Universität Duisburg-Essen
Lehrstuhl für Ökonometrie
Dr. Yannick Hoga
MSc. Martin Arnold

Wintersemester 2022/23

Übungsblatt 6 — Inferenz in Regressionsmodellen

R-Lösungen

Zu diesem Übungsblatt empfehlen wir neben der Lektüre von Kapitel 5 des Lehrbuches *Introduction to Econometrics* von *Stock & Watson* eine Aufarbeitung mithilfe der Kapitel 5 in unserem Online-Companion *Introduction to Econometrics with R*.

Aufgabe 1 — Anwendungsbeispiel: Boston Housing Data

Das Paket `MASS` enthält den Datensatz `Boston`, welcher wiederum die Variable `medv` (median house value) für 506 Gemeinden rund um Boston enthält. Im folgenden wollen wir ein lineares Modell aufstellen mit `medv` als abhängiger Variable und bis zu 13 erklärenden Variablen. Diese sind z. B. `rm` (average number of rooms per house), `age` (average age of houses), und `lstat` (percent of households with low socioeconomic status).

(a) Verschaffen Sie sich einen Überblick über den Datensatz.

```
# Paket und Datensatz laden
library(MASS)
data(Boston)

# Die ersten Zeilen des datensatzes betrachten
head(Boston)
```

```
     crim zn indus chas   nox    rm  age    dis rad tax ptratio  black lstat
1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90  4.98
2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90  9.14
3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83  4.03
4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63  2.94
5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90  5.33
6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12  5.21
  medv
1 24.0
2 21.6
3 34.7
4 33.4
5 36.2
6 28.7
```

```
# Wie viele Zeilen und Spalten?
dim(Boston)
```

```
[1] 506  14
```

```r
# Namen der Variablen im Datensatz
names(Boston)
```

```
 [1] "crim"    "zn"      "indus"   "chas"    "nox"     "rm"      "age"
 [8] "dis"     "rad"     "tax"     "ptratio" "black"   "lstat"   "medv"
```

```r
# Mehr Informationen über den Datensatz
?Boston

# Details der Variable `medv`
summary(Boston$medv)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   5.00   17.02   21.20   22.53   25.00   50.00
```

```r
# Details aller Variablen im Datensatz
summary(Boston)
```

```
      crim                zn             indus            chas
 Min.   : 0.00632   Min.   :  0.00   Min.   : 0.46   Min.   :0.00000
 1st Qu.: 0.08205   1st Qu.:  0.00   1st Qu.: 5.19   1st Qu.:0.00000
 Median : 0.25651   Median :  0.00   Median : 9.69   Median :0.00000
 Mean   : 3.61352   Mean   : 11.36   Mean   :11.14   Mean   :0.06917
 3rd Qu.: 3.67708   3rd Qu.: 12.50   3rd Qu.:18.10   3rd Qu.:0.00000
 Max.   :88.97620   Max.   :100.00   Max.   :27.74   Max.   :1.00000
      nox               rm             age              dis
 Min.   :0.3850   Min.   :3.561   Min.   :  2.90   Min.   : 1.130
 1st Qu.:0.4490   1st Qu.:5.886   1st Qu.: 45.02   1st Qu.: 2.100
 Median :0.5380   Median :6.208   Median : 77.50   Median : 3.207
 Mean   :0.5547   Mean   :6.285   Mean   : 68.57   Mean   : 3.795
 3rd Qu.:0.6240   3rd Qu.:6.623   3rd Qu.: 94.08   3rd Qu.: 5.188
 Max.   :0.8710   Max.   :8.780   Max.   :100.00   Max.   :12.127
      rad              tax           ptratio          black
 Min.   : 1.000   Min.   :187.0   Min.   :12.60   Min.   :  0.32
 1st Qu.: 4.000   1st Qu.:279.0   1st Qu.:17.40   1st Qu.:375.38
 Median : 5.000   Median :330.0   Median :19.05   Median :391.44
 Mean   : 9.549   Mean   :408.2   Mean   :18.46   Mean   :356.67
 3rd Qu.:24.000   3rd Qu.:666.0   3rd Qu.:20.20   3rd Qu.:396.23
 Max.   :24.000   Max.   :711.0   Max.   :22.00   Max.   :396.90
     lstat            medv
 Min.   : 1.73   Min.   : 5.00
 1st Qu.: 6.95   1st Qu.:17.02
 Median :11.36   Median :21.20
 Mean   :12.65   Mean   :22.53
 3rd Qu.:16.95   3rd Qu.:25.00
 Max.   :37.97   Max.   :50.00
```

```r
# Zusatz:
# Funktion auf Spalten des Datensatzes anwenden: `sd()`
apply(Boston, 2, sd)
```

```
       crim          zn       indus        chas         nox          rm
```

```
  8.6015451  23.3224530   6.8603529   0.2539940   0.1158777   0.7026171
        age         dis         rad         tax     ptratio       black
 28.1488614   2.1057101   8.7072594 168.5371161   2.1649455  91.2948644
      lstat        medv
  7.1410615   9.1971041
```

(b) Passen Sie ein einfaches lineares Modell an mit `medv` als abhängiger und `lstat` als erklärender Variable. Geben Sie die p-Werte und die Standardfehler der geschätzten Koeffizienten an. Berechnen Sie auch das R^2 und die F-Statistik für das Modell.

```r
# Lineare Regression von `medv` auf `lstat`
lm.fit <- lm(medv ~ lstat, data = Boston)
lm.fit
```

```
Call:
lm(formula = medv ~ lstat, data = Boston)

Coefficients:
(Intercept)        lstat
      34.55        -0.95
```

```r
# Mehr Informationen über das Modell erhalten
# (inkl. R^2 und F-Statistik)
summary(lm.fit)
```

```
Call:
lm(formula = medv ~ lstat, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.168  -3.990  -1.318   2.034  24.500

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 34.55384    0.56263   61.41   <2e-16 ***
lstat       -0.95005    0.03873  -24.53   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.216 on 504 degrees of freedom
Multiple R-squared:  0.5441,    Adjusted R-squared:  0.5432
F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

```r
# Konstante und Koeffizient von `lstat` sind signifikant von 0 verschieden:
# Beide p-Werte sind sehr nahe bei 0
#
# Das R^2 zeigt eine mittelmäßige Anpassung des Modells an die Daten.
#
# F-Test lehnt nicht ab: Der Koeffizient von`lstat` ist signifikant von 0 verschieden.


# Was wird in `lm.fit` gespeichert?
names(lm.fit)
```

```
 [1] "coefficients"  "residuals"     "effects"       "rank"
 [5] "fitted.values" "assign"        "qr"            "df.residual"
 [9] "xlevels"       "call"          "terms"         "model"
```

```
coef(lm.fit)
```

```
(Intercept)       lstat
 34.5538409  -0.9500494
```

Erklärung zu F-Test / F-Statistik im Output von `summary()`:

Die Nullhypothese bzgl. der im Output von `summary()` gelisteten F-Teststatistik (`F-Statistic`) lautet stets:

Die Koeffizienten aller Regressoren (Regressoren zusätzlich zur Konstante) sind null: kein Regressoren hat Erklärungskraft für die abhängige Variable.

$$H_0 : \beta_1, \ldots, \beta_k = 0$$

Es handelt sich also um einen *gemeinsamen* Test von Restriktionen bzgl. *aller* Koeffizienten $\beta_j, j = 1, \ldots, k$. Das Wichtigste zum F-Test:

- Unter der Annahme normalverteilter Fehler und Homoskedastie hat die F-Statistik eine F-Verteilung mit q Zähler- und $n-k-1$ Nennerfreiheitsgeraden. q ist die Anzahl der getesteten Restriktionen, n die Anzahl der Beobachtungen und k die Anzahl der Regressoren im *unrestringierten* Modell (exklusive Konstante).
- Das unrestringierte Modell ist das Modell unter der Alternativhypothese (die Restriktionen der Nullhypothese gelten nicht) und das restringierte Modell meint das Modell bei Gültigkeit der Nullhypothese (die Restriktionen der Nullhypothese gelten).
- Der F-Test ist ein rechtsseitiger Test, d.h. wir lehnen H_0 ab, wenn die F-Statistik das $1-\alpha$-Quantil der $F_{q,n-k-1}$-Verteilung *überschreitet*.

Im Modell in dieser Aufgabe gibt es nur einen Regressor (`lstat`). Daher lautet das unrestringierte Modell hier

$$medv_i = \beta_0 + \beta_1 lstat_i + u_i$$

und das restringierte Modell ist

$$medv_i = \beta_0 + \epsilon_i.$$

Es gilt $q = 1$, $k = 1$ und $n - k - 1 = 506 - 1 - 1 = 504$. Die F-Statistik folgt also einer F-Verteilung mit Zählerfreiheitsgrad 1 und 504 Nennerfreiheitsgeraden. Der kritische p-Wert für einen Test zum 5%-Niveau kann leicht mit `R` ermittelt werden:

```
qf(df1 = 1, df2 = 504, p = 0.95)
```

```
[1] 3.859975
```

Laut Output von Summary beträgt die Teststatistik `601.6` und liegt somit deutlich oberhalb des kritischen Werts von 3.86. Beachten Sie, dass `summary()` auch den zugehörigen p-Wert ausgibt. Dieser ist sehr klein (`p-value: <2.2e-16`). Wir können die Nullhypothese also zu jeden gängigen Signifikanzniveau ablehnen.

Hinweis:
In diesem Modell entspricht die F-Statistik der quadrierten t-Statistik für den Koeffizienten von `lstat`, d.h. beide Tests testen die selbe Hypothese (und kommen zum selben Ergebnis)! (Vergleichen Sie mit dem Exkurs zur F-Statistik auf dem letzten Übungsblatt.)

(c) Berechnen Sie ein 90%-Konfidenzintervall für die geschätzten Koeffizienten.

```
# 90%-Konfidenzintervalle der Koeffizientenschätzer
confint(lm.fit, level = 0.9)
```

```
                5 %       95 %
(Intercept) 33.626697 35.4809847
lstat       -1.013877 -0.8862212
```

(d) Schätzen Sie die Werte von `medv` für den Fall, dass die Variable `lstat` die Werte 5, 10 und 15 annimmt.

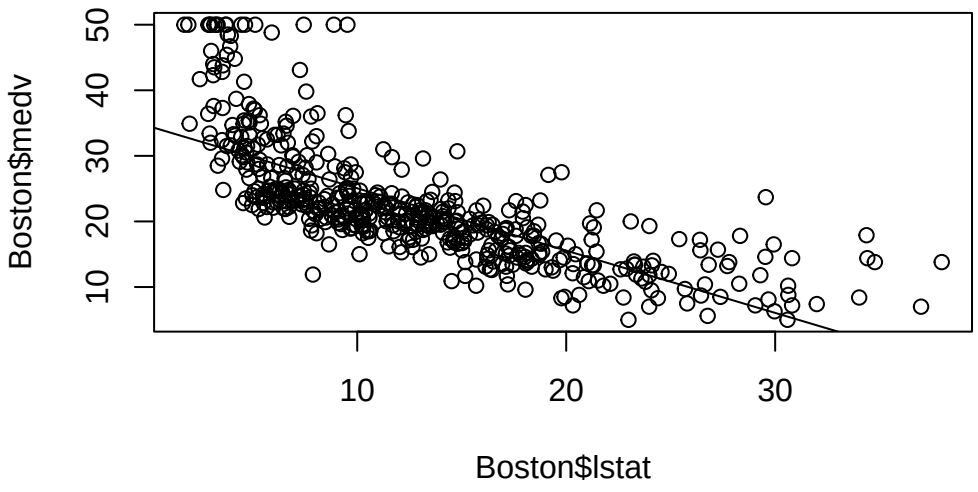
```
# Vorhersage von `medv` für die Werte 5, 10, 15 der Variable `lstat`
predict(lm.fit,
        new = data.frame(lstat = c(5, 10, 15)))
```

```
       1        2        3
29.80359 25.05335 20.30310
```

Beachten Sie, dass die Daten im Argument `new` als `data.frame` übergeben werden müssen!

(e) Plotten Sie `medv` und `lstat` zusammen mit der geschätzten Regressionsgerade.

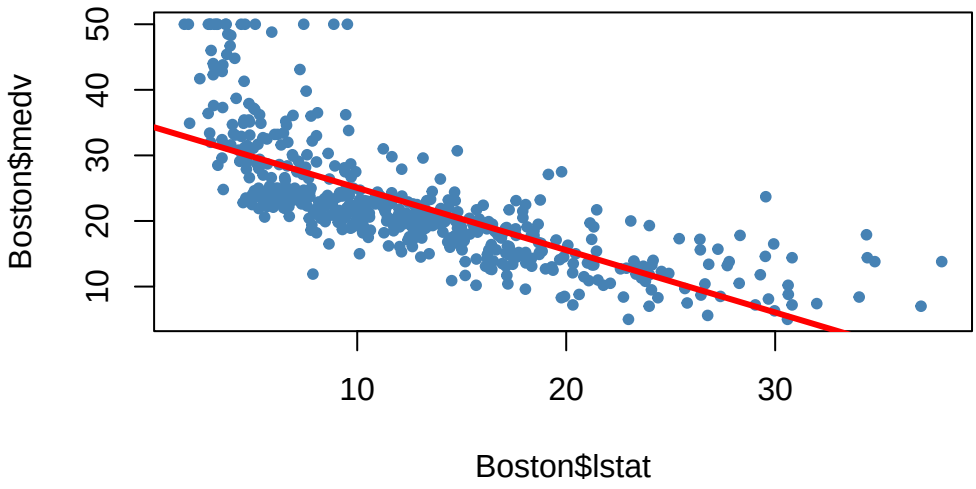
```
# Plot von `medv` und `lstat` zusammen mit der Regressionsgerade
plot(Boston$lstat, Boston$medv)
abline(lm.fit)
```




```
# Zusatz:

# blaue Punkte (Farbe "steelblue")
plot(Boston$lstat, Boston$medv, col = "steelblue", pch = 20)

# dicke, rote Regressionsgerade
abline(lm.fit, lwd = 3, col = "red")
```



(f) Wiederholen Sie die Aufgaben (b) und (c) unter Verwendung heteroskedastie-robuster Standardfehler. Ausführliche Erläuterungen zur Berechnung heteroskedastie-robuster Standardfehler in Regressionsmodellen mit Anwendungsbeispielen in R finden Sie in Kapitel 5.4 im Online-Companion.

```r
# Paket `AER` laden
library(AER)

# (b) mit hetoerskedastie-robuten Standardfehlern
coeftest(lm.fit,
         vcov. = vcovHC,
         type = "HC1")
```

```
t test of coefficients:

             Estimate Std. Error t value  Pr(>|t|)
(Intercept) 34.553841   0.754199  45.815 < 2.2e-16 ***
lstat       -0.950049   0.049605 -19.152 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# (c) mit hetoerskedastie-robuten Standardfehlern
rob.confint <- function(model, level) {

  # Robuste Varianz-Kovarianz-Matrix für `model` schätzen
  vcov_m <- vcovHC(model, type = "HC1")

  # Koeffizienten aus `model` auslesen
  coefs <- model$coefficients

  # Kritischen Wert zum niveau 1-level festlegen
  crit <- qnorm(level + (1 - level)/2)

  # Konfidenzintervalle berechnen
  l <- coefs - crit * sqrt(diag(vcov_m))
  u <- coefs + crit * sqrt(diag(vcov_m))

  # Konfidenzintervalle ausgeben
```

```
  return(
    cbind("lower" = l,
      "upper" = u)
  )
}

# Wir testen die Funktion mit `lm.fit`:
rob.confint(model = lm.fit, level = 0.9)
```

```
                lower      upper
(Intercept) 33.313295 35.7943871
lstat       -1.031642 -0.8684565
```

```
# Die geschätze Varianzmatrix bei Heteroskedastie:
# Robuste Standardfehler erhält man als Wurzel der Diagonalelemente
matrix <- vcovHC(lm.fit, type = "HC1")
matrix
```

```
            (Intercept)        lstat
(Intercept)  0.56881549 -0.035027495
lstat       -0.03502749  0.002460648
```

```
# Wurzel der Diagonalelemente
sqrt(diag(matrix))
```

```
(Intercept)       lstat
 0.75419857  0.04960492
```

- Zu (b): Die Funktion `coeftest()` erstellt standardmäßig die selbe statistische Zusammenfassung der Modellkoeffizienten wie `summary()`. Über die Argumente `vcov.` und `type` kann die Berechnung bei Heteroskedastie gültiger Standardfehler festgelegt werden.
- Zu (c): Die Funktion `confint()` berechnet nur bei Homoskedastie gültige Konfidenzintervalle. Wir schreiben daher eine Funktion `rob.confint()`, die bei Heteroskedastie gültige Intervalle berechnet. Die Vorgehensweise ist sehr ähnlich wie bei Zusatzaufgabe 2.
- Überprüfen Sie, dass die Wurzeln der Diagonalelemente der geschätzten Varianzmatrix den heteroskedastie-robusten Standardfehlern im Output von entsprechen!

(g) Passen Sie ein Regressionsmodell an, welches `lstat` und `age` als erklärende Variablen für `medv` enthält.

```
# Modell mit `lstat` und `age` als erklärende Variablen
lm.fit <- lm(medv ~ lstat + age, data = Boston)
summary(lm.fit)
```

```
Call:
lm(formula = medv ~ lstat + age, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.981  -3.978  -1.283   1.968  23.158

Coefficients:
```

```
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 33.22276    0.73085  45.458  < 2e-16 ***
lstat       -1.03207    0.04819 -21.416  < 2e-16 ***
age          0.03454    0.01223   2.826  0.00491 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.173 on 503 degrees of freedom
Multiple R-squared:  0.5513,    Adjusted R-squared:  0.5495
F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

(h) Passen Sie ein Regressionsmodell an, welches `medv` auf alle 13 Variablen regressiert.

```r
# Regression mit allen 13 Variablen im Datensatz `Boston`
lm.fit <- lm(medv ~ ., data = Boston)
summary(lm.fit)
```

```
Call:
lm(formula = medv ~ ., data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max
-15.595  -2.730  -0.518   1.777  26.199

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.646e+01  5.103e+00   7.144 3.28e-12 ***
crim        -1.080e-01  3.286e-02  -3.287 0.001087 **
zn           4.642e-02  1.373e-02   3.382 0.000778 ***
indus        2.056e-02  6.150e-02   0.334 0.738288
chas         2.687e+00  8.616e-01   3.118 0.001925 **
nox         -1.777e+01  3.820e+00  -4.651 4.25e-06 ***
rm           3.810e+00  4.179e-01   9.116  < 2e-16 ***
age          6.922e-04  1.321e-02   0.052 0.958229
dis         -1.476e+00  1.995e-01  -7.398 6.01e-13 ***
rad          3.060e-01  6.635e-02   4.613 5.07e-06 ***
tax         -1.233e-02  3.760e-03  -3.280 0.001112 **
ptratio     -9.527e-01  1.308e-01  -7.283 1.31e-12 ***
black        9.312e-03  2.686e-03   3.467 0.000573 ***
lstat       -5.248e-01  5.072e-02 -10.347  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.745 on 492 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7338
F-statistic: 108.1 on 13 and 492 DF,  p-value: < 2.2e-16
```

(i) Passen Sie ein Regressionsmodell an, welches `medv` auf alle Variablen im Datensatz außer `age` regressiert.

```r
# Modell mit allen Regressoren außer `age`
lm.fit <- lm(medv ~ . -age, data = Boston)
summary(lm.fit)
```

```
Call:
lm(formula = medv ~ . - age, data = Boston)

Residuals:
     Min       1Q   Median       3Q      Max
-15.6054  -2.7313  -0.5188   1.7601  26.2243

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  36.436927   5.080119   7.172 2.72e-12 ***
crim         -0.108006   0.032832  -3.290 0.001075 **
zn            0.046334   0.013613   3.404 0.000719 ***
indus         0.020562   0.061433   0.335 0.737989
chas          2.689026   0.859598   3.128 0.001863 **
nox         -17.713540   3.679308  -4.814 1.97e-06 ***
rm            3.814394   0.408480   9.338  < 2e-16 ***
dis          -1.478612   0.190611  -7.757 5.03e-14 ***
rad           0.305786   0.066089   4.627 4.75e-06 ***
tax          -0.012329   0.003755  -3.283 0.001099 **
ptratio      -0.952211   0.130294  -7.308 1.10e-12 ***
black         0.009321   0.002678   3.481 0.000544 ***
lstat        -0.523852   0.047625 -10.999  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.74 on 493 degrees of freedom
Multiple R-squared:  0.7406,    Adjusted R-squared:  0.7343
F-statistic: 117.3 on 12 and 493 DF,  p-value: < 2.2e-16
```

```r
# Zweite Möglichkeit:
lm.fit <- update(lm.fit, ~ . -age)
```

Das Alter hat keinen signifikanten Einfluss auf `medv` in der zweiten Regression. Das könnte an einer möglichen Korrelation zwischen der Variable `age` und einer anderen Variable im Datensatz liegen, die im zweiten Modell benutzt wurde aber nicht im ersten (mehr zu dieser Problematik auf Übungsblatt 6 der Theorieübung).

(j) Wiederholen Sie die Aufgabenteile (g) - (i) unter Verwendung bei Heteroskedastie gültiger Standardfehler.

```r
# (g) mit heteroskedastie-robusten Standardfehlern
lm.fit <- lm(medv ~ lstat + age, data = Boston)
coeftest(lm.fit, vcov. = vcovHC, type = "HC1")
```

```
t test of coefficients:

             Estimate Std. Error  t value Pr(>|t|)
(Intercept) 33.222761   0.729488  45.5426  < 2e-16 ***
lstat       -1.032069   0.077501 -13.3169  < 2e-16 ***
age          0.034544   0.016883   2.0461  0.04126 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# (h) mit heteroskedastie-robusten Standardfehlern
lm.fit <- lm(medv ~ ., data = Boston)
coeftest(lm.fit, vcov. = vcovHC, type = "HC1")
```

```
t test of coefficients:

              Estimate  Std. Error t value  Pr(>|t|)
(Intercept)  3.6459e+01  8.0010e+00  4.5569 6.558e-06 ***
crim        -1.0801e-01  2.8944e-02 -3.7317 0.0002124 ***
zn           4.6420e-02  1.3765e-02  3.3722 0.0008043 ***
indus        2.0559e-02  5.0380e-02  0.4081 0.6834006
chas         2.6867e+00  1.2938e+00  2.0766 0.0383600 *
nox         -1.7767e+01  3.7858e+00 -4.6930 3.495e-06 ***
rm           3.8099e+00  8.4490e-01  4.5093 8.142e-06 ***
age          6.9222e-04  1.6464e-02  0.0420 0.9664807
dis         -1.4756e+00  2.1471e-01 -6.8724 1.918e-11 ***
rad          3.0605e-01  6.1436e-02  4.9816 8.744e-07 ***
tax         -1.2335e-02  2.6909e-03 -4.5838 5.798e-06 ***
ptratio     -9.5275e-01  1.1744e-01 -8.1126 3.985e-15 ***
black        9.3117e-03  2.6786e-03  3.4763 0.0005534 ***
lstat       -5.2476e-01  9.9650e-02 -5.2660 2.087e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# (i) mit heteroskedastie-robusten Standardfehlern
lm.fit <- lm(medv ~ . -age, data = Boston)
coeftest(lm.fit, vcov. = vcovHC, type = "HC1")
```

```
t test of coefficients:

              Estimate  Std. Error t value  Pr(>|t|)
(Intercept) 36.4369266   7.7394270  4.7080 3.256e-06 ***
crim        -0.1080056   0.0289183 -3.7349 0.0002098 ***
zn           0.0463337   0.0133502  3.4706 0.0005649 ***
indus        0.0205622   0.0503408  0.4085 0.6831137
chas         2.6890262   1.2954334  2.0758 0.0384327 *
nox        -17.7135399   3.3772862 -5.2449 2.325e-07 ***
rm           3.8143936   0.7896299  4.8306 1.820e-06 ***
dis         -1.4786116   0.2251148 -6.5683 1.295e-10 ***
rad          0.3057859   0.0591366  5.1708 3.394e-07 ***
tax         -0.0123287   0.0026932 -4.5777 5.960e-06 ***
ptratio     -0.9522112   0.1145732 -8.3109 9.281e-16 ***
black        0.0093207   0.0026692  3.4919 0.0005227 ***
lstat       -0.5238518   0.0849697 -6.1652 1.467e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Die Vorgehensweise ist genau wie in Aufgabenteil (f): Wir verwenden `coeftest()` anstatt `summary()`. Hinsichtlich der Signifikanz der Koeffizienten ergeben sich keine wesentlichen Unterschiede.