Kolokwium

Piotr Chlebicki

2023-12-04

Pakiety:

```r
library(lmtest)
library(tidyverse) # dplyr + ggplot
```

Zadanie 1

```r
head(iris)
```

a)

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
iris %>%
  select(Sepal.Length, Sepal.Width) %>%
  # Wyświetlam tylko początek, żeby było łatwiej czytać
  head()
```

```
##   Sepal.Length Sepal.Width
## 1          5.1         3.5
## 2          4.9         3.0
## 3          4.7         3.2
## 4          4.6         3.1
## 5          5.0         3.6
## 6          5.4         3.9
```

```r
iris %>%
  select(starts_with("S")) %>%
  head()
```

b)

```
##   Sepal.Length Sepal.Width Species
## 1          5.1         3.5  setosa
## 2          4.9         3.0  setosa
## 3          4.7         3.2  setosa
## 4          4.6         3.1  setosa
```

```
## 5          5.0         3.6  setosa
## 6          5.4         3.9  setosa
```

```
iris %>%
  filter(Sepal.Length >= 3.5,
         Petal.Width  >= .8) %>%
  head()
```

c)

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1          7.0         3.2          4.7         1.4 versicolor
## 2          6.4         3.2          4.5         1.5 versicolor
## 3          6.9         3.1          4.9         1.5 versicolor
## 4          5.5         2.3          4.0         1.3 versicolor
## 5          6.5         2.8          4.6         1.5 versicolor
## 6          5.7         2.8          4.5         1.3 versicolor
```

```
iris %>%
  select(Sepal.Length, Sepal.Width, Species) %>%
  arrange(Sepal.Length, Sepal.Width) %>%
  head()
```

d)

```
##   Sepal.Length Sepal.Width Species
## 1          4.3         3.0  setosa
## 2          4.4         2.9  setosa
## 3          4.4         3.0  setosa
## 4          4.4         3.2  setosa
## 5          4.5         2.3  setosa
## 6          4.6         3.1  setosa
```

```
iris %>%
  mutate(proportion = Petal.Length / Petal.Width) %>%
  head()
```

e)

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species proportion
## 1          5.1         3.5          1.4         0.2  setosa       7.00
## 2          4.9         3.0          1.4         0.2  setosa       7.00
## 3          4.7         3.2          1.3         0.2  setosa       6.50
## 4          4.6         3.1          1.5         0.2  setosa       7.50
## 5          5.0         3.6          1.4         0.2  setosa       7.00
## 6          5.4         3.9          1.7         0.4  setosa       4.25
```

f) Wszystkie cechy poza `Species` są numeryczne (gdyby tak nie było można dać inny warunek logiczny).

Dla każdej zmiennej pierwsza kolumna to średnia druga to mediana trzecia to odchylebnie standardowe.

```
iris %>%
  group_by(Species) %>%
  summarise_if(is.numeric, ~ cbind(mean(.x), median(.x), sd(.x)))
```

```
## # A tibble: 3 x 5
##   Species    Sepal.Length[,1] Sepal.Width[,1] Petal.Length[,1] Petal.Width[,1]
##   <fct>                 <dbl>           <dbl>            <dbl>           <dbl>
## 1 setosa                 5.01            3.43             1.46           0.246
## 2 versicolor             5.94            2.77             4.26           1.33
## 3 virginica              6.59            2.97             5.55           2.03
## # i 4 more variables: Sepal.Length[2:3] <dbl>, Sepal.Width[2:3] <dbl>,
## #   Petal.Length[2:3] <dbl>, Petal.Width[2:3] <dbl>
```

Zadanie 2

```r
set.seed(1234567890)
fn <- function(n = 1000) {
  Y <- replicate(
    n    = n,
    expr = {
      # gdy rpois(n = 1, lambda = 10) - rpois(n = 1, lambda = 2) < 0
      # czyli liczebnośc X-sów
      # zwracane jest zero
      sum(sample(
        x       = c(100, 1000, 10000),
        size    = max(0, rpois(n = 1, lambda = 10) - rpois(n = 1, lambda = 2)),
        prob    = c(.2, .75, .05),
        replace = TRUE
      ) ^ 2)
    }
  )

  c("mean" = mean(Y), "sd" = sd(Y))
}
fn()
```

```
##     mean       sd
## 47223870 66564247
```

Zadanie 3

Dane

```r
df <- tibble(
  x = c(100, 200, 300, 450, 600, 800, 1000),
  y = c(253, 337, 395, 451, 495, 534,  574)
)
```

a) Model kwadratowy:

```r
lm_square <- lm(y ~ poly(x, degree = 2, raw = TRUE), data = df)
# To samo co
# lm_square <- lm(y ~ x + I(x ^ 2), data = df)
```

Model sześcienny:

```r
lm_cube <- lm(y ~ poly(x, degree = 3, raw = TRUE), data = df)
```

b) Na poziomie istotności 95% wszystkie parametry (w obydwu modelach) są istotne bo p-wartości są niższe niż 0.05:

```r
summary(lm_square)
```

```
##
## Call:
## lm(formula = y ~ poly(x, degree = 2, raw = TRUE), data = df)
##
## Residuals:
##       1       2       3       4       5       6       7
## -14.420   9.192  13.624   2.060  -6.158 -12.912   8.614
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      2.002e+02  1.695e+01  11.811 0.000294 ***
## poly(x, degree = 2, raw = TRUE)1  7.062e-01  7.568e-02   9.332 0.000734 ***
## poly(x, degree = 2, raw = TRUE)2 -3.410e-04  6.754e-05  -5.049 0.007237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 13.79 on 4 degrees of freedom
## Multiple R-squared:  0.9902, Adjusted R-squared:  0.9852
## F-statistic: 201.1 on 2 and 4 DF,  p-value: 9.696e-05
```

```r
summary(lm_cube)
```

```
##
## Call:
## lm(formula = y ~ poly(x, degree = 3, raw = TRUE), data = df)
##
## Residuals:
##        1        2        3        4        5        6        7
## -2.35639  3.52782  1.83769 -4.43416  0.01945  2.21560 -0.81001
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      1.555e+02  8.182e+00  19.003 0.000318 ***
## poly(x, degree = 3, raw = TRUE)1  1.119e+00  6.454e-02  17.332 0.000419 ***
## poly(x, degree = 3, raw = TRUE)2 -1.254e-03  1.360e-04  -9.220 0.002699 **
## poly(x, degree = 3, raw = TRUE)3  5.550e-07  8.184e-08   6.782 0.006552 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.941 on 3 degrees of freedom
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9988
## F-statistic:  1658 on 3 and 3 DF,  p-value: 2.512e-05
```

wykorzystanie standardowych testów jest uzasadnione, bo po pierwsze nie występuje (albo przynajmniej nie ma podstawy twierdzić, że występuje) heteroskedastyczność i nie ma potrzeby korekty błędów standardowych przez macierze zgodne z heteroskedastycznością:

```r
bptest(lm_square) %>% print()
```

```
##
##  studentized Breusch-Pagan test
##
```

```
## data:  lm_square
## BP = 2.0255, df = 2, p-value = 0.3632
```

```r
bptest(lm_cube)   %>% print()
```

```
##
##  studentized Breusch-Pagan test
##
## data:  lm_cube
## BP = 1.9875, df = 3, p-value = 0.575
```

Reszty regresji mają też rozkład normalny:

```r
resid(lm_square) %>% shapiro.test() %>% print()
```

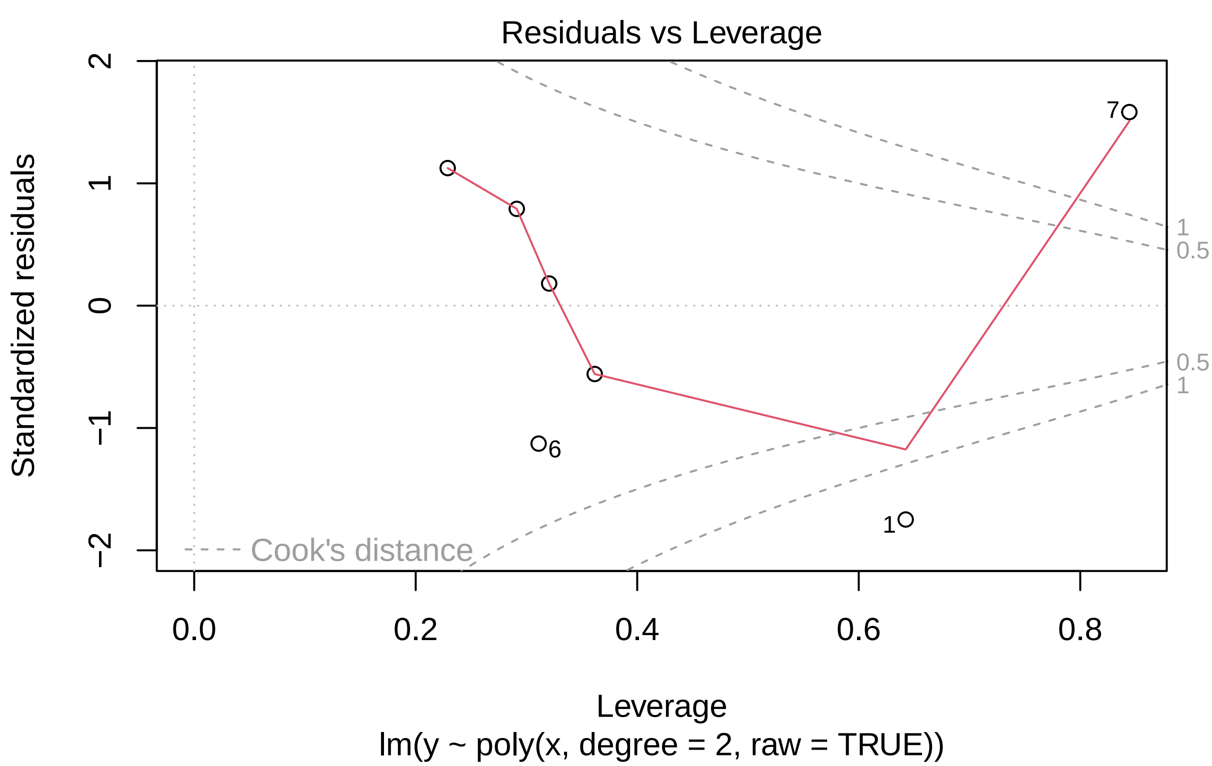
```
##
##  Shapiro-Wilk normality test
##
## data:  .
## W = 0.9058, p-value = 0.3676
```

```r
resid(lm_cube) %>% shapiro.test() %>% print()
```

```
##
##  Shapiro-Wilk normality test
##
## data:  .
## W = 0.96908, p-value = 0.8918
```

Istnieje co prawda problem z obserwacjami wpływowymi ale w tak małej próbie ciężko tego uniknąć:

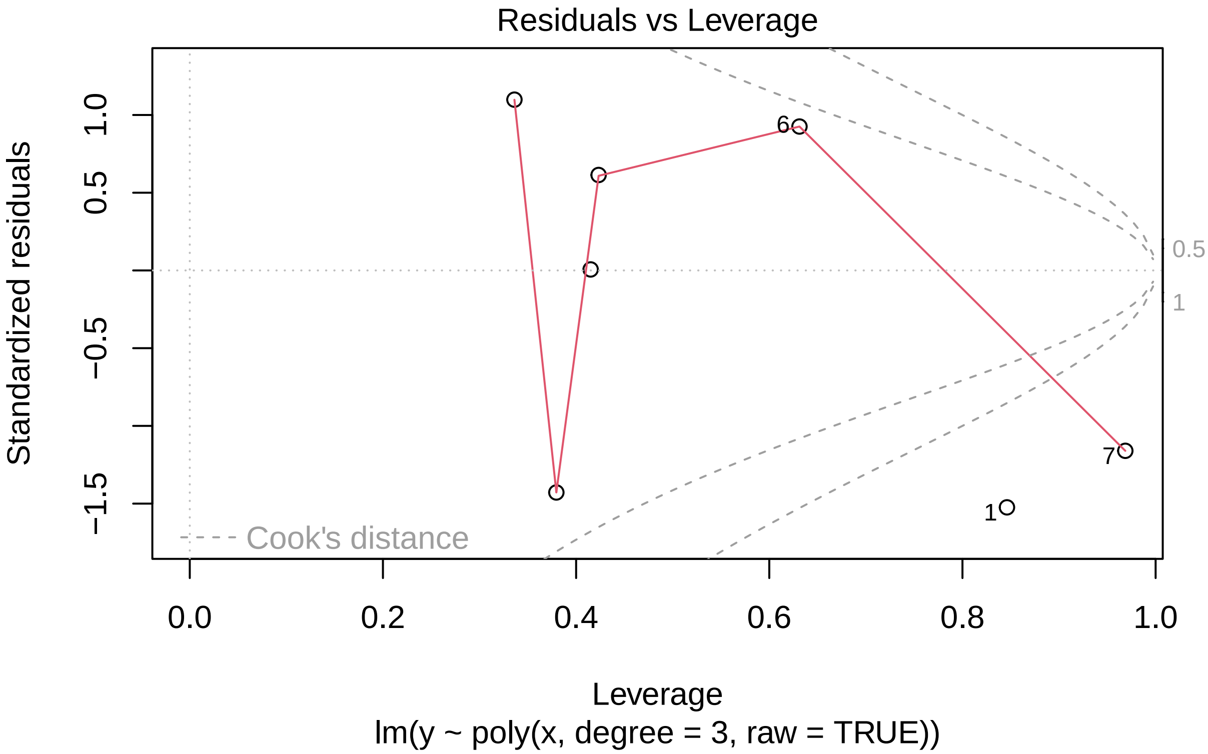
```r
plot(lm_square, which = 5)
```



```r
plot(lm_cube, which = 5)
```

```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced

## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



c) Kryterium `BIC` sugeruje, że model sześcienny jest lepszy od modelu kwadratowego (podobnie jak kryterium `AIC`):

```r
BIC(lm_square, lm_cube)
```

```
##           df      BIC
## lm_square  4 60.47123
## lm_cube    5 42.86487
```

```r
AIC(lm_square, lm_cube)
```

```
##           df      AIC
## lm_square  4 60.68759
## lm_cube    5 43.13532
```

```r
df %>%
  ggplot(aes(y = y, x = x)) +
  geom_point() +
  geom_smooth(formula = y ~ 1 + x + I(x ^ 2),
              method = "lm",
              se = FALSE,
              aes(col = "Kwadratowa"),
              linewidth = 1.1) +
  geom_smooth(formula = y ~ 1 + x + I(x ^ 2) + I(x ^ 3),
```

```
              method = "lm",
              se = FALSE,
              aes(col = "Sześcienna"),
              linewidth = 1.1) +
  scale_color_manual(values = c("Kwadratowa" = "salmon",
                                "Sześcienna" = "lightgreen")) +
  labs(color = "Model Regresji",
       x = "Wysokość [m]",
       y = "Zasięg [m]") +
  ggtitle("Regresja kwadratowa i sześcienna") +
  theme_bw()
```

d)

e) Prognoza:

```
predict(lm_cube, newdata = data.frame(x = 1100))
```

```
##        1
## 606.9814
```