# LAB Prostate LinReg 3.R Focus on the topic estimation of the general- ization error.

Behrooz Filzadeh

2025-05-14

#Einleitung Dieses Dokument analysiert das R-Skript LAB Prostate LinReg 3.R. Aufbauend auf der Modellauswahl aus dem vorherigen Skript (2.R), liegt der zentrale Fokus hier auf der Schätzung des Generalisierungsfehlers. Der Generalisierungsfehler gibt an, wie gut unser trainiertes Modell voraussichtlich auf neuen, ungesehenen Daten performen wird. Die primäre Methode zur Schätzung des Generalisierungsfehlers ist die Bewertung des final ausgewählten Modells auf einem separaten Testdatensatz. Wir werden also das Modell, das wir mittels “Best Subset Selection” und analytischen Kriterien (z.B. BIC) als optimal identifiziert haben, auf den zurückgehaltenen Testdaten anwenden und dessen Leistung (z.B. Mean Squared Error - MSE) berechnen. Dieser MSE auf den Testdaten dient als unsere direkte Schätzung des Generalisierungsfehlers. Zusätzlich vergleichen wir diesen Wert mit den analytischen Schätzern. Die Schritte der Datenaufbereitung und der initialen Modellauswahl mittels regsubsets sind identisch zu LAB Prostate LinReg 2.R. Code-Blöcke und Interpretation #1. Bibliotheken laden & 2. Datenaufbereitung

```
# Laden der Bibliotheken
library(data.table)
library(ggplot2)
library(leaps)   # Für regsubsets
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-8
```

```
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(psych)
```

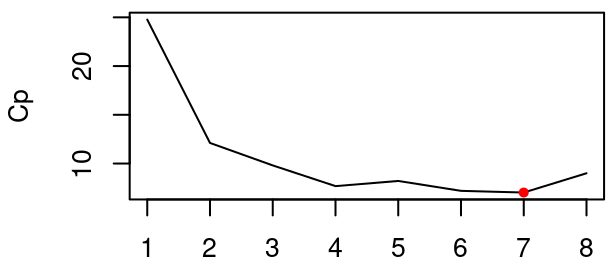
```
##
## Attaching package: 'psych'
```

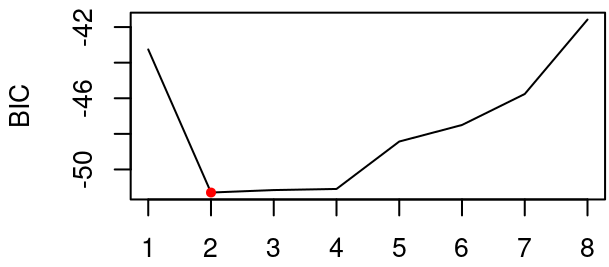
```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

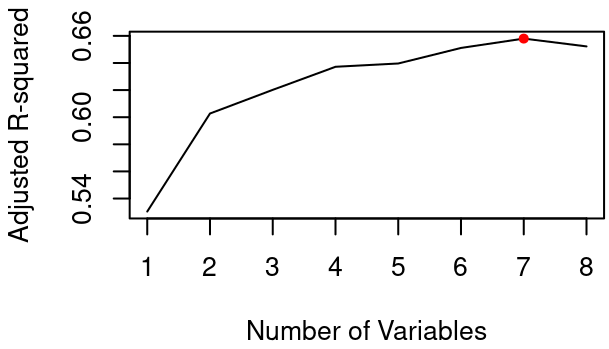
```
library(DataExplorer)

# Daten laden und vorbereiten
# Stellen Sie sicher, dass der Pfad und Dateiname korrekt sind.
# Beispiel für .data Datei:
prostateData <- read.table(file="prostate_data.csv", header = TRUE, sep = "")
prostateData <- as.data.table(prostateData)
prostateData_train <- prostateData[train==TRUE]
prostateData_test <- prostateData[train==FALSE]

prostateData_train$train <- NULL
prostateData_test$train <- NULL

prostateData_train_scaled <- scale(prostateData_train[, 1:8])
prostateData_train_scaled <- as.data.table(prostateData_train_scaled)
prostateData_train_scaled[, lpsa:=prostateData_train$lpsa] # Original lpsa zum Training hinzufüg
en

# Korrekte Hinzufügung von lpsa zum skalierten Testdatensatz
# Es wird die *originale* lpsa-Spalte aus prostateData_test benötigt.
temp_lpsa_test <- prostateData_test$lpsa
prostateData_test_scaled <- scale(prostateData_test[, 1:8])
prostateData_test_scaled <- as.data.table(prostateData_test_scaled)
prostateData_test_scaled[, lpsa:=temp_lpsa_test]
```

#Interpretation: Die Daten werden geladen und aufbereitet. Wichtig ist die korrekte Erstellung von prostateData_train_scaled und prostateData_test_scaled, insbesondere dass die korrekte lpsa-Spalte (nicht-skaliert) für die spätere Fehlerberechnung vorhanden ist. #3. Durchführung der Best Subset Selection (Basis für Modellauswahl)

```
# Volles lineares Modell (wird für sigma^2 Schätzung in Cp benötigt)
lmFit_all <- lm(lpsa~., data=prostateData_train_scaled)

# Best Subset Selection durchführen
best_subset_selection <- regsubsets(lpsa~., data=prostateData_train_scaled[,1:9], nvmax=8)
summary_best_subset <- summary(best_subset_selection)

 #Die Plots der Auswahlkriterien sind hier optional, da der Fokus auf der Testset-Evaluation lie
gt.

 par(mfrow = c(2, 2))
 plot(summary_best_subset$cp, xlab = "Number of Variables", ylab = "Cp", type = "l"); points(whi
ch.min(summary_best_subset$cp), summary_best_subset$cp[which.min(summary_best_subset$cp)], col =
"red", pch = 20)
 plot(summary_best_subset$bic, xlab = "Number of Variables", ylab = "BIC", type = "l"); points(w
hich.min(summary_best_subset$bic), summary_best_subset$bic[which.min(summary_best_subset$bic)],
col = "red", pch = 20)
 plot(summary_best_subset$adjr2, xlab = "Number of Variables", ylab = "Adjusted R-squared", type
= "l"); points(which.max(summary_best_subset$adjr2), summary_best_subset$adjr2[which.max(summary
_best_subset$adjr2)], col = "red", pch = 20)
 par(mfrow = c(1,1))
```




#Interpretation: Die “Best Subset Selection” liefert uns eine Reihe von Kandidatenmodellen und die dazugehörigen analytischen Schätzer (Cp, BIC, adjR²). Diese Informationen werden im nächsten Schritt verwendet, um ein finales

Modell für die Bewertung auszuwählen. Das lmFit_all (volle Modell) wird später für die Berechnung von sigma^2 benötigt, falls wir Cp-Werte genauer interpretieren wollen. # 4. Auswahl des finalen Modells, Training und Schätzung des Generalisierungsfehlers

```
# Auswahl des finalen Modells, z.B. basierend auf dem minimalen BIC
num_vars_final_model <- which.min(summary_best_subset$bic)
cat("Das Modell mit minimalem BIC hat", num_vars_final_model, "Variablen.\n")
```

```
## Das Modell mit minimalem BIC hat 2 Variablen.
```

```
# Extrahieren der Koeffizienten und Variablennamen für dieses Modell
final_model_coefs_summary <- coef(best_subset_selection, id = num_vars_final_model)
final_model_vars <- names(final_model_coefs_summary)[-1] # Intercept entfernen

cat("Ausgewählte Variablen für das finale Modell:", paste(final_model_vars, collapse=", "),
"\n")
```

```
## Ausgewählte Variablen für das finale Modell: lcavol, lweight
```

```
# Erstellen der Formel für das finale Modell
if (length(final_model_vars) > 0) {
  final_model_formula <- as.formula(paste("lpsa ~", paste(final_model_vars, collapse = " + ")))
} else { # Falls das beste Modell nur der Intercept ist (selten bei BIC mit echten Prädiktoren)
  final_model_formula <- as.formula("lpsa ~ 1")
}
cat("Formel des finalen Modells:", deparse(final_model_formula), "\n")
```

```
## Formel des finalen Modells: lpsa ~ lcavol + lweight
```

```
# Trainieren des finalen ausgewählten Modells auf den *gesamten* Trainingsdaten
lmFit_final_selected <- lm(final_model_formula, data = prostateData_train_scaled)

# A. Fehler auf Trainingsdaten für das finale Modell (Trainings-MSE)
mse_train_final_selected <- mean(residuals(lmFit_final_selected)^2)
cat("MSE auf Trainingsdaten (finales ausgewähltes Modell):", round(mse_train_final_selected, 4),
"\n")
```

```
## MSE auf Trainingsdaten (finales ausgewähltes Modell): 0.5536
```

```
# B. Schätzung des Generalisierungsfehlers auf den Testdaten (Test-MSE)
# Dies ist der Kernfokus dieses Skripts.
predictions_on_test <- predict(lmFit_final_selected, newdata = prostateData_test_scaled)
mse_on_test <- mean((predictions_on_test - prostateData_test_scaled$lpsa)^2)

cat("------------------------------------------------------------------------\n")
```

```
## ------------------------------------------------------------------------
```

```
cat("SCHÄTZUNG DES GENERALISIERUNGSFEHLERS (MSE auf Testdaten):\n")
```

```
## SCHÄTZUNG DES GENERALISIERUNGSFEHLERS (MSE auf Testdaten):
```

```
cat("   MSE auf Testdaten für das via BIC ausgewählte Modell:", round(mse_on_test, 4), "\n")
```

```
##    MSE auf Testdaten für das via BIC ausgewählte Modell: 0.5484
```

```
cat("------------------------------------------------------------------------\n")
```

```
## ------------------------------------------------------------------------
```

```
# C. Vergleich mit analytischen Schätzern (aktiviert)
if (exists("lmFit_all")) { # Stellt sicher, dass lmFit_all existiert
  # Cp-Wert des ausgewählten Modells (aus regsubsets)
  # Dieser Cp-Wert ist RSSp/sigma_hat_full^2 - (n - 2p_model)
  cp_value_from_regsubsets <- summary_best_subset$cp[num_vars_final_model]
  cat("Mallows' Cp des ausgewählten Modells (von regsubsets):", round(cp_value_from_regsubsets,
2), "\n")
}
```

```
## Mallows' Cp des ausgewählten Modells (von regsubsets): 12.11
```

```
bic_value_selected_model <- summary_best_subset$bic[num_vars_final_model]
cat("BIC des ausgewählten Modells (von regsubsets):", round(bic_value_selected_model, 2), "\n")
```

```
## BIC des ausgewählten Modells (von regsubsets): -51.3
```

```
# Adjusted R-squared des ausgewählten Modells
adjr2_value_selected_model <- summary_best_subset$adjr2[num_vars_final_model]
cat("Adjusted R-squared des ausgewählten Modells (von regsubsets):", round(adjr2_value_selected_
model, 4), "\n")
```

```
## Adjusted R-squared des ausgewählten Modells (von regsubsets): 0.6027
```

#Interpretation des Prozesses zur Schätzung des Generalisierungsfehlers und Vergleich: Auswahl und Training des finalen Modells: Ein Modell wird basierend auf einem analytischen Kriterium (hier: minimaler BIC) ausgewählt. Die spezifischen Variablen werden identifiziert, eine Formel wird erstellt, und dieses Modell (lmFit_final_selected) wird auf den gesamten Trainingsdaten trainiert. Trainings-MSE: Der mse_train_final_selected zeigt, wie gut das ausgewählte Modell auf den Daten passt, auf denen es gelernt hat. Er dient als Referenz, ist aber typischerweise eine optimistische Schätzung der Leistung auf neuen Daten. Test-MSE als Schätzung des

Generalisierungsfehlers: Der mse_on_test wird durch Anwendung des trainierten Modells auf die ungesehenen Testdaten berechnet. Dieser Wert ist unsere primäre, datengestützte Schätzung des Generalisierungsfehlers. Er gibt an, welchen Fehler wir im Durchschnitt erwarten, wenn dieses Modell auf neue Daten angewendet wird. Vergleich mit analytischen Schätzern: Der mse_on_test wird mit den Werten von Cp, BIC und Adjusted $R^2$ des ausgewählten Modells verglichen (die Werte stammen aus dem summary_best_subset-Objekt). BIC: Da das Modell aufgrund des minimalen BIC ausgewählt wurde, dient dieser Wert hier als Bestätigung. Mallows' Cp: Der Cp-Wert (z.B. r if(exists("cp_value_from_regsubsets")) round(cp_value_from_regsubsets, 2) else "NA") für ein Modell mit r num_vars_final_model Variablen wird betrachtet. Ein Wert nahe num_vars_final_model (plus Intercept, also num_vars_final_model + 1 für p in der Cp-Definition, wenn p die Gesamtzahl der Koeffizienten ist) deutet auf geringen Bias hin. Adjusted $R^2$: Ein höherer Adjusted $R^2$ (r if(exists("adjr2_value_selected_model")) round(adjr2_value_selected_model, 4) else "NA") ist besser. Dieser Vergleich hilft zu beurteilen, ob die analytischen Schätzer, die zur Auswahl des Modells verwendet wurden, auch gute Indikatoren für die tatsächliche Leistung auf Testdaten waren. Im Idealfall sollte ein Modell, das nach Cp/BIC/adj$R^2$ gut ist, auch einen niedrigen Test-MSE haben. Schlussfolgerung: Die zentrale Aufgabe dieses Skripts ist die Quantifizierung des Generalisierungsfehlers. Dies geschieht durch die sorgfältige Auswahl eines Modells auf den Trainingsdaten (mithilfe von Kriterien wie BIC) und dessen anschließende, ehrliche Bewertung auf einem unabhängigen Testdatensatz. Der resultierende Test-MSE (r if(exists("mse_on_test")) round(mse_on_test, 4) else "NA") ist die Schlüsselmetrik. Der Vergleich mit den analytischen Schätzern liefert zusätzlichen Kontext und hilft, die Zuverlässigkeit dieser Schätzer in der Praxis zu bewerten. Ein niedriger Test-MSE, der idealerweise nicht dramatisch höher als der Trainings-MSE ist, signalisiert ein Modell, das gut generalisiert.