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

Wintersemester 2022/23

Übungsblatt 5 — Inferenz im einfachen Regressionsmodell

Aufgabe 3 — Anwendungsbeispiel: Hauspreise

Der Datensatz `br.dat` (Variablenbeschreibung in `br.def`) enthält die Daten von 1080 verkauften Häusern im Ort Baton Rouge für das Jahr 2005.

(a) Schätzen Sie die Koeffizienten aus dem Regressionsansatz $\text{Price} = \beta_0 + \beta_1 \text{Age} + \epsilon$ und konstruieren Sie ein 95%-Konfidenzintervall für den unbekannten Koeffizienten β_1.

Zunächst lesen wir die Daten ein. Die Funktion `read.table()` erzeugt ein Objekt des Typs `data.frame`. Da die Datei `br.dat` keine Variablennamen enthält, weisen wir diese manuell zu (siehe `br.def` für eine Beschreibung des Datensatzes). Wir ändern hierzu die Spaltennamen von `baton` mit der Funktion `colnames()`.

```r
# Der Datnsatz 'br.dat' ist in Moodle verfügbar
# (ggf. korrektes Arbeitsverzeichnis setzen)
# Daten einlesen; erzeugt einen data.frame:
# baton <- read.table("br.dat")

# Ist 'baton' ein data.frame?
is.data.frame(baton)
```

```
[1] TRUE
```

```r
# Variablennamen gem. der Beschreibung setzen
colnames(baton) <- c("Price", "sqft","Bedrooms",
                     "Baths", "Age", "Occupancy",
                     "Pool", "Style", "Fireplace",
                     "Waterfront", "DOM")
# Ueberblick
head(baton)
```

```
   Price sqft Bedrooms Baths Age Occupancy Pool Style Fireplace Waterfront DOM
1  66500  741        1     1  18         1    1     1         1          0   6
2  66000  741        1     1  18         2    1     1         0          0  23
3  68500  790        1     1  18         1    0     1         1          0   8
4 102000 2783        2     2  18         1    0     1         1          0  50
5  54000 1165        2     1  35         2    0     1         0          0 190
6 143000 2331        2     2  25         1    0     1         1          0  86
```

```r
# statistische Zusammenfassung
summary(baton)
```

```
     Price             sqft         Bedrooms        Baths      
 Min.   :  22000   Min.   : 662   Min.   :1.00   Min.   :1.000  
 1st Qu.:  99000   1st Qu.:1604   1st Qu.:3.00   1st Qu.:2.000  
 Median : 130000   Median :2186   Median :3.00   Median :2.000  
 Mean   : 154863   Mean   :2326   Mean   :3.18   Mean   :1.973  
 3rd Qu.: 170162   3rd Qu.:2800   3rd Qu.:4.00   3rd Qu.:2.000  
 Max.   :1580000   Max.   :7897   Max.   :8.00   Max.   :5.000  
      Age          Occupancy          Pool             Style       
 Min.   : 1.00   Min.   :1.000   Min.   :0.00000   Min.   : 1.000  
 1st Qu.: 5.00   1st Qu.:1.000   1st Qu.:0.00000   1st Qu.: 1.000  
 Median :18.00   Median :2.000   Median :0.00000   Median : 1.000  
 Mean   :19.57   Mean   :1.565   Mean   :0.07963   Mean   : 3.753  
 3rd Qu.:25.00   3rd Qu.:2.000   3rd Qu.:0.00000   3rd Qu.: 7.000  
 Max.   :80.00   Max.   :3.000   Max.   :1.00000   Max.   :11.000  
   Fireplace       Waterfront           DOM        
 Min.   :0.000   Min.   :0.00000   Min.   :  0.00  
 1st Qu.:0.000   1st Qu.:0.00000   1st Qu.: 14.00  
 Median :1.000   Median :0.00000   Median : 40.00  
 Mean   :0.563   Mean   :0.07222   Mean   : 74.06  
 3rd Qu.:1.000   3rd Qu.:0.00000   3rd Qu.:100.25  
 Max.   :1.000   Max.   :1.00000   Max.   :728.00  
```

```r
# Modell schätzen
mod <- lm(Price ~ Age, data = baton)
mod
```

```

Call:
lm(formula = Price ~ Age, data = baton)

Coefficients:
(Intercept)          Age  
     184053        -1491  
```

```r
# Statistischer Zusammenfassung der Schätzunh
summary(mod)
```

```

Call:
lm(formula = Price ~ Age, data = baton)

Residuals:
    Min      1Q  Median      3Q     Max 
-117210  -54383  -27562   16930 1415333 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 184052.8     5546.5  33.183  < 2e-16 ***
Age          -1491.2      212.9  -7.003 4.39e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 120300 on 1078 degrees of freedom
Multiple R-squared:  0.04352,	Adjusted R-squared:  0.04263 
F-statistic: 49.05 on 1 and 1078 DF,  p-value: 4.392e-12
```

```r
# Genauere Ergebnisse erhalten wir durch Auslesen der
# Koeffizienten mit coef()
coef(mod)
```

```
(Intercept)         Age
 184052.764   -1491.237
```

Wir berechnen das Konfidenzintervall zunächst per Hand. Hierzu verwenden wir die Statistiken aus dem Output von `summary(mod)` weiter oben.

Für das 95%-Konfidenzintervall für $\widehat{\beta}_1$ benutzen wir die Formel

$$95\%\text{-KI} = \left[\widehat{\beta}_1 - \text{KritischerWert}(5\%) \cdot SE(\widehat{\beta}_1),\ \widehat{\beta}_1 + \text{KritischerWert}(5\%) \cdot SE(\widehat{\beta}_1)\right]$$

von Folie 4-16 der VL. Wir verwenden zunächst den kritischen Wert der t-Verteilung mit $n-2$ Freiheitsgraden.

```r
#### Das 95%-KI per Hand berechnen ####
n <- nrow(baton)                          # Anzahl Beobachtungen
k <- 2                                    # Anzahl der Regressoren + Konstante
alpha <- 0.05                             # Signifikanzniveau
t.factor <- qt(p = 1-alpha/2, df = n-2)  # 1 - alpha/2 Quantil der t-Verteilung
baton.coef <- summary(mod)$coefficients  # Koeffizientenmatrix
b.dach <- baton.coef[2, 1]               # Schätzung des Koeffizienten
b.err <- baton.coef[2, 2]                # Standardfehler des Schätzers

# Intervall berechnen:
c(b.dach - t.factor * b.err, b.dach + t.factor * b.err)
```

```
[1] -1909.047 -1073.427
```

Praktischer ist die Funktion `confint()`, welche genau diese Berechnung für uns durchführt:

```r
# Mit 'confint()' das 95%-KI für Koeffizienten von 'Age' berechnen
confint(mod, parm = "Age")
```

```
        2.5 %    97.5 %
Age -1909.047 -1073.427
```

Wie bei der Funktion `t.test()` rechnet `confint()` mit dem 97.5%-Quantil der entsprechenden t-Verteilung. Wir können 1.96, dass 97.5%-Quantil der Standardnormalverteilung verwenden, da die t-Verteilung für $n - k = 1080 - 2 = 1078$ Freiheitsgrade sehr gut durch die Standardnormalverteilung approximiert wird (s. Folie 4-10 ff.), d.h. wir approximieren 1.962167 mit 1.96 (Vergleich `qt(0.975, df = 1078)` und `qnorm(0.975)`).

Per Hand:

```r
# 95%-KI wenn Standardnormalverteilung unterstellt wird:
c(b.dach - 1.96 * b.err, b.dach + 1.96 * b.err)
```

```
[1] -1908.586 -1073.889
```

Die beiden Intervalle sind tatsächlich sehr ähnlich.

(b) Testen Sie die Nullhypothese, dass ein zusätzliches Jahr im Alter des Hauses zu einer Reduktion des Preises um 1000 Dollar führt. Bestimmen Sie dazu den p-Wert des geeigneten zweiseitigen t-Tests.

Wir können `R` nutzen, um mit den berechneten Größen (t-Statistik / p-Wert) zu überprüfen, ob H_0 : $\beta_1 = -1000$ zum Signifikanzniveau 5% abgelehnt wird:

```
## Test per Hand durchführen
# Nullhypothese
Age.H0 <- -1000

# Teststatistik
t.Stat <- (b.dach - Age.H0)/b.err
```

Wir berechnen den p-Wert für den Test gem. den Erläuterungen auf Folie 4-10.

```
# P-Wert (t-Verteilung mit 1078 Freiheitsgraden wird unterstellt)
p.Wert <- 2 * (1 - pt(abs(t.Stat), df = n - k))
p.Wert
```

```
[1] 0.02124343
```

Hypothesentest durchführen…

…durch Vergleich mit kritischem Wert (liegt die Teststatistik im Ablehnbereich?)

```
abs(t.Stat) >= t.factor
```

```
[1] TRUE
```

```
# => Wir lehnen H_0: beta_1 = -1000 ab!
```

…mit dem p-Wert: ist dieser kleiner als das Signifikanzniveau $\alpha = 0.05$?

```
p.Wert <= 0.05
```

```
[1] TRUE
```

```
# => Wir lehnen H_0: beta_1 = -1000 ab!
```

Da $p = 0.02124 \leq \alpha = 0.05$ kann die Nullyhypothese $H_0 : \beta_1 = -1000$ also zum Signifikanzniveau 5% verworfen werden.

Exkurs:

Für den Hypothesentest können wir die Funktion `linearHypothesis()` benutzen. Diese ist Bestandteil des Pakets `AER`.

```r
# Paket AER installieren
# install.packages("AER") # einmalig

# Paket AER laden
library(AER) # jedes mal, wenn R neu gestartet wird

# Hypothesentest mit linearHypothesis() durchführen
linearHypothesis(mod, "Age = -1000")
```

```
Linear hypothesis test

Hypothesis:
Age = - 1000

Model 1: restricted model
Model 2: Price ~ Age

  Res.Df        RSS Df  Sum of Sq      F  Pr(>F)
1   1079 1.5669e+13
2   1078 1.5592e+13  1 7.6979e+10 5.3223 0.02124 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Die Teststatistik hat den Wert 5.3223. Auch hier beträgt der p-Wert etwa 0.02124. Auch hier argumentieren wir, dass die Nullhypothese abgelehnt wird, denn $p = 0.02124 < \alpha = 0.05$.

Genauer führt `linearHypothesis()` keinen t-Test, sondern einen F-Test durch:

Unter der Annahme $H_0 : \beta_1 = \beta_{1,0}$ und normalverteilter KQ-Schätzer ist

$$t\text{-Statistik} = \left[\widehat{\beta}_1 - \beta_{1,0}\right] / SE(\widehat{\beta}_1) \sim t_{n-k},$$

d.h. die t-Statistik ist t-verteilt mit $n - k$ Freiheitsgraden, und es gilt

$$F\text{-Statistik} = \left[\frac{\widehat{\beta}_1 - \beta_{1,0}}{SE(\widehat{\beta}_1)}\right]^2 \sim F_{1,n-k},$$

wobei $F_{1,n-k}$ die F-Verteilung mit Zählerfreiheitsgrad 1 und Nennerfreiheitsgraden $n - k$ ist.

Für die Teststatistik der Hypothese $H_0 : \beta_1 = -1000$ mit $n = 1080$ und $k = 2$ folgt also

$$\left[\frac{\widehat{\beta}_1 - (-1000)}{SE(\widehat{\beta}_1)}\right]^2 \sim F_{1,1078}.$$

Aufgrund des Quadrates sind Realisationen der Teststatistiken nicht-negativ. Wir lehnen ab, wenn die Teststatistik den kritischen Wert in der rechten Flanke der $F_{1,1078}$-Verteilung *überschreitet*.

```r
# kritischer Wert der F_1,1078-Verteilung für alpha = 0.05
qf(0.95, df1 = 1, df2 = 1078)
```

```
[1] 3.850099
```

Die Grafik unten zeigt die Dichtefunktion der $F_{1,1078}$-Verteilung sowie den Ablehnbereich für einen Test zum Signifikanzniveau $\alpha = 0.05$. Offenbar liegt 5.3223, der Wert unserer Teststatistik, im Ablehnbereich, sodass $H_0 : \beta_1 = -1000$ zum 5%-Niveau angelehnt wird.

```
# Dichtefunktion der F_1,1078-Verteilung zeichnen
curve(df(x, df1 = 1, df2 = 1078),
      yaxs = "i",
      cex.lab = 0.75,
      cex.axis = 0.75,
      lwd = 2,
      from = 0,
      to = 6,
      xlab = "Quadrierte t-Statistik bei n=1080 und k=2",
      ylab = "Dichte der F(1,1078)-Verteilung")

# Kritischen Wert berechnen
kritWert <- qf(0.95, df1 = 1, df2 = 1078)

# Ablehnbereich in die Grafik einzeichnen
lines(x = seq(kritWert, 7, length.out = 60), y = rep(0.01, 60), col = "red", lwd = 5)
```