Affäre.
- Das Risiko für eine Affäre ist um den Faktor 1.106 erhöht für jedes Jahr, das man länger verheiratet ist (wobei alle anderen Faktoren konstant gehalten werden).
- 10 Ehejahre ergeben einen Faktor von $1.106^10 \approx 2.7$.

Einfluss von Faktoren auf die Wahrscheinlichkeit einer Affäre

- Vorhersage der Wahrscheinlichkeiten
- Beispiel: Variationen im Beziehungsstatus

```
> testdata <- data.frame(rating=seq(1:5),
    age=mean(Affairs$age),
    yearsmarried=mean(Affairs$yearsmarried),
    religiousness = mean(Affairs$religiousness))
> testdata$prob <- predict(fit.reduced, newdata=testdata,
    type="response")
> testdata
  rating      age yearsmarried religiousness      prob
1      1 32.48752     8.177696      3.116473 0.5302296
2      2 32.48752     8.177696      3.116473 0.4157377
3      3 32.48752     8.177696      3.116473 0.3096712
4      4 32.48752     8.177696      3.116473 0.2204547
5      5 32.48752     8.177696      3.116473 0.1513079
```

Funktionen in R

Funktionen in R

- Funktionen helfen Code zu strukturieren und Duplikationen zu vermeiden
- Weniger Fehler durch Wiederverwendung von Operationen
- Maßgeschneiderte Ausgabe und Anpassungen

Syntax

- Allgemeine Syntax

```
name.of.function <- function(arg1, arg2, ...) {
    statements
    return(something)
}
```

- Beispiel: Quadratfunktion

```
square.it <- function(x) {
    square <- x * x
    return(square)
}
```

- Aufruf

```
# square a number
> square.it(5)
[1]  25
# square a vector
> square.it(c(1, 4, 2))
[1]  1 16  4
```

Lokale und globale Umgebung

- Explizites Return-Statement ist nicht notwendig

```
fun1 <- function(x) {
    3 * x - 1
}
fun1(5)
## [1] 14
```

- Objekte, die innerhalb von Funktionen erzeugt werden, sind nur innerhalb der Funktion verfügbar, falls sie nicht explizit herausgegeben werden

```
fun2 <- function(x) {
    y <- 3 * x - 1
}
fun2(5)
print(y)
## Error: object 'y' not found
```

Lokale und globale Umgebung

- Beispiel: Lokale Variablen und Parameter

```
my.fun <- function(X.matrix, y.vec, z.scalar) {
   # Quadriere den Skalarwert
   sq.scalar <- z.scalar^2

    # Multipliziere Matrix mit Vektor
    mult <- X.matrix %*% y.vec

    # Multipliziere das Ergebnis mit dem quadrierten Skalarwert
    final <- mult * sq.scalar

    # Rueckgabe des Ergebnisses
    return(final)
}
```

Rückgabe mehrerer Objekte

- Trick: Zusammenfassen in eine Liste

```
another.fun <- function(sq.matrix, vector) {
    step1 <- t(sq.matrix)
    step2 <- vector * vector
    final <- list(step1, step2)
    return(final)
}
outcome <- another.fun(sq.matrix = cbind(c(1, 2), c(3, 4)),
                       vector = c(2, 3))
print(outcome)
## [[1]]
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
##
## [[2]]
## [1] 4 9
```

Rückgabe mehrerer Objekte

- Separieren der einzelnen Elemente mit dem [[ ]]-Operator

```
outcome[[1]]
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4

outcome[[2]]
## [1] 4 9
```

Fehlermeldungen

- Stop der Ausführung und Ausgabe einer Fehlermeldung

```
fun.with.check <- function(matrix, vector) {
    if (dim(matrix)[2] != length(vector)) {
        stop("Can't multiply matrix%*%vector because
              the dimensions are wrong")
    }
    product <- matrix %*% vector
    return(product)
}
```

- Ausführung

```
my.mat <- cbind(c(1, 3, 4), c(5, 4, 3))
fun.with.check(my.mat, c(6, 5))
##      [,1]
## [1,]   31
## [2,]   38
## [3,]   39

fun.with.check(my.mat, c(6, 5, 7))
## Error: Can't multiply matrix%*%vector because the
## dimensions are wrong
```

Style-Guide: Wie schreibt man Funktionen?

- Unterteile umfangreiche Funktionen in kleinere Einheiten und weitere Funktionen auf!
- Kommentiere die Verwendung von Parametern und Rückgabewert!
- Teste Funktionen mit verschiedenen Eingaben!

Beispiel: Eigene Zusammenfassungen

- Ausgabe von `summary()` ist unübersichtlich, wenn eine Datensatz sehr viele Spalten hat
- Idee: Anpassung der Zusammenfassung, so dass die Ausgabe kompakter ist

```
#function to summarize the variables in the data
summarize.vars<-function(data){

  #use dummies package to turn all factors into dummies
  require(dummies, quietly=TRUE)
  dat.d<-dummy.data.frame(data, dummy.class="factor")

  #use apply to calculate statistics for each variable
  mat<-t(apply(dat.d, 2, function(x) c(length(x),
                                       round(mean(x, na.rm=TRUE),2),
                                       round(sd(x, na.rm=TRUE),2),
                                       round(min(x, na.rm=TRUE),2),
                                       round(max(x, na.rm=TRUE),2),
                                       length(x)-length(x[!is.na(x)]))))

  #assign column names and rownames to output table
  colnames(mat)<-c("N","Mean","SD","Min","Max","Num Missing")
  rownames(mat)<-colnames(dat.d)
  return(mat)
}
```

Erläuterungen

- `dummy.data.frame` wandelt alle Faktoren in numerische Werte um (`dummy.class="factor"`))
- `apply` wendet hier eine anonyme Funktion auf die Spalten (2) des Dataframes an
- Die Funktion `t` transponiert eine Matrix bzw. Dataframe

Beispiel: Affären

```
> summarize.vars(Affairs)
                 N  Mean   SD   Min Max Num Missing
affairs        601  1.46 3.30  0.00  12           0
genderfemale   601  0.52 0.50  0.00   1           0
gendermale     601  0.48 0.50  0.00   1           0
age            601 32.49 9.29 17.50  57           0
yearsmarried   601  8.18 5.57  0.12  15           0
childrenno     601  0.28 0.45  0.00   1           0
childrenyes    601  0.72 0.45  0.00   1           0
religiousness 601  3.12 1.17  1.00   5           0
education      601 16.17 2.40  9.00  20           0
occupation     601  4.19 1.82  1.00   7           0
rating         601  3.93 1.10  1.00   5           0
```