Programmieren in Anwendungen

Annette Bieniusa

Technische Universität Kaiserslautern

bieniusa@cs.uni-kl.de

03.07.2014

Überblick

Regressionsanalyse

Empfehlungssyteme

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*
  - *ŷhat* - Data Science Operations Platform: `https://docs.yhathq.com/r/examples/movie-recommender`

Regressionsanalyse

Regressionsanalyse

- Kernkonzept der Statistik
- Bestimmung der Art und Stärke der Beziehung zwischen einer abhängigen und anderer unabhängigen Variablen
- Prognose anhand eines Vorhersagemodels, das auf dieser Beziehung beruht
- Beispiele
  - Welche Faktoren (Alter, Verkehrsaufkommen, Design, Material, Wetter) haben Einfluss auf die Abnutzung von Straßen?
  - Welchen Einfluss hat der familiäre Hintergrund (Einkommen, Ausbildung der Eltern, Herkunftsland, Familienzusammensetzung) auf die Schulnoten von Kindern?
  - Wie ist das Verhältnis von Bluthochdruck, Salzkonsum und Alter von Menschen?
- R bietet mehr als 200 verschiedene Regressionsmethoden an!
- Literatur: *Rob Kabacoff. R in Action - Data Analysis and Graphics with R. Manning, 2010*

Mathematisches Modell

- Allgemeines Modell für unabhängige Varialen $x_1, \ldots, x_k$ und abhängige Variable y

$$Y = f(x_1, \ldots, x_k) + \epsilon$$

- Die abhängige Variable wird als Zufallsvariable modelliert, $x_1, \ldots, x_k$ sind hingegen Messstellen
- ϵ bezeichnet die Störgröße bzw. das Residuum des Modells
- ϵ ist eine normalverteile Zufallsvariable mit Erwartungswert 0 und konstanter Varianz σ^2

OLS Regression

- In dieser Vorlesung: Lineare Regression mittles Methode der kleinsten Quadrate (OLS = Ordinary least squares)

$$Y = f(x^n, \ldots, x^k) + \epsilon = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k$$

- Mathematisches Modell für die Vorhersage:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_{1,i} + \cdots + \hat{\beta}_k X_{k,i}, \qquad i = 1 \ldots n$$

- n ist die Anzahl der Beobachtungen bzw. Messungen
- $\hat{Y}_i$ ist die Vorhersage für Y
- $X_{1,i}, \ldots, X_{k,i}$ sind die Messwerte für Messung i
- $\hat{\beta}_i$ sind die Regressionskoeffizienten

Lineare Regression

Ziel: Wähle die Regressionskoeffizienten $\hat{\beta}_j, j = 0, \ldots, k$, so dass das Quadrat der Differenz zwischen der Beobachung Y_i und der Prognose $\hat{Y}$ minimiert wird

$$\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 = \sum_{i=1}^{n}\left(Y_i - \hat{\beta}_0 + \hat{\beta}_1 X_{1,i} + \cdots + \hat{\beta}_k X_{k,i}\right)^2 = \sum_{i=1}^{n}\epsilon^2$$

Lineare Regression in R

- Lineare Regression

  `fitted.model <- lm (formula,data)`
- `formula` beschreibt das Modell, `data` den Datensatz
- Typische Formel: `Y` $\sim$ `X1 + X2 + .... + Xk`
- Bedeutung: Y wird durch Variablen $X1, \ldots, Xk$ beschrieben

Weitere Formelelemente

Gegeben sei ein Datenframe mit Variablen `X,Y,Z,W`.

- `:` beschreibt Interaktion von Variablen, z.B. für `Y ~ X + Z + X:Z` wird Y als Kombination von X, Z und einer Interaktion von X und Z modelliert
- `.` umfasst alle Variablen ausser der abhängigen Variablen, `Y ~ .` bedeutet `Y ~ X + Z + W`
- `I()` beschreibt arithmetische Beziehungen, z.B. `Y ~ X + I((Z+W)^2)` modelliert Y als Kombination von X und einer neuen Variable bestehend aus dem Quadrat des Summe von Z und W

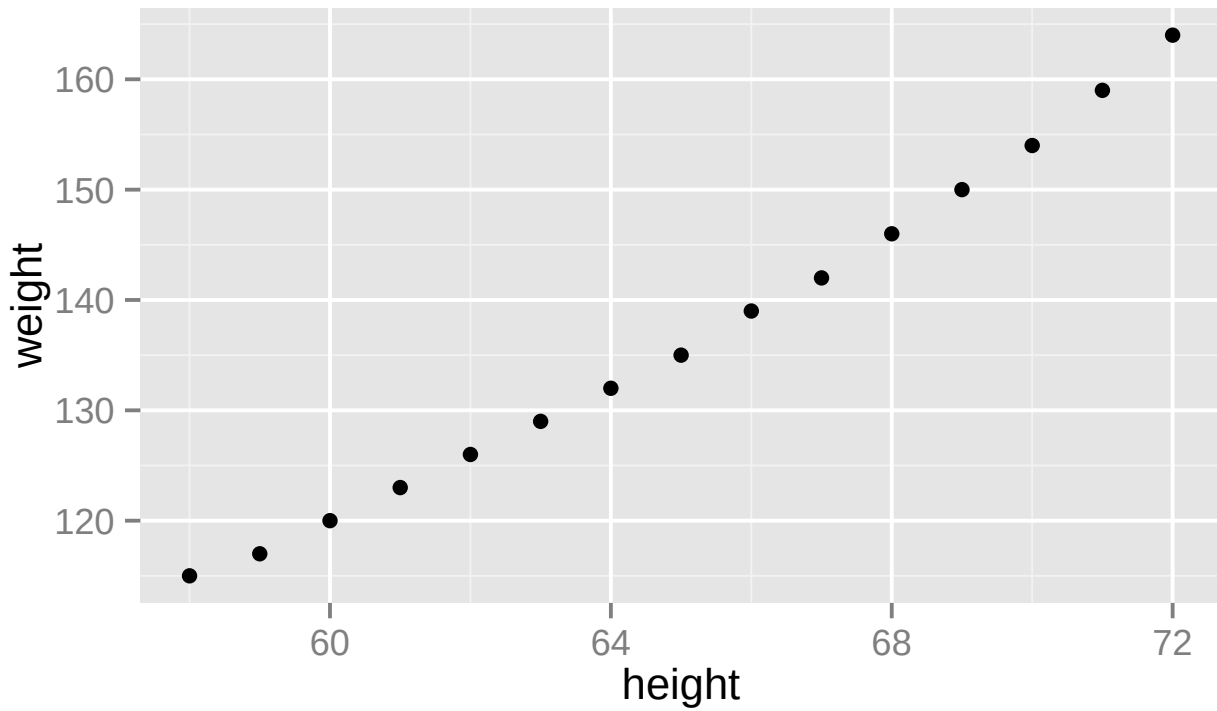
Beispiel: Größe und Gewicht

- Datensatz `women` liefert Messdaten zu Größe und Gewicht von Frauen

```
> summary(women)
     height          weight
 Min.   :58.0   Min.   :115.0
 1st Qu.:61.5   1st Qu.:124.5
 Median :65.0   Median :135.0
 Mean   :65.0   Mean   :136.7
 3rd Qu.:68.5   3rd Qu.:148.0
 Max.   :72.0   Max.   :164.0
```

Visualisierung

```
> p <- ggplot(women,aes(height,weight)) + geom_point()
> p
```



Einfache lineare Regression

```
> fit <- lm(weight ~ height, data=women)
> summary(fit)

Call:
lm(formula = weight ~ height, data = women)

Residuals:
    Min      1Q  Median      3Q     Max
-1.7333 -1.1333 -0.3833  0.7417  3.1167

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -87.51667    5.93694  -14.74 1.71e-09 ***
height        3.45000    0.09114   37.85 1.09e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.525 on 13 degrees of freedom
Multiple R-squared:  0.991, Adjusted R-squared:  0.9903
F-statistic:  1433 on 1 and 13 DF,  p-value: 1.091e-14
```

Interpretation

- Mathematisches Modell

$$\widehat{Weight} = -87.52 + 3.45 \times Height$$

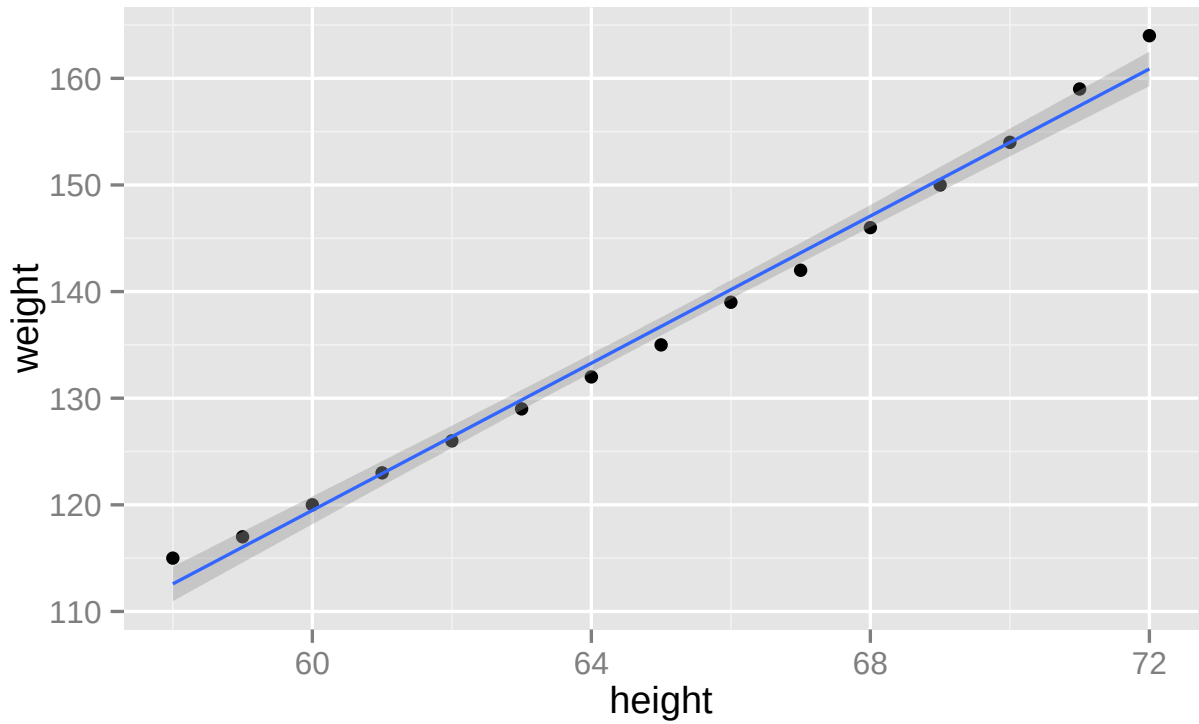
- Regressionskoeffizient von 3.45 ist signifikant unterschiedlich von 0 ($Pr(>|t|) = 1.09e-14$)
- `Multiple R-squared: 0.991` gibt die Korrelation zwischen tatsächlichem und vorhergesagtem Wert
- D.h. 99.1% des Wertes können durch das Modell erklärt werden
- `Residual standard error: 1.525` kann als Standardabweichung bei der Vorhersage gedeutet werden.
- `p-value: 1.091e-14` der F-Statistik testet, ob die unabhängigen Variablen eine zuverlässige Vorhersage bieten

Funktionen für das Modell

- `summary(fit)` zeigt eine detaillierte Übersicht
- `coefficients(fit)` gibt die Modellparameter wieder
- `residuals(fit)` liefert die Residuen, `fitted()` die Vorhersagen für die Messpunkte

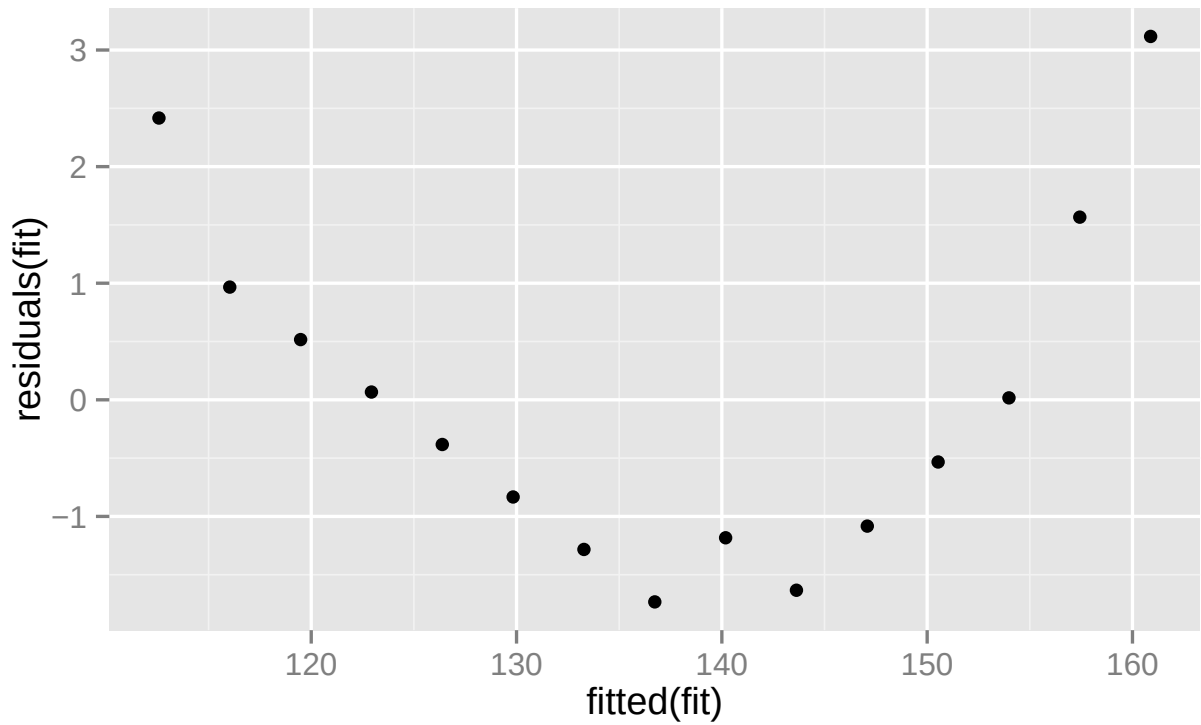
Visualisierung des linearen Modells

```
> p + stat_smooth(method="lm", se=TRUE)
```



Visualisierung der Residuuen

```
> qplot(fitted(fit),residuals(fit))
```



- Residuen sind scheinbar nicht normalverteilt!

Polynomielle Regression

- Abhängigkeit zu vorherzusagender Variable ist ein Polynom n-ten Grades
- Beispiel für quadratische Abhängigkeit:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1^2$$

- Es handelt sich dennoch um ein **lineares** Modell!

Polynomielle Regression in R

```
> fit2 <- lm(weight ~ height + I(height^2), data=women)
> summary(fit2)

Call:
lm(formula = weight ~ height + I(height^2), data = women)

Residuals:
     Min       1Q   Median       3Q      Max
-0.50941 -0.29611 -0.00941  0.28615  0.59706

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 261.87818   25.19677  10.393 2.36e-07 ***
height       -7.34832    0.77769  -9.449 6.58e-07 ***
I(height^2)   0.08306    0.00598  13.891 9.32e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3841 on 12 degrees of freedom
Multiple R-squared:  0.9995, Adjusted R-squared:  0.9994
F-statistic: 1.139e+04 on 2 and 12 DF,  p-value: < 2.2e-16
```

Beobachtung

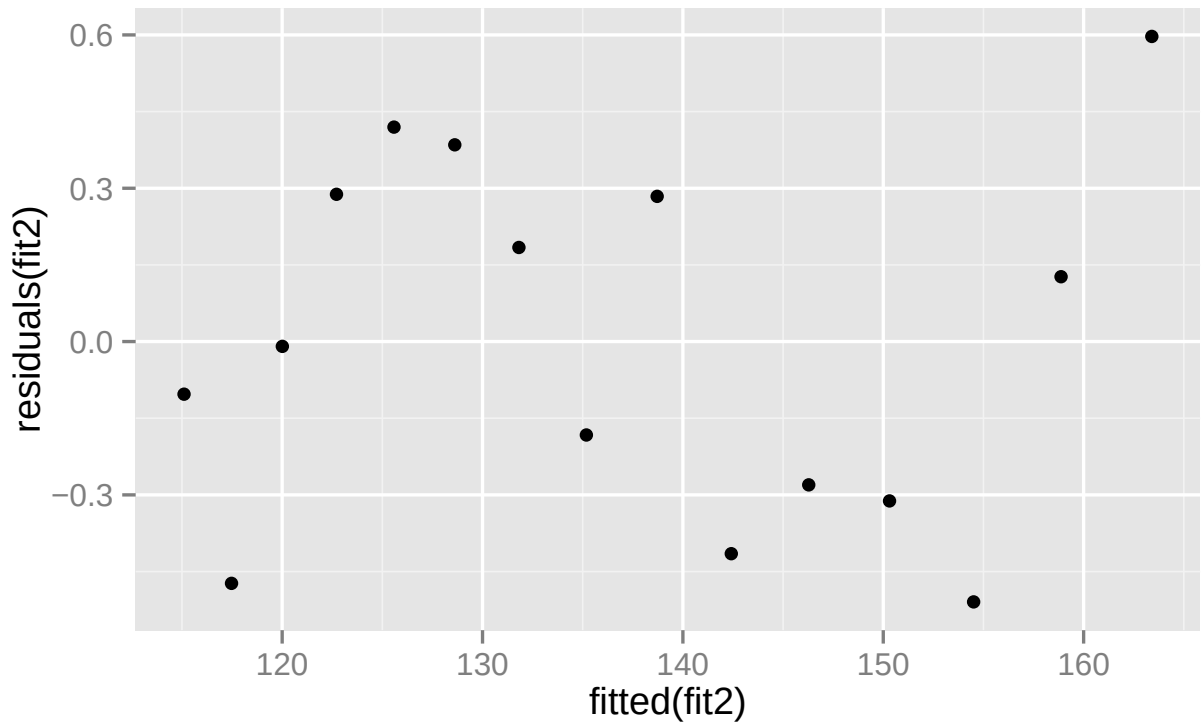
- Neues Modell:

$$\widehat{Weight} = 261.88 - 7.35 \times Height + 0.083 \times Height^2$$

- Alle Regressionskoeffizienten sind statistisch relevant ($Pr(>|t|) < 0.001$)
- `Multiple R-squared: 0.9995` zeigt noch höhere Korrelation

Visualisierung der Residuuen

```
> qplot(fitted(fit2),residuals(fit2))
```

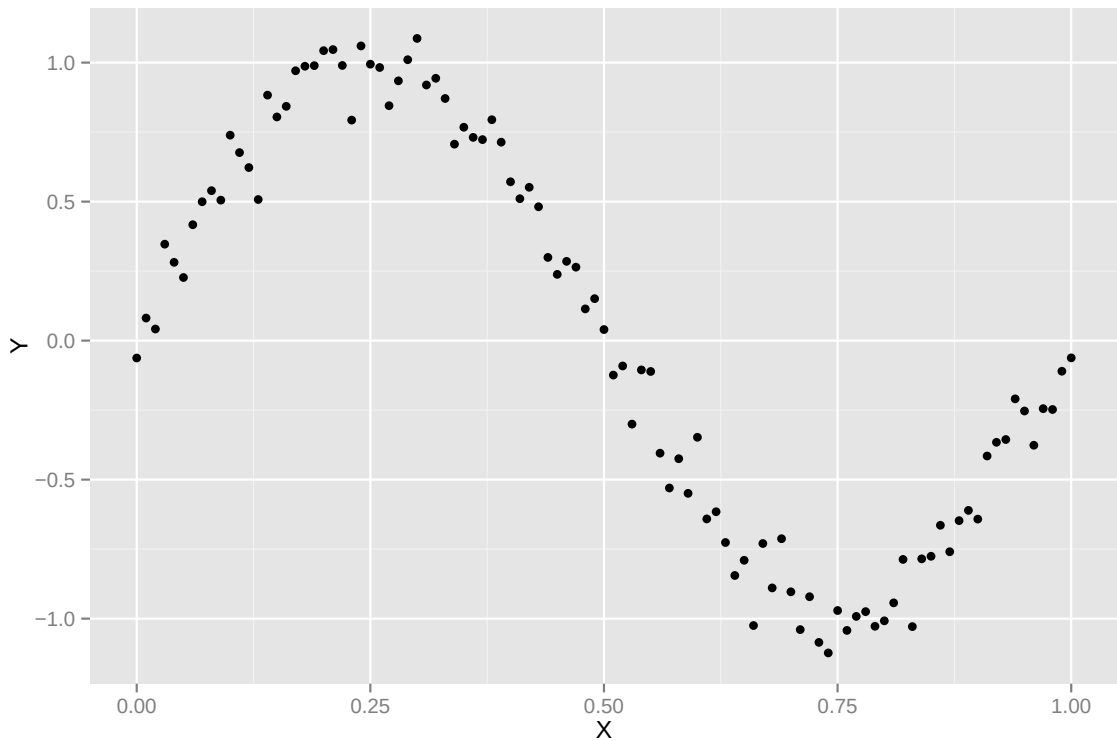


- Residuen sind scheinbar nicht normalverteilt!

Zum Umgang mit der Polynomiellen Regression

- Einfaches Beispiel: Simulierter Datensatz

```
set.seed(1) #Deterministische Wiederholung
x <- seq(0, 1, by = 0.01)
y <- sin(2 * pi * x) + rnorm (length(x),0, 0.1)
df <- data.frame(X = x, Y = y)
ggplot(df, aes(x = X, y = Y)) + geom_point()
```



Zum Umgang mit der Polynomiellen Regression

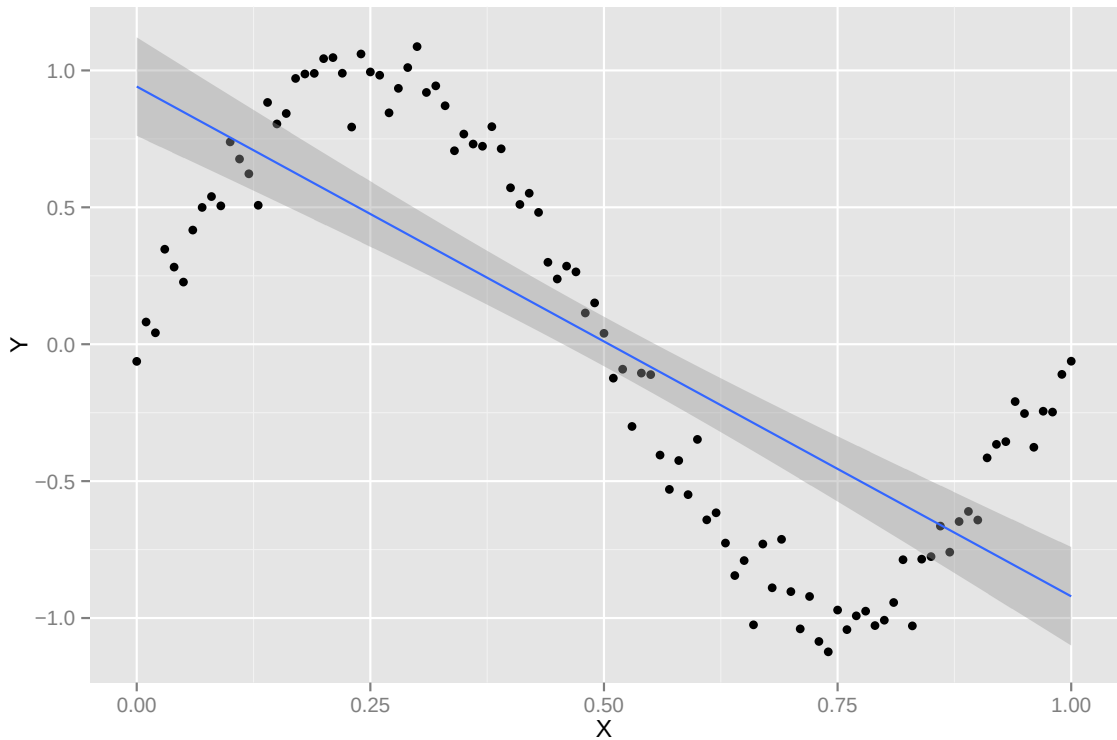
- Einfaches lineares Modell ist scheinbar gar nicht so schlecht (Fit von fast 60%)

```
> summary(lm(Y ~ X, data = df))

Call:
lm(formula = Y ~ X, data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-1.00376 -0.41253 -0.00409  0.40664  0.85874

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.94111    0.09057   10.39   <2e-16 ***
X           -1.86189    0.15648  -11.90   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4585 on 99 degrees of freedom
Multiple R-squared:  0.5885, Adjusted R-squared:  0.5843
F-statistic: 141.6 on 1 and 99 DF,  p-value: < 2.2e-16
```

Zum Umgang mit der Polynomiellen Regression

- Visualisierung schafft Klarheit

```
ggplot(df, aes(x = X, y = Y)) + geom_point()
    + geom_smooth(method = "lm", se = TRUE)
```



Zum Umgang mit der Polynomiellen Regression

- Idee: Fitten eines Polynoms 3. Grades

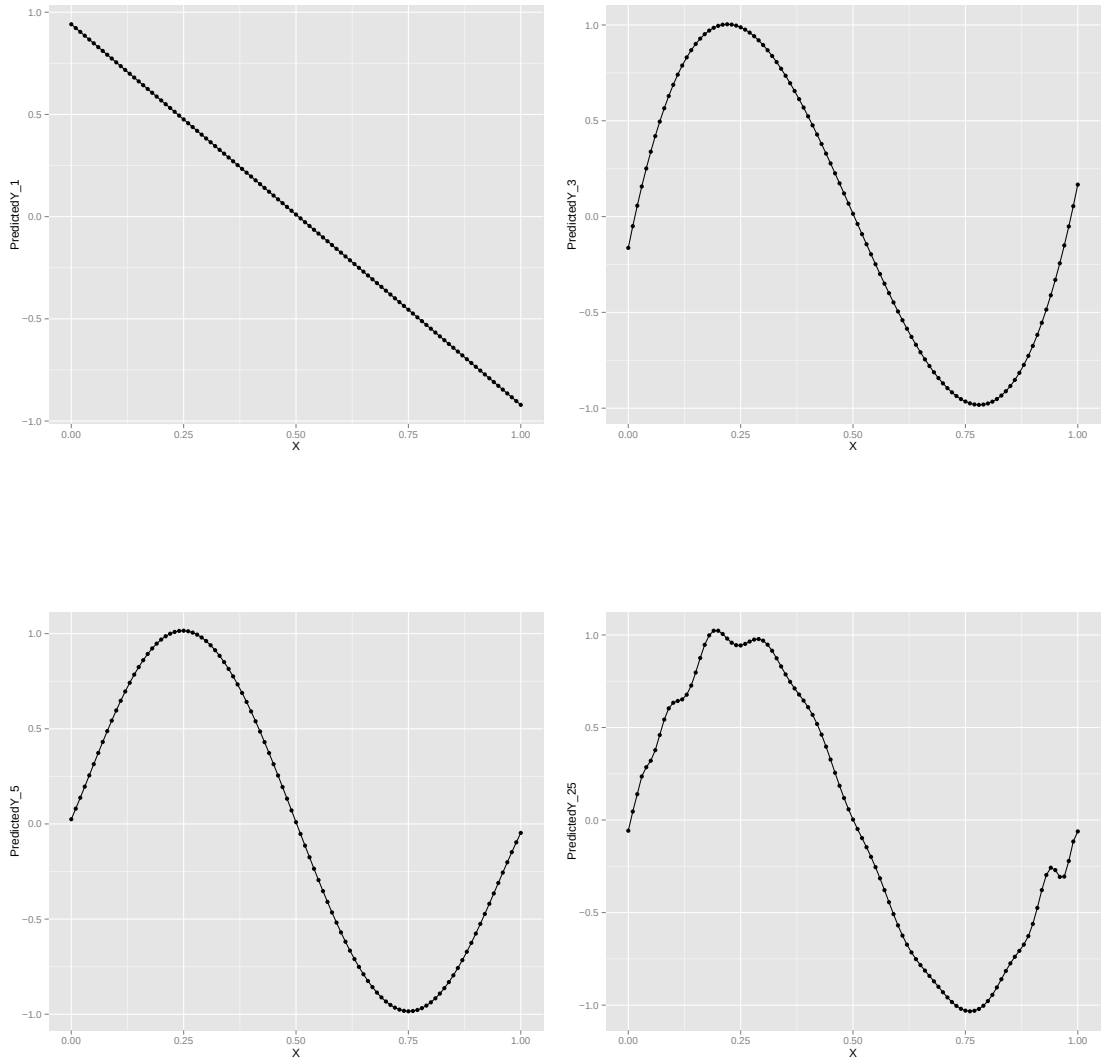
```
> summary(lm(Y ~ poly(X,3), data = df))

Call:
lm(formula = Y ~ poly(X, 3), data = df)

Residuals:
     Min       1Q   Median       3Q      Max
-0.32331 -0.08538  0.00652  0.08320  0.20239

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.01017    0.01148   0.886    0.378
poly(X, 3)1 -5.45536    0.11533 -47.302   <2e-16 ***
poly(X, 3)2 -0.03939    0.11533  -0.342    0.733
poly(X, 3)3  4.41805    0.11533  38.308   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1153 on 97 degrees of freedom
Multiple R-squared:  0.9745, Adjusted R-squared:  0.9737
F-statistic:  1235 on 3 and 97 DF,  p-value: < 2.2e-16
```

Zum Umgang mit der Polynomiellen Regression

- Je höher der Grad, desto besser der Fit!

Polynomgrad	Fit
1	0.5885
2	0.5885
3	0.9745
5	0.9844
10	0.9855
25	0.9876

Gefahr der Überanpassung an den Datensatz



Kreuzvalidierung

- Modell soll das Signal abbilden, nicht das zufällige Rauschen, um Vorhersagen zu ermöglichen
- Trick: Aufteilen des Datensatzes in Trainingsdaten und Testdaten!

```
set.seed(1)
x <- seq(0, 0.99, by = 0.01)
y <- sin(2 * pi * x) + rnorm (length(x),0, 0.1)

n <- length(x)
indices <- sort(sample(1:n, round(0.5 * n)))
training.x <- x [indices]
training.y <- y [indices]

test.x <- x [-indices]
test.y <- y [-indices]

training.df <- data.frame(X = training.x, Y = training.y)
test.df <- data.frame(X = test.x, Y = test.y)
```

Kreuzvalidierung

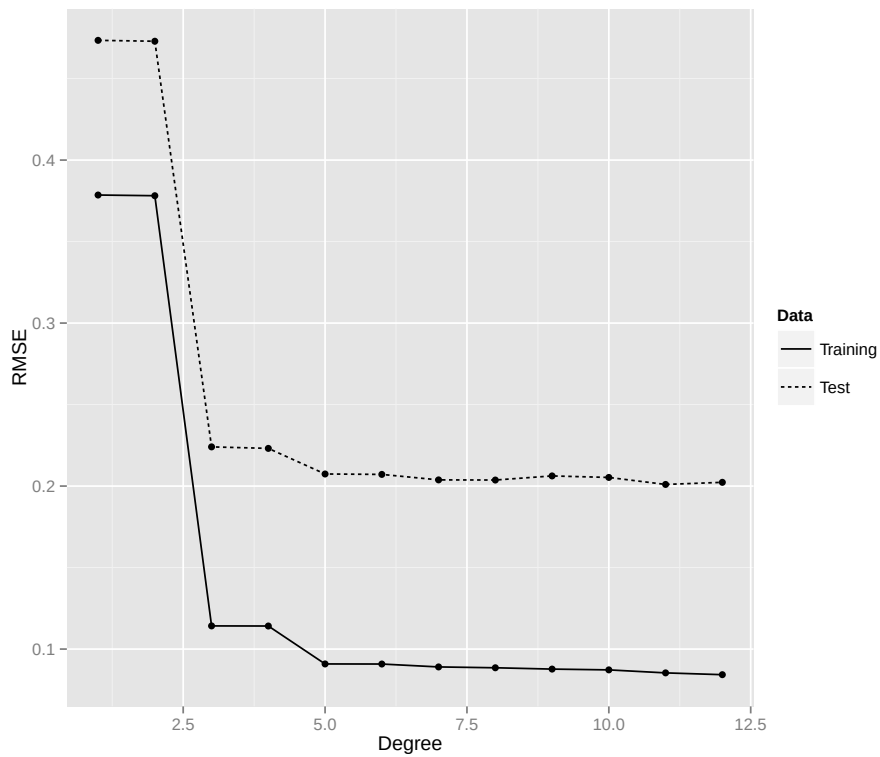
- Berechne Wurzel der mittleren quadratischen Abweichung der Test- bzw. Trainingsdaten
- Erstelle Übersicht für den Fit der Modell-Polynome für Grade 1 bis 12

```
rmse <- function(y,h) { return (sqrt(mean((y-h)^2)))}
performance <- data.frame()
for (d in 1:12) {
  poly.fit <- lm(Y ~ poly(X, degree = d), data = training.df)
  performance <- rbind(performance,
       data.frame(Degree=d,
                  Data = "Training",
                  RMSE = rmse(training.y,predict(poly.fit))))
  performance <- rbind(performance,
       data.frame(Degree=d,
                  Data = "Test",
                  RMSE = rmse(test.y, predict(poly.fit))))
}
```

Kreuzvalidierung

- Polynomgrad 1 und 2 liefern sehr schwache Modelle
- Polynomegrade ¿ 6 führen zu Überanpassung

```
ggplot(performance,aes(x = Degree,y = RMSE, linetype = Data))
   + geom_point() + geom_line()
```



Multiple lineare Regression

- Bisher basierte das Modell nur auf einer Variablen
- Was passiert, wenn mehrere unabhängige Variablen in Betracht kommen?

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
```

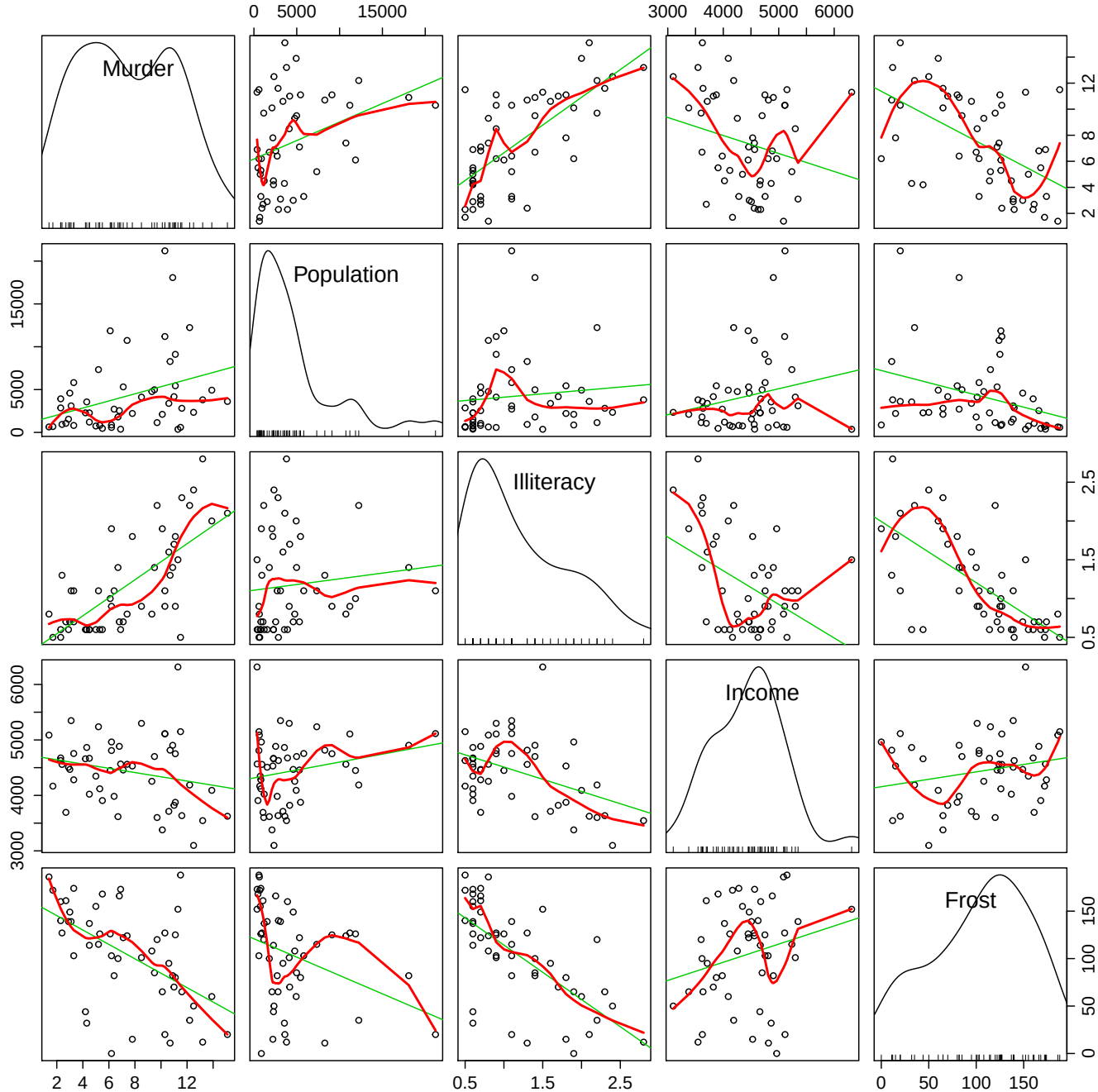
Korrelation der Faktoren

```
> 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
```

Beobachtungen

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

```
install.packages("car")
require(car)
scatterplotMatrix(states, spread = FALSE)
#spread=FALSE verhindert Plotten der Variance
```


Murder
Population
Illiteracy
Income
Frost
0 5000 15000
3000 4000 5000 6000
2 4 6 8 12
0 5000 15000
3000 4000 5000 6000
0.5 1.5 2.5
0 50 100 150

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

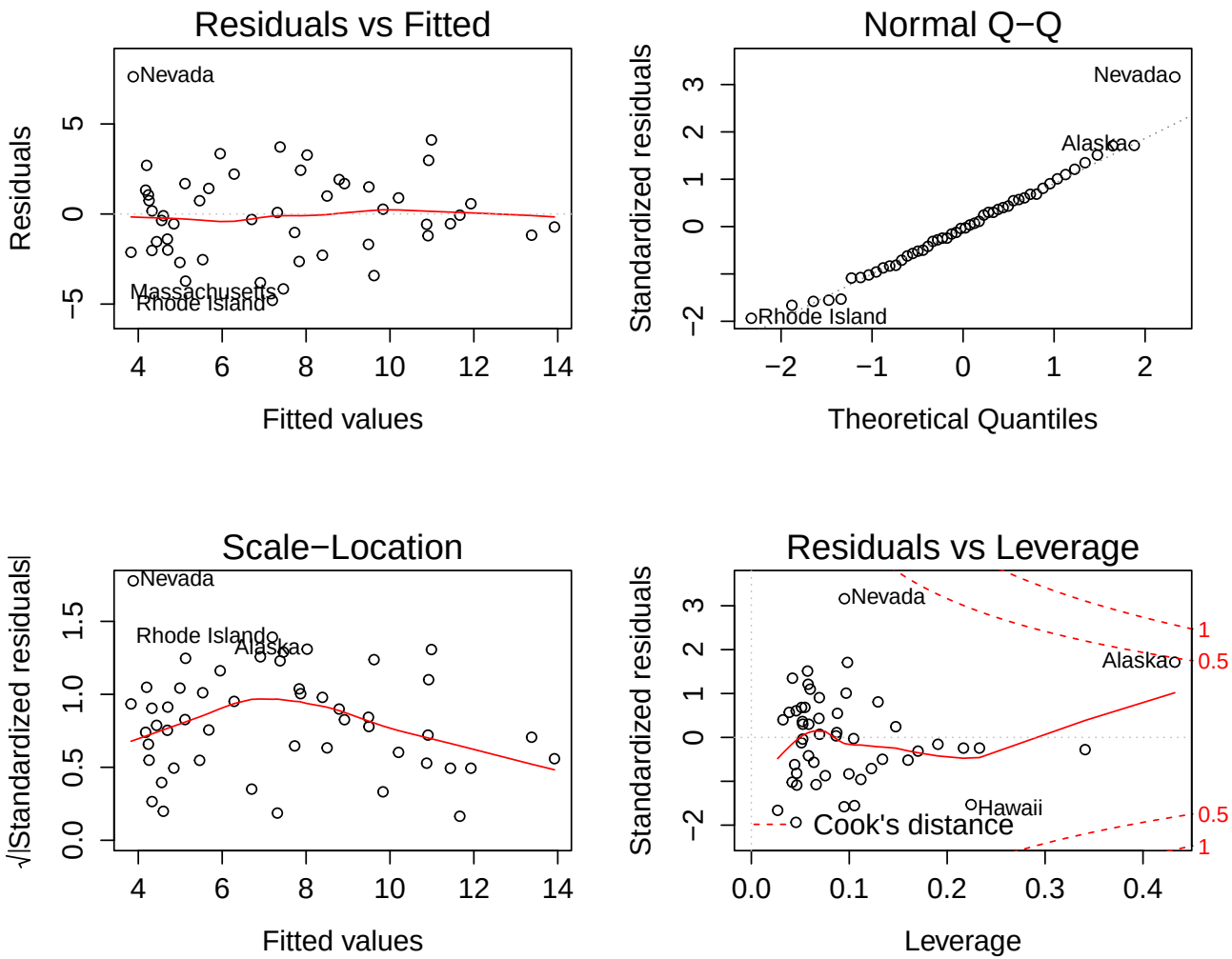
- Statistisch signifikant sind der Einfluss von Analphabetismus und Populationsgröße auf die Mordrate
- Die Mordrate ist nicht linear abhängig von Einkommen oder Temperatur
- Insgesamt erklären die Variablen etwa einen Anteil von 56%

Diagnostik

- Wie gut ist nun das Modell? ($\rightarrow$ siehe nächste Vorlesung)
- Gelten eigentlich die Vorraussetzungen?
  - Normalverteilte Residuen mit konstanter Varianz
  - Unabhängigkeit der unabhängigen Faktoren
  - Lineare Abhängigkeit der abhängigen Variablen von den einzelnen Faktoren

Diagnose-Plots

```
> fit <- lm (Murder ~ Population + Illiteracy + Income + Frost, data = states)
> par (mfrow=c(2,2))
> plot(fit)
```



Ausreißer - Outlier

- Nevada fällt etwas aus dem Rahmen
- Diskrepanz zwischen Modell und tatsächlichem Wert
- Nevada, Alaska, Hawaii haben einen hohen Einfluss auf die Regressionsparameter
- Entfernen dieser Datenpunkte?

```
> states["Nevada",]
       Murder Population Illiteracy Income Frost
Nevada   11.5        590        0.5   5149   188
> fitted(fit)["Nevada"]
  Nevada
3.878958
> residuals(fit)["Nevada"]
  Nevada
7.621042
```

Empfehlungssyteme

Kunden, die diesen Artikel gekauft haben, kauften auch...

- Bei einem sehr großen Angebot an Objekten muss einem Nutzer/Käufer Hilfestellung geben werden, um sich zurecht zu finden
- Einfachste Möglichkeit: Die allgemein beliebtesten Objekte angeben
- Empfehlungssyteme (*Recommendation systems*) sind ein popärer Zweig des Machine Learning
- Anwendung in Verkaufsplattformen, Sozialen Netzen, Musikbibliotheken, etc.
- Beispiel: Film-Empfehlungen basierend auf Bewertungen
- Quelle: `https://docs.yhathq.com/r/examples/movie-recommender`

Film-Empfehlungssystem: Vorbereitung

- Installieren des plyr-Packets zum Splitten und Rekombinieren von Datensätzen

```
install.packages("plyr")
library(plyr)
```

- Data laden von `http://www.grouplens.org/node/73`
- `movielens.txt` beinhaltet die Bewertungen für MovieIDs, `movies.txt` bildet die Id auf den Filmnamen ab

```
data <- read.table("./data/movielens.txt", header=TRUE,
                   stringsAsFactors=FALSE)
data <- data[order(data$user_id, data$movie_id),]

movies <- read.csv("./data/movies.txt", header=TRUE,
                   stringsAsFactors=FALSE, sep="|")
```

Usern, die diesen Film mochte, gefällt auch...

- Suche nach gemeinsamen Reviewern für zwei Filme

```
find_common_reviewers <- function(movieA, movieB) {
  movieAViewers <- subset(data, movie_id==movieA)$user_id
  movieBViewers <- subset(data, movie_id==movieB)$user_id
  commonViewers <- intersect(movieAViewers, movieBViewers)
  commonViewers
}
```

- Korrelation des Ratings

```
get_review_correlation <- function(movieA, movieB) {
  common <- find_common_reviewers(movieA, movieB)
  cor(subset(data, user_id %in% common & movie_id==movieA)$rating,
      subset(data, user_id %in% common & movie_id==movieB)$rating)
}
```

Verkleinern des Datensatzes

- Suchen nach Filmen mit mehr als 300 Bewertungen

```
movies.count <- ddply(data, .(movie_id), nrow)
movies.count <- movies.count[movies.count$V1 > 300,]
```

- Reduzieren des Datensatzes auf diese Einträge

```
data <- subset(data, movie_id %in% movies.count$movie_id)
movies <- subset(movies, id %in% unique(data$movie_id))
```

Ermitteln der Empfehlungen

- Berechnen der Korrelation zwischen der Anfrage und allen anderen Filmen

```
rec_movies <- function(movieTitle) {
  id <- subset(movies, title==movieTitle)$id
  # compute the correlation between the movie a user specified
  # and each title
  ddply(movies, .(id, title), function(movie) {
    c("similarity"=get_review_correlation(id, movie$id))
  })
}
```

Testabfrage

- Höchste Korrelation mit dem Eintrag selbst
- Ausgabe nach Movie ID sortiert

```
> rec_movies("Toy Story (1995)")
   id                                    title  similarity
1   1                        Toy Story (1995) 1.000000000
2   2                          Jumanji (1995) 0.187466745
3   3                 Grumpier Old Men (1995) 0.160649192
4   6                             Heat (1995) 0.051096916
5   7                          Sabrina (1995) 0.149535844
6  10                        GoldenEye (1995) 0.143598390
7  11         American President, The (1995) 0.178131972
8  16                           Casino (1995) 0.043230419
9  17           Sense and Sensibility (1995) 0.243104673
10 19 Ace Ventura: When Nature Calls (1995) 0.014298850
11 21                       Get Shorty (1995) 0.125379159
12 22                          Copycat (1995) 0.200846065
13 24                           Powder (1995) 0.122250972
```

Weitere Schritte

- Sortieren nach `similarity`
- Entfernen des Anfrageparameters aus dem Ergebnis

Hinweise zum Verfahren

- Qualitativ hochwertige Empfehlungen
- Algorithmus ist vielseitig anwendbar
- Neue Einträge können direkt berücksichtigt werden
- Aber: Algorithmus ist sehr langsam, da für jeden Film der gesamte Datenbestand verwendet wird, um die Korrelationsmatrix zu berechnen
- Algorithmus funktioniert nur, wenn der Datensatz partitioniert ist und User nur Bewertungen für nicht überlappende Untergruppen abgegeben haben