Universität Leipzig
Institut für Meteorologie

Seminararbeit im Modul „TM4 Nichtlineare Statistik“

Kreuzvalidierung

Felix Dietzsch

1473979

5. März 2012

eingereicht bei Prof. Dr. Uwe Schlink,
Umweltforschungszentrum Leipzig

Inhaltsverzeichnis

1 Einleitung

Ein wichtiger und häufig angewendeter Bereich der Statistik ist die Qualitätsanalyse von Regressionsmodellen. Um das in dieser Arbeit beschriebene Kreuzvalidierungsverfahren einzuordnen, wird zunächst ein Überblick über mehrere verschiedene Verifikations- bzw. Testmethoden gegeben. Dieser erhebt aber nicht den Anspruch auf Vollständigkeit.

Es existiert eine Reihe von Verfahren zum Testen der Güte statistischer Modelle. Die meisten solcher Verfahren liefern Kennzahlen, deren Quantität eine Aussage über die Modellgüte treffen. Ein Beispiel hierfür ist das Bestimmtheitsmaß R^2. Dieses Maß erklärt den Anteil einer Variablen Y an der durch ein statistisches Modell erklärten Varianz (Yule 1897). Andere Gütetests beziehen sich auf die Beurteilung binärer Klassifikatoren. Als Beispiel sei die Verifikation numerischer Wettervorhersagemodelle genannt. An ihnen wird u.a. untersucht, wie genau ein voprhergesagter Parameter mit tatsächlich eingetretenen Ereignissen übereinstimmt (z.B. Niederschlag ja/nein). Es werden Häufigkeitszahlen bestimmt, die angeben, wie oft eine Vorhersage bei eingetretenen Ereignissen korrekt war und wie oft inkorrekt. Äquivalent dazu das ganze für Nicht-Ereignisse. Aus den erhaltenen vier Häufigkeiten lassen sich kategorische Gütemaße definieren. Als Beispiele seien die Kennzahlen *Probability Of Detection*, *False Alarm Rate* und *True Skill Statistics* genannt (DWD 2006a). Kontinuierliche Gütemaße beziehen sich dagegen auf kontinuierliche Parameter wie z.B. die Temperatur. Hier finden übliche statistische Maße wie *mittlerer Fehler*, *mittlerer absoluter Fehler*, *RMSE*[1] und die *Standardabweichung* Anwendung (DWD 2006b).

Die Methodik, von der einige Verfahren erwähnt wurden, bezieht sich u.a. immer auf die Analyse eines kompletten Datensatzes, auf dem ein statistisches Modell beruht. Einen anderen Ansatz verfolgt dagegen die Kreuzvalidierung. Ein Datensatz wird in mehrere Teile aufgeteilt. Auf Basis eines Teils des Datensatzes wird ein statistisches Modell abgeleitet. Das Modell wird auf den Rest des Datensatzes angewendet und die Abwei-

[1] Root Mean Square Error; Wurzel des mittleren quadratischen Fehlers

chung zu den tatsächlichen Werten untersucht. Das Verfahren dient also in erster Linie der Überprüfung der *Prognosegüte* eines Modells.

2 Theoretische Grundlagen der Kreuzvalidierung

2.1 Einfache Kreuzvalidierung

Bei der einfachen Kreuzvalidierung wird ein Datensatz in K Teildatensätze unterteilt. Ein Datensatz wird als Testdatensatz ausgewählt. An die restlichen $K - 1$ Datensätze wird ein Regressionsmodell angepasst. Die Auswahl dieser Datensätze erfolgt nach dem Zufallsprinzip. In der Praxis verwendet man in der Regel einen softwareseitigen Zufallsgenerator.

Mit Hilfe des angepassten Modells wird eine Vorhersage für die Werte des k-ten Datensatzes durchgeführt und der mittlere quadratische Fehler bestimmt. Enthält jeder Teildatensatz N_k Daten, so ergibt sich der mittlere quadratische Fehler η_k für den Datensatz k zu

$$\eta_k = \frac{1}{N} \sum_{i=1}^{N} \left(\widetilde{y}_{k,i} - y_{k,i} \right)^2 \tag{2.1}$$

mit den vorhergesagten Daten $\widetilde{y}_{k,i}$ und den tatsächlichen Daten $y_{k,i}$ des k-ten Datensatzes. Führt man das Verfahren für alle k Teildatensätze durch, so erhält man k Testdatensätze sowie k verschiedene Regressionsgleichungen, die getestet werden. Die jeweiligen Fehler werden noch einmal gemittelt und man erhält zusammenfassend den mittleren quadratischen Fehler

$$\bar{\eta} = \frac{1}{K} \sum_{k=1}^{K} \eta_k = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{N_k} \sum_{i=1}^{N} \left(\widetilde{y}_{k,i} - y_{k,i} \right)^2 \tag{2.2}$$

(Wilks 2006). Man spricht bei der einfachen Kreuzvalidierung auch von der K-fachen Kreuzvalidierung oder von der K-fach gefalteten Kreuzvalidierung in Abhängigkeit der gewählten Anzahl der Unterteilungen eines Datensatzes in K Teildatensätze.

2.2 Leave-One-Out

Das Leave-one-out-Verfahren ist ein Spezialfall der einfachen Kreuzvalidierung. Hierbei wird jeder einzelne Datensatz einmal als Testdatensatz benutzt und der restliche Datensatz als Trainingsdatensatz. Das Verfahren wird für jeden einzelnen Datensatz wiederholt. Vorteile der Methode sind die optimale Ausnutzung der Daten sowie die Vermeidung einer zufälligen Stichprobenauswahl. Nachteilig sind der sehr hohe Rechenaufwand bei entsprechend großen Datensätzen sowie die nicht vorhandene Möglichkeit der vorherigen Stratifikation (vgl. Kapitel 2.3) der Daten (Witten und Frank 2005).

2.3 Stratifizierte Kreuzvalidierung

Bei der stratifizierten Kreuzvalidierung werden Teildatensätze benutzt, die annähernd gleich verteilt sind. Dadurch erfolgt eine Anpassung des Modells über die gesamte Spannbreite der Daten und die Varianz der Fehler verringert sich. Ein in der Praxis gängiges Verfahren ist die *zehnfache stratifizierte Kreuzvalidierung* (Spiliopoulou 2009).

2.4 Bootstrap

Das Bootstrap-Verfahren ist ein statistisches Resampling-Verfahren. Dies bedeutet, dass statistische Eigenschaften eines Datensatzes auf Basis einer oder mehrerer Stichproben abgeleitet werden, und somit Aussagen über Verteilungseigenschaften getroffen werden können. Eine Anwendung im Bereich der Kreuzvalidierung ist das 0.632-Bootstrapping.

Aus einem vorhandenen Datensatz des Umfangs N werden ebensoviele Stichproben mit Zurücklegen gezogen. Aus diesem Datensatz wird die *Menge der Trainingsdaten* gebildet. Alle Daten des Ursprungsdatensatzes, die nicht im Trainingsdatensatz enthalten sind, bilden die *Testmenge*. Grund für dieses Vorgehen ist die sonst zu optimistische Schätzung des Vorhersagefehlers, da identische Daten sowohl in Trainings- als

auch Testmenge enthalten wären. Die Wahrscheinlichkeit, dass sich ein Datensatz in der Trainingsmenge T befindet, beträgt

$$\mathcal{P}\left\{y_i \in T\right\} = \lim_{N\to\infty} 1 - \left(1 - \frac{1}{N}\right)^N = 1 - \frac{1}{\mathrm{e}} \approx 0.632 \tag{2.3}$$

(Heumann und Schmid 2010). Etwa 63.2 % der Daten werden demnach zum Trainieren des Modells verwendet, während die restlichen 36.8 % zum Testen verwendet werden. Der Anteil der Trainingsdaten am Gesamtdatensatz ist recht gering. Dementsprechend pessimistisch wird der Fehler für die Testdaten geschätzt. Um dies zu vermeiden, wird der Testdatenfehler mit dem Resubstitutionsfehler der Trainingsinstanzen gewichtet nach Anteil verrechnet (Witten und Frank 2005):

$$\eta = 0.632\,\tilde{\eta} + 0.368\,\hat{\eta}, \tag{2.4}$$

wobei $\tilde{\eta}$ den Fehler der Testmenge und $\hat{\eta}$ den Fehler der Trainingsmenge bezeichnet.

3 Anwendung in R

3.1 Möglichkeiten der Durchführung einer Kreuzvalidierung

Routinen zur Durchführung einer Kreuzvalidierung werden in R durch die Pakete DAAG, `bootstrap` und `boot` bereitgestellt. DAAG bezieht sich hierbei speziell auf die Kreuzvalidierung linearer Regressionsmodelle, während mit Hilfe der Pakete `boot` und `bootstrap` die Kreuzvalidierung allgemeiner linearer Modelle möglich ist (Wollschläger 2010; Maindonald und Braun 2012; Tibshirani 2009; Canty und Ripley 2012).

3.1.1 Das DAAG-Paket

Das R-Paket DAAG bietet eine Funktion `cv.lm()` zur Durchführung einer Kreuzvalidierung für ein lineares Regressionsmodell. Die Syntax lautet

```
> cv.lm(df = <dataframe>, form.lm = <formula>, m=<number>,
   dots = <boolean>, seed = <number>, plotit = <boolean>,
   printit = <boolean>)
```

Hierbei bezeichnet `df` einen Dataframe, in dem die erste Spalte die Response-Variable und die zweite Spalte den Prädiktor enthält. `form.lm` erwartet ein Objekt des Typs *formula*. Ein solches Objekt erhält man zum Beispiel durch Anwendung der linearen Regression `lm()` auf einen Datensatz. Mit `m` wird die Anzahl der Faltungen angegeben. Die restlichen Parameter spielen eine untergeordnete Rolle und dienen u.a. der Darstellung eines Plots. Die Zuordnung der Daten zu Test- und Trainingsdatensatz erfolgt mittels eines einfachen Zufallsgenerators (Maindonald und Braun 2012). Das resultierende Objekt liefert die Parameter `ss` und `df`. `ss` liefert die Summe der quadratischen Residuen. `df` gibt die Anzahl der Freiheitsgrade an. Man erhält also den mittleren quadratischen Fehler aus

$$\eta = \frac{\texttt{ss}}{\texttt{df}}. \tag{3.1}$$

Die Funktion `cv.binary` liefert eine Kreuzvalidierung für Datensätze mit einer binären Response-Variable. Die Syntax lautet

```
> cv.binary(obj, rand=NULL, nfolds=<number>, print.details=
  TRUE)
```

`obj` verlangt hier ein generalisiertes lineares Modell (`glm`). `rand` ist ein Vektor, der jede Beobachtung einer Faltung zuordnet, wobei `nfolds` die Anzahl der Faltungen angibt. Mit `print.details` wird die Ausführlichkeit der Ausgabe festgelegt.

Die Funktion `CVlm()` ist syntaktisch identisch zu `cv.lm()`, wird aber zur Validierung von multiplen linearen Regressionsmodellen verwendet.

3.1.2 Das bootstrap-Paket

Flexibler als die Funktionen des DAAG-Paketes ist die Funktion `crossval()` des `boot`-Paketes. Eine detaillierte Dokumentation findet sich in Efron und Tibshirani 1994. Die die wichtigsten Parameter beinhaltende Syntax lautet:

```
> crossval(x, y, theta.fit, theta.predict, ..., ngroup=n)
```

Hierbei bezeichnet `x` eine Matrix mit den Prädiktoren x_n im Falle multipler Regression. Für den Spezialfall der einfachen linearen Regression reduziert `x` sich auf einen Vektor. `y` bezeichnet einen Vektor mit den jeweiligen Response-Werten. `theta.fit` ist eine Funktion, die kreuzvalidiert werden soll. Ein Beispiel hierfür findet sich in (Canty und Ripley 2012). `theta.predict` ist eine Funktion, die Response-Werte für `theta.fit` liefert. Die Anzahl der Faltungen wird mit `ngroup` festgelegt.

3.1.3 Das boot-Paket

Eine weitere Möglichkeit, die Kreuzvalidierung generalisierter linearer Regressionsmodelle durchzuführen, wird durch das Paket `boot` in Form der Funktion `cv.glm()` bereitgestellt. Die entsprechende Syntax lautet hier:

```
> cv.glm(data=<dataset>, glmfit=<glm-model>, K=<number>)
```

`dataset` ist ein Datensatz in Form einer Matrix oder eines Dataframes, der die Daten für ein generalisiertes lineares Modell enthält. Das Modell selber wird dem Parameter `glmfit` übergeben. Mit K wird die Anzahl der Faltungen des Datensatzes angegeben (Wollschläger 2010). Detaillierte Angaben und ein Beispiel finden sich in Canty und Ripley 2012.

3.2 Auswertung eines Beispieldatensatzes

Für eine beispielhafte Durchführung einer Kreuzvalidierung steht ein Datensatz des Leipziger Umeltforschungszentrums zur Verfügung. Dieser enthält die Temperaturen sowie Feuchte an zehn verschiedenen über das Leipziger Stadtgebiet verteilten Stationen. Als Referenz stehen außerdem die Daten des Leipziger Instituts für Meteorologie (LIM) zur Verfügung.

3.2.1 Einfache lineare Regression

Es soll ein einfaches lineares Regressionsmodell validiert werden, das die Temperaturzeitreihe von LIM und Leonhard-Frank-Straße miteinander in Beziehung setzt. Zunächst erfolgt das Einlesen der Daten:

```
1 >  mydata.df = read.csv2("klimadaten-leipzig-sommer2010.csv",
     header = TRUE, sep = ";", dec = ".")
```

Die Spalten `T1` und `T2m` enthalten die gesuchten Daten. Diese werden in einen Prädiktor- und einen Response-Vektor überführt:

```
2 >  x = mydata.df$T2m # Prädiktor-Variable
3 >  Y = mydata.df$T1 # Response-Variable
```

Es soll also mit Hilfe des linearen Regressionsmodells eine Vorhersage der Temperatur in der Leonhard-Frank-Straße auf Grundlage der Temperatur am LIM getroffen werden. Die beiden Temperaturreihen werden im nächsten Schritt in einen Dataframe überführt:

```
4 >  mydata = data.frame(x=x, Y=Y)
```

In dem resultierenden Dataframe `mydata` müssen nun die ungültigen Datensätze eliminiert werden. Dies geschieht mittels des Befehls

```
5 >  mydata = na.omit(mydata)
```

Jetzt kann mit Hilfe des Datensatzes ein einfaches lineares Regressionsmodell angepasst werden:

```
6 >  fit = lm(Y~x, data = mydata)
```

Als Resultat erhält man die Regressionskoeffizienten

$$\text{slope} = 1.046 \qquad \text{intercept} = -1.125, \tag{3.2}$$

das heißt es ist

$$T_1 = 1.046 \cdot T_{2\text{m}} - 1.125. \tag{3.3}$$

Für dieses Modell kann nun die Kreuzvalidierung durchgeführt werden. Dazu wird hier das Paket `DAAG` verwendet.

```
7 >  kv = cv.lm(df = mydata, fit, m = 10)
```

Der Funktion wird der zuvor definierte Datensatz `mydata` übergeben, zusammen mit dem darauf angepassten linearen Modell `fit` aus Schritt 6. Die Kreuzvalidierung soll zehnfach gefaltet durchgeführt werden. Das heißt, dass der Datensatz in zehn gleiche zufällig gewählte Teile unterteilt wird, von denen ein Teil den Testdatensatz darstellt. Die Ausgabe von `kv` enthält die berechneten Residuen für jeden einzelnen Durchlauf. Als Beispiel sei ein Ausschnitt gezeigt[1]:

Codestück 3.1: Ausschnitt der Ausgabe während der Ausführung der Kreuzvalidierung.

```
            X5954 X5955 X5956 X5957 X5958 X5959 X5960 X5961
x          11.000 10.50 10.30 10.20 10.10  9.90  9.80  9.70
Predicted 10.387  9.86  9.65  9.55  9.45  9.24  9.13  9.03
Y          11.100 11.20 11.10 11.00 11.00 10.80 10.50 10.40
Residual    0.713  1.34  1.45  1.45  1.55  1.56  1.37  1.37
```

[1] Voraussetzung: Der Parameter `printit` ist auf `TRUE` gesetzt, dies ist auch der Default-Wert und deswegen hier nicht angegeben

Der anschließende Aufruf des Objektes `kv` liefert die Parameter `ss` und `df` für die Summe der Fehlerquadrate und die Anzahl der Freiheitsgrade. Das hier gezeigte Beispiel liefert das folgende Resultat:

```
> kv
    ss     df
131277  60072
```

Den resultierenden mittleren quadratischen Fehler erhält man dann mit `ss/df` (vgl. Gleichung 3.1):

```
> kv[1]/kv[2]
  ss
2.19
```

Das Ergebnis mit dem Wert 2.19 ist als relatives Maß zu verstehen. Zum Beispiel wäre es jetzt möglich, ein anderes Regressionsmodell anzupassen, dafür die Kreuzvalidierung durchzuführen und die Werte zu vergleichen. Eine Interpretation des jetzt alleine stehenden Wertes ist nicht möglich.

Problematisch bei dieser Betrachtung ist die Tatsache, dass die Temperaturzeitreihen autokorreliert sind. Autokorrelation eines Datensatzes führt zu einer Verzerrung der statistischen Schätzer wie Standardabweichung und Mittelwert sowie zu autokorrelierten Residuen (Stein, Pavetić und Noack 2009). Der Datensatz liegt in minütlicher Auflösung vor und erstreckt sich über einen Zeitraum von etwa vier Wochen. Bei der Bildung von Tagesmitteln wäre die Zeitreihe noch immer, wenn auch nicht im selben Maße, autokorreliert. Die Bildung von Wochenmitteln lohnt sich nicht, da dafür die Zahl der Datensätze zu klein würde. Deswegen wird hier die Verzerrung durch Autokorrelation in Kauf genommen und auf die Bildung von Mittelwerten verzichtet.

3.2.2 Multiple lineare Regression

Um die Möglichkeiten von R noch tiefergehend zu nutzen, soll in diesem Schritt für die Kreuzvalidierung ein multiples lineares Regressionsmodell unter Einbezug der LIM-Daten von Temperatur, Feuchte, Global- und Himmelsstrahlung, Druck sowie Windrichtung und -stärke zur Anpassung an die Temperatur in der Leonhard-Frank-Straße verwendet werden. Der entsprechende Quelltext in R dafür ist ähnlich zu dem in Kapitel 3.2.1.

Codestück 3.2: Kreuzvalidierung eines multiplen linearen Regressionsmodells in R.

```
1  > library(boot)
2  > mydata.df = read.csv2("klimadaten-leipzig-sommer2010.csv",
       header = TRUE, sep = ";", dec = ".")
3  > Y = mydata.df$T1
4  > temp = mydata.df$T2m
5  > wind = mydata.df$Windgeschw
6  > richtung = mydata.df$Windrichtg
7  > globalstr = mydata.df$Globalstr
8  > himmelsstr = mydata.df$Himmelsstr
9  > druck = mydata.df$Luftdr
10 > feuchte = mydata.df$RH
11 > mydata = data.frame(Y, temp, wind, richtung, globalstr,
       himmelsstr, druck, feuchte)
12 > mydata = na.omit(mydata)
13 > modell = glm(Y ~ temp + feuchte + wind + richtung +
       globalstr + himmelsstr + druck, data = mydata)
14 > kv = cv.glm(mydata, glmfit = modell, K = 10)
15 > kv$delta
```

Das multiple lineare Regressionsmodell wird in Zeile 13 mit Hilfe der Funktion `glm()` definiert. Die Ausgabe des Objektes `modell` liefert die resultierenden Regressionskoeffizienten:

```
> modell

Call:  glm(formula = Y ~ temp + feuchte + wind + richtung +
    globalstr + himmelsstr + druck, data = mydata)

Coefficients:
(Intercept)         temp      feuchte         wind
 19.7764260    1.0250430   -0.0104249    0.1817297
   richtung    globalstr   himmelsstr        druck
  0.0020554   -0.0002669   -0.0002901   -0.0205358
```

Für die Kreuzvalidierung wurde die Funktion `cv.glm()` aus dem `boot`-Paket in Anlehnung an Wollschläger 2010 verwendet. Die Funktion `cv.glm()` liefert eine Liste

zurück. Diese wurde hier analog zu Kapitel 3.2.1 in der Variable `kv` gespeichert. Die Komponente `delta` (vgl. Zeile 15) enthält die Fehlerkoeffizienten der Kreuzvalidierung. Für diesen Fall liefert die Ausgabe folgende Werte:

```
> kv$delta
[1] 2.088745 2.088696
```

Der erste Wert enthält den mittleren quadratischen Fehler der Kreuzvalidierung. Der zweite Wert berücksichtigt eine durch Wahl von K entstehende Verzerrung (Wollschläger 2010).

Zunächst sei auf die in Kapitel 3.2.1 erwähnte Verfälschung der Ergebnisse durch Autokorrelation hingewiesen, die auch hier einen Einfluss hat. Des Weiteren hat die hier getroffene Wahl von K (hier: 10) offenbar nur geringen Einfluss auf den Fehlerkoeffizienten. Die Berücksichtigung der Verzerrung bewirkt einen relativen Unterschied von weniger als 1 ‰. Der Vergleich der Fehlerkoeffizienten der einfachen und multiplen linearen Regression zeigt eine offenbar leicht bessere Vorhersage des multiplen linearen Modells. Allerdings ist der relative Unterschied mit etwa 5% relativ gering. Außerdem ist zu diskutieren, welchen Einfluss die Unterschiede der Algorithmen der verschiedenen hier verwendeten Kreuzvalidierungsfunktionen haben. Eine Auflösung ist in diesem Rahmen nicht möglich, da eine vollständige Dokumentation der entsprechenden Bibliotheken nicht vorliegt.

4 Zusammenfassung

Kreuzvalidierung ist ein Verfahren zur Überprüfung der Prognosegüte von Regressionsmodellen. Grundprinzip ist die Aufspaltung eines zu untersuchenden bzw. anzupassenden Datensatzes in einen Trainings- und einen Testdatensatz. Das Modell wird an den Trainingsdatensatz angepasst und führt eine Vorhersage mit Hilfe der Prädiktoren des Testdatensatzes durch. Die Abweichung zwischen Vorhersage- und tatsächlichem Wert wird in ein Fehlermaß überführt, das eine Aussage über die Prognosegüte des Modells trifft. Je nach Anzahl der Unterteilungen und Wiederholungen des Verfahrens bezeichnet man dieses auch als *mehrfache Kreuzvalidierung*. Ein weiterer Spezialfall ist die *stratifizierte Kreuzvalidierung*, bei der auf die Gleichverteilung der ausgewählten Datensätze geachtet wird. Eine weitere Alternative der Wahl der entsprechenden Datensätze stellt das *Bootstrapping* dar, bei dem aus dem Ausgangsdatensatz einzelne Datensätze nach dem Prinzip „Ziehen mit Zurücklegen“ ausgewählt werden, die dann den Trainingsdatensatz bilden. Insbesondere kommt hier das *0.632-Bootstrapping* zum Tragen.

Die Statistiksoftware R bietet die Möglichkeit der Durchführung einer Kreuzvalidierung sowohl für einfache als auch multiple lineare Regressionsmodelle. Entsprechende Funktionen werden durch die Pakete `boot`, `bootstrap` sowie DAAG bereitgestellt. Gezeigt wird die Kreuzvalidierung eines einfachen und eines multiplen linearen Regressionsmodells am Beispiel eines Datensatzes, der verschiedene Temperaturmessungen im Leipziger Stadtgebiet sowie die Messungen sämtlicher gängiger meteorologischer Parameter am Leipziger Institut für Meteorologie enthält. Die Ergebnisse werden erläutert und auch auf verschiedene zu berücksichtigende Probleme wird eingegangen. Insbesondere führt die Autokorrelation der Werte des verwendeten Datensatzes zu einer begrenzten Verwertbarkeit der Ergebnisse.

Literatur

Canty, A. und B. Ripley (2012). *Package 'boot'*. http://cran.r-project.org/web/packages/boot/boot.pdf.

DWD (2006a). *Kategorische Gütemaße*. http://www.dwd.de/bvbw/generator/DWDWWW/Content/Oeffentlichkeit/FE/FE1/Interpretation/KategoriScores,templateId=raw,property=publicationFile.pdf/KategoriScores.pdf.

– (2006b). *Kontinuierliche Gütemaße*. http://www.dwd.de/bvbw/generator/DWDWWW/Content/Oeffentlichkeit/FE/FE1/Interpretation/KontiScores,templateId=raw,property=publicationFile.pdf/KontiScores.pdf.

Efron, B. und R.J. Tibshirani (1994). *An Introduction to the Bootstrap*. 1. Aufl. Chapman und Hall/CRC.

Heumann, C. und V. Schmid (2010). *Schätzen und Testen II*. http://www.statistik.lmu.de/~semwiso/schaetzentesten2-ss10/skript/ST2-ss10-skript.pdf, S. 23–27.

Maindonald, J. und W.J. Braun (2012). *Data Analysis And Graphics data and functions*. http://cran.r-project.org/web/packages/DAAG/DAAG.pdf.

Spiliopoulou, M. (2009). *Evaluationsmethoden*. http://omen.cs.uni-magdeburg.de/itikmd/cms/upload/datamining/DM_2_2.pdf.

Stein, P., M. Pavetić und M. Noack (2009). *Multivariate Analyseverfahren*. http://www.uni-due.de/imperia/md/content/soziologie/stein/multivariate.pdf. Universität Duisburg-Essen.

Tibshirani, R.J. (2009). *Package 'bootstrap'*. http://cran.r-project.org/web/packages/bootstrap/bootstrap.pdf.

Wilks, D.S. (2006). „Cross Validation“. In: *Statistical methods in the atmospheric sciences*. Academic Press, S. 215–217.

Witten, I.H. und E. Frank (2005). *Data Mining*. Hrsg. von N. Fuhr. http://www.is.informatik.uni-duisburg.de/courses/dm_ws05/folien/Kapitel5.ppt.

Wollschläger, D. (2010). „Kreuzvalidierung“. In: *Grundlagen der Datenanalyse in R*. www.uni-kiel.de/psychologie/dwoll/r/doc/06_05_regrCV.pdf. Springer, S. 179–181.

Yule, G.U. (1897). „On the theory of correlation“. In: *Journal of the Royal Statistical Society* 62, S. 812–854.