

1 Wprowadzenie do R

1.1 Skróty klawiszowe programu RStudio

- **CTRL+SHIFT+n** - tworzy nowy plik źródłowy
- **CTRL+ENTER** - przekazuje kod z edytora do konsoli R
- **CTRL+1** i **CTRL+2** - przenoszą karetkę między edytorem a konsolą
- **CTRL+F11** i **CTRL+F12** - przenoszą karetkę między otwartymi skryptami

1.2 System pomocy

```
?mean
```

```
help(mean)
```

1.3 Pakiety

- Pakiet to zestaw narzędzi, takich jak nowe funkcje wraz z dokumentacją oraz nowe zbiory danych, rozszerzających funkcjonalność programu R. Większość z nich znajduje się w repozytorium CRAN (*Comprehensive R Archive Network*).
- **install.packages(nazwa pakietu, dependencies = TRUE)** - instalacja pakietu
- **library(nazwa pakietu)** - ładowanie pakietu
- **detach(package:nazwa pakietu)** - usunięcie pakietu

```
install.packages("car")
library(car)
detach(package:car)
```

1.4 Wektory atomowe

1.4.1 Wektory wartości logicznych

- W R zdefiniowane są dwie stałe logiczne:
 - TRUE - prawda,
 - FALSE - fałsz.

```
FALSE
```

```
## [1] FALSE
```

- Wektory można tworzyć przez łączenie. Wektor (ciąg) składający się z konkretnych wartości logicznych w określonej kolejności, można utworzyć za pomocą funkcji `c()` (od ang. `combine` - łącz).

```
c(TRUE, TRUE, FALSE, FALSE, TRUE)
```

```
## [1] TRUE TRUE FALSE FALSE TRUE
```

```
c(c(TRUE, TRUE, FALSE), c(FALSE, TRUE))
```

```
## [1] TRUE TRUE FALSE FALSE TRUE
```

- Długość wektora zwraca funkcja `length()`.

```
length(c(TRUE, TRUE, FALSE, FALSE, TRUE))
```

```
## [1] 5
```

1.4.2 Wektory liczbowe

```
c(1, +2, -3, 2.3, -.4, 5.)
```

```
## [1] 1.0 2.0 -3.0 2.3 -0.4 5.0
```

- Do generowania ciągów arytmetycznych w R służą:
 - operator `:` (różnica równa się 1 lub -1),
 - funkcja `seq()` (od ang. **s**equence, dowolne różnice).

```
c(-3:2, 4:0)
```

```
## [1] -3 -2 -1 0 1 2 4 3 2 1 0
```

```
seq(1, 8, by = 2)
```

```
## [1] 1 3 5 7
```

```
seq(1, 8, length.out = 6)
```

```
## [1] 1.0 2.4 3.8 5.2 6.6 8.0
```

1.4.3 Wektory napisów

- Ciągi dowolnych znaków drukowanych, zwane napisami, tworzymy wykorzystując apostrofy lub cudzysłów.

```
c("DSTA LIO", "informatyka", "", "statystyka", "!")
```

```
## [1] "DSTA LIO"      "informatyka"  ""           "statystyka"  "!"
```

```
length(c("DSTA LIO", "informatyka", "", "statystyka", "!"))
```

```
## [1] 5
```

1.4.4 Nazywanie obiektów

- W R obiekty nazywamy za pomocą jednego z następujących operatorów przypisania (ang. assignment operator):

- =
- <- (w RStudio skrót klawiszowy ALT+-)
- ->

```
x = 5
5 = x
## Error in 5 = x : invalid (do_set) left-hand side to assignment
y <- 6
6 -> y
x
## [1] 5
y
## [1] 6
```

- Wielu użytkowników programu R nie zaleca stosowania operatora = , ponieważ ma on również inne znaczenia, np. używa się go do ustalania wartości funkcji.
- Lepiej nie używać (poza ewentualnie komentarzami) polskich znaków diakrytycznych.
- Polecenie ls() podaje wszystkie aktualnie istniejące obiekty.
- Usunąć jakiś obiekt możemy za pomocą funkcji rm() .
- Wszystkie obiekty usuwamy poleceniem rm(list = ls()) .

```
x <- 1:2
y <- list(1, 2)
ls()
```

```
## [1] "x" "y"
```

```
rm(x)
ls()
```

```

## [1] "y"

rm(list = ls())
ls()

## character(0)

```

1.5 Operatory arytmetyczne

- Do działania na wektorach liczbowych (czasem również zespolonych) można używać następujących binarnych operatorów arytmetycznych:
 - + (dodawanie),
 - - (odejmowanie),
 - * (mnożenie),
 - / (dzielenie rzeczywiste),
 - ^ (potęgowanie),
 - %% (reszta z dzielenie (modulo)),
 - %/% (dzielenie całkowite (bez reszty)).
- Operatory arytmetyczne są zwektoryzowane (ang. vectorized), tzn. dla wektorów $\mathbf{x} = (x_1, x_2, \dots, x_n)$ i $\mathbf{y} = (y_1, y_2, \dots, y_n)$ o tej samej długości n w wyniku działania $\mathbf{x} \diamond \mathbf{y}$ uzyskujemy wektor

$$\mathbf{w} = (x_1 \diamond y_1, x_2 \diamond y_2, \dots, x_n \diamond y_n).$$

Czyli operacje tego typu wykonywane są element po elemencie (ang. elementwise). Unikamy w tej sposób „jawnej” pętli (pętla jest „ukryta” w kodzie operatora), co może pozwolić na przyśpieszenie obliczeń.

```

7 %% 3

## [1] 1

1:3 + c(3, 4, 5)

```

```
## [1] 4 6 8
```

- W przypadku, gdy wektory będące argumentami operatorów binarnych są różnej długości, stosowana jest tak zwana reguła zawijania (ang. recycling rule). Powiela ona niejako krótszy wektor tak, aby uzgodnić jego długość dłuższym wektorem. Niech $\mathbf{x} = (x_1, x_2, \dots, x_n)$ i $\mathbf{y} = (y_1, y_2, \dots, y_m)$, gdzie bez straty ogólności $m \geq n$. Wtedy wynikiem działania jest m -elementowy wektor postaci (dla odpowiedniego l)

$$\mathbf{x} \diamond \mathbf{y} = (x_1 \diamond y_1, \dots, x_n \diamond y_n, x_1 \diamond y_{n+1}, x_2 \diamond y_{n+2}, \dots, x_l \diamond y_m).$$

```
x <- c(1, 3, 5, 8, 1, 3, 0, 6)
x * c(1, 3)
```

```
## [1] 1 9 5 24 1 9 0 18
```

```
x <- c(1, 3, 5, 8, 1, 3, 0)
x * c(1, 3)
```

```
## Warning in x * c(1, 3): długość dłuższego obiektu nie jest wielokrotnością
## długości krótszego obiektu
```

```
## [1] 1 9 5 24 1 9 0
```

1.6 Operatory logiczne i relacyjne

- Rozważamy następujące operatory i funkcje logiczne:
 - $\neg x$ (negacja),
 - $x \mid y$ (alternatywa),
 - $x \& y$ (koniunkcja).
- Do porównywania wektorów służą następujące operatory relacyjne:
 - $x < y$ (czy mniejsze?),
 - $x > y$ (czy większe?),
 - $x \leq y$ (czy nie większy?),

- $x \geq y$ (czy nie mniejszy?),
 - $x == y$ (czy równy?),
 - $x != y$ (czy nierówny?).
- Można je stosować na wektorach dowolnych typów. Jednak wynikiem ich działania jest zawsze wektor logiczny.

```
(1:7) == (7:1)
```

```
## [1] FALSE FALSE FALSE TRUE FALSE FALSE FALSE
```

```
c(TRUE, FALSE) < 1
```

```
## [1] FALSE TRUE
```

1.7 Indeksowanie wektorów

- Wartości elementów każdego wektora leżą na ścisłe określonych pozycjach oznaczonych kolejnymi liczbami naturalnymi (`1:length(x)`).
- Do elementów wektora odwołujemy się poprzez nawiasy kwadratowe `[]`.

```
x <- 1:5
```

```
x[2]
```

```
## [1] 2
```

```
x[2:4]
```

```
## [1] 2 3 4
```

```
x[-2]
```

```
## [1] 1 3 4 5

x[-(2:4)]

## [1] 1 5

# x[c(1, -2)]
## Error in x[c(1, -2)] : only 0's may be mixed with negative subscripts
x[c(TRUE, TRUE, FALSE, TRUE, FALSE)]

## [1] 1 2 4
```

```
x[x < 4]
```

```
## [1] 1 2 3
```

- Nawiasów kwadratowych możemy również użyć do zmiany elementów danego wektora.

```
x[2] <- 6
x
```

```
## [1] 1 6 3 4 5
```

```
x[c(2, 4)] <- c(4, 2)
x
```

```
## [1] 1 4 3 2 5
```

```
x[c(2, 4)] <- 6
```

```
x
```

```
## [1] 1 6 3 6 5
```

1.8 Wybrane funkcje wbudowane

- Program R zawiera wiele zwektryzowanych funkcji matematycznych dla wektorów liczbowych lub czasem zespolonych. W wyniku ich działania uzyskujemy wektor tej samej długości co wektor wejściowy, którego wartości są wyznaczane przez przekształcenie każdego elementu daną funkcją.
- `abs()` - wartość bezwzględna, `sign()` - znak liczby, `floor()` - funkcja „podłoga”, `ceiling()` - funkcja „sufit”, `trunc()` - obcięcie części ułamkowej liczby ($\lfloor x \rfloor$ dla $x \geq 0$, $\lceil x \rceil$ dla $x < 0$), `round(x, digits = 0)` - zaokrąglenie x do $digits$ miejsc po kropce dziesiętnej, `sqrt()` - pierwiastek kwadratowy, `exp()` - funkcja wykładnicza, `log(x, base = exp(1))` - logarytm o podstawie `base`, `gamma()` - funkcja gamma, `beta(a, y)` - funkcja beta, `choose(n, k)` - współczynnik dwumianowy, `sin()`, `cos()`, `tan()` - funkcje trygonometryczne, `asin()`, `acos()`, `atan()` - funkcje cyklometryczne, `Conj(z)` - liczba sprzężona do z , `Re(z)` - część rzeczywista z , `Im(z)` - część urojona z , `Mod(z)` - moduł z , `Arg(z)` - argument z .

```
abs(-2:2)
```

```
## [1] 2 1 0 1 2
```

```
ceiling(c(-1.99, -0.1, 0.1, 1.99))
```

```
## [1] -1  0  1  2
```

1.9 Zadania

Zadanie 1. Otwórz program RStudio. Następnie utwórz nowy skrypt i zapisz go jako, na przykład, `wprowadzenie_do_R_zadania.R`. W tym skrypcie możesz napisać rozwiązania następujących zadań.

Zadanie 2. Użyj funkcji `rep()`, aby utworzyć wektor logiczny, zaczynając od trzech wartości prawda, następnie czterech wartości fałsz, po których następują dwie wartości prawda i wreszcie pięć wartości fałsz. Przypisz ten wektor logiczny do zmiennej `x`. Na koniec przekonwertuj ten wektor na wektor numeryczny. Jak zmieniły się wartości prawda i fałsz?

```
## [1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE TRUE FALSE FALSE  
## [12] FALSE FALSE FALSE  
  
## [1] 1 1 1 0 0 0 0 1 1 0 0 0 0 0 0
```

Zadanie 3. Palindromem nazywamy wektor, którego elementy czytane od końca tworzą ten sam wektor co elementy czytane od początku. Utwórz taki wektor 100 liczb przy czym pierwsze 20 liczb to kolejne liczby naturalne, następnie występuje 10 zer, następnie 20 kolejnych liczb parzystych, a pozostałe elementy określone są przez palindromicznosć (warunek symetrii).

```
## [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 0 0 0  
## [24] 0 0 0 0 0 0 0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32  
## [47] 34 36 38 40 40 38 36 34 32 30 28 26 24 22 20 18 16 14 12 10 8 6 4  
## [70] 2 0 0 0 0 0 0 0 0 0 20 19 18 17 16 15 14 13 12 11 10 9  
## [93] 8 7 6 5 4 3 2 1
```

Zadanie 4. Z wektora `letters` wybierz litery na pozycjach 5, 10, 15, 20, 25.

```
## [1] "e" "j" "o" "t" "y"
```

Zadanie 5. Utwórz wektor liczb naturalnych od 1 do 1000, a następnie zamień liczby parzyste na ich odwrotności.

```
## [1] 1 0.5 3 0.25 5 0.1666667 ...
```

Zadanie 6. Uporządkuj elementy wektora `(6, 3, 4, 5, 2, 3)` od największego do najmniejszego wykorzystując funkcję `order()`.

```
## [1] 6 5 4 3 3 2
```

Zadanie 7. Wyznacz znaki elementów wektora $(-1,876; -1,123; -0,123; 0; 0,123; 1,123; 1,876)$. Następnie zaokrągl j i elementy tego wektora do dwóch miejsc po przecinku. Na koniec wyznacz część całkowitą każdego elementu nowego wektora.

```
## [1] -1 -1 -1  0  1  1  1
```

```
## [1] -1.88 -1.12 -0.12  0.00  0.12  1.12  1.88
```

```
## [1] -2 -2 -1  0  0  1  1
```

Zadanie 8. Wyznacz pierwiastek kwadratowy z każdej liczby naturalnej od 1 do 100 milionów. Najpierw wykonaj to polecenie korzystając z odpowiedniej funkcji wbudowanej w R, a następnie wykorzystując potęgowanie. Który sposób działa szybciej? **Wskazówka:** Do badania długości czasu działania programu można wykorzystać funkcję `system.time()`.

```
## Time difference of 1.485525 secs
## Time difference of 9.706759 secs
## [1] 1 1.414214 1.732051 2 2.236068 2.44949 ...
```

Zadanie 9. W pakiecie `schoolmath` znajduje się zbiór danych `primlist`, który zawiera liczby pierwsze pomiędzy 1 a 9999999.

- Znajdź największą liczbę pierwszą mniejszą od 1000.
- Ile jest liczb pierwszych większych od 100 a mniejszych od 500?

```
## [1] 997
```

```
## [1] 73
```

Zadanie 10. Wyznacz wszystkie kombinacje wartości wektorów (a, b) i $(1, 2, 3)$ za pomocą funkcji `rep()` i `paste()`.

```
## [1] "a1" "a2" "a3" "b1" "b2" "b3"
```

Zadanie 11. Utwórz wektor 30 napisów następującej postaci: `liczba.litera`, gdzie liczba to kolejne liczby naturalne od 1 do 30 a litera to trzy wielkie litery X, Y, Z występujące cyklicznie.

```
## [1] "1.X"  "2.Y"  "3.Z"  "4.X"  "5.Y"  "6.Z"  "7.X"  "8.Y"  "9.Z"  "10.X"
## [11] "11.Y" "12.Z" "13.X" "14.Y" "15.Z" "16.X" "17.Y" "18.Z" "19.X" "20.Y"
## [21] "21.Z" "22.X" "23.Y" "24.Z" "25.X" "26.Y" "27.Z" "28.X" "29.Y" "30.Z"
```

Zadanie 12. W pewnych sytuacjach przydatna może się okazać tzw. kategoryzacja zmiennych, czyli inny podział na kategorie niżby wynikał z danych. Wygeneruj 100 obserwacji, które są odpowiedziami na pytania ankiety, każda odpowiedź może przyjąć jedną z wartości: 'a', 'b', 'c', 'd', 'e'. Dokonaj kategoryzacji w taki sposób, aby kategoria 1 obejmowała odpowiedzi 'a' i 'b', 2 odpowiedzi 'c' i 'd' oraz 3 odpowiedź 'e'. **Wskazówka:** Wykorzystaj funkcję `sample()` oraz funkcję `recode()` z pakietu `car`.

```
## Loading required package: carData
```

```
## [1] "d" "b" "e" "d" "a" "e" "d" "b" "b" "d" "d" "d" "e" "d" "c" "d" "e"
## [18] "b" "e" "b" "c" "d" "d" "c" "a" "c" "d" "b" "c" "b" "e" "a" "c" "a"
## [35] "e" "a" "a" "b" "a" "c" "b" "c" "a" "c" "a" "b" "e" "a" "c" "c" "b"
## [52] "e" "b" "d" "d" "a" "e" "c" "e" "c" "d" "d" "a" "d" "d" "c" "a" "d"
## [69] "a" "b" "e" "e" "a" "b" "b" "b" "e" "c" "d" "d" "c" "b" "d" "e"
## [86] "b" "a" "c" "c" "a" "e" "a" "e" "a" "b" "c" "c" "e" "d" "c"

## [1] 2 1 3 2 1 3 2 1 1 2 2 2 3 2 2 2 3 1 3 1 2 2 2 2 1 2 2 1 2 1 3 1 2 1 3
## [36] 1 1 1 1 2 1 2 1 1 3 1 2 2 1 3 1 2 2 1 3 2 3 2 2 2 1 2 2 2 1 2 1 1
## [71] 3 3 3 1 1 1 1 3 2 2 2 2 1 2 3 1 1 2 2 1 3 1 3 1 1 2 2 3 2 2
```

2 Wprowadzenie do R cd.

2.1 Listy

- Kolejnym podstawowym typem danych jest lista. Najlepiej postrzegać ją jako ciąg złożony z elementów o dowolnych typach (a więc już niekoniecznie tych samych jak w przypadku wektorów atomowych). W skład listy mogą wchodzić wektory logiczne, liczbowe i napisów, a nawet funkcje, czy też same listy.
- Listy tworzymy zazwyczaj za pomocą funkcji `list()`.

```
(x <- list(TRUE, 3.5, "DSTA"))
```

```
## [[1]]  
## [1] TRUE  
##  
## [[2]]  
## [1] 3.5  
##  
## [[3]]  
## [1] "DSTA"
```

```
(x <- list(logiczna = TRUE, liczba = 3.5, napis = "DSTA"))
```

```
## $logiczna  
## [1] TRUE  
##  
## $liczba  
## [1] 3.5  
##  
## $napis  
## [1] "DSTA"
```

```

x[[1]]

## [1] TRUE

x$logiczna

## [1] TRUE

str(x)

## List of 3
## $ logiczna: logi TRUE
## $ liczba : num 3.5
## $ napis : chr "DSTA"

x[1] <- NULL
str(x)

## List of 2
## $ liczba: num 3.5
## $ napis : chr "DSTA"

```

2.2 Macierze

- Macierz (typ złożony `matrix`) i, ogólniej, tablice (typ złożony `array`) są reprezentowane w R przez wektory atomowe. Mają one jednak ustawiony atrybut specjalny `dim`. Jego wartość może mieć on przypisany jedynie wektor liczb całkowitych dodatnich o długości nie mniejszej niż dwa.
- Wykorzystując atrybut `dim`, macierz możemy utworzyć „ręcznie”.
- Macierze zazwyczaj łatwiej utworzyć korzystając z funkcji `matrix()`.

```
(x <- matrix(1:8, nrow = 2, ncol = 4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]     1     3     5     7
## [2,]     2     4     6     8
```

```
dim(x)
```

```
## [1] 2 4
```

```
nrow(x)
```

```
## [1] 2
```

```
ncol(x)
```

```
## [1] 4
```

```
(x <- matrix(1:8, nrow = 2))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]     1     3     5     7
## [2,]     2     4     6     8
```

```
(x <- matrix(1:8, ncol = 4))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]     1     3     5     7
## [2,]     2     4     6     8
```

```
(x <- matrix(1:8, ncol = 4, byrow = TRUE))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    4
## [2,]    5    6    7    8
```

- Skoro macierze są wektorami atomowymi, do ich indeksowania możemy użyć nawiasów kwadratowych `[]`. Jednak możemy korzystać z nich na „większą” liczbę sposobów.

```
(x <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
x[2, 3]
```

```
## [1] 6
```

```
x[2, ]
```

```
## [1] 2 4 6
```

```
x[, 3]
```

```
## [1] 5 6
```

```
x[, 2:3]
```

```
##      [,1] [,2]
## [1,]    3    5
## [2,]    4    6
```

```
x[1:2, c(1, 3)]
```

```
##      [,1] [,2]
## [1,]    1    5
## [2,]    2    6
```

- Oczywiście, w R zaimplementowanych jest wiele funkcji wykonujących operacje specyficzne dla macierzy.

```
t(matrix(1:6, ncol = 3))
```

```
##      [,1] [,2]
## [1,]    1    2
## [2,]    3    4
## [3,]    5    6
```

```
(A <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
(B <- matrix(7:12, ncol = 2))
```

```
##      [,1] [,2]
## [1,]    7   10
## [2,]    8   11
## [3,]    9   12
```

```
A %*% B
```

```
##      [,1] [,2]
## [1,]    76   103
## [2,]   100   136
```

```
B %*% A
```

```
##      [,1] [,2] [,3]
## [1,]    27   61   95
## [2,]    30   68  106
## [3,]    33   75  117
```

```
(A <- matrix(1:6, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]    1     3     5
## [2,]    2     4     6
```

```
(B <- matrix(7:12, ncol = 3))
```

```
##      [,1] [,2] [,3]
## [1,]    7     9    11
## [2,]    8    10    12
```

```
A * B
```

```
##      [,1] [,2] [,3]
## [1,]    7    27    55
## [2,]   16    40    72
```

- `rankMatrix()` z pakietu `Matrix` - rząd macierzy
- `det()` - wyznacznik macierzy
- `kronecker(A, B)` - iloczyn Kroneckera macierzy
- `solve(A, b)` - rozwiązuje układy równań liniowych, jako pierwszy parametr podajemy macierz współczynników, a jako drugi wektor wyrazów wolnych. Jeśli nie podamy drugiego parametru funkcja obliczy macierz odwrotną.
- `ginv()` z pakietu `MASS` - pseudoodwrotność Moore'a-Penrose'a
- `eigen()` - wartości oraz wektory własne (rozkład spektralny macierzy symetrycznej)

2.3 Czynniki

- Czynnik (ang. factors) można postrzegać jako wektory zawierające elementy ze zbioru o z góry określonej, najczęściej względnie niewielkiej liczbie możliwych wartości. Zatem czynniki służą do reprezentowania danych jakościowych.
- W praktyce analizy danych, zmienne typu czynnikowego zazwyczaj kodują informację o zmiennych niemierzalnych, takich jak płeć, kolor oczu czy wykształcenie. Można je oczywiście zakodować za pomocą liczb.

```
(plec <- rep(c("F", "M"), c(2, 3)))
```

```
## [1] "F" "F" "M" "M" "M"
```

```
(plec_factor <- factor(plec))
```

```
## [1] F F M M M
```

```
## Levels: F M
```

```
levels(plec_factor)
```

```
## [1] "F" "M"
```

```
nlevels(plec_factor)
```

```
## [1] 2


```

```
## plec_factor
## F M
## 2 3
```

2.4 Ramki danych

- Ramki danych (ang. data frames) to obiekty przechowujące informacje w postaci macierzowej, najczęściej takie, które są np. wynikiem eksperymentów (także numerycznych). Wiersze ramki danych odpowiadają reprezentowanym obiektom, tzw. obserwacjom (ang. observations), bądź przypadkom (ang. cases), np. badanym osobom. Kolumny z kolei podają informacje na temat wartości różnych zmiennych (ang. variables) opisujących ich wybrane własności (mierzalne lub nie).
- W R, ramki danych są reprezentowane przez listy zawierające wektory atomowe o tej samej długości. Każdy element tej szczególnej listy odpowiada kolumnie ramki danych.

```
ramka <- data.frame(
  plec = c("K", "K", "M", "M", "K"),
  wyksztalcenie = c("s", "w", "w", "p", "s"),
  waga = c(60, 55, 80, 75, 62)
)
ramka
```

```
##   plec wyksztalcenie waga
## 1    K                 s   60
## 2    K                 w   55
## 3    M                 w   80
## 4    M                 p   75
## 5    K                 s   62
```

```
nrow(ramka)

## [1] 5

ncol(ramka)

## [1] 3

rownames(ramka)

## [1] "1" "2" "3" "4" "5"

colnames(ramka)

## [1] "plec"          "wyksztalcenie" "waga"

ramka[[3]] # Lub ramka$waga Lub ramka[, 3]

## [1] 60 55 80 75 62

ramka$waga <- c(58, 54, 78, 72, 60)
ramka[ramka$plec == "M", ]

##   plec wyksztalcenie waga
## 3     M             w    78
## 4     M             p    72

rbind(ramka[1:2, ], ramka[1:2, ])
```

```

##    plec wyksztalcenie waga
## 1   K           s   58
## 2   K           w   54
## 3   K           s   58
## 4   K           w   54

cbind(ramka[1:2, ], wyksztalcenie_2 = as.integer(ramka$wyksztalcenie[1:2]))
```

```

##    plec wyksztalcenie waga wyksztalcenie_2
## 1   K           s   58           2
## 2   K           w   54           3
```

2.5 Odczytywanie i zapisywanie danych

- `read.table()` , `load()` , `read.csv()` , `read.csv2()` - wczytanie zbioru danych, odpowiednio z pliku tekstopowego, pliku w formacie programu R (z rozszerzeniem `RData`), pliku `csv` , odpowiednio
- `write.table()` , `save()` , `write.csv()` , `write.csv2()` - zapis zbioru danych, odpowiednio do pliku tekstopowego, pliku w formacie programu R (z rozszerzeniem `RData`), plików `csv` , odpowiednio
- Przy odczytywaniu i zapisywaniu danych, wygodnie jest najpierw ustalić katalog bieżący na ten, w którym znajdują się lub mają znaleźć się pliki z danymi. Aktualny katalog bieżący sprawdzamy za pomocą funkcji `getwd()` , natomiast zmieniamy go używając funkcji `setwd()` .

`getwd()`

```

## [1] "/home/ls/MEGA/DYDAKTYKA/STA/DSTA_LIO/DSTA_LIO_cwiczenia_bookdown"

# setwd("/home/Ls/MEGA/DYDAKTYKA/STA/DSTA_LIO")
# (odczyt_1 <- read.table("odczyt_1.txt"))
(odczyt_1 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_1.txt"))
```

```

##          V1      V2      V3
## 1 zmienna1 zmienna2 zmienna3
## 2      1.2      1.3      1.4
## 3      2.1      2.2      2.3
## 4      3.1      3.2      3.3

(odczyt_1 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_1.txt",
                         header = TRUE))

##   zmienna1 zmienna2 zmienna3
## 1      1.2      1.3      1.4
## 2      2.1      2.2      2.3
## 3      3.1      3.2      3.3

(odczyt_2 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_2.txt",
                         header = TRUE))

##   zmienna1.zmienna2.zmienna3
## 1      1,2;1,3;1,4
## 2      2,1;2,2;2,3
## 3      3,1;3,2;3,3

(odczyt_2 <- read.table("http://ls.home.amu.edu.pl/data_sets/odczyt_2.txt",
                         header = TRUE, sep = ";", dec = ","))

##   zmienna1 zmienna2 zmienna3
## 1      1.2      1.3      1.4
## 2      2.1      2.2      2.3
## 3      3.1      3.2      3.3

```

- Pliki z danymi do powyższych przykładów: odczyt_1.txt, odczyt_2.txt

- Można też zimportować dane klikając na Import Dataset w RStudio i w otworzonym okienku ustawić potrzebne parametry.
- Podgląd danych w edytorze kodu źródłowego otrzymujemy za pomocą funkcji view() .
- Zapisywanie danych:

```
dane_1 <- data.frame(1:10, 5:14)

write.table(dane_1, "dane_1.txt")

save(dane_1, file = "dane_1.RData")

dane_1 <- read.table("dane_1.txt")

load("dane_1.RData")
```

2.6 Zadania

Zadanie 1. Skonstruuje listę o nazwie moja_lista , której pierwszym elementem będzie dwuelementowy wektor napisów zawierający Twoje imię i nazwisko, drugim elementem będzie liczba π , trzecim funkcja służąca do obliczania pierwiastka kwadratowego, a ostatni element listy to wektor złożony z liczb 0,02; 0,04; ... ; 1. Następnie usuń elementy numer jeden i trzy z tej listy. Na zakończenie, wyznacz listę zawierającą wartości funkcji gamma Eulera dla elementów listy moja_lista .

```
## List of 4
## $ : chr [1:2] "Łukasz" "Smaga"
## $ : num 3.14
## $ :function (x)
## $ : num [1:50] 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2 ...
## 
## List of 2
## $ : num 3.14
## $ : num [1:50] 0.02 0.04 0.06 0.08 0.1 0.12 0.14 0.16 0.18 0.2 ...
```

```

## [[1]]
## [1] 2.288038
##
## [[2]]
## [1] 49.442210 24.460955 16.145727 11.996566 9.513508 7.863252 6.688686
## [8] 5.811269 5.131821 4.590844 4.150482 3.785504 3.478450 3.216852
## [15] 2.991569 2.795751 2.624163 2.472735 2.338256 2.218160 2.110371
## [22] 2.013193 1.925227 1.845306 1.772454 1.705844 1.644773 1.588641
## [29] 1.536930 1.489192 1.445038 1.404128 1.366164 1.330884 1.298055
## [36] 1.267473 1.238954 1.212335 1.187471 1.164230 1.142494 1.122158
## [43] 1.103124 1.085308 1.068629 1.053016 1.038403 1.024732 1.011947
## [50] 1.000000

```

Zadanie 2. Wyznacz rząd, wyznacznik, odwrotność, wartości własne, wektory własne oraz sumy i średnie arytmetyczne dla kolejnych wierszy i kolumn dla następującej macierzy:

$$\begin{bmatrix} 1 & 5 & 3 \\ 2 & 0 & 5 \\ 1 & 2 & 1 \end{bmatrix}$$

Ponadto, pomnóż tę macierz przez jej odwrotność.

```

## [1] 3
## attr(,"method")
## [1] "tolNorm2"
## attr(,"useGrad")
## [1] FALSE
## attr(,"tol")
## [1] 6.661338e-16

```

```

## [,1]      [,2]      [,3]
## [1,] -0.5882353  0.05882353  1.47058824
## [2,]  0.1764706 -0.11764706  0.05882353
## [3,]  0.2352941  0.17647059 -0.58823529

```

```
## eigen() decomposition  
## $values  
## [1] 6.0790256 -3.2070365 -0.8719891  
##  
## $vectors  
## [,1] [,2] [,3]  
## [1,] -0.7537024 0.7058088 -0.9275678  
## [2,] -0.5472752 -0.6900345 0.1392322  
## [3,] -0.3638991 0.1602696 0.3467453
```

```
## [1] 9 7 4
```

```
## [1] 3.000000 2.333333 1.333333
```

```
## [1] 4 7 9
```

```
## [1] 1.333333 2.333333 3.000000
```

```
## [,1] [,2] [,3]  
## [1,] 1.000000e+00 -1.110223e-16 2.220446e-16  
## [2,] 0.000000e+00 1.000000e+00 0.000000e+00  
## [3,] -2.775558e-17 -2.775558e-17 1.000000e+00
```

Zadanie 3. Utwórz wektor kwadratów 100 pierwszych liczb naturalnych. Następnie zlicz, które cyfry oraz jak często występują na pozycji jedności w kolejnych elementach tego wektora.

```
## [1] 1 4 9 16 25 36 ...
```

```
##  
## 0 1 4 5 6 9  
## 10 20 20 10 20 20
```

Zadanie 4. Za pomocą funkcji `outer()` wyznacz tabliczkę mnożenia dla liczb mniejszych od 6.

```

## [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] "1 * 1 = 1" "1 * 2 = 2" "1 * 3 = 3" "1 * 4 = 4" "1 * 5 = 5"
## [2,] "2 * 1 = 2" "2 * 2 = 4" "2 * 3 = 6" "2 * 4 = 8" "2 * 5 = 10"
## [3,] "3 * 1 = 3" "3 * 2 = 6" "3 * 3 = 9" "3 * 4 = 12" "3 * 5 = 15"
## [4,] "4 * 1 = 4" "4 * 2 = 8" "4 * 3 = 12" "4 * 4 = 16" "4 * 5 = 20"
## [5,] "5 * 1 = 5" "5 * 2 = 10" "5 * 3 = 15" "5 * 4 = 20" "5 * 5 = 25"

```

Zadanie 5. Odczytaj zbiór danych `dane1.csv` a następnie:

1. Z odczytanej ramki danych wyświetl tylko parzyste wiersze.
 2. Korzystając z operatorów logicznych wyświetl tylko wiersze odpowiadające pacjentkom starszym niż 50 lat z przerzutami do węzłów chłonnych (`Wezly.chlonne = 1`).

##		Wiek	Rozmiar.guza	Wezly.chlonne	Nowotwor	Receptory.estrogenowe
##	2	29	1	0	2	(++)
##	4	32	1	0	3	(++)
##	6	33	1	1	3	(-)
##	8	35	2	1	2	(+)
##	10	36	1	1	2	(-)
##	12	37	1	0	3	(-)
##						
##		Receptory.progesteronowe	Niepowodzenia	Okres.bez.wznowy	VEGF	
##	2	(++)		brak		53 1118
##	4	(++)		brak		26 1793
##	6	(++)		wznowa		36 2776
##	8	(++)		brak		38 3827
##	10	(++)		brak		37 834
##	12	(+)		wznowa		40 3331

...

##		Wiek	Rozmiar.guza	Wezly.chlonne	Nowotwor	Receptory.estrogenowe
##	78	51	1	1	2	(++)
##	79	51	1	1	2	(+)
##	81	51	2	1	2	(+++)
##	84	51	2	1	NA	(-)
##	88	52	2	1	2	(+)
##	95	55	1	1	2	(++)
##						
##		Receptory.progesteronowe	Niepowodzenia	Okres.bez.wznowy	VEGF	
##	78	(++)		brak		33 629
##	79	(+)		brak		36 2879
##	81	(++)		brak		52 1098
##	84	(-)		brak		30 8064
##	88	(+)		wznowa		48 1927
##	95	(++)		brak		29 373

...

Zadanie 6. Poniższe dane są średnimi miesięcznymi temperaturami (w °F) w Nowym Jorku.

Styczeń – 32	Kwiecień – 52	Lipiec – 77	Październik – 58
Luty – 33	Maj – 62	Sierpień – 75	Listopad – 47
Marzec – 41	Czerwiec – 72	Wrzesień – 68	Grudzień – 35

1. Wprowadź te dane do ramki danych o jednej zmiennej `NY_F`.
2. Utwórz nową zmienną `NY_C` podającą temperaturę w stopniach Celsiusza (zaokrągloną do dwóch miejsc po przecinku). **Wskazówka:** $(x^{\circ}\text{F}) = (x - 32) \cdot 5/9(\text{ }^{\circ}\text{C})$
3. Zamień nazwy kolumn na `NY_Fahrenheit` i `NY_Celsiusz`.
4. Usuń kolumnę z temperaturą w Fahrenheitach.
5. Zapisz otrzymane dane w pliku `NY_temp.RData`.

```
##          NY_F
## Styczeń      32
## Luty        33
## Marzec      41
## Kwiecień     52
## Maj         62
## Czerwiec    72
## Lipiec       77
## Sierpień     75
## Wrzesień     68
## Październik  58
## Listopad     47
## Grudzień     35
```

```
##          NY_F  NY_C
## Styczeń   32  0.00
## Luty      33  0.56
## Marzec    41  5.00
## Kwiecień   52 11.11
## Maj        62 16.67
## Czerwiec  72 22.22
## Lipiec     77 25.00
## Sierpień   75 23.89
## Wrzesień   68 20.00
## Październik 58 14.44
## Listopad    47  8.33
## Grudzień   35  1.67
```

	NY_Fahrenheit	NY_Celsiusz
## Styczeń	32	0.00
## Luty	33	0.56
## Marzec	41	5.00
## Kwiecień	52	11.11
## Maj	62	16.67
## Czerwiec	72	22.22
## Lipiec	77	25.00
## Sierpień	75	23.89
## Wrzesień	68	20.00
## Październik	58	14.44
## Listopad	47	8.33
## Grudzień	35	1.67

	NY_Celsiusz
## Styczeń	0.00
## Luty	0.56
## Marzec	5.00
## Kwiecień	11.11
## Maj	16.67
## Czerwiec	22.22
## Lipiec	25.00
## Sierpień	23.89
## Wrzesień	20.00
## Październik	14.44
## Listopad	8.33
## Grudzień	1.67

3 Programowanie w R

3.1 Funkcje

- Korzystając z programu R, bardzo szybko odczuwa się potrzebę użycia pewnych fragmentów kodu wielokrotnie, choć być może dla różnych danych.
- Tak jak listy grupują obiekty (być może różnych typów), tak funkcje zbierają określone wyrażenia służące np. do obliczenia pewnych wartości dla zadanych danych.
- Dodatkową zaletą stosowania funkcji jest możliwość dzielenia długiego kodu na łatwiejsze do opanowania części.
- Tworzenie obiektów typu funkcja odbywa się według następującej składni

```
function(lista parametrów) ciało funkcji
```

gdzie `ciało funkcji` jest wyrażeniem do wykonania na obiektach określonych przez listę parametrów .

- Wartość obliczonego wyrażenia jest wynikiem działania funkcji. Takim wynikiem może być jeden i tylko jeden obiekt, np. lista.
- Parametrów może być jednak wiele. `lista parametrów` to ciąg oddzielonych przecinkami elementów postaci:
 - nazwa parametru (pod taką nazwą będzie dostępny w funkcji obiekt przekazany przy wywołaniu),
 - nazwa = wyrażenie (parametr z wartością domyślną),
 - ... - parametr specjalny, który pozwala przekazać dowolną liczbę argumentów w grupie.

```
szescian <- function(x) x^3 # funkcje zazwyczaj się nazywa  
szescian(2)
```

```
## [1] 8
```

```
szescian_2 <- function(x, y) {  
  x3 <- x^3  
  y3 <- y^3  
  return(c(x3, y3))  
}  
szescian_2(2, 3) # Lub szescian_2(x = 2, y = 3)
```

```
## [1] 8 27
```

```
szescian_3 <- function(x = 2, y = 2) {  
  x3 <- x^3  
  y3 <- y^3  
  return(c(x3, y3))  
}  
szescian_3()
```

```
## [1] 8 8
```

```
szescian_3(y = 3)
```

```
## [1] 8 27
```

```
str(lapply(list(1, 2, 3), function(x) x^3))
```

```
## List of 3  
## $ : num 1  
## $ : num 8  
## $ : num 27
```

```

szescian_4 <- function(x) {
  if (!is.numeric(x)) {
    stop("non-numeric argument x")
  }
  x^3
}
szescian_4(-3)
## [1] -27
szescian_4("a")
## Error in szescian_4("a") : non-numeric argument x

```

3.2 Instrukcje warunkowe

- Wyrażenie warunkowe `if` ma następującą składnię:

```
if (warunek) wyrazenieTRUE else wyrazenieFALSE
```

- Przykładowo:

```

if (is.numeric("wyrazenie")) {
  print("wyrazenieTRUE")
} else {
  print("wyrazenieFALSE")
}

```

```
## [1] "wyrazenieFALSE"
```

- W celu agregacji wektorów logicznych, można wykorzystać dwie ważne funkcje agregujące wartości logiczne: `all()` i `any()`, które zwracają wartość `TRUE` wtedy i tylko wtedy, gdy odpowiednio wszystkie lub co najmniej jedna wartość w wektorze równa jest `TRUE`.

```

x <- 1:5
any(x < 2)

```

```
## [1] TRUE

all(x < 2)

## [1] FALSE

all(x == seq(1, 5))

## [1] TRUE
```

3.3 Pętle

- Pętle umożliwiają wielokrotne wykonywanie tego samego wyrażenia (choć zapewne na różnych obiektach). W programie R mamy do dyspozycji pętle:
 - **while**
 - **repeat**
 - **for**
- Składnia pętli **while** jest następująca:

```
while (warunek) wyrazenie
```

- Zadaniem pętli **while** jest obliczanie **wyrażenia** dopóty, dopóki **warunek** jest spełniony.

```
i <- 1
while (i <= 3) {
  print(i)
  i <- i + 1
}
```

```
## [1] 1
## [1] 2
## [1] 3
```

- Aby pętla nie wykonywała się nieskończoną liczbę razy, zazwyczaj warunek będzie konstruowany na danych odczytywanych z pewnego obiektu, który jest modyfikowany za pomocą wyrażenia .
- Może się zdarzyć, że warunek testowy nigdy nie będzie spełniony i wtedy liczba wykonanych obrotów pętli będzie równa zeru.
- Pętla repeat zachowuje się tak jak while z warunkiem testowym na stałe ustawionym na TRUE . Zatem należy zawsze pamiętać o wywołaniu break , o ile chcemy doczekać wyniku.

```
i <- 0
repeat {
  i <- i + 1
  print(i)
  if (i == 3) break
}
```

```
## [1] 1
## [1] 2
## [1] 3
```

- Pętla for jest chyba najczęściej stosowaną pętlą w programie R. Szczególnie nadaje się ona do „przechodzenia” po elementach wektora atomowego lub listy bądź też wykonywania ciągu wyrażeń zadaną liczbę razy. Jej składnia jest następująca:

```
for (nazwa in wektor) wyrazenie
```

- W pętli for każdą kolejną (od pierwszej do ostatniej) wartość wektora związuje my z podaną nazwą i obliczamy wyrażenie . Pętla ta wykonuje się zawsze dokładnie length(wektor) razy, o ile nie użyte zostało wyrażenie break .

```
for (i in 1:3) print(i)
```

```
## [1] 1
## [1] 2
## [1] 3
```

3.4 Zadania

Zadanie 1. Oblicz iloczyn elementów dowolnego wektora `x` za pomocą pętli `while`, `repeat` i `for` (każdej z osobna).

```
# dla  
x <- 1:5  
  
## [1] 120
```

Zadanie 2. Ile liczb postaci $\binom{n}{r}$ jest większych od miliona dla $1 \leq r \leq n \leq 100$?

```
## [1] 4075
```

Zadanie 3. Napisz funkcję, która sprawdza czy wektor jest palindromem.

```
# dla  
x <- c(1, 2, 3, 3, 2, 1)  
  
## [1] TRUE
```

```
# dla  
x <- c(1, 2, 3, 3, 2, 2)  
  
## [1] FALSE
```

Zadanie 4. Napisz funkcję zamieniającą miarę kąta podaną w stopniach na radiany. Sprawdź działanie tej funkcji dla kątów o mierze: 0° , 30° , 45° , 60° , 90° . Następnie przygotuj ramkę danych, w której zebrane będą informacje o wartościach funkcji sinus, cosinus, tangens i cotangens dla kątów o takich miarach.

```
## [1] 0.0000000 0.5235988 0.7853982 1.0471976 1.5707963
```

```

##      sin      cos      tg      ctg
## 1 0.000000 1.000000e+00 0.000000e+00      Inf
## 2 0.500000 8.660254e-01 5.773503e-01 1.732051e+00
## 3 0.7071068 7.071068e-01 1.000000e+00 1.000000e+00
## 4 0.8660254 5.000000e-01 1.732051e+00 5.773503e-01
## 5 1.000000 6.123234e-17 1.633124e+16 6.123234e-17

```

Zadanie 5. Napisz funkcję, której argumentem będzie wektor liczbowy a wynikiem wektor zawierający trzy najmniejsze i trzy największe liczby w tym wektorze. W przypadku argumentu krótszego niż trzy liczby, funkcja ma zwracać komunikat o błędzie z komentarzem „za krótki argument”.

```

# dla
x <- c(2, 6, 1, 5, 7, 3, 4)

## [1] 1 2 3 5 6 7

# dla
x <- c(2, 6)
## Error in command 'extreme_3(x)': za krótki argument

```

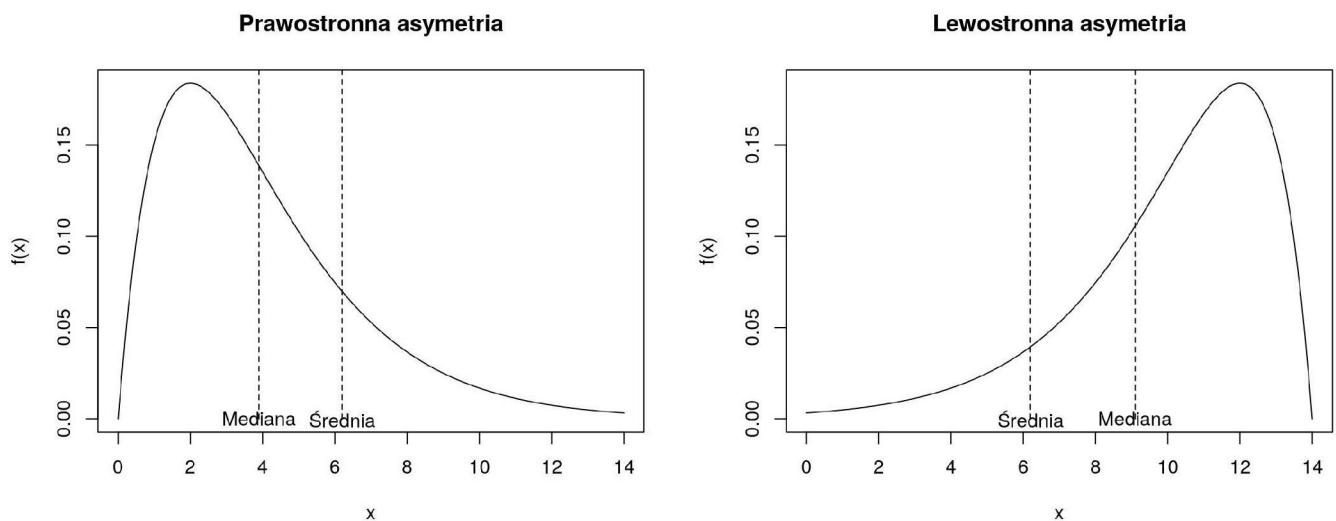
4 Statystyka opisowa

4.1 Miara asymetrii rozkładu

- współczynnik asymetrii (skośności)

$$A = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^3}{s^3}$$

- Współczynnik asymetrii
 - równy zeru oznacza symetrię rozkładu zmiennej.
 - przyjmujący wartość dodatnią oznacza prawostronną asymetrię. Prawy ogon jest dłuższy, a masa rozkładu jest skoncentrowana po lewej stronie.
 - przyjmujący wartość ujemną oznacza lewostronną asymetrię. Lewy ogon jest dłuższy, a masa rozkładu jest skoncentrowana po prawej stronie.

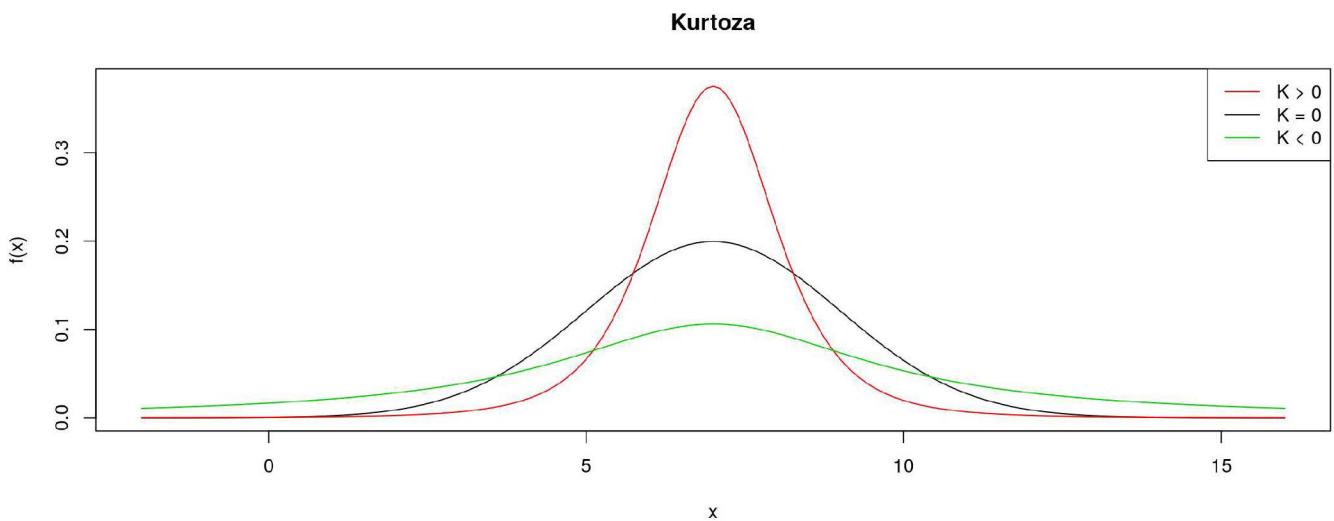


4.2 Miara koncentracji rozkładu

- kurtoza

$$K = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^4}{s^4} - 3$$

- Kurtoza jest miarą skupienia wartości zmiennej wokół średniej.
- Porównuje ona badany rozkład empiryczny z rozkładem normalnym i przyjmuje wartości większe niż -2 .
- Im większa wartość K , tym większe skupienie wartości zmiennej wokół średniej.
- Kurtoza rozkładu normalnego wynosi zero.
- Jeśli $K < 0$, wówczas rozkład jest bardziej spłaszczony niż rozkład normalny, a jeśli $K > 0$ - bardziej smukły.



4.3 Przykłady

Przykład 1. Poniższe dane podają liczbę błędów w grupie 50 osób zdających egzamin testowy. Egzamin składał się z 18 pytań (można popełnić maksymalnie dwa błędy, aby zdać egzamin).

1	1	2	0	1	3	1	4	4	4	0	1	0	0	0	2	3
4	0	1	5	2	3	5	3	2	2	4	0	2	2	0	2	2
3	3	1	3	2	2	0	0	5	4	2	1	5	2	2	0	

Zmienna X to liczba błędów. Jest to dyskretna zmienna ilościowa.

```
liczba_bledow <- c(1, 1, 2, 0, 1, 3, 1, 4, 4, 4, 0, 1, 0, 0, 0, 2, 3,
                    4, 0, 1, 5, 2, 3, 5, 3, 2, 2, 4, 0, 2, 2, 0, 2, 2,
                    3, 3, 1, 3, 2, 2, 0, 0, 5, 4, 2, 1, 5, 2, 2, 0)

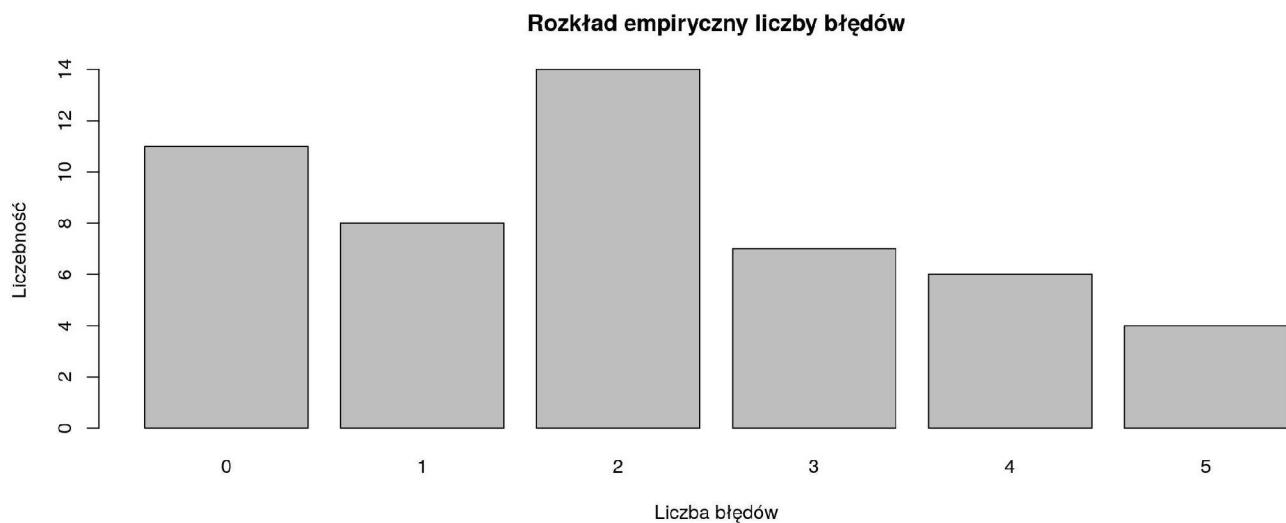
# rozkład empiryczny opisany za pomocą szeregu rozdzielczego
data.frame(cbind(liczebnosc = table(liczba_bledow),
                  procent = prop.table(table(liczba_bledow))))
```

```

##    liczebnosc procent
## 0        11  0.22
## 1         8  0.16
## 2        14  0.28
## 3         7  0.14
## 4         6  0.12
## 5         4  0.08

# wykres słupkowy
barplot(table(liczba_bledow),
        xlab = "Liczba błędów", ylab = "Liczebność",
        main = "Rozkład empiryczny liczby błędów")

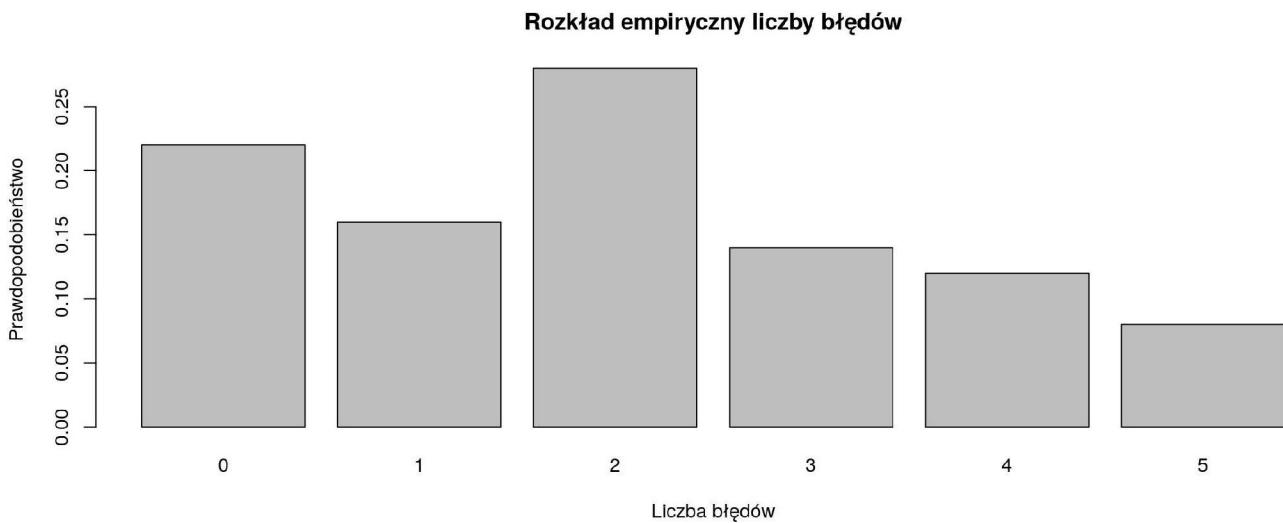
```



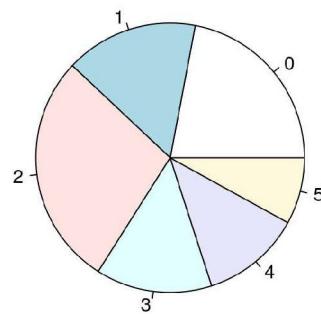
```

barplot(prop.table(table(liczba_bledow)),
        xlab = "Liczba błędów", ylab = "Prawdopodobieństwo",
        main = "Rozkład empiryczny liczby błędów")

```



```
# wykres kołowy  
pie(table(liczba_bledow))
```



```
# średnia  
mean(liczba_bledow)  
  
## [1] 2.02
```

```
# mediana  
median(liczba_bledow)
```

```

## [1] 2

# odchylenie standardowe
sd(liczba_bledow)

## [1] 1.558256

# współczynnik zmienności
sd(liczba_bledow) / mean(liczba_bledow) * 100

## [1] 77.14141

```

Przykład 2. Badano czas oczekiwania na tramwaj, który kursuje w jednakowych odstępach czasu. Plik `czas_oczek_tramwaj.RData` zawiera dane dotyczące czasu oczekiwania na tramwaj (wyrażonego w minutach) 100 osób wybranych losowo. Zmienna X to czas oczekiwania na tramwaj. Jest to zmienna ilościowa ciągła.

```

load(url("http://ls.home.amu.edu.pl/data_sets/czas_oczek_tramwaj.RData"))
head(czas_oczek_tramwaj)

## [1] 4.03 11.04 5.73 12.36 13.17 0.64

data.frame(cbind(liczebnosc = table(cut(czas_oczek_tramwaj, breaks = seq(0, 14, 2))),
procent = prop.table(table(cut(czas_oczek_tramwaj, breaks = seq(0, 14,

```

```
##           liczebnosc procent
## (0,2]            15   0.15
## (2,4]            13   0.13
## (4,6]            15   0.15
## (6,8]            15   0.15
## (8,10]           15   0.15
## (10,12]          12   0.12
## (12,14]          15   0.15

(czas_oczek_tramwaj_hist <- hist(czas_oczek_tramwaj, plot = FALSE)$breaks)

## [1] 0 2 4 6 8 10 12 14

data.frame(cbind(liczebnosc = table(cut(czas_oczek_tramwaj, breaks = czas_oczek_tramwaj,
                                         procent = prop.table(table(cut(czas_oczek_tramwaj, breaks = czas_oczek_tramwaj,
```



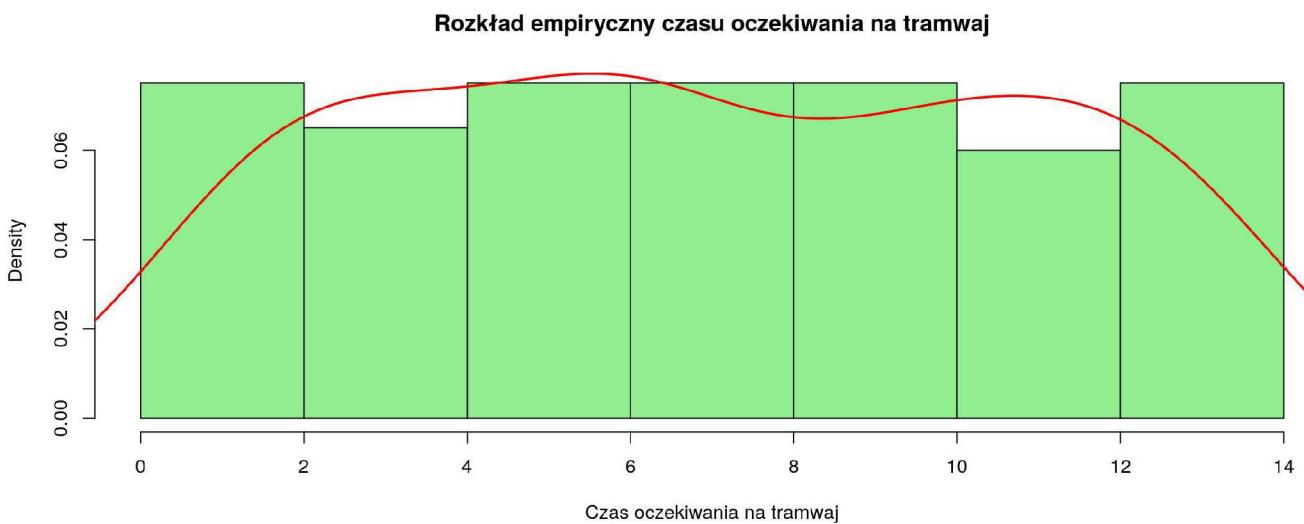
```
##           liczebnosc procent
## (0,2]            15   0.15
## (2,4]            13   0.13
## (4,6]            15   0.15
## (6,8]            15   0.15
## (8,10]           15   0.15
## (10,12]          12   0.12
## (12,14]          15   0.15
```

Histogram - zestaw sąsiadujących prostokątów, których podstawy, równe rozpiętości przedziałów klasowych, znajdują się na osi odciętych, a wysokości są liczebnosciami przedziałów.

```
# histogram
hist(czas_oczek_tramwaj,
      xlab = "Czas oczekiwania na tramwaj",
      main = "Rozkład empiryczny czasu oczekiwania na tramwaj")
rug(jitter(czas_oczek_tramwaj))
```



```
# histogram z estymatorem jądrowym gęstości
hist(czas_oczek_tramwaj,
      xlab = "Czas oczekiwania na tramwaj",
      main = "Rozkład empiryczny czasu oczekiwania na tramwaj",
      probability = TRUE,
      col = "lightgreen")
lines(density(czas_oczek_tramwaj), col = "red", lwd = 2)
```



Wykres ramkowy to metoda graficznego przedstawienia danych liczbowych za pomocą ich kwantyli. Tworzymy go poprzez umieszczenie na osi pionowej wartości niektórych parametrów rozkładu (kwantyli).

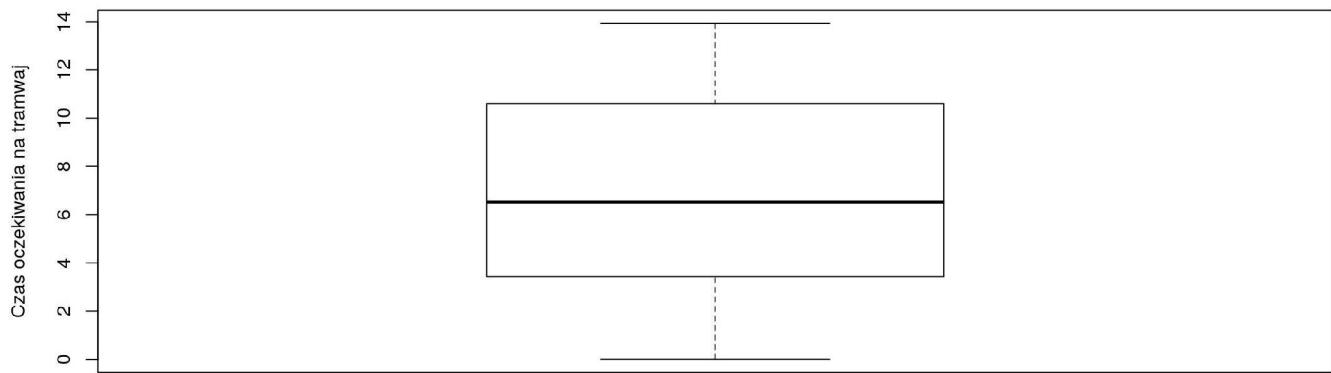
- Wewnątrz prostokąta znajduje się pogrubiona pozioma linia, która określa wartość mediany.
- Nad osią znajduje się prostokąt (ramka), którego dolny bok jest określony przez pierwszy kwartyl, a górny bok przez trzeci kwartyl. Wysokość pudełka odpowiada wartości rozstępu międzykwartylowego ($Q_3 - Q_1$).
- Pudełko jest uzupełnione od góry i od dołu segmentami (wąsami). Dolny koniec dolnego segmentu reprezentuje najmniejszą wartość w zestawie danych, zaś górny koniec górnego segmentu jest obserwacją największą. Wartości te muszą spełniać dodatkowy warunek, a mianowicie dolny koniec nie może być mniejszy niż $Q_1 - 1,5 \cdot (Q_3 - Q_1)$, a górny większy niż $Q_3 + 1,5 \cdot (Q_3 - Q_1)$. Jeśli istnieją obserwacje poza tym zakresem, są one zaznaczane na wykresie indywidualnie jako osobne punkty i są traktowane jako obserwacje odstające.

Wykres pudełkowy jako wskaźnik tendencji centralnej, dyspersji, symetrii, skośności i wielkości ogona:

- dyspersja - odstępy między różnymi częściami pudełka
- symetryczny - pogrubiona linia znajduje się blisko środka pudełka, a długości wąsów są takie same
- prawostronnie asymetryczny - górny wąs jest znacznie dłuższy niż dolny wąs, a linia jest bliższa dolnej części pudełka.
- lewostronne asymetryczny - dolny wąs jest znacznie dłuższy niż górny wąs, a linia jest bliższa górnej części pudełka
- grube ogony - długość wąsów znacznie przekracza długość pudełka
- ciekie ogony - długość wąsów jest krótsza niż długość pudełka
- bardzo krótkie ogony (populacja w kształcie litery U, z zanurzeniem w środku zamiast garbu) - wąsy są nieobecne

```
# wykres ramkowy
boxplot(czas_oczek_tramwaj,
         ylab = "Czas oczekiwania na tramwaj",
         main = "Rozkład empiryczny czasu oczekiwania na tramwaj")
```

Rozkład empiryczny czasu oczekiwania na tramwaj



- statystyki opisowe

```
# średnia
```

```
mean(czas_oczek_tramwaj)
```

```
## [1] 6.9796
```

```
# mediana
```

```
median(czas_oczek_tramwaj)
```

```
## [1] 6.525
```

```
# odchylenie standardowe
```

```
sd(czas_oczek_tramwaj)
```

```
## [1] 3.989571
```

```
# współczynnik zmienności
```

```
sd(czas_oczek_tramwaj) / mean(czas_oczek_tramwaj) * 100
```

```
## [1] 57.16046
```

```

library(e1071)
# współczynnik asymetrii
skewness(czas_oczek_tramwaj)

## [1] 0.03362109

# kurtoza
kurtosis(czas_oczek_tramwaj)

## [1] -1.250899

```

4.4 Zadania

Zadanie 1. Zmienna `wynik` w pliku `ankieta.txt` opisuje wyniki badania działalności prezydenta pewnego miasta. Wybrano losowo 100 mieszkańców miasta i zadano im następujące pytanie: Jak oceniasz działalność prezydenta miasta? Dostępne były następujące odpowiedzi: zdecydowanie dobrze (`a`), dobrze (`b`), źle (`c`), zdecydowanie źle (`d`), nie mam zdania (`e`). Jakiego typu jest ta zmienna? Jaki są możliwe wartości tej zmiennej?

1. Zaimportuj dane z pliku `ankieta.txt` do zmiennej `ankieta`.

```

## plec szkola wynik
## 1   m     p     d
## 2   m     s     e
## 3   m     w     a
## 4   m     s     d
## 5   m     p     c
## 6   m     w     c
## ...

```

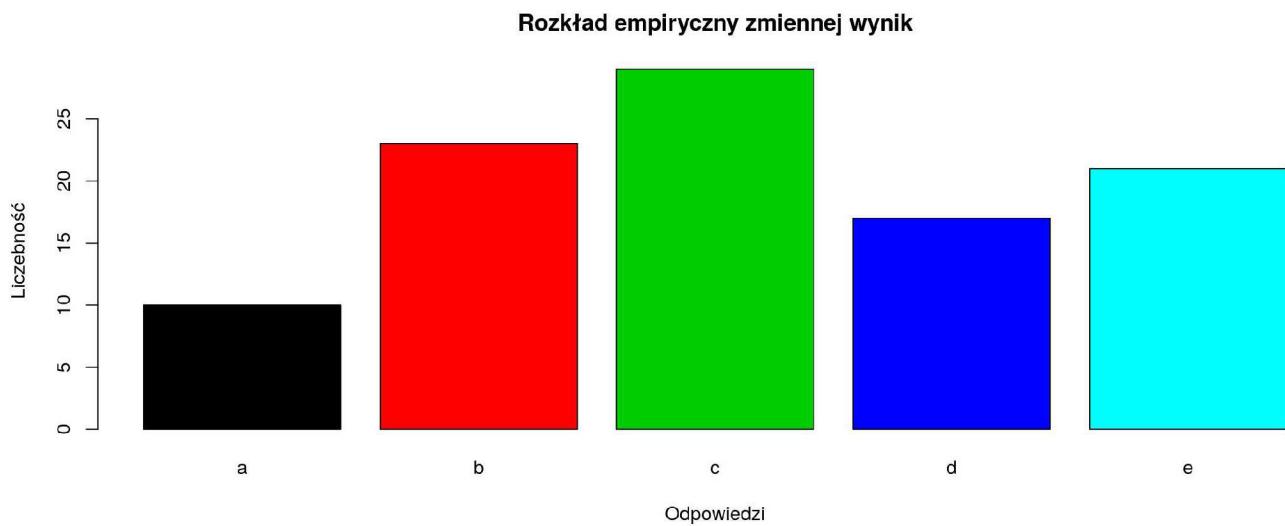
2. Przedstaw rozkład empiryczny zmiennej `wynik` za pomocą szeregu rozdzielczego.

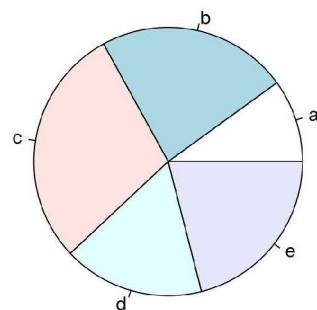
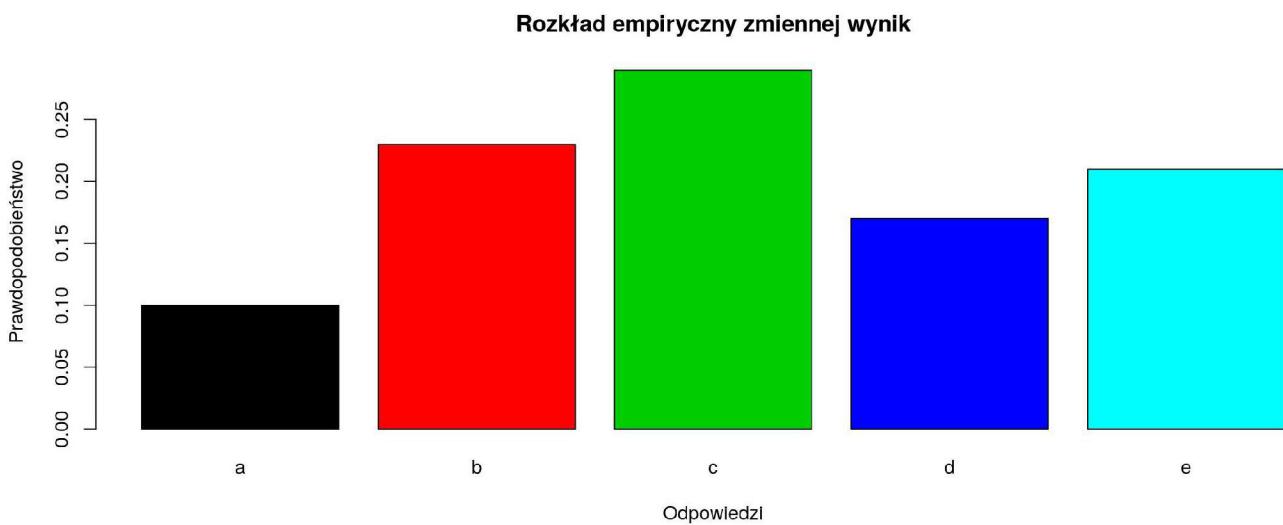
```
##  liczebnosc procent  
## a      10  0.10  
## b      23  0.23  
## c      29  0.29  
## d      17  0.17  
## e      21  0.21
```

3. Przedstaw rozkład empiryczny zmiennej wynik tylko dla osób z wykształceniem podstawowym za pomocą szeregu rozdzielczego.

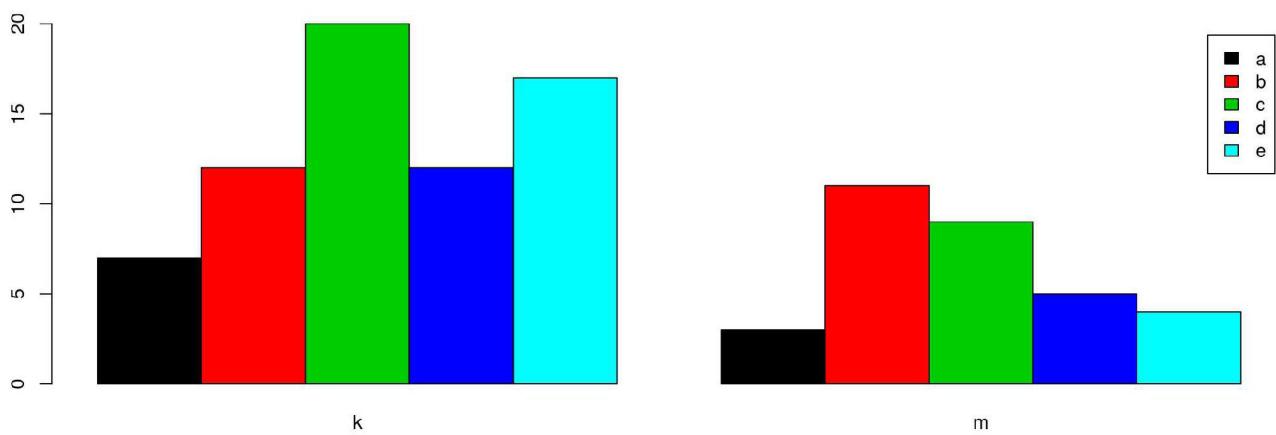
```
##  liczebnosc procent  
## a      2 0.11764706  
## b      3 0.17647059  
## c      4 0.23529412  
## d      7 0.41176471  
## e      1 0.05882353
```

4. Zilustruj wyniki ankiety za pomocą wykresu słupkowego i kołowego.





5. Zilustruj wyniki ankiety za pomocą wykresu słupkowego z podziałem na kobiety i mężczyzn.



6. Zinterpretuj powyższe wyniki (tabelaryczne i graficzne).

Zadanie 2. Przebadano 200 losowo wybranych 5-sekundowych okresów pracy centrali telefonicznej. Rejestrowano liczbę zgłoszeń. Wyniki są zawarte w pliku Centrala.RData. Jakiego typu jest ta zmienna? Jakie są możliwe wartości tej zmiennej?

1. Zaimportuj dane z pliku Centrala.RData.

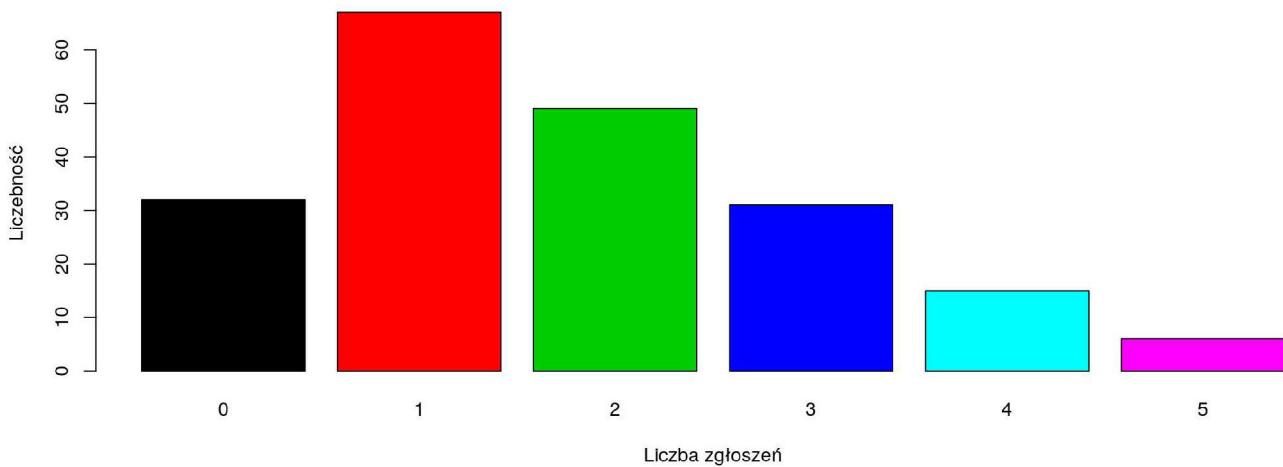
```
##    Liczba  
## 1      0  
## 2      0  
## 3      5  
## 4      1  
## 5      1  
## 6      2  
  
## ...
```

2. Przedstaw rozkład empiryczny liczby zgłoszeń za pomocą szeregu rozdzielczego.

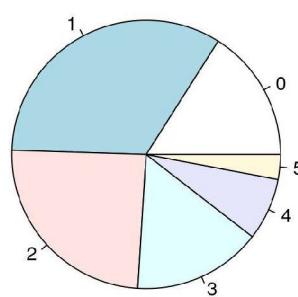
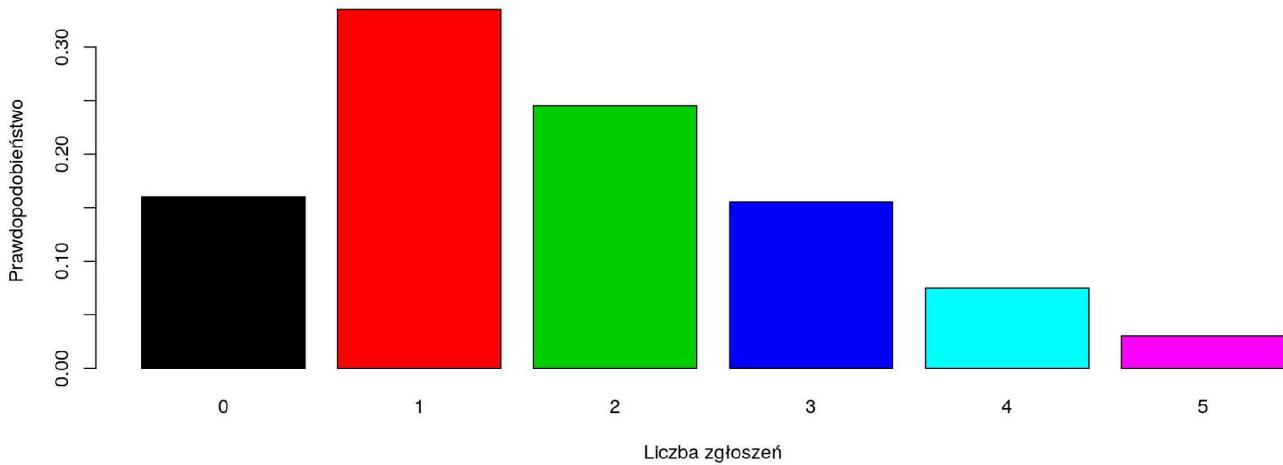
```
##    liczebnosc procent  
## 0      32  0.160  
## 1      67  0.335  
## 2      49  0.245  
## 3      31  0.155  
## 4      15  0.075  
## 5       6  0.030
```

3. Zilustruj liczbę zgłoszeń za pomocą wykresu słupkowego i kołowego.

Rozkład empiryczny liczby zgłoszeń



Rozkład empiryczny liczby zgłoszeń



4. Obliczyć średnią z liczby zgłoszeń, medianę liczby zgłoszeń, odchylenie standardowe liczby zgłoszeń i współczynnik zmienności liczby zgłoszeń.

```
## [1] 1.74
```

```
## [1] 2  
  
## [1] 1.28086
```

```
## [1] 73.61266
```

5. Zinterpretuj powyższe wyniki (tabelaryczne, graficzne i liczbowe).

Zadanie 3. Notowano pomiary średniej szybkości wiatru w odstępach 15 minutowych wokół nowo powstającej elektrowni wiatrowej. Wyniki są następujące:

0.9	6.2	2.1	4.1	7.3
1.0	4.6	6.4	3.8	5.0
2.7	9.2	5.9	7.4	3.0
4.9	8.2	5.0	1.2	10.1
12.2	2.8	5.9	8.2	0.5

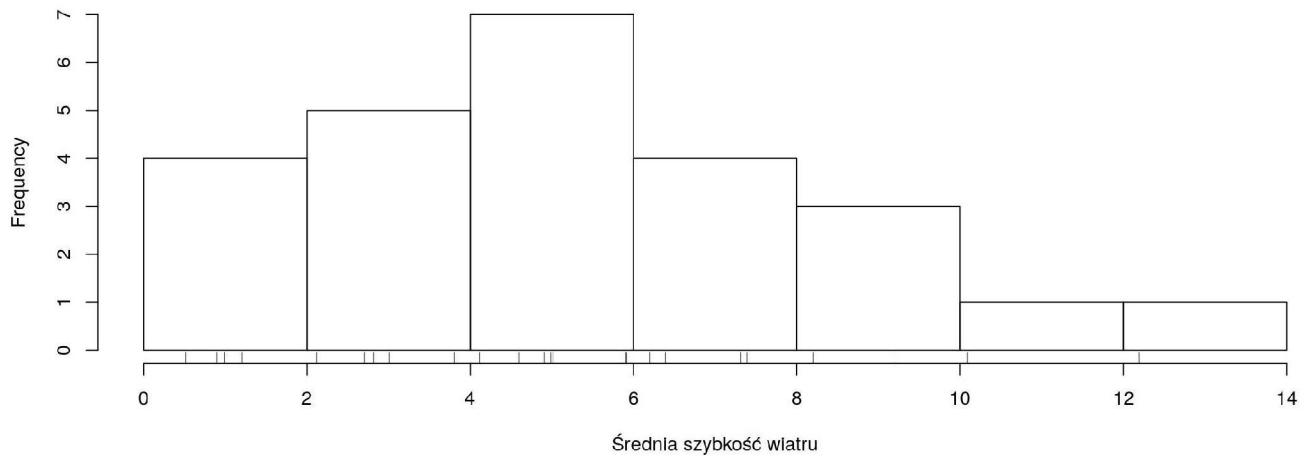
Jakiego typu jest ta zmienna? Jakie są możliwe wartości tej zmiennej?

1. Przedstaw rozkład empiryczny badanej zmiennej za pomocą szeregu rozdzielczego.

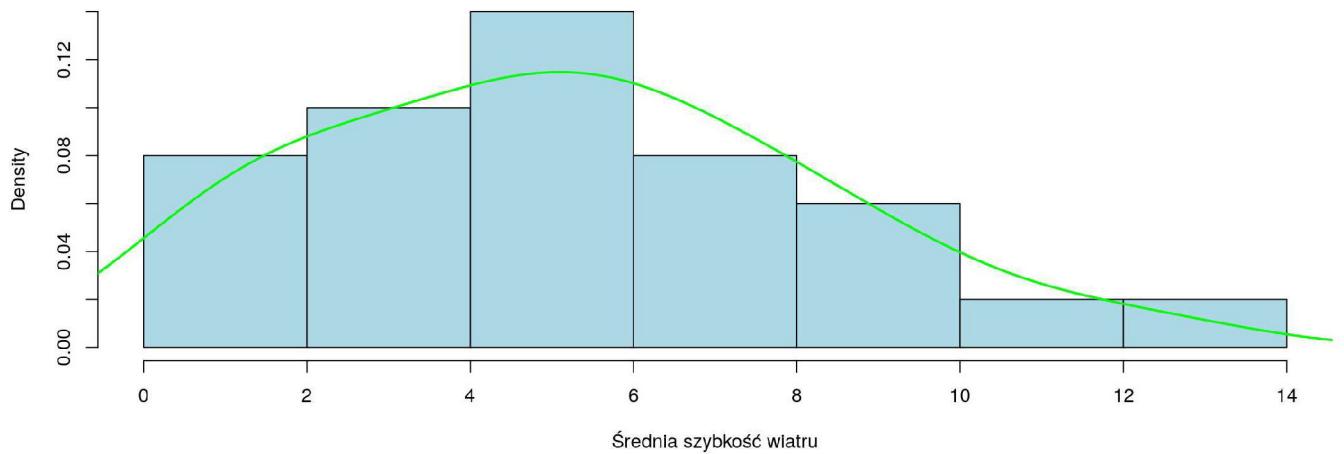
```
##      liczebnosc procent  
## (0,2]          4    0.16  
## (2,4]          5    0.20  
## (4,6]          7    0.28  
## (6,8]          4    0.16  
## (8,10]         3    0.12  
## (10,12]        1    0.04  
## (12,14]        1    0.04
```

2. Zilustruj rozkład empiryczny średniej szybkości wiatru za pomocą histogramu i wykresu pudełkowego. Jakie są zalety i wady tych wykresów?

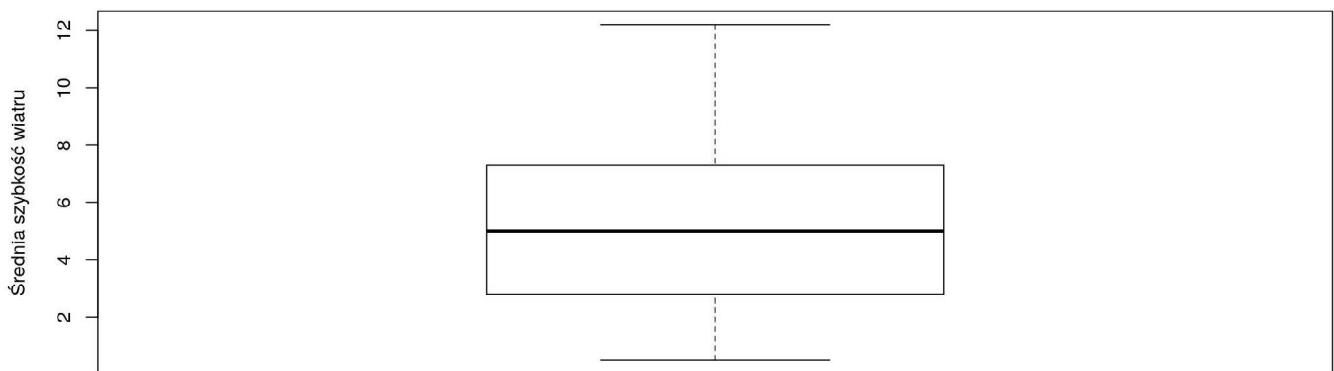
Rozkład empiryczny średniej szybkość wiatru



Rozkład empiryczny średniej szybkość wiatru



Rozkład empiryczny średniej szybkość wiatru



4. Obliczyć średnią, medianę, odchylenie standardowe, współczynnik zmienności, współczynnik asymetrii i kurtozę średniej szybkości wiatru.

[1] 5.144

```
## [1] 5  
  
## [1] 3.054925  
  
## [1] 59.38812  
  
## [1] 0.3422838  
  
## [1] -0.665667
```

5. Zinterpretuj powyższe wyniki (tabelaryczne, graficzne i liczbowe).

Zadanie 4. Napisz funkcję `wspolczynnik_zmiennosci()`, która oblicza wartość współczynnika zmienności dla danego wektora obserwacji. Funkcja powinna mieć dwa argumenty:

- `x` - wektor zawierający dane,
- `na.rm` - wartość logiczna (domyślnie `FALSE`), która wskazuje czy braki danych (obiekty `NA`) mają być zignorowane.

Funkcja zwraca wartość współczynnika zmienności wyrażoną w procentach. Ponadto funkcja sprawdza, czy wektor `x` jest wektorem numerycznym. W przeciwnym razie zostanie zwrócony błąd z następującym komunikatem: „argument nie jest liczbą”. Przykładowe wywołania i wyniki funkcji są następujące:

```
x <- c(1, NA, 3)  
wspolczynnik_zmiennosci(x)  
## [1] NA  
wspolczynnik_zmiennosci(x, na.rm = TRUE)  
## [1] 70.71068  
wspolczynnik_zmiennosci()  
## Error in wspolczynnik_zmiennosci():  
##   argument "x" is missing, with no default  
wspolczynnik_zmiennosci(c("x", "y"))  
## Error in wspolczynnik_zmiennosci(c("x", "y")) : argument nie jest liczbą
```


5 Model statystyczny i estymacja punktowa

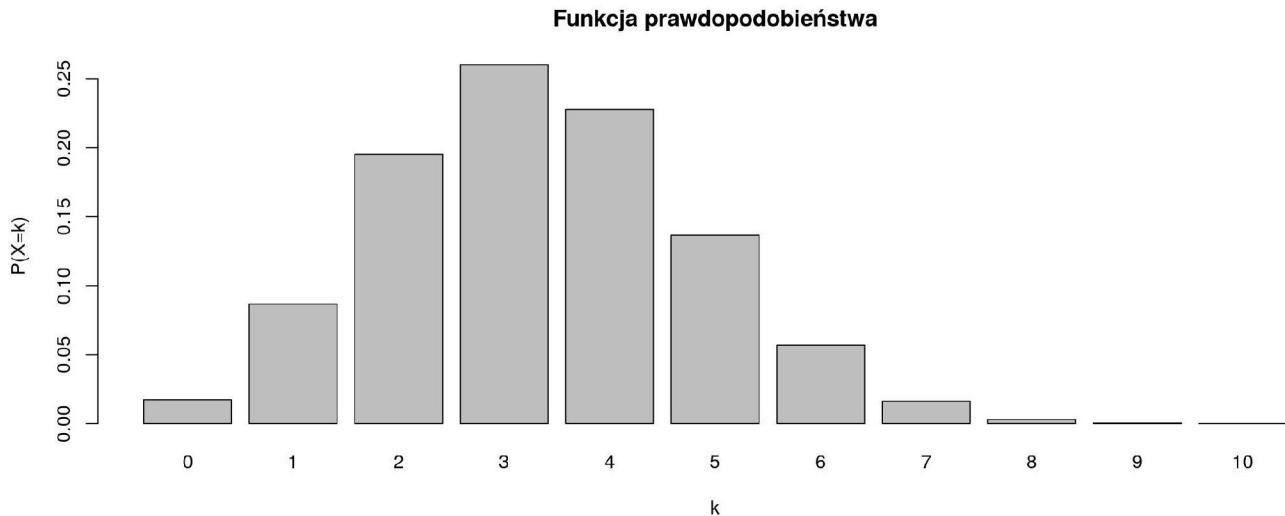
5.1 Wybrane rozkłady prawdopodobieństwa

1. rozkład dwumianowy $b(m, p)$, $m \in \mathbb{N}$, $p \in (0, 1)$

$$\mathbb{P}(X = k) = \binom{m}{k} p^k (1 - p)^{m-k}, \quad k = 0, 1, \dots, m$$

- Funkcja prawdopodobieństwa zmiennej $X \sim b(10, 1/3)$

```
barplot(dbinom(x = 0:10, size = 10, prob = 1 / 3), names.arg = 0:10,
        xlab = "k", ylab = "P(X=k)", main = "Funkcja prawdopodobieństwa")
```

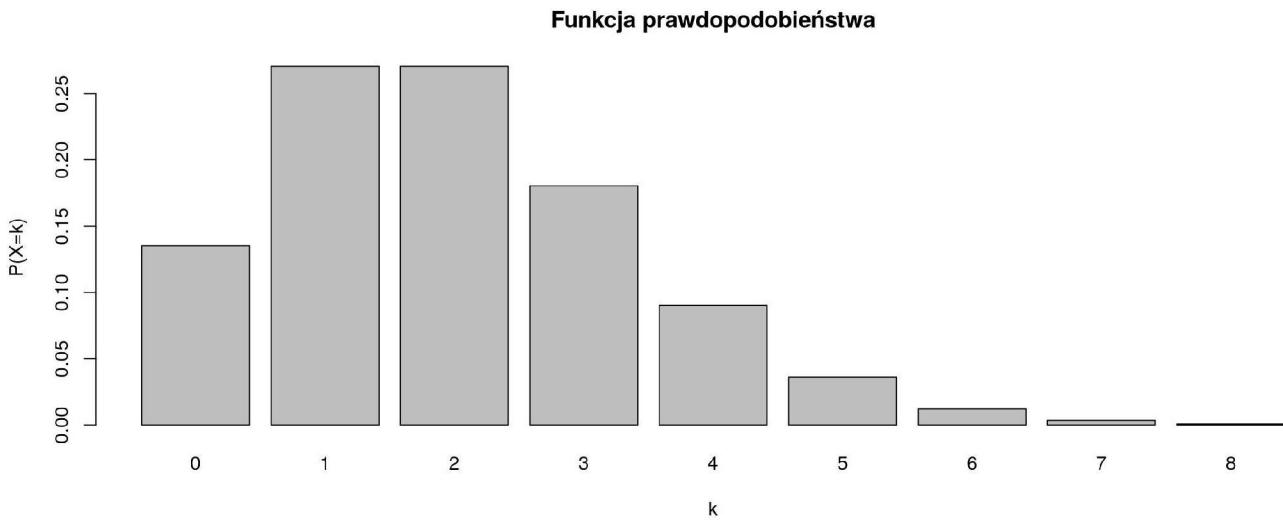


2. rozkład Poissona $\pi(\lambda)$, $\lambda > 0$

$$\mathbb{P}(X = k) = \frac{e^{-\lambda} \lambda^k}{k!}, \quad k = 0, 1, \dots$$

- Funkcja prawdopodobieństwa zmiennej $X \sim \pi(2)$

```
barplot(dpois(x = 0:8, lambda = 2), names.arg = 0:8,
        xlab = "k", ylab = "P(X=k)", main = "Funkcja prawdopodobieństwa")
```

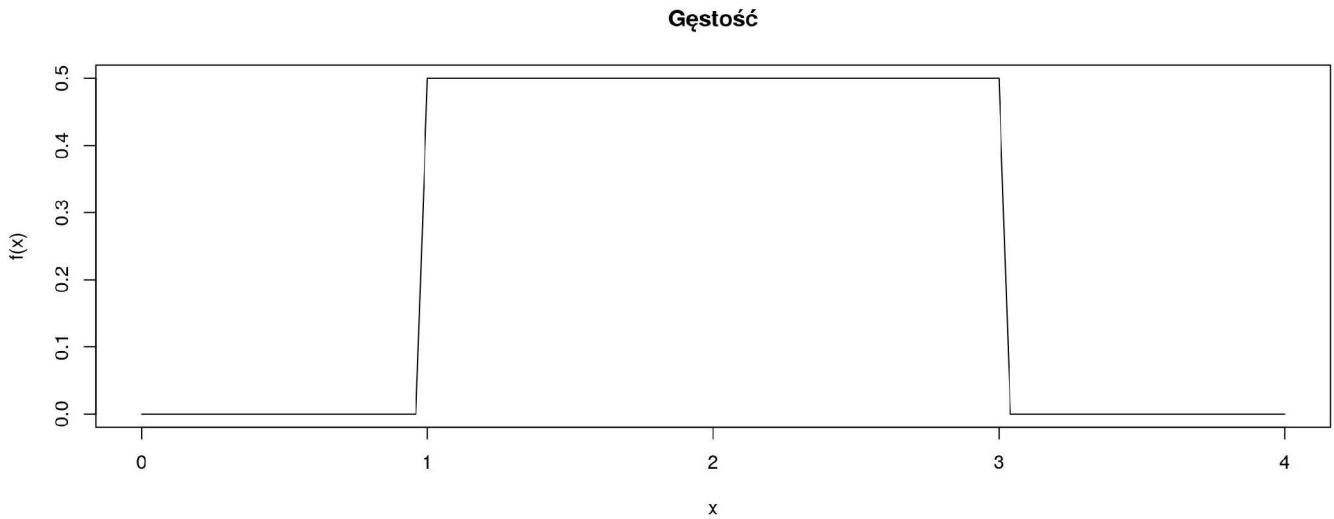


3. rozkład jednostajny $U(a, b)$, $a < b$

$$f_X(x) = \begin{cases} \frac{1}{b-a} & \text{dla } x \in (a, b) \\ 0 & \text{dla } x \notin (a, b) \end{cases}$$

- Gęstość zmiennej $X \sim U(1, 3)$

```
curve(dunif(x, min = 1, max = 3), 0, 4, ylab = "f(x)", main = "Gęstość")
```



4. rozkład normalny $N(\mu, \sigma)$, $\mu \in \mathbb{R}$, $\sigma > 0$

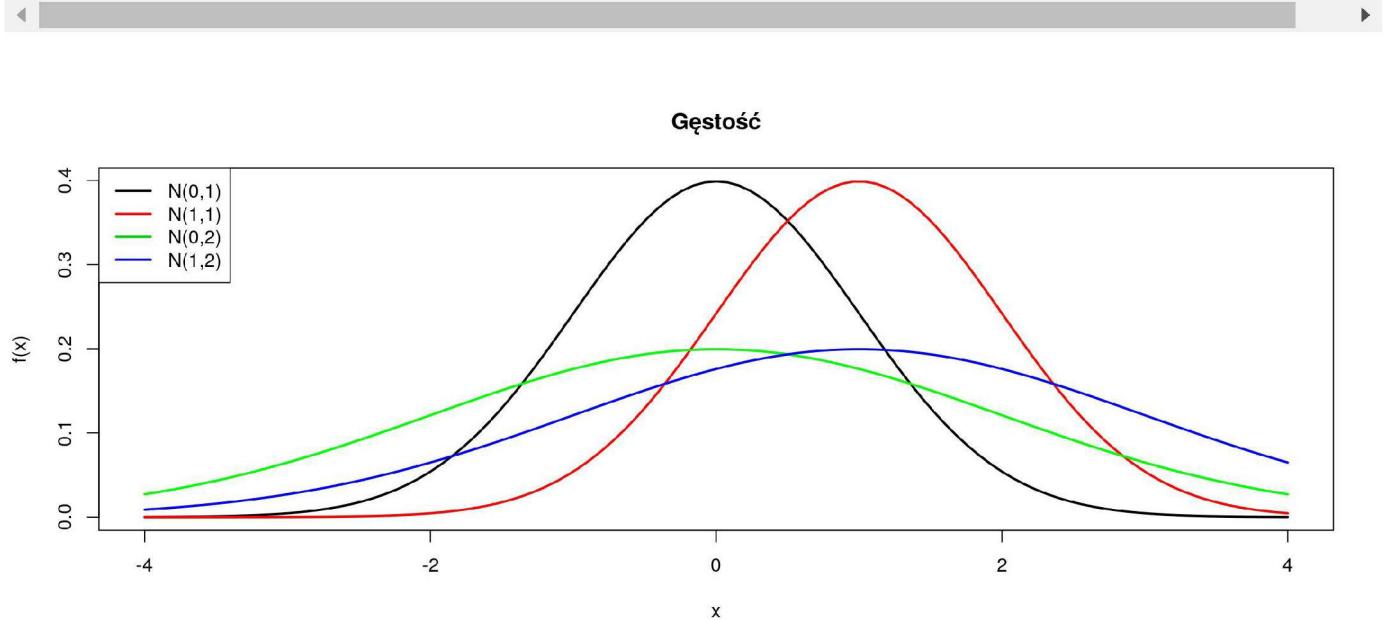
$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

- Gęstości rozkładów normalnych

```

curve(dnorm, -4, 4, ylab = "f(x)", main = "Gęstość", lwd = 2)
curve(dnorm(x, mean = 1), col = "red", add = TRUE, lwd = 2)
curve(dnorm(x, sd = 2), col = "green", add = TRUE, lwd = 2)
curve(dnorm(x, mean = 1, sd = 2), col = "blue", add = TRUE, lwd = 2)
legend("topleft", lwd = 2, col = 1:4, legend = c("N(0,1)", "N(1,1)", "N(0,2)", "N(1,2)"))

```



5. rozkład wykładniczy $Ex(\lambda)$, $\lambda > 0$

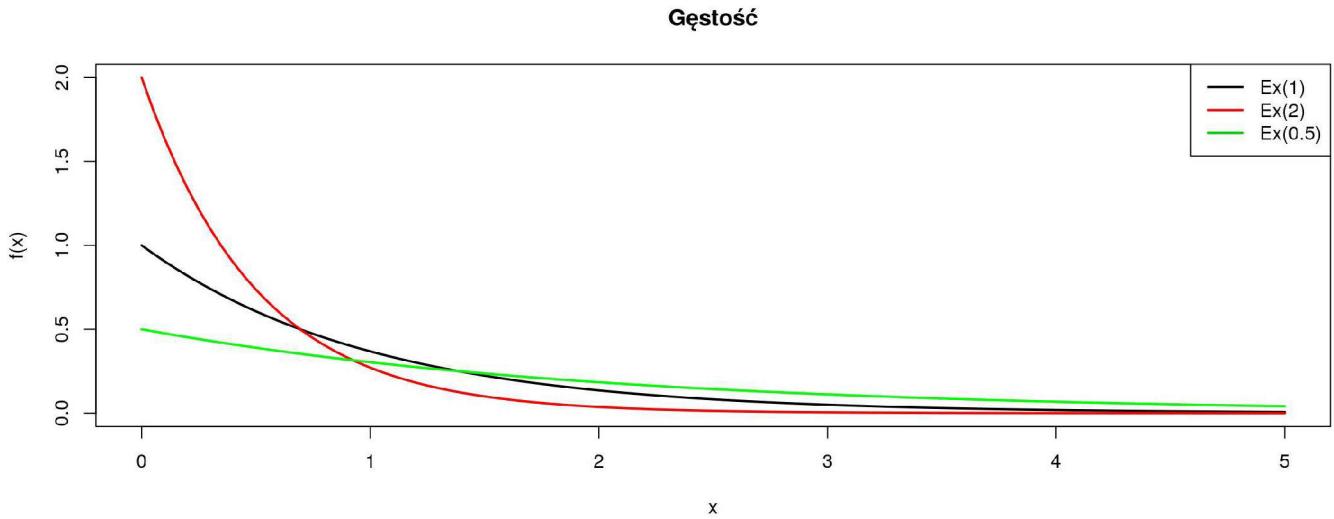
$$f_X(x) = \begin{cases} \lambda e^{-\lambda x} & \text{dla } x > 0 \\ 0 & \text{dla } x \leq 0 \end{cases}$$

- Gęstości rozkładów wykładniczych

```

curve(dexp, 0, 5, ylim = c(0, 2), ylab = "f(x)", main = "Gęstość", lwd = 2)
curve(dexp(x, rate = 2), col = "red", add = TRUE, lwd = 2)
curve(dexp(x, rate = 0.5), col = "green", add = TRUE, lwd = 2)
legend("topright", lwd = 2, col = 1:3, legend = c("Ex(1)", "Ex(2)", "Ex(0.5)"))

```



6. rozkład Rayleigha $R(\lambda)$, $\lambda > 0$

$$f_\lambda(x) = \frac{2}{\lambda} x \exp\left(-\frac{x^2}{\lambda}\right) I_{(0,\infty)}(x)$$

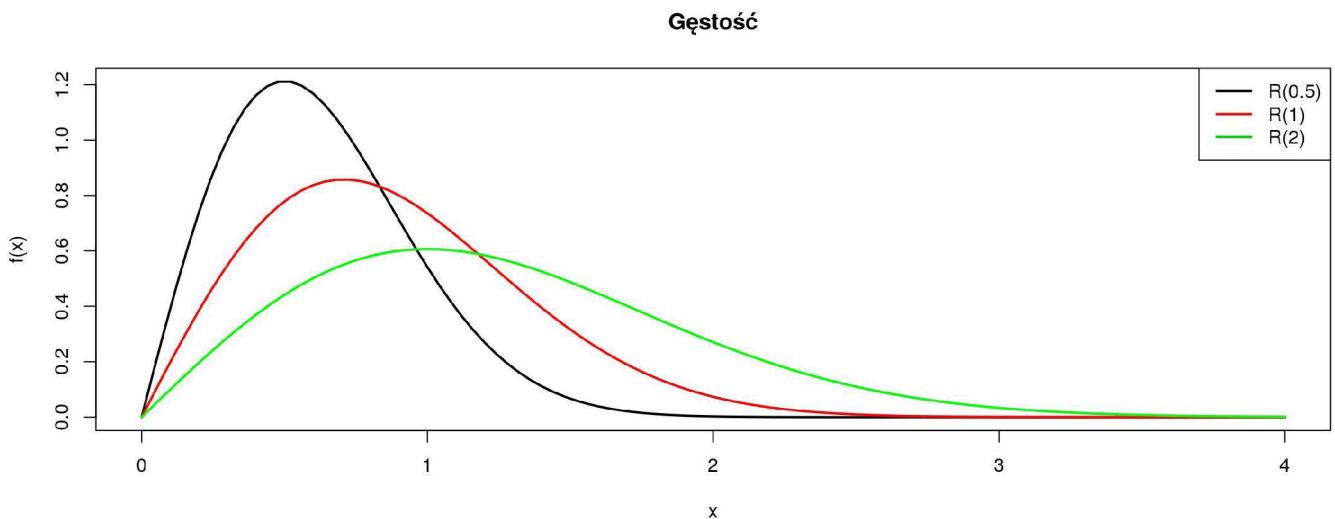
Uwaga. Rozkład Rayleigha jest zaimplementowany w pakiecie `VGAM` z następującą funkcją gęstości

$$f_\sigma(x) = \frac{x}{\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) I_{(0,\infty)}(x),$$

więc w naszej notacji $\sigma = \sqrt{\frac{\lambda}{2}}$.

- Gęstości rozkładów Rayleigha

```
lambda <- 0.5
curve(VGAM::drayleigh(x, sqrt(lambda / 2)),
      xlim = c(0, 4), ylab = "f(x)", main = "Gęstość", lwd = 2)
lambda <- 1
curve(VGAM::drayleigh(x, sqrt(lambda / 2)),
      col = "red", add = TRUE, lwd = 2)
lambda <- 2
curve(VGAM::drayleigh(x, sqrt(lambda / 2)),
      col = "green", add = TRUE, lwd = 2)
legend("topright", lwd = 2, col = 1:3, legend = c("R(0.5)", "R(1)", "R(2)"))
```



Rozkłady prawdopodobieństwa w programie R

Rozkład	Dystrybuanta	Gęstość/Funkcja prawd.	Kwantyl	Gen
dwumianowy	pbinom	dbinom	qbinom	
Poissona	ppois	dpois	qpois	
ujemny dwumianowy	pnbinom	dnbinom	qnbinom	r
geometryczny	pgeom	dgeom	qgeom	
hipergeometryczny	phyper	dhyper	qhyper	
jednostajny	punif	dunif	qunif	
beta	pbeta	dbeta	qbeta	
wykładniczy	pexp	dexp	qexp	
gamma	pgamma	dgamma	qgamma	r
normalny	pnorm	dnorm	qnorm	
logarytmiczno-normalny	plnorm	dlnorm	qlnorm	
Weibulla	pweibull	dweibull	qweibull	
chi-kwadrat	pchisq	dchisq	qchisq	
t-Studenta	pt	dt	qt	
Cauchy'ego	pcauchy	dcauchy	qcauchy	i
F-Snedecora	pf	df	qf	
Rayleigha	VGAM::prayleigh	VGAM::drayleigh	VGAM::qrayleigh	VGAM::ri

5.2 Przykłady

Przykład 1. Poniższe dane podają liczbę błędów w grupie 50 osób zdających egzamin testowy. Egzamin składał się z 18 pytań (można popełnić maksymalnie dwa błędy, aby zdać egzamin).

```

1 1 2 0 1 3 1 4 4 4 0 1 0 0 0 2 3
4 0 1 5 2 3 5 3 2 2 4 0 2 2 0 2 2
3 3 1 3 2 2 0 0 5 4 2 1 5 2 2 0

```

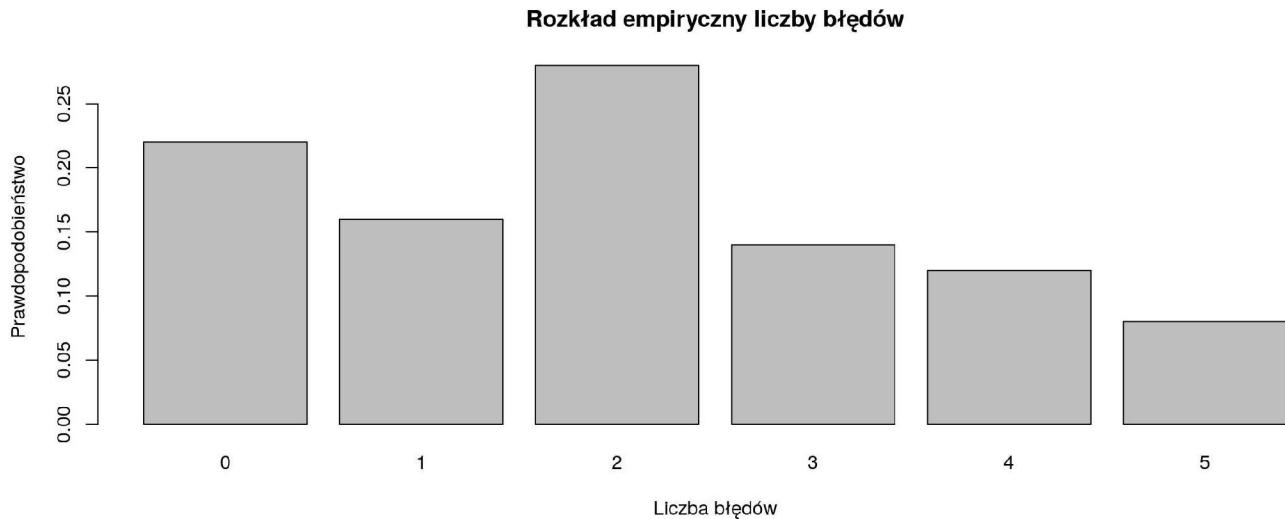
Zmienna X to liczba błędów. Jest to dyskretna zmienna ilościowa.

```

liczba_bledow <- c(1, 1, 2, 0, 1, 3, 1, 4, 4, 4, 0, 1, 0, 0, 0, 2, 3,
                  4, 0, 1, 5, 2, 3, 5, 3, 2, 2, 4, 0, 2, 2, 0, 2, 2,
                  3, 3, 1, 3, 2, 2, 0, 0, 5, 4, 2, 1, 5, 2, 2, 0)

# wykres słupkowy
barplot(prop.table(table(liczba_bledow)),
        xlab = "Liczba błędów", ylab = "Prawdopodobieństwo",
        main = "Rozkład empiryczny liczby błędów")

```



- model: rozkład dwumianowy z $m = 18$
- $\mathcal{P} = \{b(18, p) : p \in (0, 1)\}$
- $\Theta = (0, 1)$ oraz $\theta = p$

```

liczba_bledow <- c(1, 1, 2, 0, 1, 3, 1, 4, 4, 4, 0, 1, 0, 0, 0, 2, 3,
                  4, 0, 1, 5, 2, 3, 5, 3, 2, 2, 4, 0, 2, 2, 0, 2, 2,
                  3, 3, 1, 3, 2, 2, 0, 0, 5, 4, 2, 1, 5, 2, 2, 0)

m <- 18
# estymator
(p_est <- mean(liczba_bledow) / m)

## [1] 0.1122222

```

```

probs <- dbinom(sort(unique(liczba_bledow)), size = m, prob = p_est)
sum(probs)

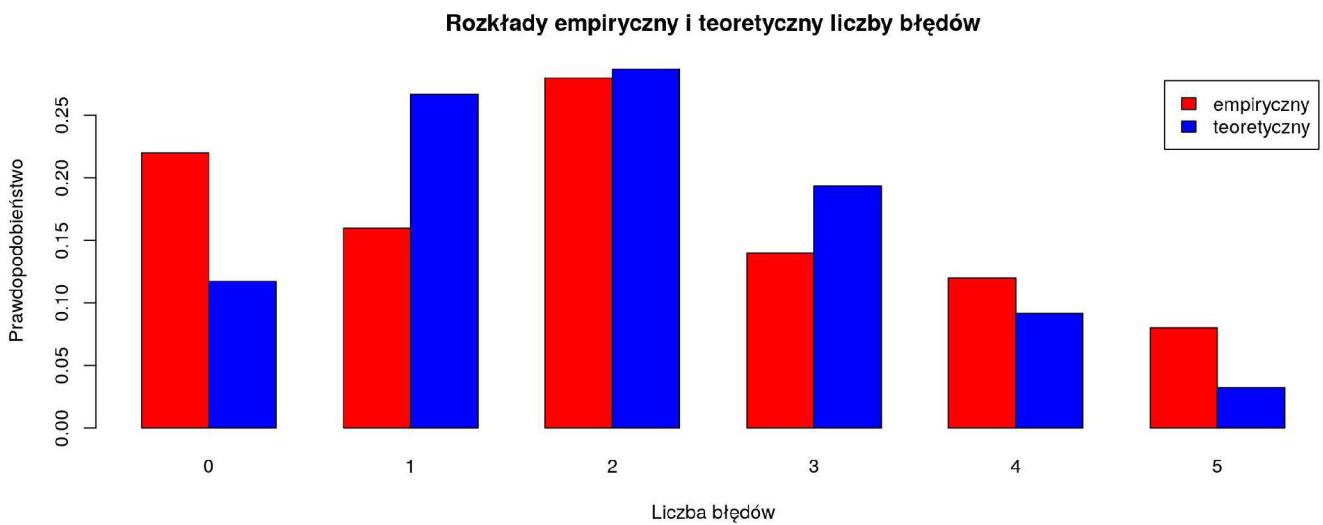
## [1] 0.9887985

counts <- matrix(c(prop.table(table(liczba_bledow)), probs), nrow = 2, byrow = TRUE)
rownames(counts) <- c("empiryczny", "teoretyczny")
colnames(counts) <- sort(unique(liczba_bledow))
counts

##          0      1      2      3      4      5 
## empiryczny 0.2200000 0.1600000 0.2800000 0.1400000 0.12000000 0.08000000
## teoretyczny 0.1173483 0.2670078 0.2868914 0.1934153 0.09168466 0.03245109

barplot(counts,
        xlab = "Liczba błędów", ylab = "Prawdopodobieństwo",
        main = "Rozkłady empiryczny i teoretyczny liczby błędów",
        col = c("red", "blue"), legend = rownames(counts), beside = TRUE)

```



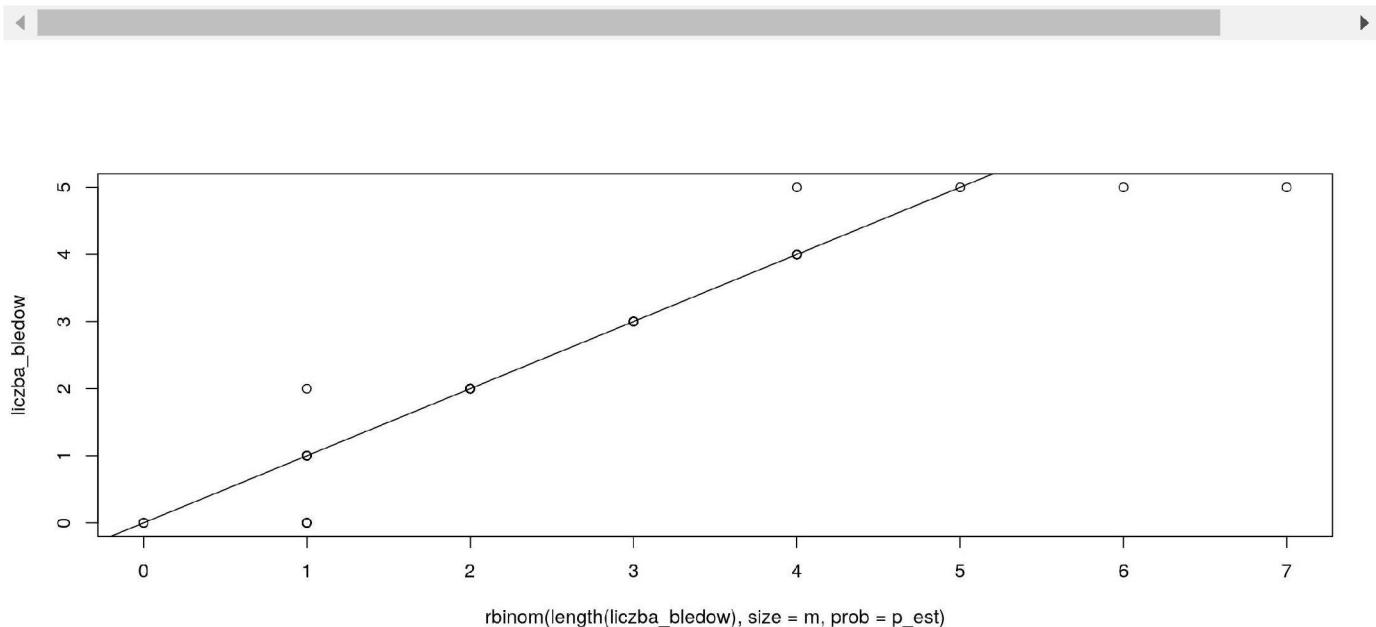
Wykres kwantyl-kwantyl

- Wykres kwantyl-kwantyl (Q-Q plot), jest wykresem zaobserwowanych statystyk porządkowych z losowej próby (kwantyle empiryczne) do odpowiadającym im

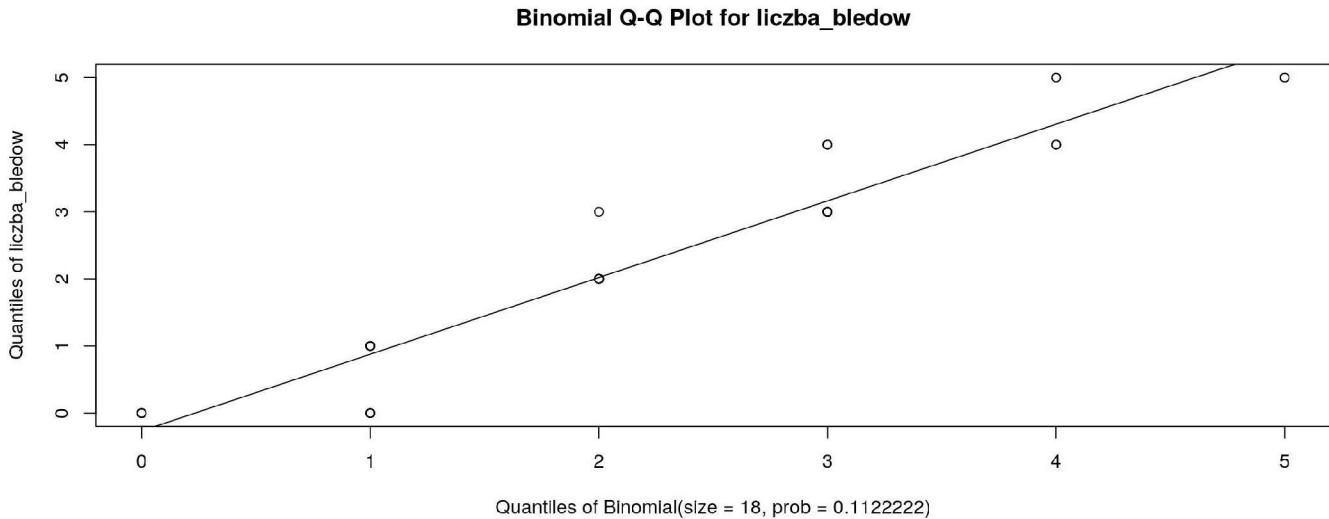
(oszacowanym) wartościom średniej lub mediany w oparciu o założony rozkład lub w stosunku do empirycznych kwantylów innego zestawu danych.

- Wykresy kwantyl-kwantyl służą do oceny, czy dane pochodzą z określonego rozkładu lub czy dwa zestawy danych mają ten sam rozkład. Jeśli rozkłady mają ten sam kształt (ale niekoniecznie te same parametry położenia lub skali), wówczas wykres układu się mniej więcej na linii prostej. Jeśli rozkłady są dokładnie takie same, wówczas wykres układu się mniej więcej na linii prostej $y = x$.
- Najpierw wybiera się zbiór kwantylów pewnych rzędów. Punkt (x, y) na wykresie odpowiada jednemu z kwantylów drugiego rozkładu (współrzędna y) wykreślonym względem kwantyla tego samego rzędu pierwszego rozkładu (współrzędna x).
- „qqline” dodaje linię do „teoretycznego” wykresu kwantyl-kwantyl, która przechodzi przez kwantyle rzędów `probs = c(0.25, 0.75)`, czyli domyślnie pierwszy i trzeci kwartył.

```
# wykres kwantyl-kwantyl
qqplot(rbinom(length(liczba_bledow), size = m, prob = p_est), liczba_bledow)
qqline(liczba_bledow, distribution = function(probs) { qbinom(probs, size = m, prob = p_
```



```
# Lub
library(EnvStats)
EnvStats::qqPlot(liczba_bledow,
                 distribution = "binom",
                 param.list = list(size = m, prob = p_est),
                 add.line = TRUE)
```



Przykład 2. Badano czas oczekiwania na tramwaj, który kursuje w jednakowych odstępach czasu. Plik `czas_oczek_tramwaj.RData` zawiera dane dotyczące czasu oczekiwania na tramwaj (wyrażonego w minutach) 100 osób wybranych losowo. Zmienna X to czas oczekiwania na tramwaj. Jest to zmienna ilościowa ciągła.

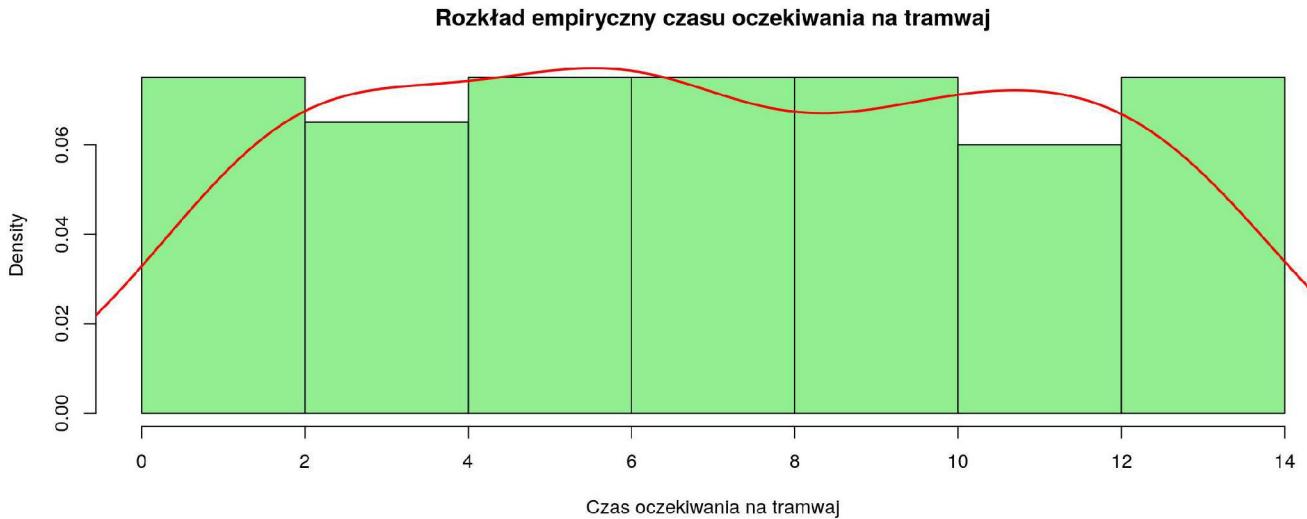
```

load(url("http://ls.home.amu.edu.pl/data_sets/czas_oczek_tramwaj.RData"))
head(czas_oczek_tramwaj)

## [1] 4.03 11.04 5.73 12.36 13.17 0.64

# histogram z estymatorem jądrowym gęstości
hist(czas_oczek_tramwaj,
      xlab = "Czas oczekiwania na tramwaj",
      main = "Rozkład empiryczny czasu oczekiwania na tramwaj",
      probability = TRUE,
      col = "lightgreen")
lines(density(czas_oczek_tramwaj), col = "red", lwd = 2)

```



- model: rozkład jednostajny
- $\mathcal{P} = \{U(a, b) : a, b \in \mathbb{R}, a < b\}$
- $\Theta = \{(a, b) \in \mathbb{R}^2 : a < b\}$ oraz $\theta = (a, b)$

```
load(url("http://ls.home.amu.edu.pl/data_sets/czas_oczek_tramwaj.RData"))
```

```
# estmatory
```

```
(a_est <- min(czas_oczek_tramwaj))
```

```
## [1] 0.01
```

```
(b_est <- max(czas_oczek_tramwaj))
```

```
## [1] 13.92
```

```
library(EnvStats)
```

```
EnvStats::eunif(czas_oczek_tramwaj, method = "mle")
```

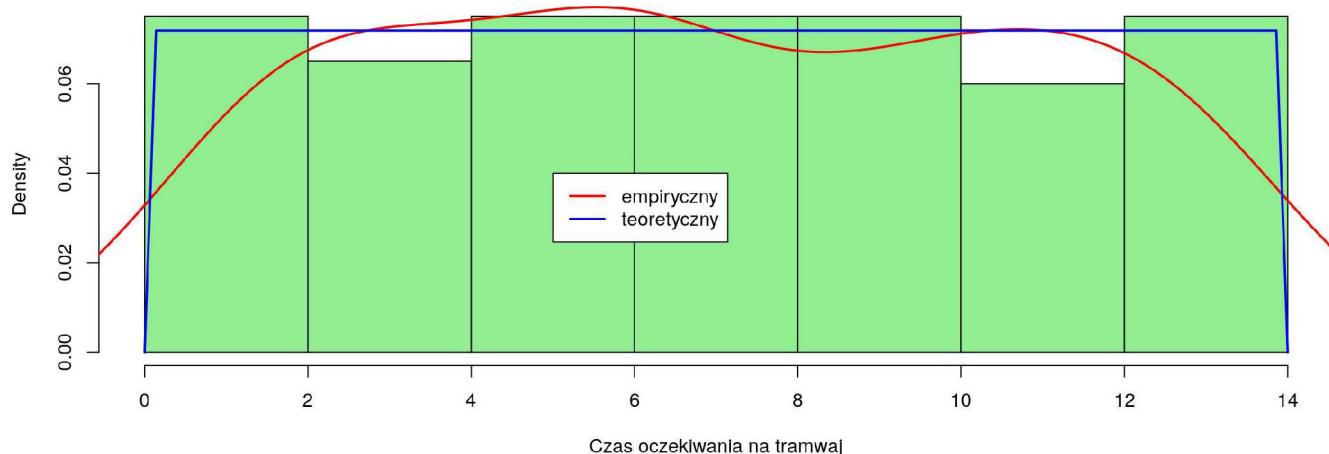
```

## 
## Results of Distribution Parameter Estimation
## -----
## 
## Assumed Distribution:           Uniform
## 
## Estimated Parameter(s):       min =  0.01
##                               max = 13.92
## 
## Estimation Method:            mle
## 
## Data:                         czas_oczek_tramwaj
## 
## Sample Size:                  100

# histogram z estymatorem jądrowym gęstości
hist(czas_oczek_tramwaj,
      xlab = "Czas oczekiwania na tramwaj",
      main = "Rozkład empiryczny i teoretyczny czasu oczekiwania na tramwaj",
      probability = TRUE,
      col = "lightgreen")
lines(density(czas_oczek_tramwaj), col = "red", lwd = 2)
curve(dunif(x, a_est, b_est),
      add = TRUE, col = "blue", lwd = 2)
legend(x = 5, y = 0.04, legend = c("empiryczny", "teoretyczny"), col = c("red", "blue"),

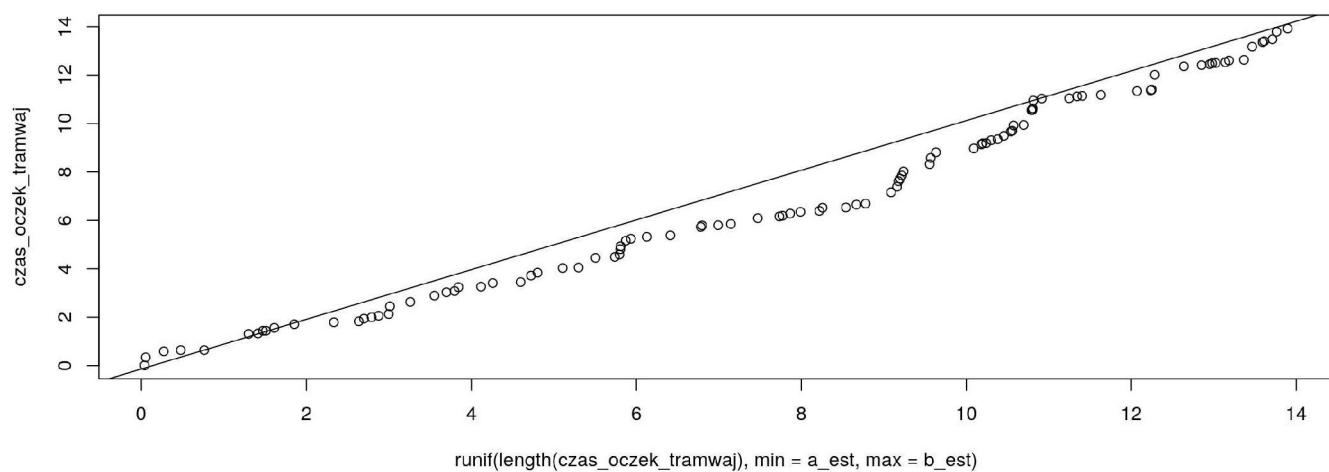
```

Rozkład empiryczny i teoretyczny czasu oczekiwania na tramwaj



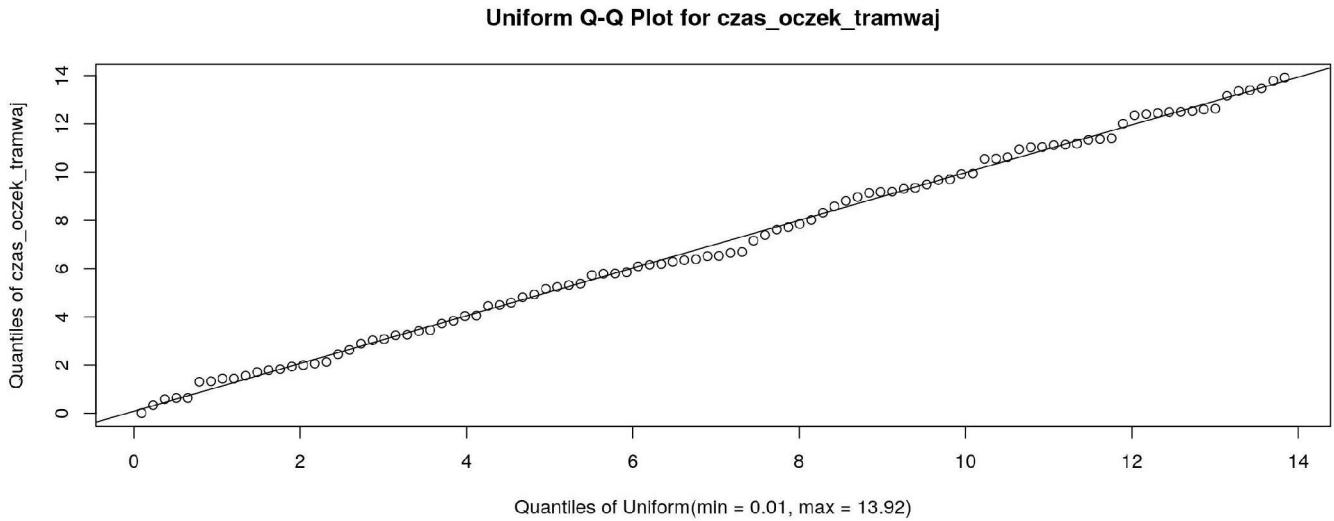
```
# wykres kwantyl-kwantyl
```

```
qqplot(runif(length(czas_oczek_tramwaj), min = a_est, max = b_est), czas_oczek_tramwaj)
qqline(czas_oczek_tramwaj, distribution = function(probs) { qunif(probs, min = a_est, ma
```



```
# Lub
```

```
library(EnvStats)
EnvStats::qqPlot(czas_oczek_tramwaj,
                 distribution = "unif",
                 param.list = list(min = a_est, max = b_est),
                 add.line = TRUE)
```



- Empiryczne i teoretyczne prawdopodobieństwo, że czas oczekiwania na tramwaj jest większy niż 10 minut, można obliczyć w następujący sposób:

```
# empirycznie
mean(czas_oczek_tramwaj > 10)
```

```
## [1] 0.27
```

```
# teoretycznie:  $X \sim U(a_{\text{est}}, b_{\text{est}})$  oraz  $P(X > 10) = 1 - P(X \leq 10) = 1 - F(10)$ 
1 - punif(10, min = a_est, max = b_est)
```

```
## [1] 0.2818116
```

5.3 Zadania

Zadanie 1. Niech $\mathbf{X} = (X_1, \dots, X_n)^\top$ będzie próbą prostą z populacji o rozkładzie jednostajnym $U(a, b)$.

1. Pokaż, że estymatory metody momentów parametrów a i b w rozkładzie jednostajnym $U(a, b)$ są postaci:

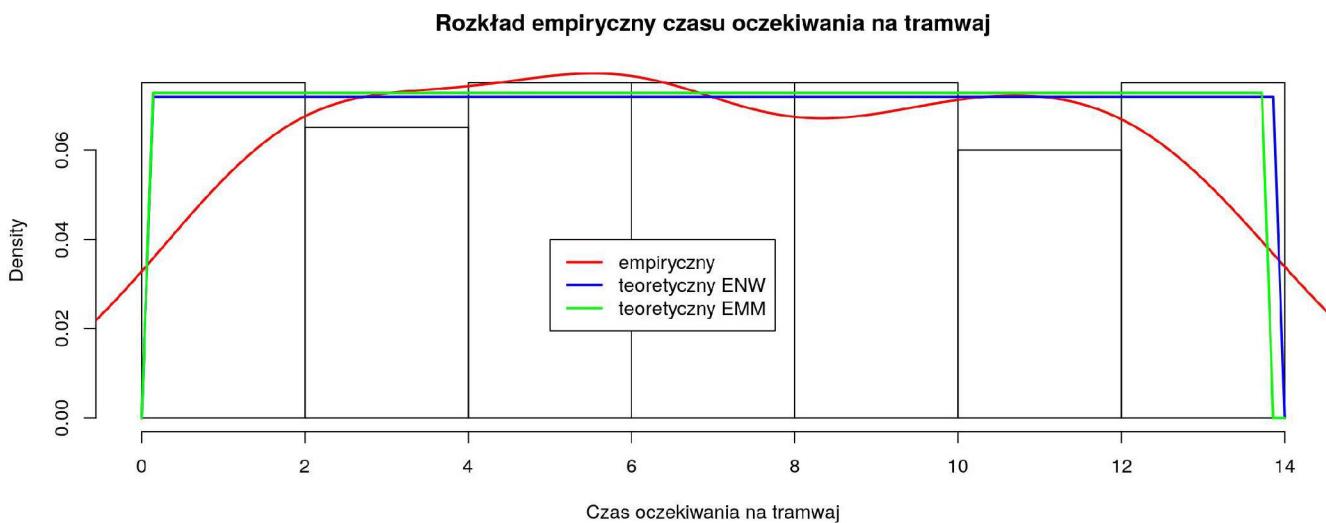
$$\hat{a} = \bar{X} - \sqrt{3}\tilde{S}, \quad \hat{b} = \bar{X} + \sqrt{3}\tilde{S},$$

gdzie $\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i$ oraz $\tilde{S} = \sqrt{\frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})^2}$.

2. Oblicz wartości tych estymatorów dla danych z przykładu dotyczącego czasu oczekiwania na tramwaj.

```
##  
## Results of Distribution Parameter Estimation  
## -----  
##  
## Assumed Distribution: Uniform  
##  
## Estimated Parameter(s): min = 0.1040974  
## max = 13.8551026  
##  
## Estimation Method: mme  
##  
## Data: czas_oczek_tramwaj  
##  
## Sample Size: 100
```

3. Zilustruj otrzymane teoretyczne funkcje gęstości korzystające z ENW i EMM na histogramie.



Zadanie 2. Przebadano 200 losowo wybranych 5-sekundowych okresów pracy centrali telefonicznej. Rejestrowano liczbę zgłoszeń. Wyniki są zawarte w pliku [Centrala.RData](#).

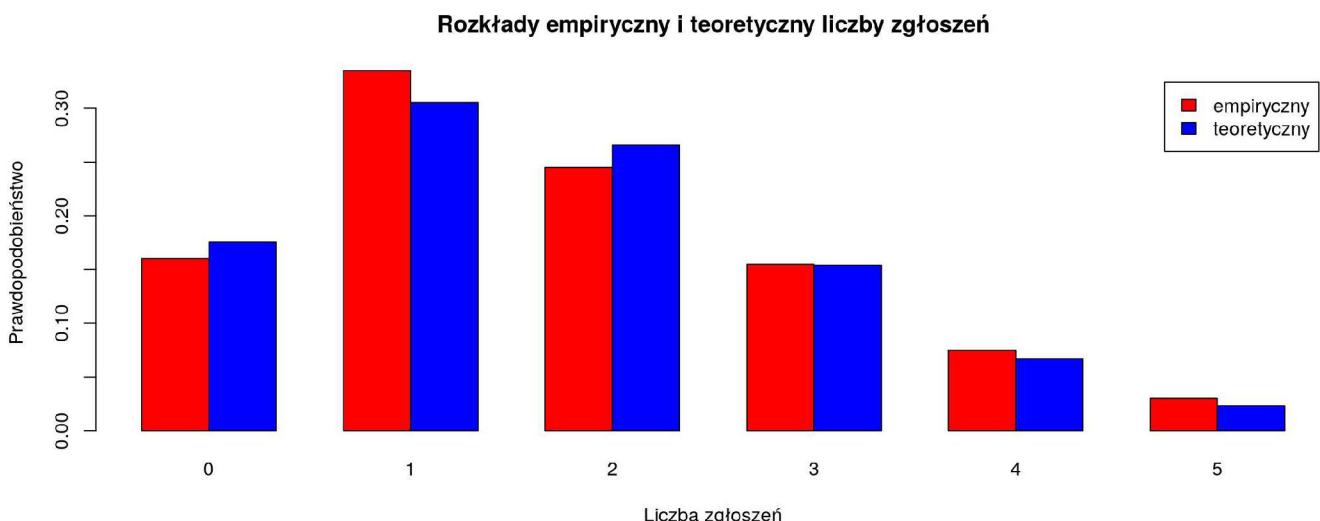
1. Zasugeruj rozkład teoretyczny badanej zmiennej.
2. Oblicz wartość estymatora parametru rozkładu teoretycznego.

```
## [1] 1.74
```

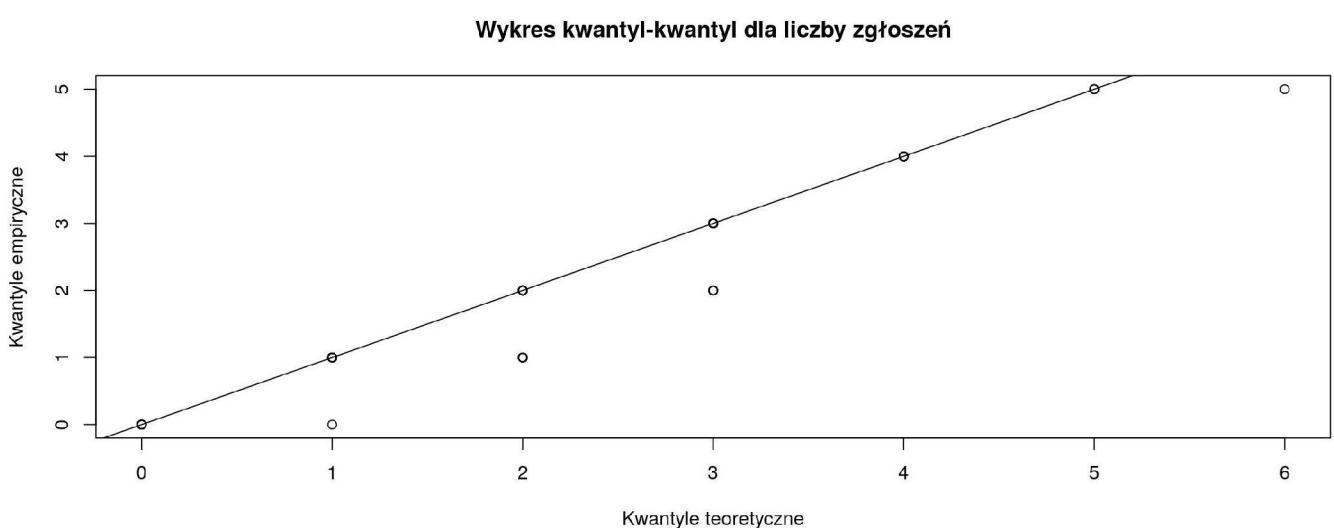
3. Porównaj empiryczne prawdopodobieństwa wystąpienia poszczególnych wartości liczby zgłoszeń w próbie z wartościami teoretycznymi uzyskanymi na podstawie rozkładu teoretycznego.

```
## [1] 0.9911019
```

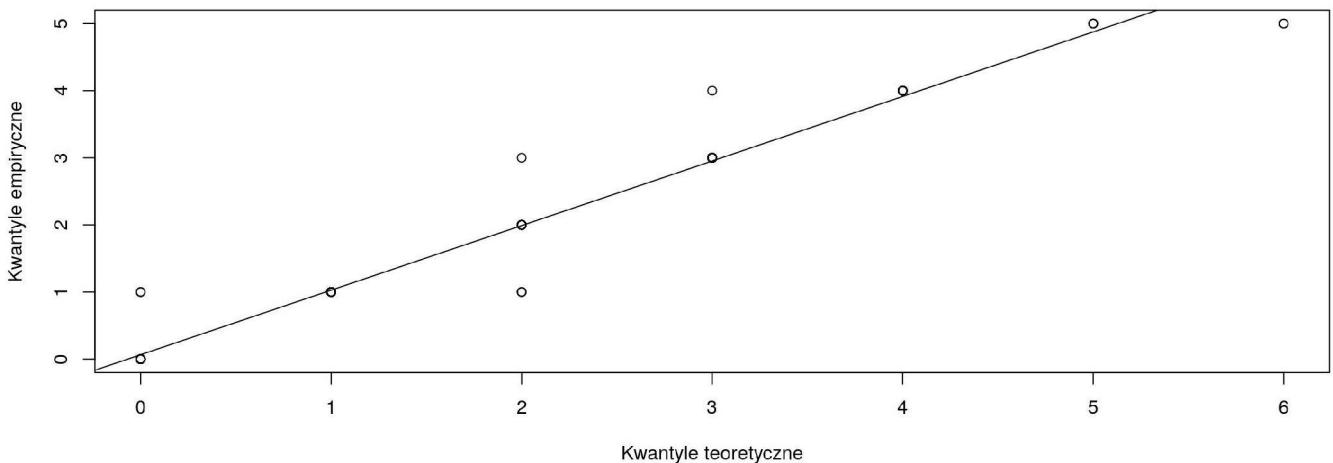
```
##          0      1      2      3      4      5 
## empiryczny 0.1600000 0.3350000 0.2450000 0.1550000 0.07500000 0.03000000
## teoretyczny 0.1755204 0.3054055 0.2657028 0.1541076 0.06703681 0.02332881
```



4. Sprawdź dopasowanie rozkładu teoretycznego za pomocą wykresy kwantyl-kwantyl.



Wykres kwantyl-kwantyl dla liczby zgłoszeń



5. Czy na podstawie powyższych rozważań rozkład teoretyczny wydaje się odpowiedni?
6. Oblicz prawdopodobieństwo empiryczne i teoretyczne, że liczba zgłoszeń jest mniejsza niż 4.

```
## [1] 0.895
```

```
## [1] 0.9007363
```

Zadanie 3. Niech $\mathbf{X} = (X_1, \dots, X_n)^\top$ będzie próbą prostą z rozkładu Rayleigha o gęstości:

$$f_\lambda(x) = \frac{2}{\lambda}x \exp\left(-\frac{x^2}{\lambda}\right) I_{(0,\infty)}(x), \quad \lambda > 0.$$

Pokaż, że ENW parametru λ jest postaci:

$$\frac{1}{n} \sum_{i=1}^n X_i^2.$$

W tym celu przeprowadź następujące kroki:

1. Pokaż, że funkcja wiarodności wynosi:

$$L(\lambda; \mathbf{x}) = f_\lambda(\mathbf{x}) = \prod_{i=1}^n f_\lambda(x_i) = \left(\frac{2}{\lambda}\right)^n \left(\prod_{i=1}^n x_i\right) \exp\left(-\frac{1}{\lambda} \sum_{i=1}^n x_i^2\right).$$

2. Wprowadź pomocniczą funkcję:

$$l = \ln L(\lambda; \mathbf{x}) = n \ln 2 - n \ln \lambda + \ln \left(\prod_{i=1}^n x_i \right) - \frac{1}{\lambda} \sum_{i=1}^n x_i^2.$$

3. Wyznacz pochodną funkcji l względem λ :

$$\frac{\partial l}{\partial \lambda} = \frac{1}{\lambda^2} \sum_{i=1}^n x_i^2 - \frac{n}{\lambda}.$$

4. Przyrównaj powyższą pochodną do zera i rozwiąż otrzymane równanie.

Zadanie 4. Notowano pomiary średniej szybkości wiatru w odstępach 15 minutowych wokół nowo powstającej elektrowni wiatrowej. Wyniki są następujące:

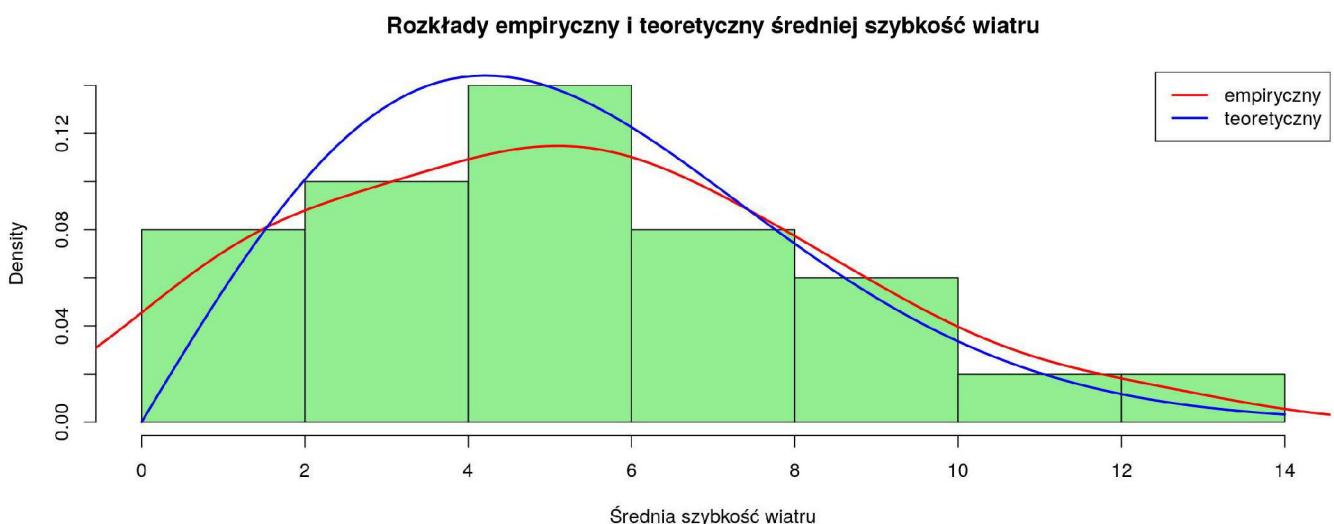
0.9	6.2	2.1	4.1	7.3
1.0	4.6	6.4	3.8	5.0
2.7	9.2	5.9	7.4	3.0
4.9	8.2	5.0	1.2	10.1
12.2	2.8	5.9	8.2	0.5

1. Zasugeruj rozkład teoretyczny badanej zmiennej.

2. Oblicz wartość ENW parametru rozkładu teoretycznego.

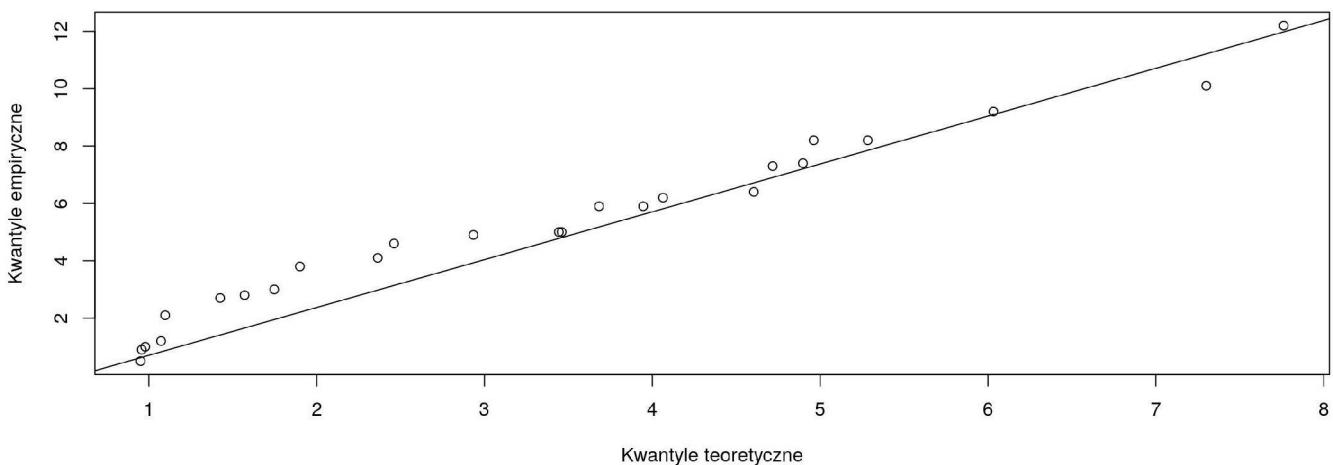
```
## [1] 35.42
```

3. Porównaj rozkład empiryczny wystąpienia poszczególnych wartości średniej szybkości wiatru w próbie z wartościami teoretycznymi uzyskanymi na podstawie rozkładu teoretycznego.



4. Sprawdź dopasowanie rozkładu teoretycznego za pomocą wykresy kwantyl-kwantyl.

Wykres kwantyl-kwantyl dla średniej szybkość wiatru



5. Czy na podstawie powyższych rozważań rozkład teoretyczny wydaje się odpowiedni?
6. Oblicz empiryczne i teoretyczne prawdopodobieństwo, że średnia szybkość wiatru jest zawarta w przedziale [4, 8].

```
## [1] 0.44
```

```
## [1] 0.3782218
```

7. Oblicz wartość ENW dla wartości oczekiwanej i wariancji rozkładu teoretycznego.

```
## [1] 5.274353
```

```
## [1] 7.601197
```

6 Przedziały ufności

6.1 Przykład

Przykład. Badano czas oczekiwania na tramwaj, który kursuje w jednakowych odstępach czasu. Plik `czas_oczek_tramwaj.RData` zawiera dane dotyczące czasu oczekiwania na tramwaj (wyrażonego w minutach) 100 osób wybranych losowo. Zmienna X to czas oczekiwania na tramwaj. Jest to zmienna ilościowa ciągła.

- model: rozkład jednostajny
- $\mathcal{P} = \{U(0, b) : b \in (0, \infty)\}$
- $\Theta = (0, \infty)$ oraz $\theta = b$

Niech $\mathbf{X} = (X_1, \dots, X_n)^\top$ będzie próbą prostą z populacji o rozkładzie jednostajnym $U(0, b)$. $100(1 - \alpha)\%$ przedziałem ufności dla parametru b jest przedział losowy postaci:

$$\left(\frac{\max\{X_1, \dots, X_n\}}{\sqrt[n]{1 - \frac{\alpha}{2}}}, \frac{\max\{X_1, \dots, X_n\}}{\sqrt[n]{\frac{\alpha}{2}}} \right).$$

```
load(url("http://ls.home.amu.edu.pl/data_sets/czas_oczek_tramwaj.RData"))
# estymator b
(b_est <- max(czas_oczek_tramwaj))

## [1] 13.92
```

```

# przedział ufności dla b
b_conf_int <- function(x, conf_level = 0.95) {
  alpha <- 1 - conf_level
  n <- length(x)
  l <- max(x) / (1 - alpha / 2)^(1 / n)
  u <- max(x) / (alpha / 2)^(1 / n)
  return(c(l, u))
}
b_conf_int(czas_oczek_tramwaj)

## [1] 13.92352 14.44308

```

6.2 Zadania

Zadanie 1. Przebadano 200 losowo wybranych 5-sekundowych okresów pracy centrali telefonicznej. Rejestrowano liczbę zgłoszeń. Wyniki są zawarte w pliku Centrala.RData. Wykorzystując przyjęty wcześniej model statystyczny dla tych danych, wyznacz (trzema metodami) przedział ufności dla parametru rozkładu teoretycznego.

```

##      LCL      UCL
## 1.561968 1.932765

##      LCL      UCL
## 1.561968 1.932765

##      LCL      UCL
## 1.557187 1.922813

```

Zadanie 2. Zmienna w pliku awarie.txt opisuje wyniki 50 pomiarów czasu bezawaryjnej pracy danego urządzenia (w godzinach). Wykorzystując przyjęty na wykładzie model statystyczny dla tych danych wyznacz granice przedziału ufności dla wartości oczekiwanej i wariancji rozkładu teoretycznego.

```

##      UCL      LCL
## 850.0693 1483.8742

```

```

##      UCL      LCL
## 722617.9 2201882.5

```

Zadanie 3. Niech $\mathbf{X} = (X_1, \dots, X_n)^\top$ będzie próbą prostą z populacji o rozkładzie Rayleigha $R(\lambda)$, $\lambda > 0$.

1. Napisz funkcję `median_cint()`, która implementuje następujący przybliżony przedział ufności dla mediany $\sqrt{\lambda \ln 2}$ tego rozkładu:

$$\left(\sqrt{\ln(2) \frac{1}{n} \sum_{i=1}^n X_i^2 \left(1 - \frac{z(1-\alpha/2)}{\sqrt{n}} \right)}, \sqrt{\ln(2) \frac{1}{n} \sum_{i=1}^n X_i^2 \left(1 + \frac{z(1-\alpha/2)}{\sqrt{n}} \right)} \right),$$

gdzie $z(\beta)$ oznacza kwantyl rzędu β z rozkładu normalnego $N(0, 1)$. Funkcja ta powinna mieć dwa argumenty: `x` - wektor zawierający dane, `conf_level` - poziom ufności. Funkcja zwraca obiekt typu `list` klasy `confint` o następujących elementach: `title` - nazwa estymowanej funkcji parametrycznej, `est` - wartość ENW funkcji parametrycznej, `l` - lewy krańec przedziału ufności, `r` - prawy krańec przedziału ufności, `conf_level` - poziom ufności.

2. Następujące dane to pomiary średniej szybkości wiatru w odstępach 15 minutowych odnotowane wokół nowo powstającej elektrowni wiatrowej:

0.9	6.2	2.1	4.1	7.3
1.0	4.6	6.4	3.8	5.0
2.7	9.2	5.9	7.4	3.0
4.9	8.2	5.0	1.2	10.1
12.2	2.8	5.9	8.2	0.5

Teoretyczny rozkład średniej szybkości wiatru to rozkład Rayleigha $R(\lambda)$, $\lambda > 0$. Używając funkcji `median_cint()`, oblicz wartość ENW i krańce 95% przedziału ufności dla mediany średniej szybkości wiatru. **Wskazówka:** Przed wywołaniem funkcji `median_cint()`, najpierw załaduj następujące funkcje przeciążone `print()` i `summary()`:

```

print.confint <- function(x) {
  cat(x$conf_level * 100, "percent confidence interval:", "\n")
  cat(x$l, " ", x$r, "\n")
}

summary.confint <- function(x) {
  cat("\\n", "Confidence interval of", x$title, "\\n", "\\n")
  cat(x$conf_level * 100, "percent confidence interval:", "\\n")
  cat(x$l, " ", x$r, "\\n")
  cat("sample estimate", "\\n")
  cat(x$est, "\\n")
}

## 95 percent confidence interval:
## 3.863593 5.845955

## Confidence interval of mediana
## 95 percent confidence interval:
## 3.863593 5.845955
## sample estimate
## 4.954924

```

Zadanie 4. Dla danego wektora obserwacji i poziomu ufności napisz funkcję określającą granice przedziału ufności na poziomie ufności $1 - \alpha$, $\alpha \in (0, 1)$ dla wartości oczekiwanej w rozkładzie normalnym. Domyślny poziom ufności powinien wynosić 0,95. Następnie przeprowadź symulacje (z liczbą powtórzeń `nr = 1000`) sprawdzając prawdopodobieństwo pokrycia tego przedziału ufności (tj. prawdopodobieństwo, że ten przedział ufności zawiera wartość oczekiwana) dla rozkładów $N(1, 3)$, $\chi^2(3)$ i $Ex(3)$ osobno. Rozważ liczby obserwacji $n = 10, 50, 100$. Zinterpretuj wyniki. **Wskazówka:** Symulacja powinna przebiegać według następujących kroków:

1. Przyjmij poziom istotności, n , nr , rozkład generowanych danych oraz `temp = 0`.
2. Wygeneruj n obserwacji z zadanego rozkładu.
3. Wyznacz granice przedziału ufności dla danych wygenerowanych w kroku 2.

4. Jeśli teoretyczna wartość oczekiwana należy do przedziału otrzymanego w kroku 3, zwiększ temp o jeden.
5. Powtórz kroki 2-4 nr razy.
6. Wyznacz temp / nr .

```
## n = 10
```

```
## [1] 0.959
```

```
## [1] 0.901
```

```
## [1] 0.899
```

```
## n = 50
```

```
## [1] 0.941
```

```
## [1] 0.944
```

```
## [1] 0.941
```

```
## n = 100
```

```
## [1] 0.946
```

```
## [1] 0.942
```

```
## [1] 0.946
```


7 Testowanie hipotez statystycznych

7.1 Przykłady

Test t-Studenta dla jednej próby

Przykład. Automat produkuje tabliczki czekolady o nominalnej wadze $250g$. Podczas kontroli technicznej pobrano 16-elementową próbę tabliczek czekolady otrzymując wyniki:

```
242.2 243.8 252.8 245.4 245.6 253.6 247.3 238.7 241.6 242.8 251.1 246.8 247 245.6 242.2
```

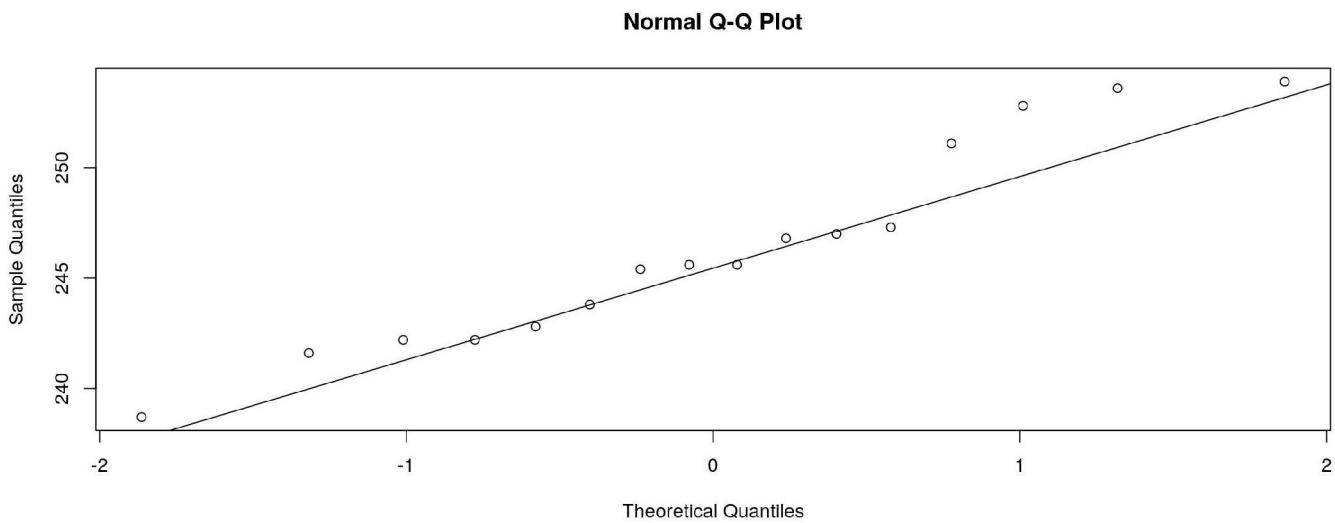


Na poziomie istotności $\alpha = 0,05$ zweryfikuj hipotezę, że automat rozleguował się i produkuje tabliczki czekolady o istotnie różnej wadze od nominalnej wagi.

```
x <- c(242.2, 243.8, 252.8, 245.4, 245.6, 253.6, 247.3, 238.7,
      241.6, 242.8, 251.1, 246.8, 247.0, 245.6, 242.2, 253.9)
shapiro.test(x)
```

```
## 
## Shapiro-Wilk normality test
##
## data: x
## W = 0.93622, p-value = 0.3052
```

```
qqnorm(x)
qqline(x)
```



```
mean(x)
```

```
## [1] 246.275
```

```
t.test(x, mu = 250, alternative = "less")
```

```
##
##  One Sample t-test
##
## data: x
## t = -3.2679, df = 15, p-value = 0.002595
## alternative hypothesis: true mean is less than 250
## 95 percent confidence interval:
##       -Inf 248.2732
## sample estimates:
## mean of x
## 246.275
```

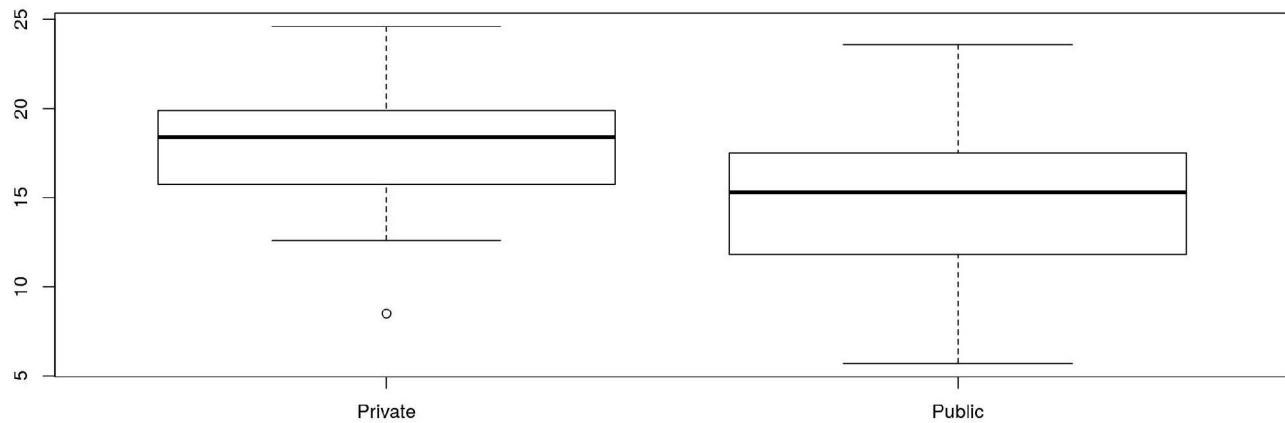
Test t-Studenta dla dwóch prób niezależnych i test F-Snedecora dla wariancji w dwóch próbach niezależnych

Przykład. Zbiór danych `homework` z pakietu `UsingR` zawiera informacje o ilości czasu poświęconego na odrabianie pracy domowej przez uczniów szkół prywatnych i publicznych. Naszym celem jest sprawdzenie, czy uczniowie obu typów szkół spędzają tyle samo czasu na

odrabianiu zadań domowych.

```
library(UsingR)  
head(homework)  
  
##    Private Public  
## 1    21.3   15.3  
## 2    16.8   17.4  
## 3     8.5   12.3  
## 4    12.6   10.7  
## 5    15.8   16.4  
## 6    19.3   11.3
```

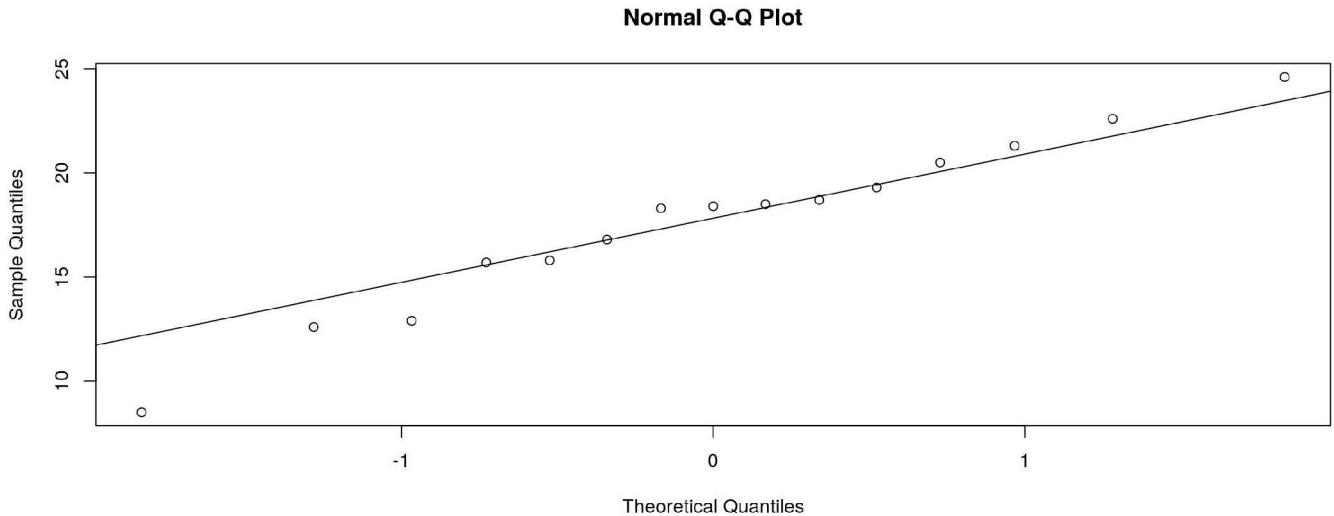
```
boxplot(homework)
```



```
shapiro.test(homework$Private)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: homework$Private  
## W = 0.97017, p-value = 0.8606
```

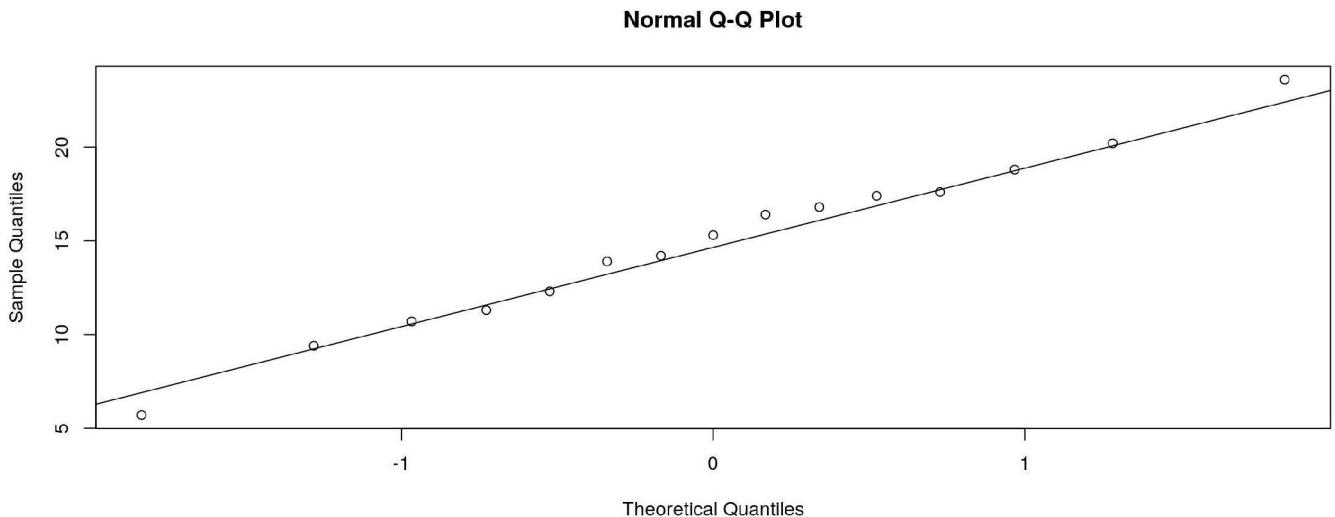
```
qqnorm(homework$Private)
qqline(homework$Private)
```



```
shapiro.test(homework$Public)
```

```
##
##  Shapiro-Wilk normality test
##
## data: homework$Public
## W = 0.99275, p-value = 0.9999
```

```
qqnorm(homework$Public)
qqline(homework$Public)
```



```
var(homework$Private)
```

```
## [1] 17.1081
```

```
var(homework$Public)
```

```
## [1] 20.87781
```

```
var.test(homework$Private, homework$Public, alternative = "less")
```

```
##
## F test to compare two variances
##
## data: homework$Private and homework$Public
## F = 0.81944, num df = 14, denom df = 14, p-value = 0.3573
## alternative hypothesis: true ratio of variances is less than 1
## 95 percent confidence interval:
## 0.000000 2.035262
## sample estimates:
## ratio of variances
## 0.8194392
```

```

mean(homework$Private)

## [1] 17.63333

mean(homework$Public)

## [1] 14.90667

t.test(homework$Private, homework$Public,
       var.equal = TRUE, alternative = 'greater')

## 
##  Two Sample t-test
##
## data: homework$Private and homework$Public
## t = 1.7134, df = 28, p-value = 0.04884
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  0.01957252      Inf
## sample estimates:
## mean of x mean of y
## 17.63333 14.90667

```

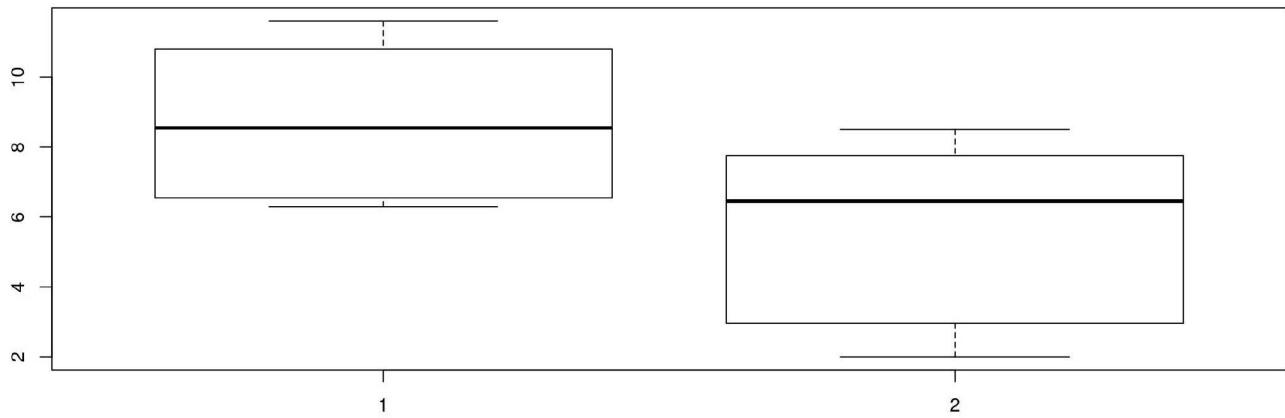
Test t-Studenta dla prób zależnych

Przykład. Badano wpływ hipnozy na redukcję bólu. Notowano poziom odczuwalnego bólu:

- przed hipnozą: 6.6, 6.5, 9.0, 10.3, 11.3, 8.1, 6.3, 11.6,
- po hipnozie: 6.8, 2.5, 7.4, 8.5, 8.1, 6.1, 3.4, 2.0.

Czy na poziomie istotności 0,05 możemy stwierdzić, że hipnoza redukuje poziom odczuwalnego bólu?

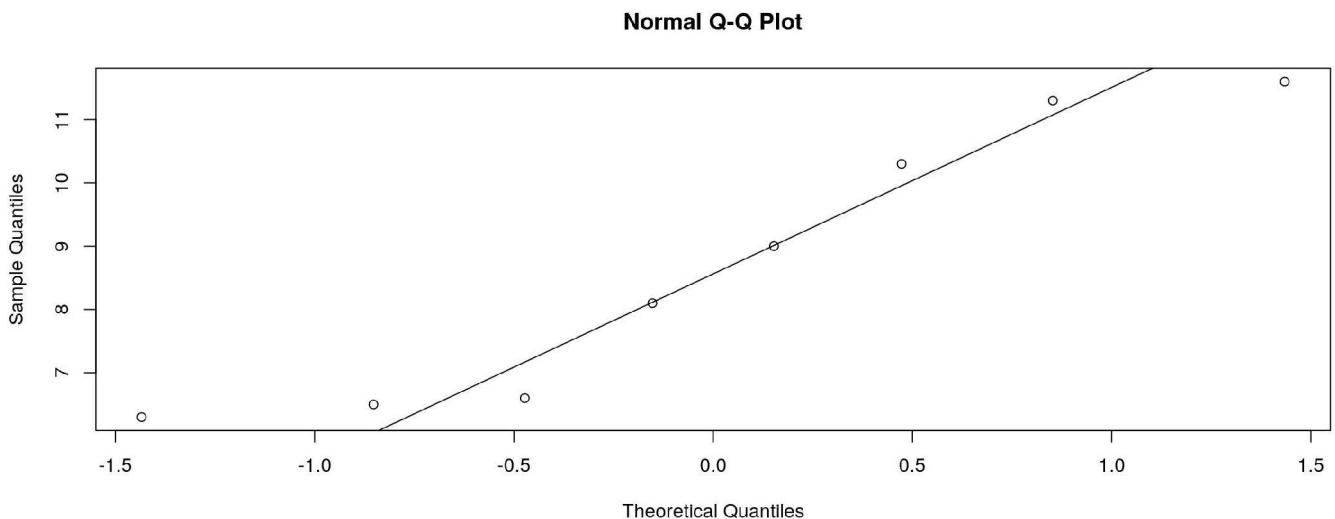
```
a <- c(6.6, 6.5, 9.0, 10.3, 11.3, 8.1, 6.3, 11.6)
b <- c(6.8, 2.5, 7.4, 8.5, 8.1, 6.1, 3.4, 2.0)
boxplot(a, b)
```



```
shapiro.test(a)
```

```
##
##  Shapiro-Wilk normality test
##
## data: a
## W = 0.88638, p-value = 0.2165
```

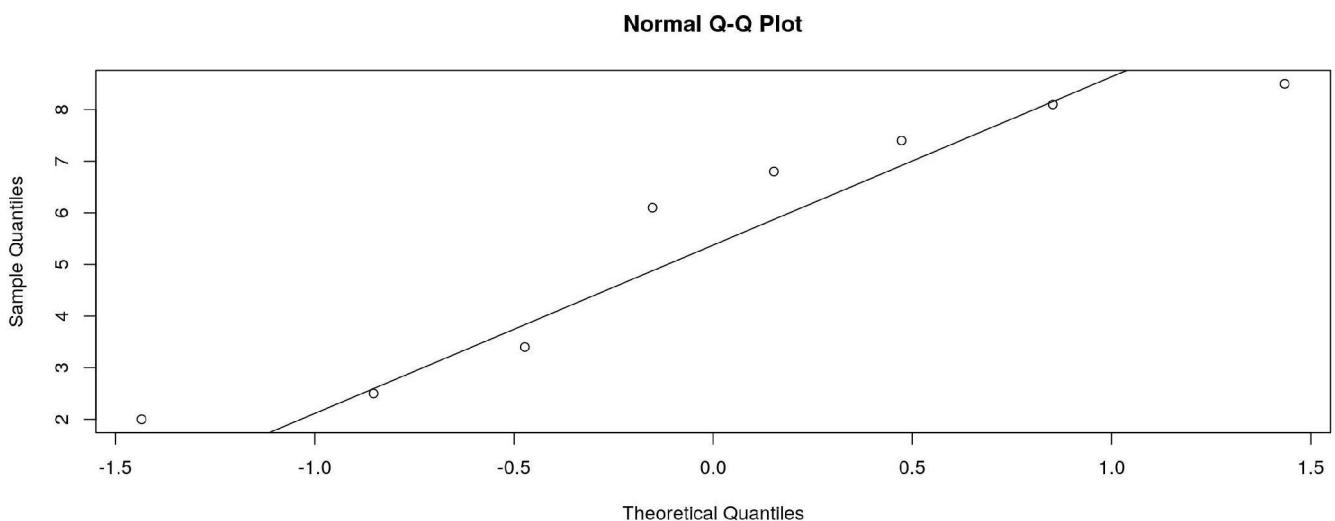
```
qqnorm(a)
qqline(a)
```



```
shapiro.test(b)
```

```
##  
## Shapiro-Wilk normality test  
##  
## data: b  
## W = 0.88356, p-value = 0.2036
```

```
qqnorm(b)  
qqline(b)
```



```
mean(a)
```

```

## [1] 8.7125

mean(b)

## [1] 5.6

t.test(a, b, alternative = 'greater', paired = TRUE)

## 
##  Paired t-test
##
## data: a and b
## t = 3.0285, df = 7, p-value = 0.009577
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
##  1.165386      Inf
## sample estimates:
## mean of the differences
##                         3.1125

```

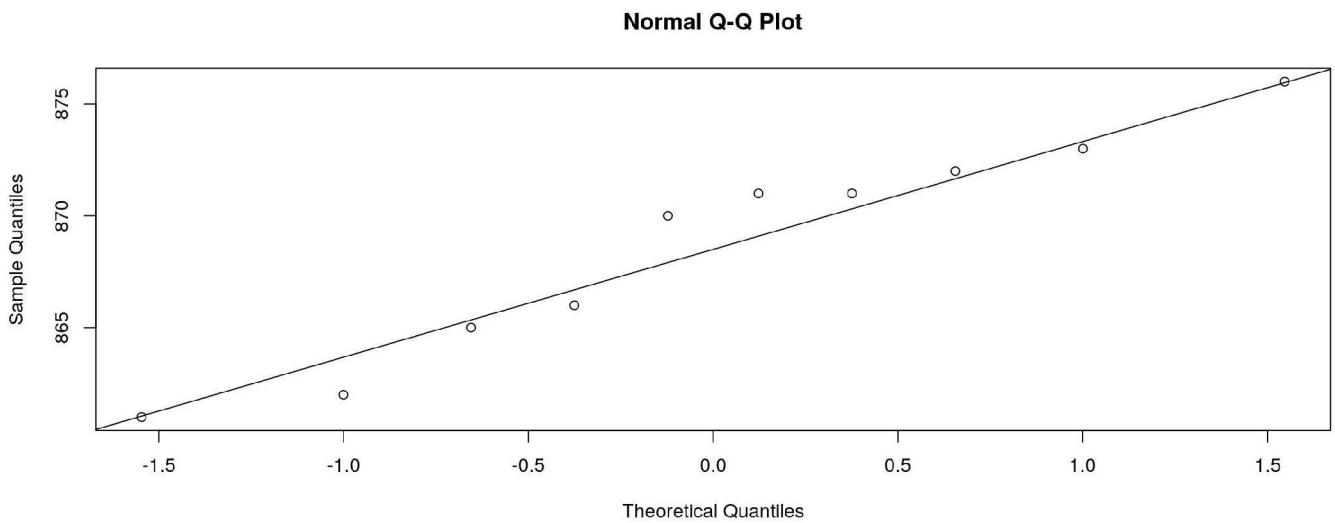
7.2 Zadania

Zadanie 1. W pewnym regionie wykonano dziesięć niezależnych pomiarów głębokości morza i uzyskano następujące wyniki: 862, 870, 876, 866, 871, 865, 861, 873, 871, 872. Na poziomie istotności $\alpha = 0,05$ zweryfikuj hipotezę, że średnia głębokość morza w tym regionie wynosi 870m.

```

## [1] 0.545861

```



```
## [1] 868.7
```

```
## [1] 0.2136555
```

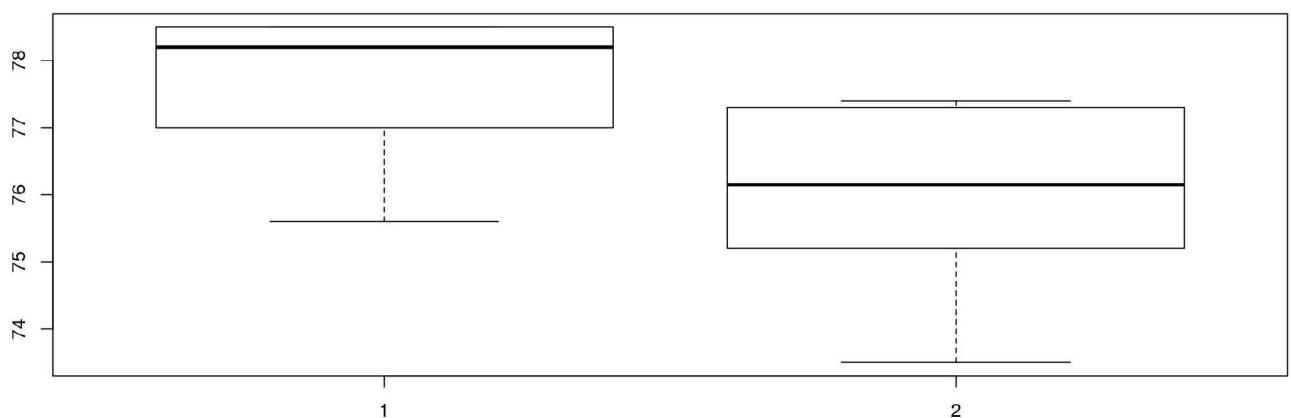
Zadanie 2. Producent proszku do prania *A* twierdzi, że jego produkt jest znacznie lepszy niż konkurencyjny proszek *B*. Aby zweryfikować to zapewnienie, CTA (Consumer Test Agency) przetestowało oba proszki do prania. W tym celu przeprowadzono pomiary stopnia wyprania 7 kawałków tkaniny z proszkiem *A* i uzyskano wyniki (w %):

78,2; 78,5; 75,6; 78,5; 78,5; 77,4; 76,6,

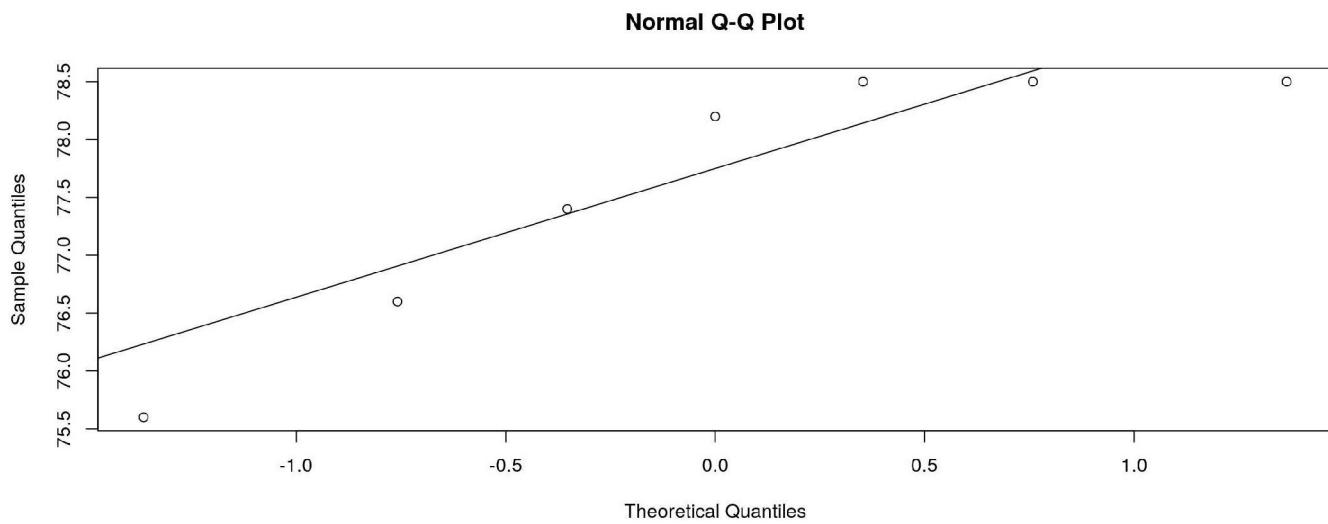
i 10 kawałków tkaniny z proszkiem *B* otrzymując wyniki (w %):

76,1; 75,2; 75,8; 77,3; 77,3; 77,0; 74,4; 76,2; 73,5; 77,4.

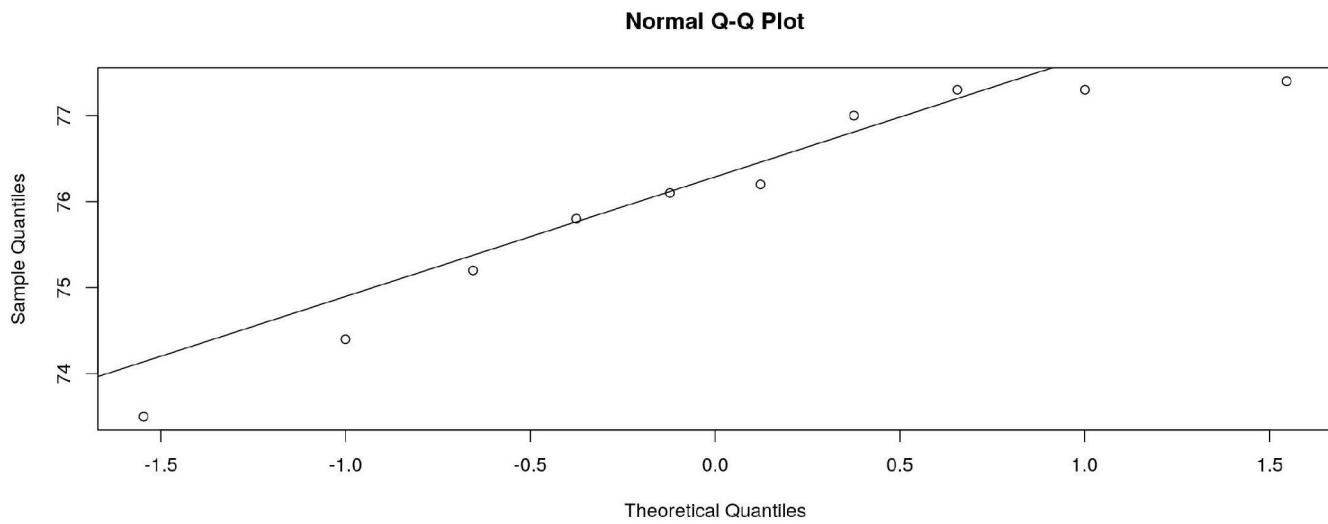
Jaki powinien być wniosek CTA na temat jakości tych proszków?



```
## [1] 0.06832755
```



```
## [1] 0.2558752
```



```
## [1] 1.304762
```

```
## [1] 1.764
```

```
## [1] 0.3683809
```

```
## [1] 77.61429
```

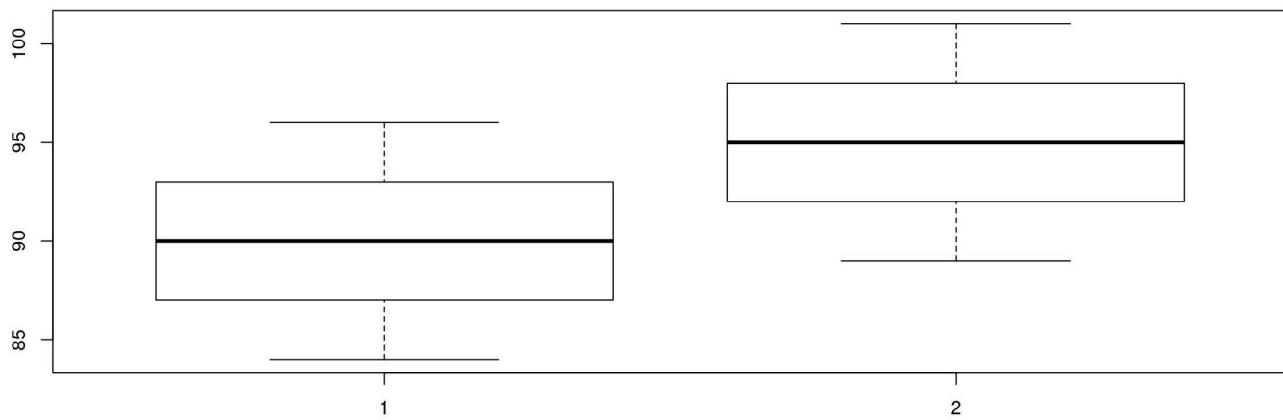
```
## [1] 76.02
```

```
## [1] 0.01059375
```

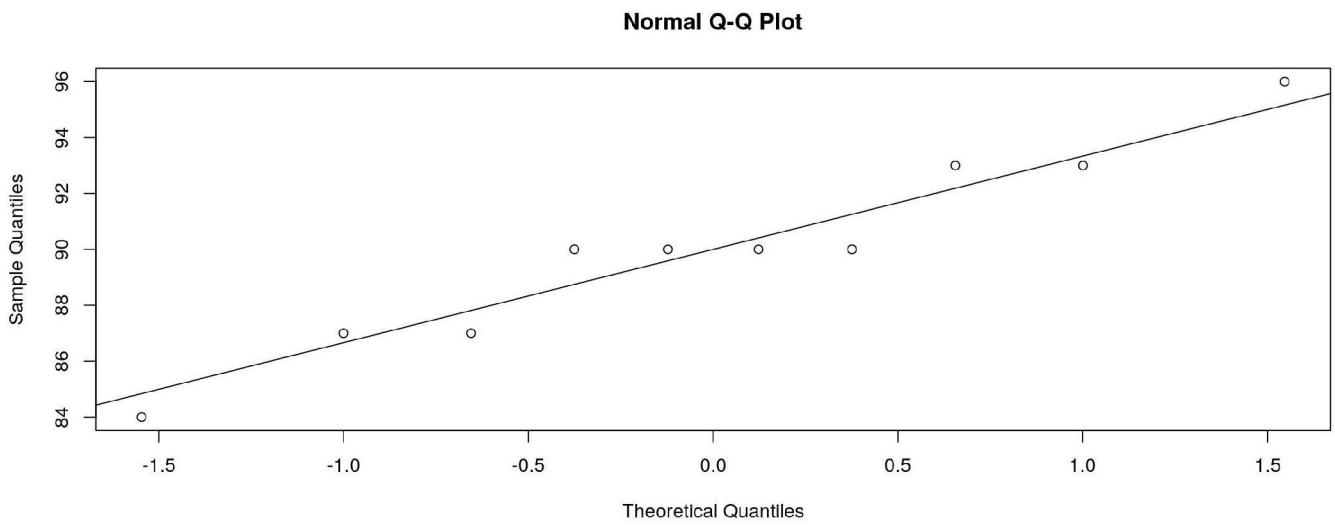
Zadanie 3. Grupa 10 osób została poddana badaniu mającym na celu zbadanie stosunku do szkół publicznych. Następnie grupa obejrzała film edukacyjny mający na celu poprawę podejścia do tego typu szkół. Wyniki są następujące (wyższa wartość oznacza lepsze podejście):

- przed: 84, 87, 87, 90, 90, 90, 90, 93, 93, 96,
- po: 89, 92, 98, 95, 95, 92, 95, 92, 98, 101.

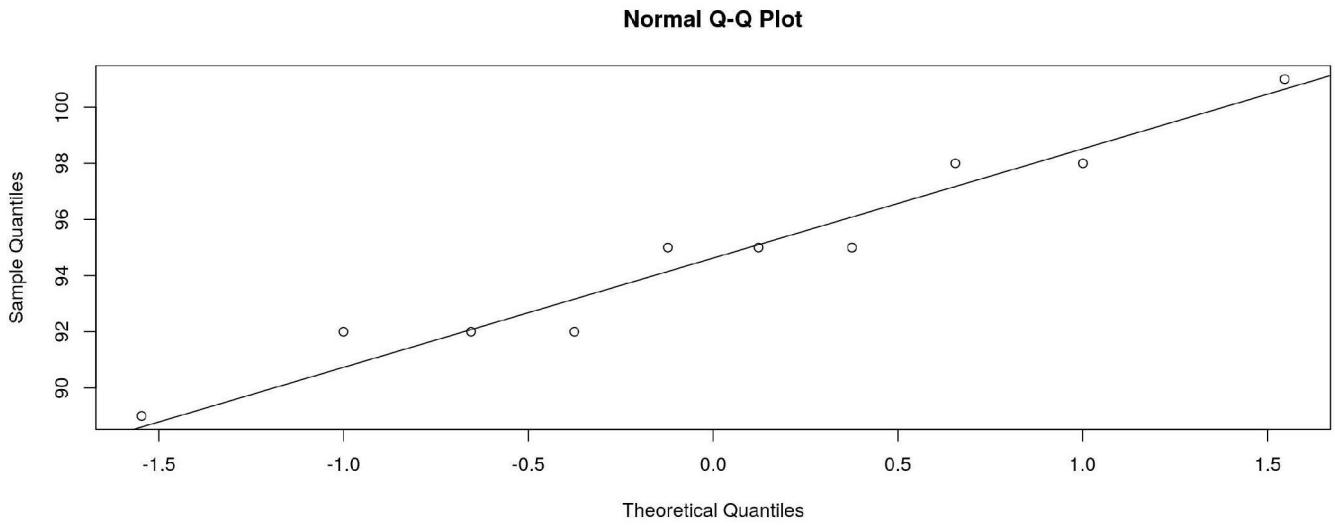
Zweryfikuj, czy film znacznie poprawia stosunek do szkół publicznych.



```
## [1] 0.7025892
```



```
## [1] 0.691489
```



```
## [1] 90
```

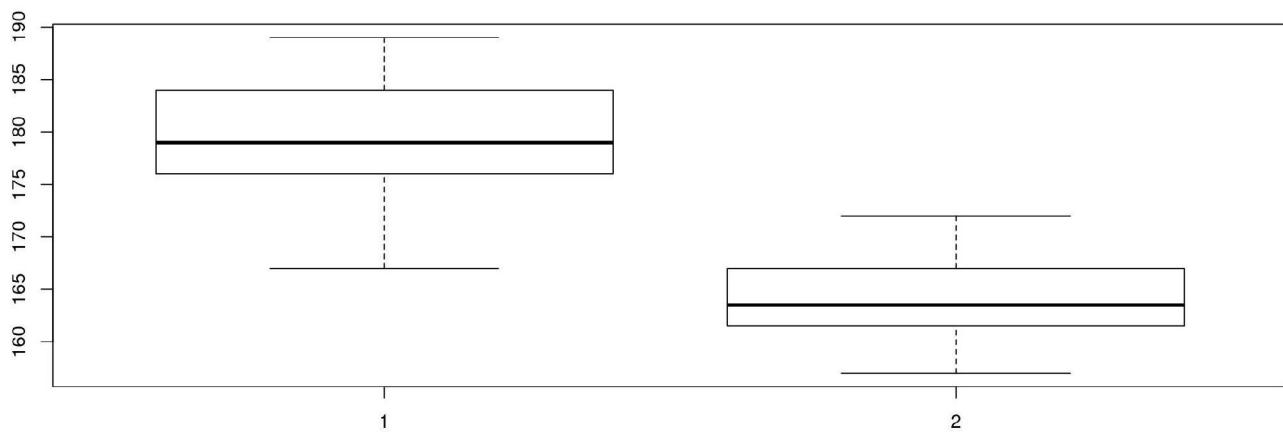
```
## [1] 94.7
```

```
## [1] 0.0003786878
```

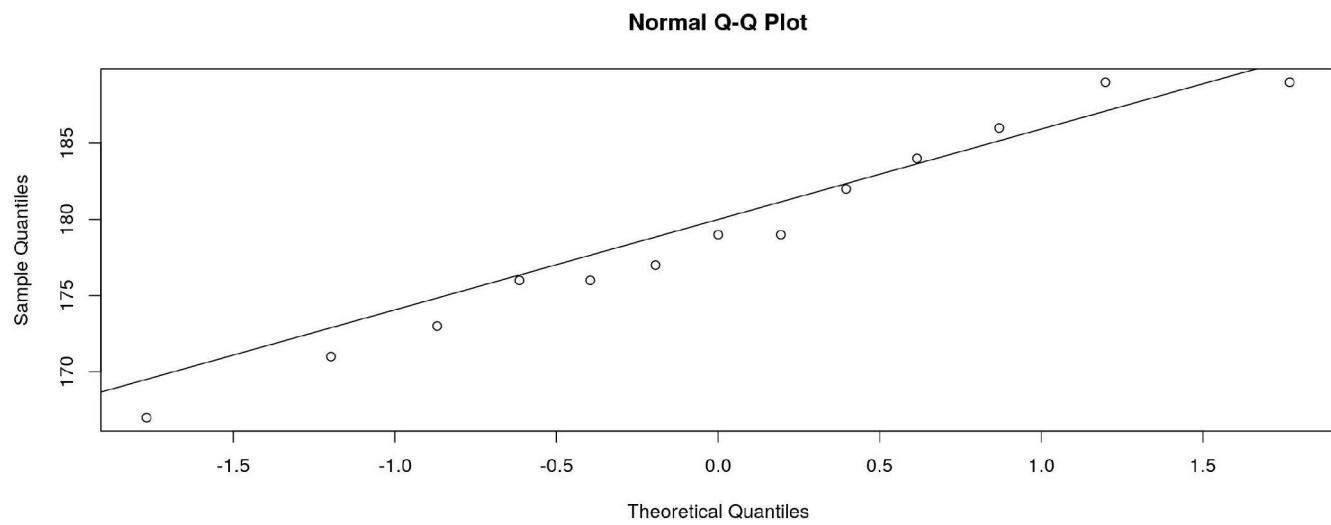
Zadanie 4. Zbadano wzrost 13 mężczyzn i 12 kobiet w pewnym centrum sportowym. Wyniki są następujące:

- mężczyźni: 171, 176, 179, 189, 176, 182, 173, 179, 184, 186, 189, 167, 177,
- kobiety: 161, 162, 163, 162, 166, 164, 168, 165, 168, 157, 161, 172.

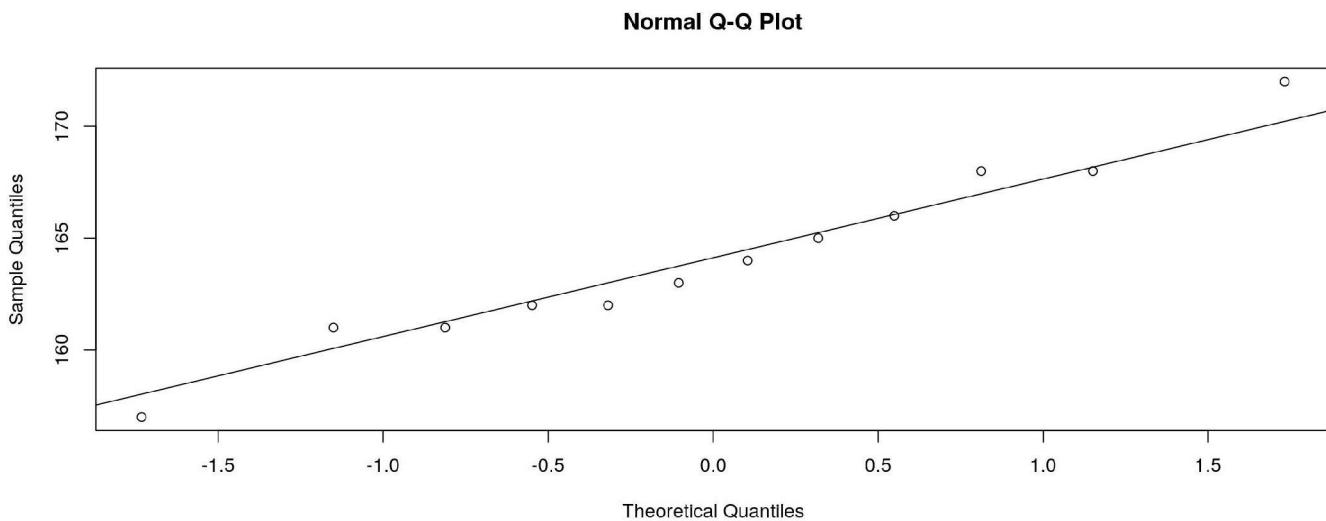
Czy możemy stwierdzić, że średni wzrost mężczyzn jest znacznie większy niż wzrost kobiet?



```
## [1] 0.8595396
```



```
## [1] 0.9447828
```



```
## [1] 45.74359
```

```
## [1] 16.08333
```

```
## [1] 0.04689163
```

```
## [1] 179.0769
```

```
## [1] 164.0833
```

```
## [1] 6.928802e-07
```

Zadanie 5.

- a. Napisz funkcję `w_test()` implementującą test χ^2 w modelu wykładniczym, który jest opisany we wskazówce. Funkcja ta powinna mieć trzy argumenty: `x` - wektor zawierający dane, `lambda_zero` - wartość λ_0 w hipotezie zerowej oraz `alternative` - typ hipotezy alternatywnej, która może mieć trzy możliwe wartości: "two.sided" (wartość domyślna), "greater", "less". Funkcja zwraca obiekt będący listą klasy `htest` o elementach: `statistic` - wartość statystyki testowej, `parameter` - liczba stopni swobody, `p.value` - p -wartość, `alternative` - wybrana hipoteza alternatywna, `method` - nazwa testu, `data.name` - nazwa zbioru danych (użyj `deparse(substitute(x))`). Dla

obiektów klasy `htest` funkcja `print()` istnieje w programie R, więc nie trzeba jej tworzyć.

Wskazówka. Niech $\mathbf{X} = (X_1, \dots, X_n)^\top$ będzie próbą prostą z populacji o rozkładzie wykładniczym $Ex(\lambda)$, gdzie $\lambda > 0$ jest nieznanym parametrem. Testy χ^2 w modelu wykładniczym weryfikują hipotezę zerową $H_0 : \lambda = \lambda_0$, gdzie $\lambda_0 > 0$ jest ustaloną liczbą. Ich obszary krytyczne są następujące:

1. dla $H_1^{(1)} : \lambda > \lambda_0$

$$R = \{\mathbf{x} : T(\mathbf{x}) \leq \chi^2(\alpha, 2n)\},$$

2. dla $H_1^{(2)} : \lambda < \lambda_0$

$$R = \{\mathbf{x} : T(\mathbf{x}) \geq \chi^2(1 - \alpha, 2n)\},$$

3. dla $H_1^{(3)} : \lambda \neq \lambda_0$

$$R = \{\mathbf{x} : T(\mathbf{x}) \geq \chi^2(1 - \alpha/2, 2n) \text{ or } T(\mathbf{x}) \leq \chi^2(\alpha/2, 2n)\},$$

gdzie

$$T(\mathbf{X}) = 2\lambda_0 n \bar{X} \Big|_{H_0} \sim \chi^2(2n)$$

jest statystyką testową, a $\chi^2(\beta, m)$ oznacza kwantyl rzędu β z rozkładu chi-kwadrat $\chi^2(m)$ z m stopniami swobody.

b. Wykorzystując funkcję `w_test()` zastosuj test χ^2 w modelu wykładniczym do danych dotyczących czasu bezawaryjnej pracy dostępnych w pliku `awarie.txt` i hipotezy zerowej $H_0 : \lambda = 0.001$.

```
## [1] 0.0009079683
```

```
##
## Test chi-kwadrat w modelu wykładniczym
##
## data:  awarie$V1
## T = 110.14, num df = 100, p-value = 0.2295
## alternative hypothesis: less
```

8 Analiza wariancji

8.1 Przykład

Przykład. Zbiór danych `vaccination` z pakietu `PBImisc` zawiera dane opisujące reakcję organizmu na zwalczanie wirusa po podaniu określonej dawki leku. Problem praktyczny dotyczy ustalenia, jaką najmniejszą możliwą dawkę leku należy podać, aby wywołać pożądaną reakcję organizmu (zagadnienie najmniejszej dawki leku). Rozważane jest również zagadnienie maksymalnej bezpiecznej dawki, którego celem jest określenie, jaka maksymalna dawka może być przyjmowana bez dużego ryzyka wystąpienia efektów ubocznych.

```
library(PBImisc)
head(vaccination)

##      response dose
## 1      88.9 0 ml
## 2     105.0 0 ml
## 3     138.4 0 ml
## 4      98.1 0 ml
## 5     107.2 0 ml
## 6      57.9 0 ml

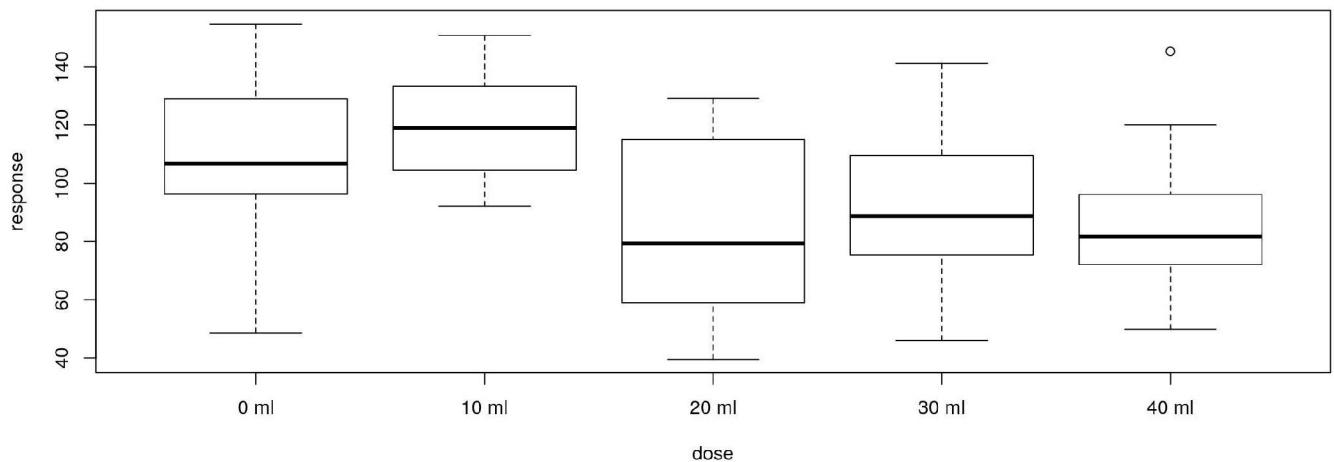
summary(vaccination)

##      response          dose
##  Min.   : 39.50   0 ml:20
##  1st Qu.: 77.30   10 ml:20
##  Median : 99.25   20 ml:20
##  Mean   : 97.89   30 ml:20
##  3rd Qu.:117.70   40 ml:20
##  Max.   :154.70
```

```
aggregate(vaccination$response,
  list(DOSE = vaccination$dose),
  FUN = mean)
```

```
##      DOSE      x
## 1  0 ml 108.570
## 2 10 ml 119.265
## 3 20 ml  84.025
## 4 30 ml  92.370
## 5 40 ml  85.220
```

```
boxplot(response ~ dose, data = vaccination)
```



```
summary(aov(response ~ dose, data = vaccination))
```

```
##           Df Sum Sq Mean Sq F value    Pr(>F)
## dose        4 19084   4771   7.929 1.47e-05 ***
## Residuals  95 57164     602
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# założenia

shapiro.test(lm(response ~ dose, data = vaccination)$residuals)

## 
## Shapiro-Wilk normality test
##
## data: lm(response ~ dose, data = vaccination)$residuals
## W = 0.99244, p-value = 0.8524


bartlett.test(response ~ dose, data = vaccination)

## 
## Bartlett test of homogeneity of variances
##
## data: response by dose
## Bartlett's K-squared = 5.6387, df = 4, p-value = 0.2278


fligner.test(response ~ dose, data = vaccination)

## 
## Fligner-Killeen test of homogeneity of variances
##
## data: response by dose
## Fligner-Killeen:med chi-squared = 4.8066, df = 4, p-value = 0.3077


library(car)

leveneTest(response ~ dose, data = vaccination)

## Levene's Test for Homogeneity of Variance (center = median)
##      Df F value Pr(>F)
## group  4  1.3679 0.2509
##          95
```

```
leveneTest(response ~ dose, data = vaccination, center = "mean")

## Levene's Test for Homogeneity of Variance (center = "mean")
##       Df F value Pr(>F)
## group  4  1.6203 0.1755
##      95

# testy post hoc
attach(vaccination)
pairwise.t.test(response, dose, data = vaccination)

##
##  Pairwise comparisons using t tests with pooled SD
##
## data: response and dose
##
##      0 ml   10 ml   20 ml   30 ml
## 10 ml 0.68485 -      -      -
## 20 ml 0.01463 0.00016 -      -
## 30 ml 0.19718 0.00633 0.85424 -
## 40 ml 0.02007 0.00027 0.87790 0.85424
##
## P value adjustment method: holm

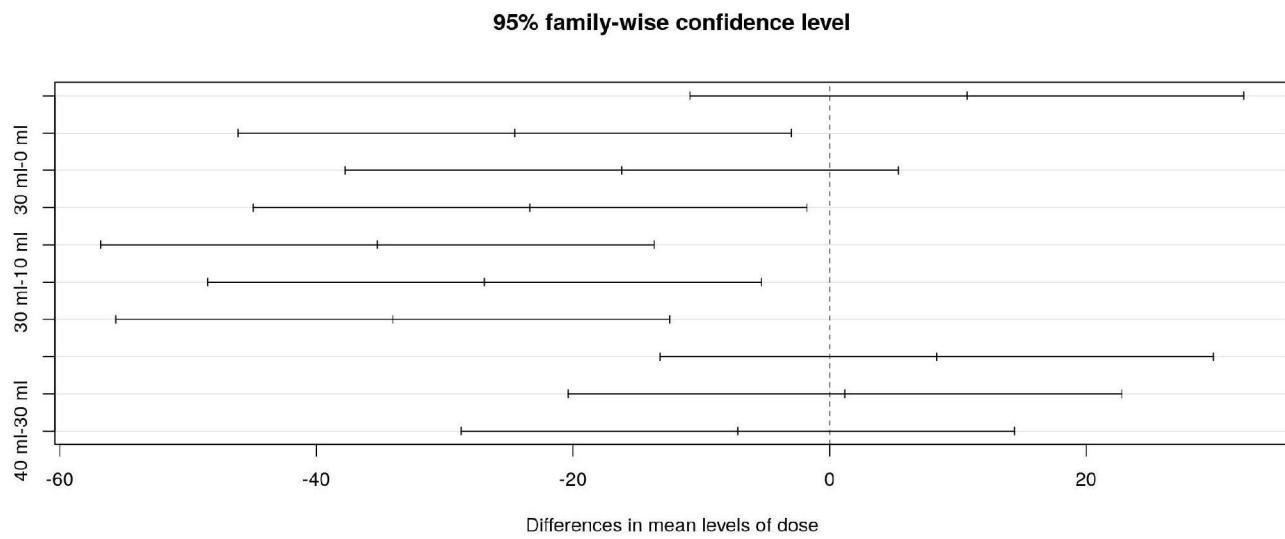
model_aov <- aov(response ~ dose, data = vaccination)
TukeyHSD(model_aov)
```

```

## Tukey multiple comparisons of means
## 95% family-wise confidence level
##
## Fit: aov(formula = response ~ dose, data = vaccination)
##
## $dose
##          diff      lwr       upr     p adj
## 10 ml-0 ml 10.695 -10.87643 32.266431 0.6426874
## 20 ml-0 ml -24.545 -46.11643 -2.973569 0.0174170
## 30 ml-0 ml -16.200 -37.77143  5.371431 0.2336465
## 40 ml-0 ml -23.350 -44.92143 -1.778569 0.0270291
## 20 ml-10 ml -35.240 -56.81143 -13.668569 0.0001562
## 30 ml-10 ml -26.895 -48.46643 -5.323569 0.0069317
## 40 ml-10 ml -34.045 -55.61643 -12.473569 0.0002808
## 30 ml-20 ml   8.345 -13.22643 29.916431 0.8185005
## 40 ml-20 ml   1.195 -20.37643 22.766431 0.9998712
## 40 ml-30 ml  -7.150 -28.72143 14.421431 0.8878461

```

```
plot(TukeyHSD(model_aov))
```



```

library(agricolae)
HSD.test(model_aov, "dose", console = TRUE)

```

```

## 
## Study: model_aov ~ "dose"
## 
## HSD Test for response
## 
## Mean Square Error: 601.7253
## 
## dose, means
## 
##      response      std   r   Min   Max
## 0 ml  108.570 25.91789 20 48.6 154.7
## 10 ml 119.265 17.64743 20 92.1 150.8
## 20 ml  84.025 30.42350 20 39.5 129.1
## 30 ml  92.370 24.27206 20 46.0 141.1
## 40 ml  85.220 22.59946 20 49.8 145.3
## 
## Alpha: 0.05 ; DF Error: 95
## Critical Value of Studentized Range: 3.932736
## 
## Minimun Significant Difference: 21.57143
## 
## Treatments with the same letter are not significantly different.
## 
##      response groups
## 10 ml    119.265     a
## 0 ml    108.570    ab
## 30 ml    92.370    bc
## 40 ml    85.220     c
## 20 ml    84.025     c

SNK.test(model_aov, "dose", console = TRUE)

```

```
##  
## Study: model_aov ~ "dose"  
##  
## Student Newman Keuls Test  
## for response  
##  
## Mean Square Error: 601.7253  
##  
## dose, means  
##  
##      response     std   r   Min   Max  
## 0 ml  108.570 25.91789 20 48.6 154.7  
## 10 ml 119.265 17.64743 20 92.1 150.8  
## 20 ml  84.025 30.42350 20 39.5 129.1  
## 30 ml  92.370 24.27206 20 46.0 141.1  
## 40 ml  85.220 22.59946 20 49.8 145.3  
##  
## Alpha: 0.05 ; DF Error: 95  
##  
## Critical Range  
##      2       3       4       5  
## 15.39978 18.46964 20.28552 21.57143  
##  
## Means with the same letter are not significantly different.  
##  
##      response groups  
## 10 ml  119.265     a  
## 0 ml  108.570     a  
## 30 ml  92.370     b  
## 40 ml  85.220     b  
## 20 ml  84.025     b
```



```
LSD.test(model_aov, "dose", p.adj = "holm", console = TRUE)
```

```

## 
## Study: model_aov ~ "dose"
## 
## LSD t Test for response
## P value adjustment method: holm
## 
## Mean Square Error: 601.7253
## 
## dose, means and individual ( 95 %) CI
## 
##      response     std   r      LCL      UCL   Min   Max
## 0 ml  108.570 25.91789 20  97.68071 119.45929 48.6 154.7
## 10 ml 119.265 17.64743 20 108.37571 130.15429 92.1 150.8
## 20 ml  84.025 30.42350 20  73.13571  94.91429 39.5 129.1
## 30 ml  92.370 24.27206 20  81.48071 103.25929 46.0 141.1
## 40 ml  85.220 22.59946 20  74.33071  96.10929 49.8 145.3
## 
## Alpha: 0.05 ; DF Error: 95
## Critical Value of t: 2.874073
## 
## Minimum Significant Difference: 22.29446
## 
## Treatments with the same letter are not significantly different.
## 
##      response groups
## 10 ml  119.265      a
## 0 ml  108.570     ab
## 30 ml  92.370     bc
## 40 ml  85.220      c
## 20 ml  84.025      c

```

```
scheffe.test(model_aov, "dose", console = TRUE)
```

```

## 
## Study: model_aov ~ "dose"
## 
## Scheffe Test for response
## 
## Mean Square Error : 601.7253
## 
## dose, means
## 
##      response      std   r   Min   Max
## 0 ml  108.570 25.91789 20  48.6 154.7
## 10 ml 119.265 17.64743 20  92.1 150.8
## 20 ml  84.025 30.42350 20  39.5 129.1
## 30 ml  92.370 24.27206 20  46.0 141.1
## 40 ml  85.220 22.59946 20  49.8 145.3
## 
## Alpha: 0.05 ; DF Error: 95
## Critical Value of F: 2.467494
## 
## Minimum Significant Difference: 24.37009
## 
## Means with the same letter are not significantly different.
## 
##      response groups
## 10 ml  119.265      a
## 0 ml  108.570     ab
## 30 ml  92.370     bc
## 40 ml  85.220     bc
## 20 ml  84.025      c

# analiza kontrastów
# przykładowe kontrasty wbudowane w programie R
contr.helmert(5)

```

```
##   [,1] [,2] [,3] [,4]
## 1    -1    -1    -1    -1
## 2     1    -1    -1    -1
## 3     0     2    -1    -1
## 4     0     0     3    -1
## 5     0     0     0     4

library(multcomp)
# kontrasty dla postępujących różnic
contr.sdif(5)

##   2-1 3-2 4-3 5-4
## 1 -0.8 -0.6 -0.4 -0.2
## 2  0.2 -0.6 -0.4 -0.2
## 3  0.2  0.4 -0.4 -0.2
## 4  0.2  0.4  0.6 -0.2
## 5  0.2  0.4  0.6  0.8

contrasts(vaccination$dose) <- contr.sdif(5)
vaccination$dose
```

```

## [1] 0 ml 0 ml
## [12] 0 ml 10 ml 10 ml
## [23] 10 ml 10 ml
## [34] 10 ml 20 ml 20 ml 20 ml 20 ml
## [45] 20 ml 20 ml
## [56] 20 ml 20 ml 20 ml 20 ml 30 ml
## [67] 30 ml 30 ml
## [78] 30 ml 30 ml 40 ml
## [89] 40 ml 40 ml
## [100] 40 ml

## attr(,"contrasts")
##      2-1 3-2 4-3 5-4
## 0 ml -0.8 -0.6 -0.4 -0.2
## 10 ml 0.2 -0.6 -0.4 -0.2
## 20 ml 0.2 0.4 -0.4 -0.2
## 30 ml 0.2 0.4 0.6 -0.2
## 40 ml 0.2 0.4 0.6 0.8
## Levels: 0 ml 10 ml 20 ml 30 ml 40 ml

```

```

model.2 <- aov(response ~ dose, data = vaccination)
summary(model.2,
        split = list(dose = list('C1' = 1, 'C2' = 2, 'C3' = 3, 'C4' = 4)))

##          Df Sum Sq Mean Sq F value    Pr(>F)
## dose       4 19084   4771   7.929 1.47e-05 ***
## dose: C1   1   2852   2852   4.739   0.032 *
## dose: C2   1 15418   15418  25.622 2.03e-06 ***
## dose: C3   1    303     303   0.504   0.479
## dose: C4   1    511     511   0.850   0.359
## Residuals 95 57164    602
## ---
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

8.2 Zadania

Zadanie 1. Zadanie to zostało opracowane na podstawie eksperymentu Smitha (1979). Głównym jego celem było pokazanie, że bycie w tym samym kontekście psychicznym w czasie nauki i podczas jej sprawdzania (test, egzamin) daje lepsze wyniki niż bycie w odmiennych kontekstach. Podczas fazy uczącej uczniowie uczyli się 80 słów w pokoju pomalowanym na pomarańczowo, ozdobionym plakatami, obrazami i dużą ilością dodatkowych akcesoriów. Pierwszy sprawdzian pamięci został przeprowadzony aby dać uczniom wrażenie, że eksperyment się zakończył. Następnego dnia, uczniowie zostali niespodziewanie poddani testowi ponownie. Mieli napisać wszystkie słowa, które zapamiętali. Test został przeprowadzony w 5 różnych warunkach. 50 uczniów zostało losowo podzielonych na 5 równolicznych grup:

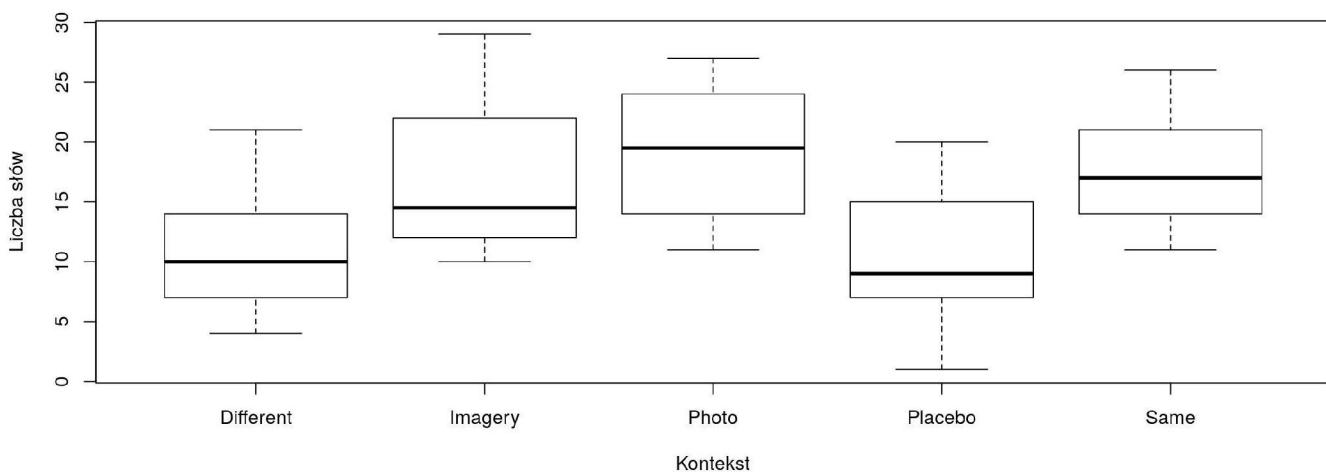
- „Same context” - test odbywał się w tym samym pokoju, w którym się uczyli.
- „Different context” - test odbywał się w bardzo odmiennym pomieszczeniu, w innej części kampusu, pomalowanym na szaro i wyglądającym bardzo surowo.
- „Imaginary context” - test odbywał się w tym samym pomieszczeniu, co w punkcie poprzednim. Dodatkowo, uczniowie mieli przypomnieć sobie pokój, w którym się uczyli. Aby im w tym pomóc badacz zadawał dodatkowe pytania o pokoju i jego wyposażeniu.
- „Photographed context” - test odbywał się w tych samych warunkach, co w punkcie poprzednim. Dodatkowo pokazano im zdjęcie pokoju, w którym się uczyli.
- „Placebo context” - test odbywał się w tym samych warunkach co grupy „Different context”. Dodatkowo uczniowie wykonali ćwiczenia „rozgrzewające” (przypominanie sobie swojego salonu).

Liczba zapamiętanych słów została zawarta w poniższej tabeli.

Same	Different	Imagery	Photo	Placebo
25	11	14	25	8
26	21	15	15	20
17	9	29	23	10
15	6	10	21	7
14	7	12	18	15
17	14	22	24	7
14	12	14	14	1
20	4	20	27	17
11	7	22	12	11
21	19	12	11	4

1. Wyznacz średnie liczby zapamiętanych słów w grupach. Ponadto, przedstaw otrzymane dane za pomocą wykresu ramkowego dla każdej grupy z osobna.

```
##      CONTEXT x
## 1 Different 11
## 2 Imagery 17
## 3 Photo 19
## 4 Placebo 10
## 5 Same 18
```

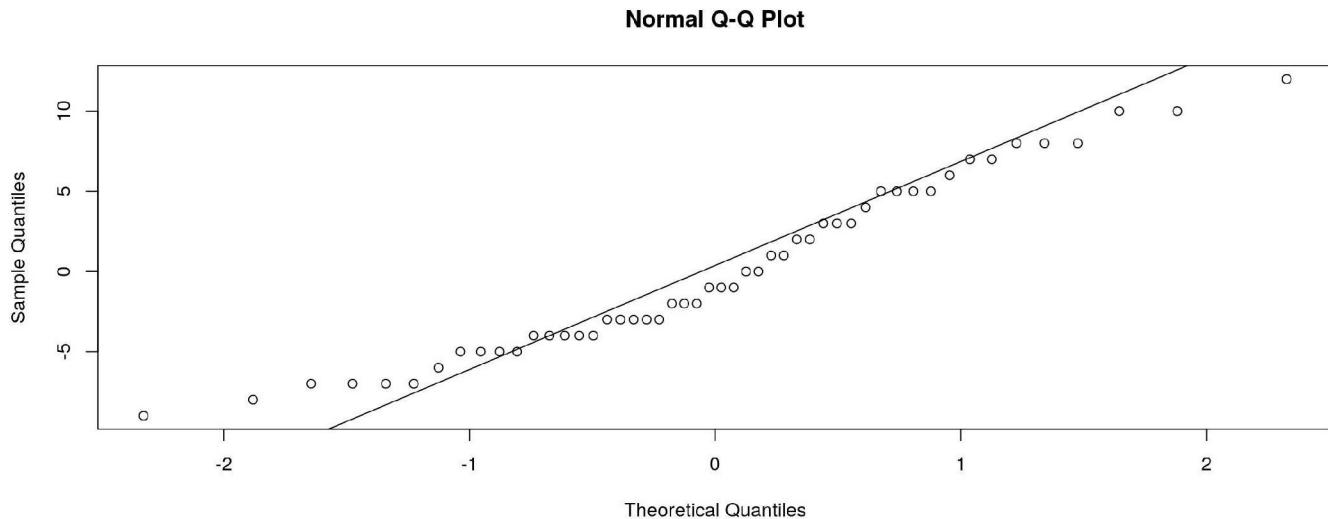


2. Wykonaj test analizy wariancji w celu sprawdzenia, czy liczba zapamiętanych słów zależy od kontekstu sprawdzania wiedzy.

```
##               Df Sum Sq Mean Sq F value    Pr(>F)
## context        4    700     175   5.469 0.00112 **
## Residuals    45   1440      32
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

3. Sprawdź założenia modelu jednoczynnikowej analizy wariancji.

```
## [1] 0.05635956
```



```
## [1] 0.9817694
```

```
## [1] 0.9759731
```

```
## [1] 0.9550502
```

```
## [1] 0.9281122
```

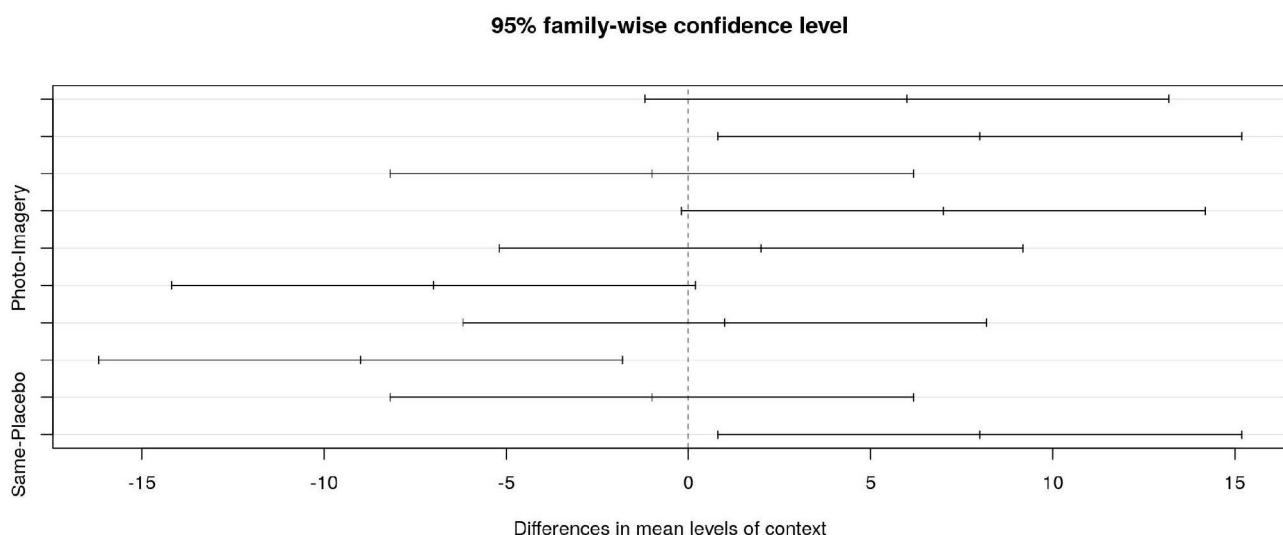
4. Wykonaj testy post hoc w celu sprawdzenia, które konteksty sprawdzania wiedzy różnią się między sobą.

```

## 
##  Pairwise comparisons using t tests with pooled SD
## 
## data:  number and context
## 
##          Different Imagery Photo Placebo
## Imagery 0.110      -     -     -
## Photo    0.025  1.000     -     -
## Placebo 1.000  0.057  0.009     -
## Same     0.057  1.000  1.000  0.025
## 
## P value adjustment method: holm

```

	diff	lwr	upr	p adj
## Imagery-Different	6	-1.188363	13.188363	0.14198584
## Photo-Different	8	0.811637	15.188363	0.02232998
## Placebo-Different	-1	-8.188363	6.188363	0.99466042
## Same-Different	7	-0.188363	14.188363	0.05967870
## Photo-Imagery	2	-5.188363	9.188363	0.93203553
## Placebo-Imagery	-7	-14.188363	0.188363	0.05967870
## Same-Imagery	1	-6.188363	8.188363	0.99466042
## Placebo-Photo	-9	-16.188363	-1.811637	0.00759672
## Same-Photo	-1	-8.188363	6.188363	0.99466042
## Same-Placebo	8	0.811637	15.188363	0.02232998



```
##  
## Study: model_aov ~ "context"  
##  
## HSD Test for number  
##  
## Mean Square Error: 32  
##  
## context, means  
##  
##          number      std   r Min Max  
## Different     11 5.617433 10    4  21  
## Imagery       17 6.000000 10   10  29  
## Photo         19 5.773503 10   11  27  
## Placebo       10 5.906682 10    1  20  
## Same          18 4.921608 10   11  26  
##  
## Alpha: 0.05 ; DF Error: 45  
## Critical Value of Studentized Range: 4.018417  
##  
## Minimun Significant Difference: 7.188363  
##  
## Treatments with the same letter are not significantly different.  
##  
##          number groups  
## Photo        19      a  
## Same         18      ab  
## Imagery      17      abc  
## Different    11      bc  
## Placebo      10      c
```

```

##  

## Study: model_aov ~ "context"  

##  

## Student Newman Keuls Test  

## for number  

##  

## Mean Square Error: 32  

##  

## context, means  

##  

##          number      std   r Min Max  

## Different     11 5.617433 10   4  21  

## Imagery       17 6.000000 10  10  29  

## Photo         19 5.773503 10   11  27  

## Placebo       10 5.906682 10    1  20  

## Same          18 4.921608 10   11  26  

##  

## Alpha: 0.05 ; DF Error: 45  

##  

## Critical Range  

##          2        3        4        5  

## 5.095323 6.131311 6.748805 7.188363  

##  

## Means with the same letter are not significantly different.  

##  

##          number groups  

## Photo        19     a  

## Same         18     a  

## Imagery      17     a  

## Different    11     b  

## Placebo      10     b

```

```

##  

## Study: model_aov ~ "context"  

##  

## LSD t Test for number  

## P value adjustment method: holm  

##  

## Mean Square Error: 32  

##  

## context, means and individual ( 95 %) CI  

##  

##          number      std   r      LCL      UCL Min Max  

## Different     11 5.617433 10  7.397062 14.60294   4  21  

## Imagery       17 6.000000 10 13.397062 20.60294  10  29  

## Photo         19 5.773503 10 15.397062 22.60294  11  27  

## Placebo       10 5.906682 10  6.397062 13.60294   1  20  

## Same          18 4.921608 10 14.397062 21.60294  11  26  

##  

## Alpha: 0.05 ; DF Error: 45  

## Critical Value of t: 2.952079  

##  

## Minimum Significant Difference: 7.468235  

##  

## Treatments with the same letter are not significantly different.  

##  

##          number groups  

## Photo        19      a  

## Same         18      ab  

## Imagery      17      abc  

## Different    11      bc  

## Placebo      10      c

```

```

## 
## Study: model_aov ~ "context"
## 
## Scheffe Test for number
## 
## Mean Square Error : 32
## 
## context, means
## 
##          number      std   r Min Max
## Different     11 5.617433 10   4  21
## Imagery       17 6.000000 10  10  29
## Photo         19 5.773503 10   11  27
## Placebo       10 5.906682 10    1  20
## Same          18 4.921608 10   11  26
## 
## Alpha: 0.05 ; DF Error: 45
## Critical Value of F: 2.578739
## 
## Minimum Significant Difference: 8.125006
## 
## Means with the same letter are not significantly different.
## 
##          number groups
## Photo        19      a
## Same         18      ab
## Imagery      17      ab
## Different    11      ab
## Placebo      10      b

```

5. Chcemy przetestować następujące hipotezy szczegółowe:

- Grupy o takim samym kontekście podczas uczenia i testowania („Same” lub „Imaginary” lub „Photographed”) wypadają lepiej od grup o różnym kontekście („Different” lub „Placebo”).
- Grupa „Same” różni się od grup „Imaginary” i „Photographed”.
- Grupa „Imaginary” różni się od grupy „Photographed”.
- Grupa „Different” różni się od grupy „Placebo”.

W tym celu wykonaj następujące polecenia:

- Zapisz odpowiednie hipotezy.
- Wyraź je za pomocą kontrastów.
- Czy ten układ kontrastów jest ortogonalny?
- Przetestuj zaproponowane kontrasty.

```
##                Df Sum Sq Mean Sq F value    Pr(>F)
## context          4   700   175   5.469  0.00112 **
## context: C1    1   675   675  21.094 3.52e-05 ***
## context: C2    1     0     0  0.000  1.00000
## context: C3    1    20    20  0.625  0.43334
## context: C4    1     5     5  0.156  0.69450
## Residuals      45  1440    32
## ---
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Zadanie 2. W 1974 roku Michael Eysenck opublikował w czasopiśmie Developmental Psychology wyniki badań dotyczących ubocznego uczenia verbalnego. W eksperymencie wzięło udział 100 osób, z czego połowę stanowili młodzi ludzie (w wieku studenckim), a drugą połowę osoby starsze (w wieku pięćdziesięciu i sześćdziesięciu lat). W obrębie każdej grupy wiekowej, pacjenci zostali przydzieleni do jednej z pięciu grup „Instrukcji”. Następnie podano im listę słów i powiedziano, aby postępowali zgodnie z instrukcjami podanymi wcześniej. Instrukcje były następujące:

- Liczenie - liczenie liter w każdym wymienionym słowie,
- Rymowanie - pomyśleć o słowie, które rymuje się z wskazanym słowem,
- Przymiotnik - pomyśleć o przymiotniku, który mógłby zmodyfikować dane słowo,
- Wyobraźnia - wyobrazić sobie obraz obiektu opisanego przez wymienione słowo,
- Kontrola - pamiętaj wymienione słowa aby później je powtórzyć.

Każdy pacjent widział tę samą listę wyrazów trzy razy i powtarzał te instrukcje trzy razy. Instrukcje Liczenie i Rymowanie mają dać informację o powierzchownym poziomie przetwarzania semantycznego. Instrukcje Przymiotnik i Wyobraźnia mają informować o głębszym poziomie przetwarzania semantycznego, tj. liczenie i rymowanie nie wymagają od pacjenta znajomości sensu słów z listy, podczas gdy instrukcje Przymiotnik i Wyobraźnia wymagają znajomości znaczenia słów. Pacjenci w grupie kontrolnej mieli tylko zapamiętać

9 Regresja liniowa

9.1 Przykład

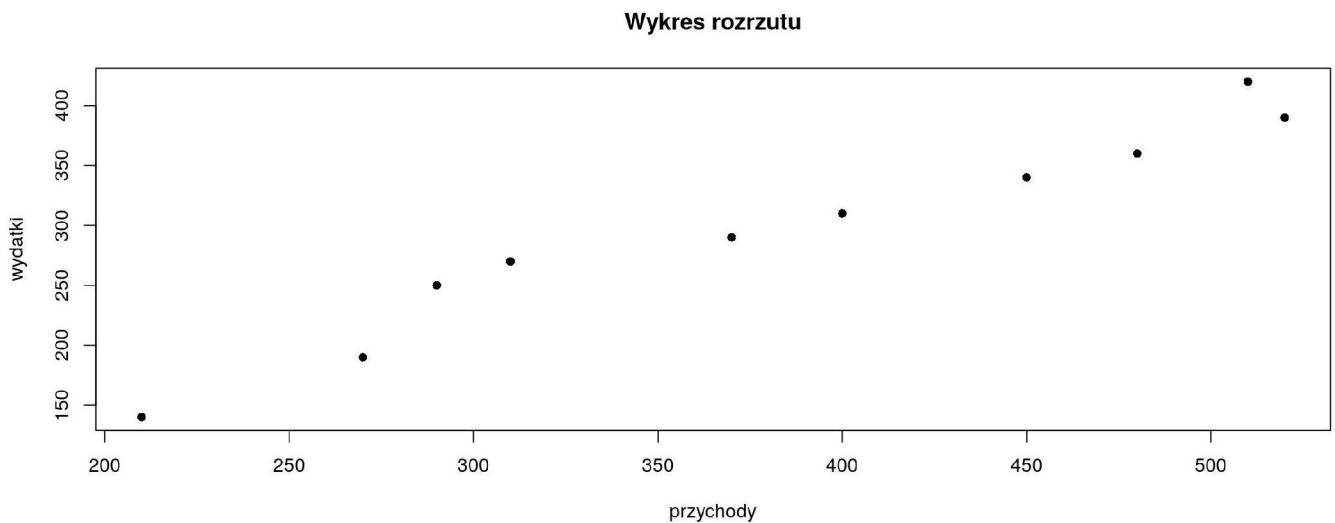
Przykład. Za pomocą regresji liniowej chcemy opisać związek między miesięcznym dochodem rodziny na jedną osobę a miesięczną wartością wydatków na jedną osobę. Dane dotyczące tych dwóch cech dla dziesięciu rodzin podano w poniższej tabeli.

rodzina	1	2	3	4	5	6	7	8	9	10
przychody	210	270	290	310	370	400	450	480	510	520
wydatki	140	190	250	270	290	310	340	360	420	390

```
# dane
przychody <- c(210, 270, 290, 310, 370, 400, 450, 480, 510, 520)
wydatki <- c(140, 190, 250, 270, 290, 310, 340, 360, 420, 390)
data_set <- data.frame(przychody = przychody, wydatki = wydatki)
head(data_set)

##  przychody wydatki
## 1      210     140
## 2      270     190
## 3      290     250
## 4      310     270
## 5      370     290
## 6      400     310

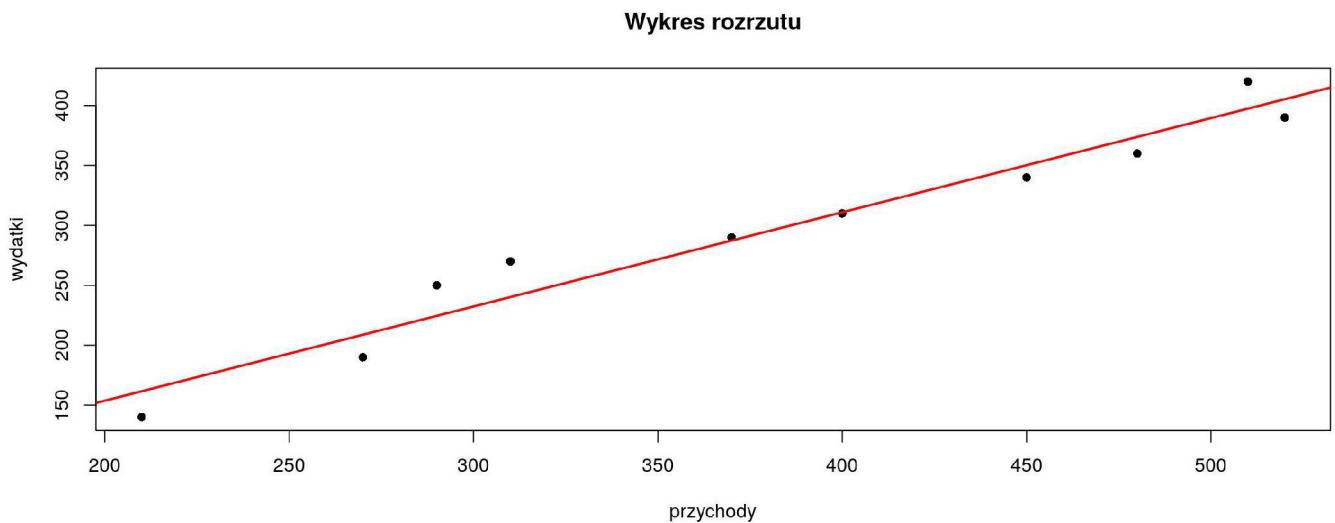
# Wykres rozrzutu
plot(data_set, main = "Wykres rozrzutu", pch = 16)
```



```
# model
model <- lm(wydatki ~ przychody, data = data_set)
model

##
## Call:
## lm(formula = wydatki ~ przychody, data = data_set)
##
## Coefficients:
## (Intercept)    przychody
##           -3.5036       0.7861

plot(data_set, main = "Wykres rozrzutu", pch = 16)
abline(model, col = "red", lwd = 2)
```



```
# estymacja parametrów
coef(model)

## (Intercept)  przychody
## -3.5036358  0.7860988

confint(model)

##               2.5 %    97.5 %
## (Intercept) -61.2027257 54.1954540
## przychody     0.6398962  0.9323013

# podsumowanie modelu
# tj. reszty, estymacja punktowa, testy istotności dla współczynników regresji,
# R^2, test istotności modelu
summary(model)
```

```

## 
## Call:
## lm(formula = wydatki ~ przychody, data = data_set)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -21.577 -14.907  -5.588  17.607  29.813 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -3.5036    25.0212   -0.14    0.892    
## przychody     0.7861     0.0634   12.40 1.67e-06 ***  
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 20.63 on 8 degrees of freedom 
## Multiple R-squared:  0.9505, Adjusted R-squared:  0.9444 
## F-statistic: 153.7 on 1 and 8 DF,  p-value: 1.67e-06

```

wartości dopasowane przez model

fitted(model)

```

##      1       2       3       4       5       6       7       8 
## 161.5771 208.7430 224.4650 240.1870 287.3529 310.9359 350.2408 373.8238 
##      9      10      
## 397.4067 405.2677

```

reszty

residuals(model)

```

##      1       2       3       4       5       6 
## -21.5771083 -18.7430352  25.5349891  29.8130135   2.6470866 -0.9358769 
##      7       8       9      10      
## -10.2408159 -13.8237794  22.5932572 -15.2677307

```

```

# sprawdzenie
wydatki - fitted(model)

##          1           2           3           4           5           6
## -21.5771083 -18.7430352 25.5349891 29.8130135  2.6470866 -0.9358769
##          7           8           9          10
## -10.2408159 -13.8237794 22.5932572 -15.2677307

```

przedziały ufności dla predykcji

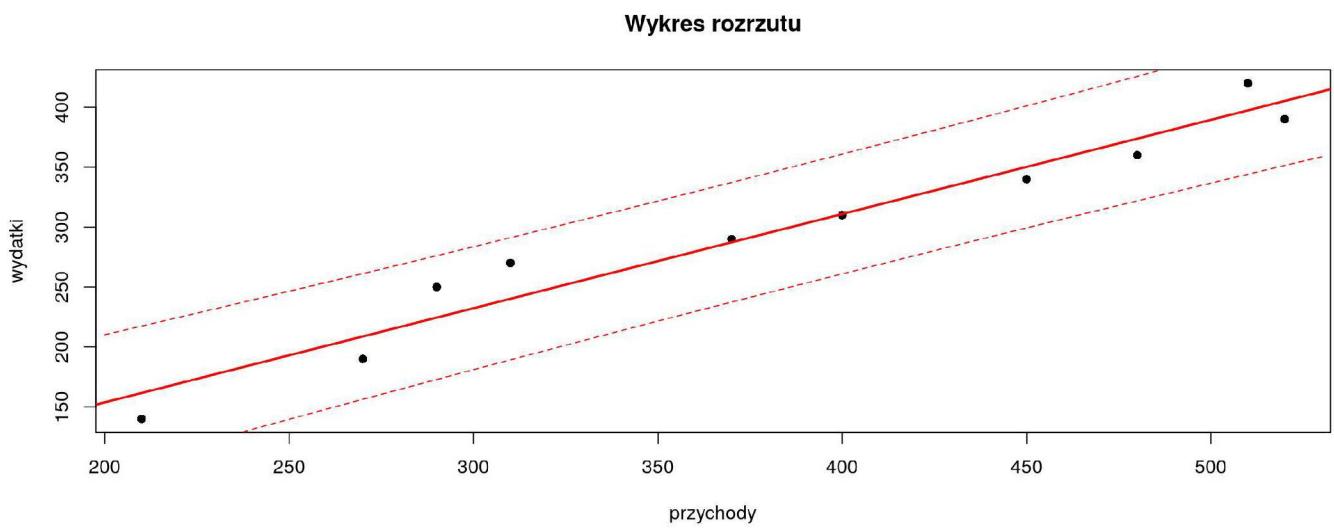
```

temp_przychody <- data.frame(przychody = seq(min(data_set$przychody) - 10,
                                               max(data_set$przychody) + 10,
                                               length = 100))

pred <- predict(model, temp_przychody, interval = "prediction")

plot(data_set, main = "Wykres rozrzutu", pch = 16)
abline(model, col = "red", lwd = 2)
lines(temp_przychody$przychody, pred[, 2], lty = 2, col = "red")
lines(temp_przychody$przychody, pred[, 3], lty = 2, col = "red")

```



predykcja wydatków dla przychodu = 350

```

nowy_przychod <- data.frame(przychody = 350)

predict(model, nowy_przychod, interval = 'prediction')

```

```
##      fit      lwr      upr
## 1 271.6309 221.528 321.7338
```

- model bez wyrazu wolnego

```
model_bez_ww <- lm(wydatki ~ przychody - 1, data = data_set)
```

```
model_bez_ww
```

```
##
```

```
## Call:
```

```
## lm(formula = wydatki ~ przychody - 1, data = data_set)
```

```
##
```

```
## Coefficients:
```

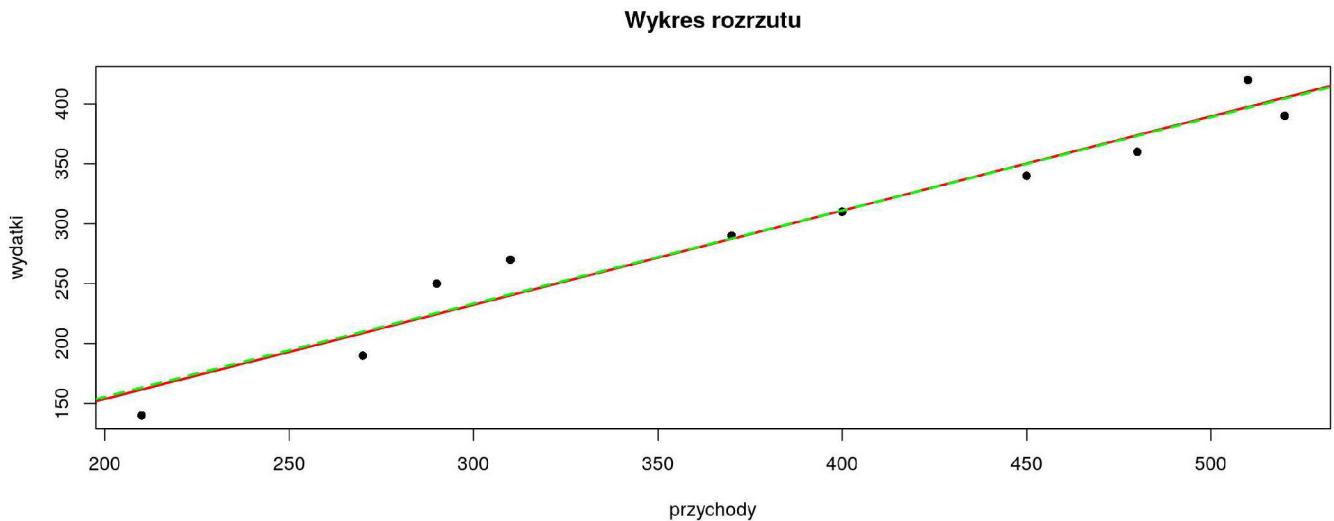
```
## przychody
```

```
##     0.7775
```

```
plot(data_set, main = "Wykres rozrzutu", pch = 16)
```

```
abline(model, col = "red", lwd = 2)
```

```
abline(model_bez_ww, col = "green", lwd = 2, lty = 2)
```



```
coef(model_bez_ww)
```

```

## przychody
## 0.7775281

confint(model_bez_ww)

##          2.5 %    97.5 %
## przychody 0.7422271 0.812829

summary(model_bez_ww)

##
## Call:
## lm(formula = wydatki ~ przychody - 1, data = data_set)
##
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -23.281 -14.039  -5.449  18.174  28.966 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## przychody    0.7775     0.0156   49.83 2.65e-12 ***
## ---    
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.48 on 9 degrees of freedom
## Multiple R-squared:  0.9964, Adjusted R-squared:  0.996 
## F-statistic: 2483 on 1 and 9 DF,  p-value: 2.651e-12

fitted(model_bez_ww)

```

```

##      1      2      3      4      5      6      7      8
## 163.2809 209.9326 225.4831 241.0337 287.6854 311.0112 349.8876 373.2135
##      9     10
## 396.5393 404.3146

```

```
residuals(model_bez_ww)
```

```

##      1      2      3      4      5      6
## -23.280899 -19.932584 24.516854 28.966292 2.314607 -1.011236
##      7      8      9      10
## -9.887640 -13.213483 23.460674 -14.314607

```

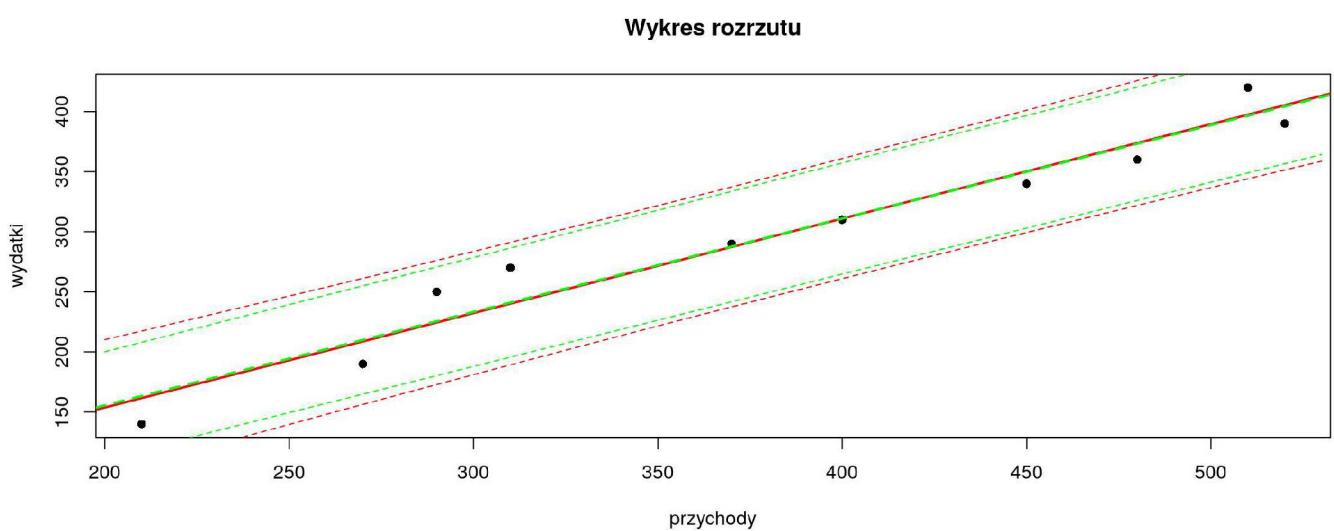
```

temp_przychody <- data.frame(przychody = seq(min(data_set$przychody) - 10,
                                         max(data_set$przychody) + 10,
                                         length = 100))

pred1 <- predict(model_bez_ww, temp_przychody, interval = "prediction")

plot(data_set, main = "Wykres rozrzutu", pch = 16)
abline(model, col = "red", lwd = 2)
abline(model_bez_ww, col = "green", lwd = 2, lty = 2)
lines(temp_przychody$przychody, pred[, 2], lty = 2, col = "red")
lines(temp_przychody$przychody, pred[, 3], lty = 2, col = "red")
lines(temp_przychody$przychody, pred1[, 2], lty = 2, col = "green")
lines(temp_przychody$przychody, pred1[, 3], lty = 2, col = "green")

```



```

nowy_przychod <- data.frame(przychody = 350)
predict(model_bez_ww, nowy_przychod, interval = 'prediction')

##          fit      lwr      upr
## 1 272.1348 226.3796 317.8901

```

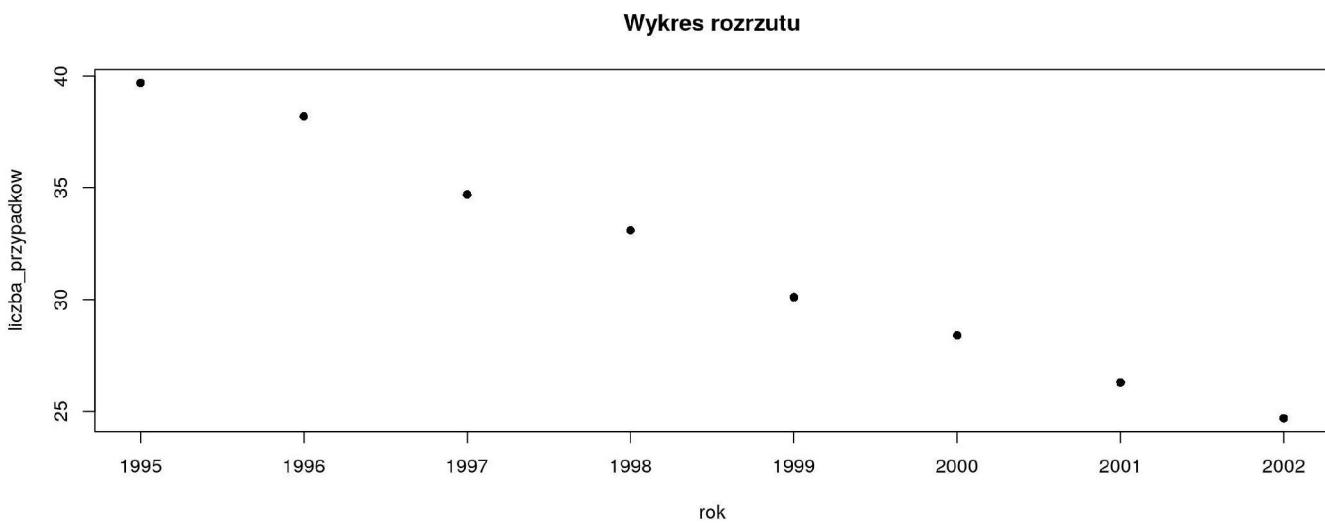
9.2 Zadania

Zadanie 1. Poniższa tabela przedstawia liczbę przypadków gruźlicy układu oddechowego w latach 1995-2002. Podano liczbę przypadków na 100.000 ludności. Zakładając liniową zależność między rokiem a liczbą przypadków, wykonaj kompleksową analizę regresji.

Dane

rok	1995	1996	1997	1998	1999	2000	2001	2002
liczba przypadków	39.7	38.2	34.7	33.1	30.1	28.4	26.3	24.7

1. Przedstaw dane na wykresie rozrzutu. Czy model regresji liniowej wydaje się adekwatny?

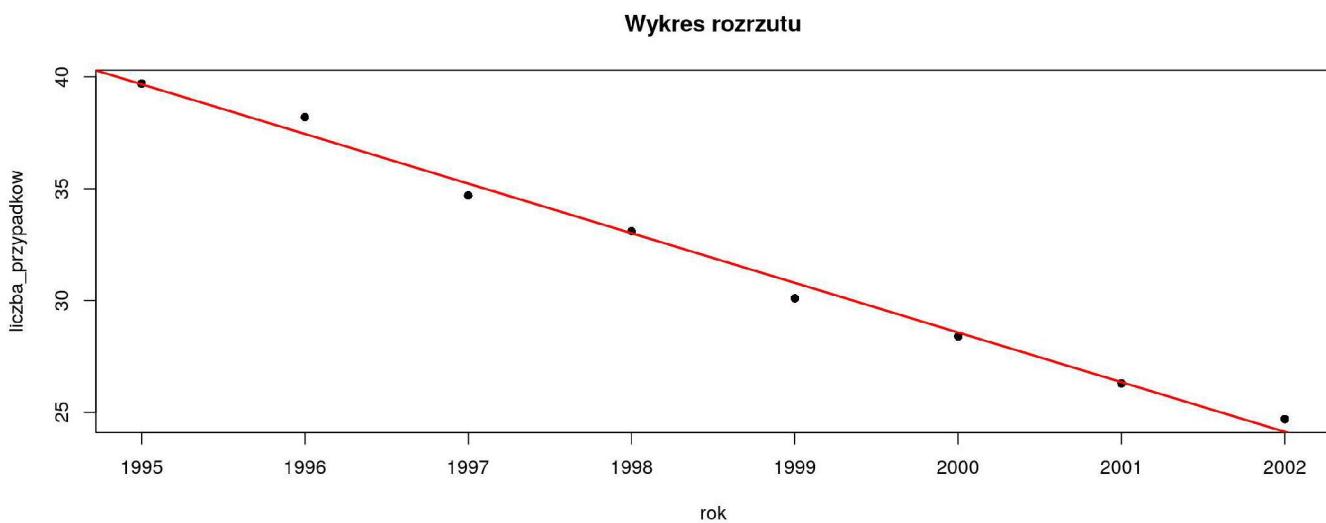


2. Dopasuj model regresji liniowej do tych danych. Jakie są wartości estymatorów współczynników regresji i przedziały ufności? Narysuj uzyskaną prostą regresji na schemacie punktowym.

```

## (Intercept)      rok
## 4466.666667 -2.219048

```



```
## (Intercept)      rok  
## 4466.666667 -2.219048
```

```
##                  2.5 %      97.5 %  
## (Intercept) 4066.82158 4866.511749  
## rok         -2.41912   -2.018975
```

3. Które współczynniki są istotne statystycznie w skonstruowanym modelu? Jakie jest dopasowanie modelu?

```

## 
## Call:
## lm(formula = liczba_przypadkow ~ rok, data = data_set)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.69048 -0.26071 -0.00952  0.20952  0.75238 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 4466.66667  163.40805   27.33 1.58e-07 ***
## rok         -2.21905    0.08177  -27.14 1.65e-07 ***  
## ---        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.5299 on 6 degrees of freedom
## Multiple R-squared:  0.9919, Adjusted R-squared:  0.9906 
## F-statistic: 736.5 on 1 and 6 DF,  p-value: 1.654e-07

```

4. Oblicz wartości dopasowane przez model, a także reszty.

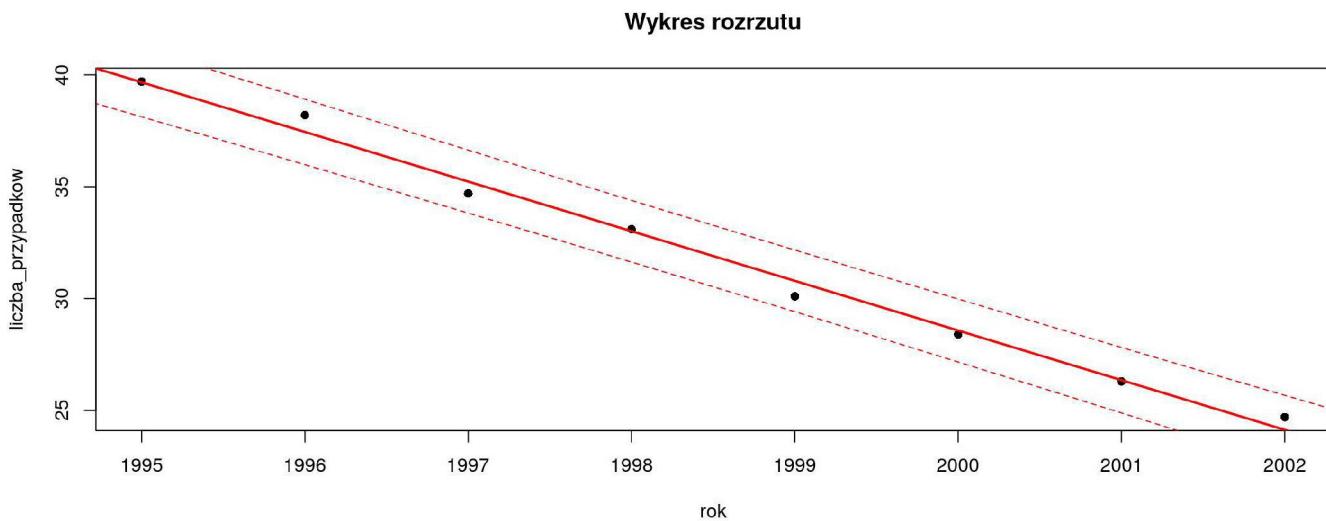
```

##      1       2       3       4       5       6       7       8
## 39.66667 37.44762 35.22857 33.00952 30.79048 28.57143 26.35238 24.13333

##      1       2       3       4       5       6
## 0.03333333 0.75238095 -0.52857143 0.09047619 -0.69047619 -0.17142857
##      7       8
## -0.05238095 0.56666667

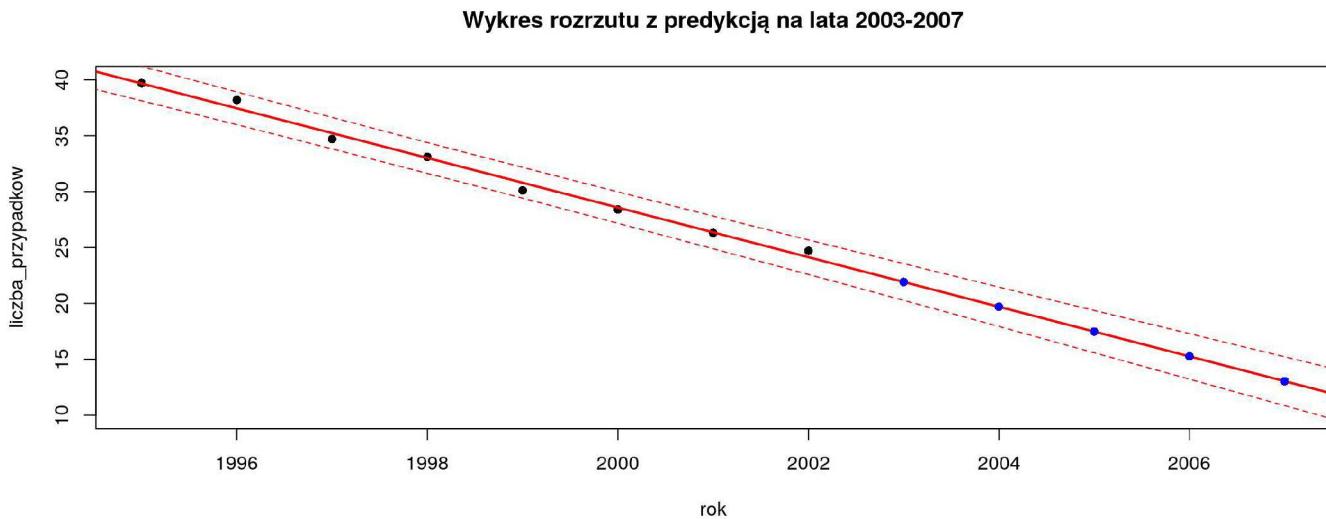
```

5. Na wykresie rozrzutu przedstaw granice przedziału prognozy 95%.



6. Dokonaj predykcji liczby przypadków gruźlicy układu oddechowego w latach 2003-2007. Zilustruj wyniki na wykresie rozrzutu.

```
##          fit      lwr      upr
## 1 21.91429 20.27052 23.55805
## 2 19.69524 17.93392 21.45656
## 3 17.47619 15.58342 19.36896
## 4 15.25714 13.22171 17.29258
## 5 13.03810 10.85098 15.22521
```



7. Czy miałoby sens usunięcie wyrazu wolnego z modelu? Jeśli tak, wykonaj powyższe polecenia dla modelu regresji liniowej bez wyrazu losowego.

Zadanie 2. Zbiór danych zawarty w pliku [braking.RData](#) zawiera informacje o długości drogi hamowania przy danej prędkości określonego modelu samochodu. W tym zbiorze danych występuje obserwacja odstająca. Zidentyfikuj ją za pomocą wykresu rozrzutu. Korzystając z

modelu regresji liniowej, opisz związek między długością drogi hamowania a prędkością przy użyciu pełnych danych i danych bez obserwacji odstającej. Jakie są wyniki dla obu modeli? Który model jest lepszy? Dokładniej, wykonaj polecenia 2-7 Zadania 1 dla każdego modelu osobno. W punkcie 6 przeprowadź predykcję długości drogi hamowania dla prędkości 30, 31, ..., 50.

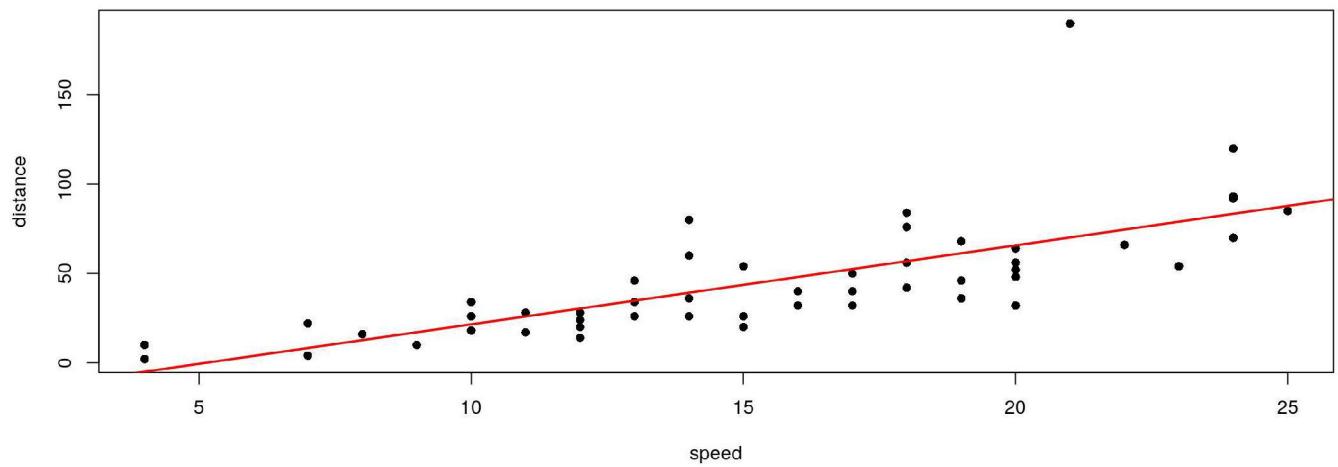
```
##   speed distance
## 1     4       2
## 2     4      10
## 3     7       4
## 4     7      22
## 5     8      16
## 6     9      10
```



Model dla pełnych danych

2.

Wykres rozrzutu



```
## (Intercept)      speed
```

```
## -22.726854    4.422338
```

```
##                   2.5 %   97.5 %
```

```
## (Intercept) -43.105778 -2.347930
```

```
## speed        3.177543  5.667134
```

3.

```

## 
## Call:
## lm(formula = distance ~ speed, data = braking)
## 
## Residuals:
##    Min     1Q Median     3Q    Max 
## -33.720 -13.298 -3.186  7.814 119.858 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -22.7269   10.1409  -2.241   0.0296 *  
## speed        4.4223    0.6194   7.139 4.04e-09 *** 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
## 
## Residual standard error: 23.18 on 49 degrees of freedom 
## Multiple R-squared:  0.5099, Adjusted R-squared:  0.4999 
## F-statistic: 50.97 on 1 and 49 DF,  p-value: 4.037e-09

```

4.

```

##      1       2       3       4       5       6       7 
## -5.037501 -5.037501 8.229514 8.229514 12.651852 17.074190 21.496528 
##      8       9      10      11      12      13      14 
## 21.496528 21.496528 25.918867 25.918867 30.341205 30.341205 30.341205 
##      15      16      17      18      19      20      21 
## 30.341205 34.763543 34.763543 34.763543 34.763543 39.185881 39.185881 
##      22      23      24      25      26      27      28 
## 39.185881 39.185881 43.608220 43.608220 43.608220 70.142249 48.030558 
##      29      30      31      32      33      34      35 
## 48.030558 52.452896 52.452896 52.452896 56.875234 56.875234 56.875234 
##      36      37      38      39      40      41      42 
## 56.875234 61.297573 61.297573 61.297573 65.719911 65.719911 65.719911 
##      43      44      45      46      47      48      49 
## 65.719911 65.719911 74.564587 78.986926 83.409264 83.409264 83.409264 
##      50      51 
## 83.409264 87.831602

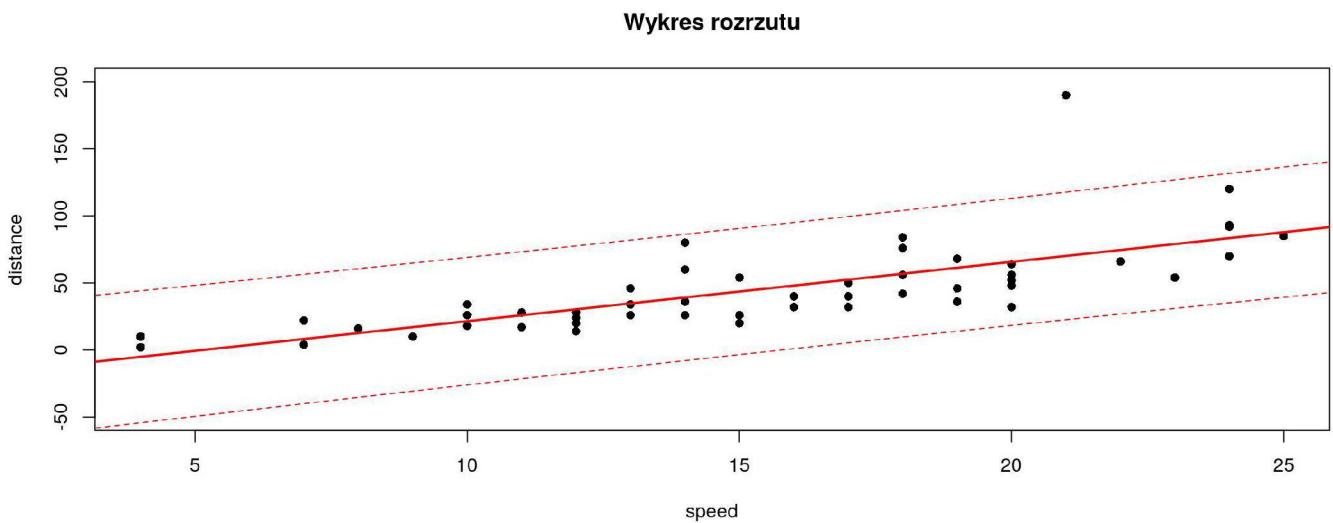
```

```

##          1           2           3           4           5           6
## 7.0375010 15.0375010 -4.2295137 13.7704863 3.3481480 -7.0741902
##          7           8           9          10          11          12
## -3.4965285 4.5034715 12.5034715 -8.9188667 2.0811333 -16.3412050
##          13          14          15          16          17          18
## -10.3412050 -6.3412050 -2.3412050 -8.7635432 -0.7635432 -0.7635432
##          19          20          21          22          23          24
## 11.2364568 -13.1858815 -3.1858815 20.8141185 40.8141185 -23.6082197
##          25          26          27          28          29          30
## -17.6082197 10.3917803 119.8577508 -16.0305580 -8.0305580 -20.4528962
##          31          32          33          34          35          36
## -12.4528962 -2.4528962 -14.8752345 -0.8752345 19.1247655 27.1247655
##          37          38          39          40          41          42
## -25.2975727 -15.2975727 6.7024273 -33.7199110 -17.7199110 -13.7199110
##          43          44          45          46          47          48
## -9.7199110 -1.7199110 -8.5645875 -24.9869257 -13.4092640 8.5907360
##          49          50          51
## 9.5907360 36.5907360 -2.8316022

```

5.



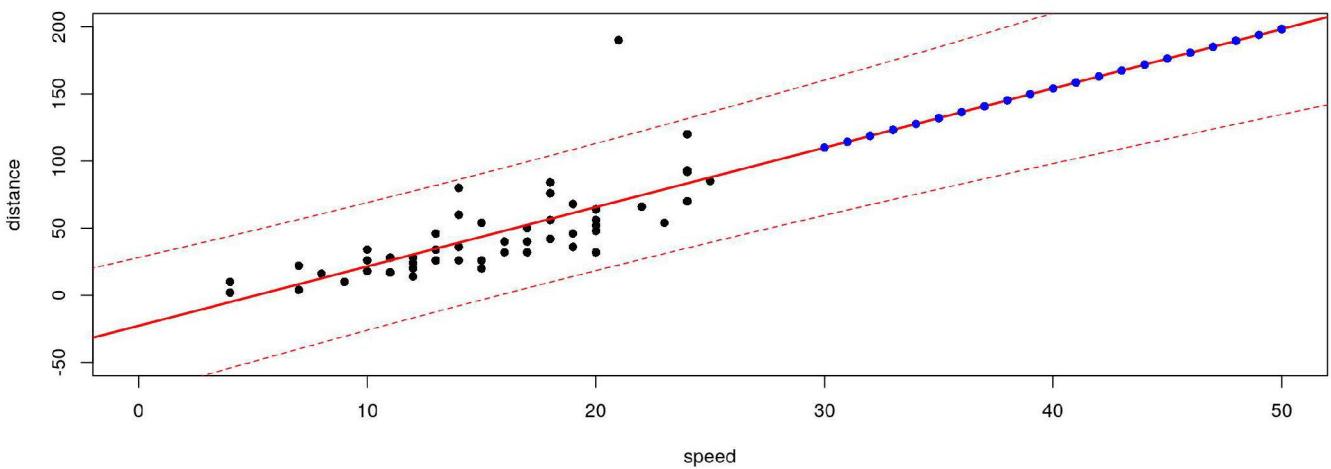
6.

```

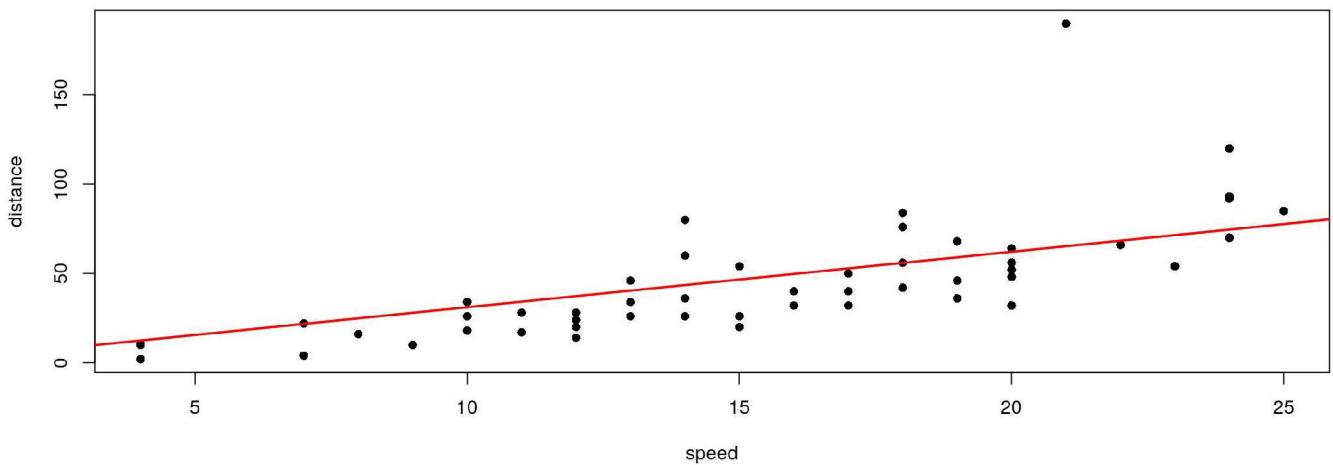
##          fit        lwr       upr
## 1  109.9433  59.56096 160.3256
## 2  114.3656  63.52436 165.2069
## 3  118.7880  67.46167 170.1143
## 4  123.2103  71.37362 175.0470
## 5  127.6326  75.26095 180.0043
## 6  132.0550  79.12441 184.9856
## 7  136.4773  82.96475 189.9899
## 8  140.8997  86.78270 195.0166
## 9  145.3220  90.57902 200.0650
## 10 149.7443  94.35444 205.1342
## 11 154.1667  98.10968 210.2237
## 12 158.5890 101.84545 215.3326
## 13 163.0114 105.56245 220.4603
## 14 167.4337 109.26136 225.6060
## 15 171.8560 112.94285 230.7692
## 16 176.2784 116.60757 235.9492
## 17 180.7007 120.25614 241.1453
## 18 185.1230 123.88919 246.3569
## 19 189.5454 127.50730 251.5835
## 20 193.9677 131.11105 256.8244
## 21 198.3901 134.70099 262.0791

```

Wykres rozrzutu z predykcja dla prędkości 30, 31, ..., 50



Wykres rozrzutu



```
##      speed
## 3.107177

##          2.5 %   97.5 %
## speed 2.693185 3.521169

##
## Call:
## lm(formula = distance ~ speed - 1, data = braking)
##
## Residuals:
##    Min     1Q   Median     3Q    Max 
## -30.144 -15.786 - 7.500  2.392 124.749 
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## speed     3.1072     0.2061   15.07   <2e-16 ***  
## ---      
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 
##
## Residual standard error: 24.1 on 50 degrees of freedom
## Multiple R-squared:  0.8197, Adjusted R-squared:  0.8161 
## F-statistic: 227.3 on 1 and 50 DF,  p-value: < 2.2e-16
```

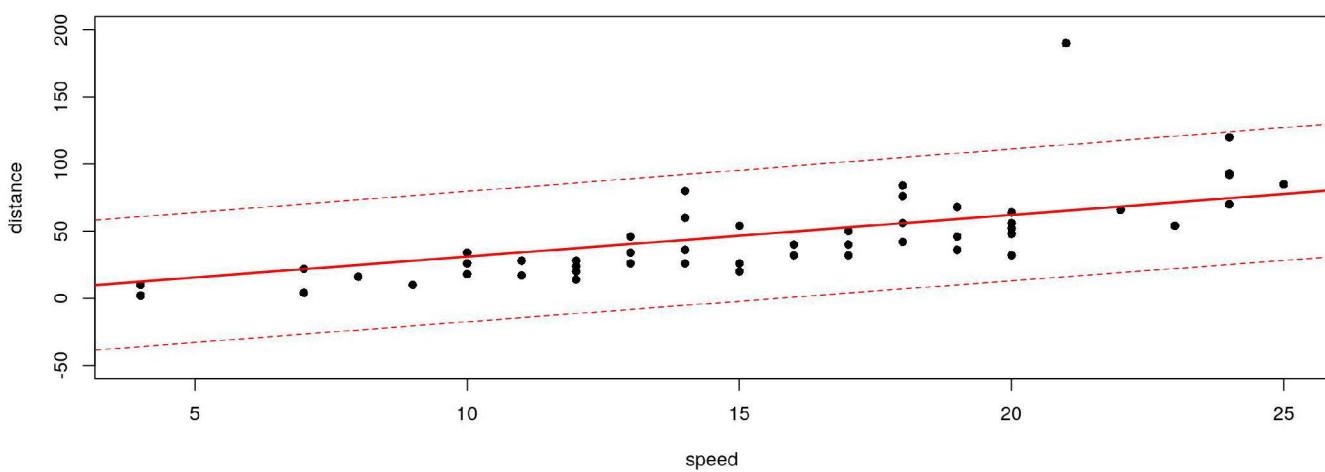
```
##      1      2      3      4      5      6      7      8
## 12.42871 12.42871 21.75024 21.75024 24.85741 27.96459 31.07177 31.07177
##      9     10     11     12     13     14     15     16
## 31.07177 34.17895 34.17895 37.28612 37.28612 37.28612 37.28612 40.39330
##     17     18     19     20     21     22     23     24
## 40.39330 40.39330 40.39330 43.50048 43.50048 43.50048 43.50048 46.60765
##     25     26     27     28     29     30     31     32
## 46.60765 46.60765 65.25071 49.71483 49.71483 52.82201 52.82201 52.82201
##     33     34     35     36     37     38     39     40
## 55.92918 55.92918 55.92918 55.92918 59.03636 59.03636 59.03636 62.14354
##     41     42     43     44     45     46     47     48
## 62.14354 62.14354 62.14354 62.14354 68.35789 71.46507 74.57224 74.57224
##     49     50     51
## 74.57224 74.57224 77.67942
```

```

##          1           2           3           4           5
## -10.42870729 -2.42870729 -17.75023776  0.24976224 -8.85741459
##          6           7           8           9          10
## -17.96459141 -13.07176823 -5.07176823  2.92823177 -17.17894506
##          11          12          13          14          15
## -6.17894506 -23.28612188 -17.28612188 -13.28612188 -9.28612188
##          16          17          18          19          20
## -14.39329871 -6.39329871 -6.39329871  5.60670129 -17.50047553
##          21          22          23          24          25
## -7.50047553 16.49952447 36.49952447 -26.60765235 -20.60765235
##          26          27          28          29          30
## 7.39234765 124.74928671 -17.71482918 -9.71482918 -20.82200600
##          31          32          33          34          35
## -12.82200600 -2.82200600 -13.92918282  0.07081718 20.07081718
##          36          37          38          39          40
## 28.07081718 -23.03635965 -13.03635965  8.96364035 -30.14353647
##          41          42          43          44          45
## -14.14353647 -10.14353647 -6.14353647  1.85646353 -2.35789012
##          46          47          48          49          50
## -17.46506694 -4.57224376 17.42775624 18.42775624 45.42775624
##          51
##    7.32057941

```

Wykres rozrzutu



10 Regresja wielokrotna i krokowa

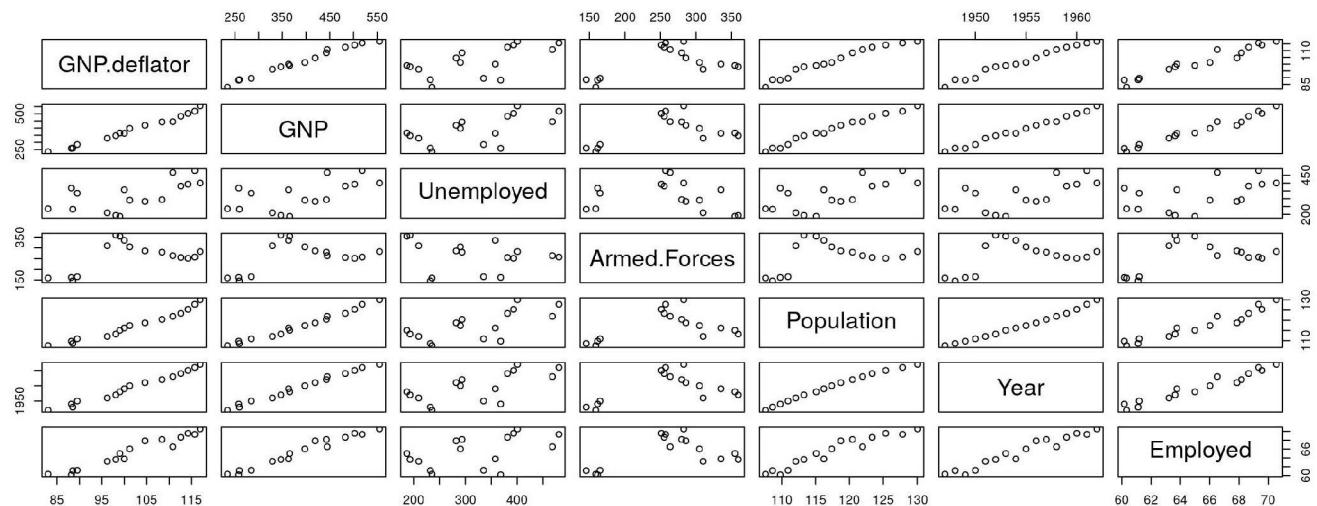
10.1 Przykład

Przykład. Zbiór danych `longley` zawiera 7 zmiennych makroekonomicznych. Chcemy modelować liczbę zatrudnionych za pomocą innych (niekoniecznie wszystkich) zmiennych przy użyciu modelu regresji wielokrotnej.

```
head(longley)
```

##	GNP.deflator	GNP	Unemployed	Armed.Forces	Population	Year	Employed
## 1947	83.0	234.289	235.6	159.0	107.608	1947	60.323
## 1948	88.5	259.426	232.5	145.6	108.632	1948	61.122
## 1949	88.2	258.054	368.2	161.6	109.773	1949	60.171
## 1950	89.5	284.599	335.1	165.0	110.929	1950	61.187
## 1951	96.2	328.975	209.9	309.9	112.075	1951	63.221
## 1952	98.1	346.999	193.2	359.4	113.270	1952	63.639

```
pairs(longley)
```



```

# model pełny

model_1 <- lm(Employed ~ ., data = longley)

# model_1 <- lm(Employed ~ GNP.deflator + GNP + Unemployed +
#                           Armed.Forces + Population + Year,
#                           data = Longley)

model_1

##

## Call:

## lm(formula = Employed ~ ., data = longley)

##

## Coefficients:
## (Intercept)  GNP.deflator          GNP  Unemployed  Armed.Forces
## -3.482e+03   1.506e-02   -3.582e-02  -2.020e-02  -1.033e-02
## Population      Year
## -5.110e-02   1.829e+00

# estymacja parametrów

coef(model_1)

## (Intercept)  GNP.deflator          GNP  Unemployed  Armed.Forces
## -3.482259e+03  1.506187e-02  -3.581918e-02  -2.020230e-02  -1.033227e-02
## Population      Year
## -5.110411e-02  1.829151e+00

confint(model_1)

```

```
##                      2.5 %      97.5 %
## (Intercept) -5.496529e+03 -1.467988e+03
## GNP.deflator -1.770290e-01  2.071528e-01
## GNP          -1.115811e-01  3.994274e-02
## Unemployed   -3.125067e-02 -9.153930e-03
## Armed.Forces -1.517949e-02 -5.485050e-03
## Population   -5.625172e-01  4.603090e-01
## Year         7.987875e-01  2.859515e+00
```

```
# podsumowanie modelu
# tj. reszty, estymacja punktowa, testy istotności dla współczynników regresji,
# R_adj^2, test istnotności modelu (test analizy wariancji w regresji)
summary(model_1)
```

```

## 
## Call:
## lm(formula = Employed ~ ., data = longley)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41011 -0.15767 -0.02816  0.10155  0.45539
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.482e+03  8.904e+02 -3.911 0.003560 **
## GNP.deflator 1.506e-02  8.492e-02  0.177 0.863141
## GNP          -3.582e-02  3.349e-02 -1.070 0.312681
## Unemployed   -2.020e-02  4.884e-03 -4.136 0.002535 **
## Armed.Forces -1.033e-02  2.143e-03 -4.822 0.000944 ***
## Population    -5.110e-02  2.261e-01 -0.226 0.826212
## Year         1.829e+00  4.555e-01  4.016 0.003037 **
## ---
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3049 on 9 degrees of freedom
## Multiple R-squared:  0.9955, Adjusted R-squared:  0.9925
## F-statistic: 330.3 on 6 and 9 DF,  p-value: 4.984e-10

```

wartości dopasowane przez model

fitted(model_1)

	1947	1948	1949	1950	1951	1952	1953	1954
##	60.05566	61.21601	60.12471	61.59711	62.91129	63.88831	65.15305	63.77418
##	1955	1956	1957	1958	1959	1960	1961	1962
##	66.00470	67.40161	68.18627	66.55206	68.81055	69.64967	68.98907	70.75776

reszty

residuals(model_1)

```
##          1947      1948      1949      1950      1951      1952
##  0.26734003 -0.09401394  0.04628717 -0.41011462  0.30971459 -0.24931122
##          1953      1954      1955      1956      1957      1958
## -0.16404896 -0.01318036  0.01430477  0.45539409 -0.01726893 -0.03905504
##          1959      1960      1961      1962
## -0.15554997 -0.08567131  0.34193151 -0.20675783
```

predykcja

```
new_data <- data.frame(GNP.deflator = 115.4,
                        GNP = 518.163,
                        Unemployed = 480.3,
                        Armed.Forces = 257.4,
                        Population = 127.857,
                        Year = 1963)

predict(model_1, new_data, interval = "prediction")
```

```
##       fit     lwr     upr
## 1 72.64695 70.55039 74.74351
```

redukcja modelu pełnego

```
summary(model_1)
```

```

## 
## Call:
## lm(formula = Employed ~ ., data = longley)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41011 -0.15767 -0.02816  0.10155  0.45539
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.482e+03  8.904e+02 -3.911 0.003560 ***
## GNP.deflator 1.506e-02  8.492e-02  0.177 0.863141
## GNP          -3.582e-02  3.349e-02 -1.070 0.312681
## Unemployed   -2.020e-02  4.884e-03 -4.136 0.002535 ***
## Armed.Forces -1.033e-02  2.143e-03 -4.822 0.000944 ***
## Population   -5.110e-02  2.261e-01 -0.226 0.826212
## Year         1.829e+00  4.555e-01  4.016 0.003037 ***
## ---
## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3049 on 9 degrees of freedom
## Multiple R-squared:  0.9955, Adjusted R-squared:  0.9925
## F-statistic: 330.3 on 6 and 9 DF,  p-value: 4.984e-10

model_2 <- lm(Employed ~ Unemployed + Armed.Forces + Year, data = longley)
# model_2 <- update(model_1, . ~ . - GNP.deflator - GNP - Population)
summary(model_2)

```

```

## 
## Call:
## lm(formula = Employed ~ Unemployed + Armed.Forces + Year, data = longley)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.57285 -0.11989  0.04087  0.13979  0.75303
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.797e+03  6.864e+01 -26.183 5.89e-12 ***
## Unemployed   -1.470e-02  1.671e-03  -8.793 1.41e-06 ***
## Armed.Forces -7.723e-03  1.837e-03  -4.204  0.00122 **
## Year         9.564e-01  3.553e-02  26.921 4.24e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3321 on 12 degrees of freedom
## Multiple R-squared:  0.9928, Adjusted R-squared:  0.9911
## F-statistic: 555.2 on 3 and 12 DF,  p-value: 3.916e-13

```

Regresja krokowa

- Istnieje również inna metoda konstrukcji modeli z dużą liczbą zmiennych objaśniających niż konstruowanie pełnego modelu i szacowanie jego parametrów (tak jak robimy to w regresji wielokrotnej).
- Jest to procedura regresji krokowej, w której możemy odrzucić lub dodać zmienną na każdym kroku.
- Powiedzmy, że zaczynamy od modelu zawierającego tylko stałą - „regresja w przód” (możemy też zacząć od pełnego modelu - „regresja w tył”). W kolejnym kroku dodajemy najlepszą zmienną w sensie kryterium (np. test istotności, AIC, BIC). W kolejnym dodamy ponownie, ale możemy również sprawdzić co się dzieje, jakbyśmy usunęli z modelu zmienną dodaną w poprzednim kroku, itd.

Przykład (cd.).

```

model_1 <- lm(Employed ~ ., data = longley)
model_2 <- lm(Employed ~ Unemployed + Armed.Forces + Year, data = longley)
# AIC (z wyrazem stałym)
AIC(model_1, model_2)

##          df      AIC
## model_1  8 14.18670
## model_2  5 15.52741

n <- nrow(longley)
p <- 6
n * log(mean(model_1$residuals^2)) + 2 * (p + 1) + n * log(2 * pi) + n + 2

## [1] 14.1867

p <- 3
n * log(mean(model_2$residuals^2)) + 2 * (p + 1) + n * log(2 * pi) + n + 2

## [1] 15.52741

# BIC (z wyrazem stałym)
AIC(model_1, model_2, k = log(nrow(longley)))

##          df      AIC
## model_1  8 20.36741
## model_2  5 19.39035

# AIC i BIC (bez wyrazu stałego)
extractAIC(model_1)[2]

```

```
## [1] -33.21933

extractAIC(model_1, k = log(n))[2]

## [1] -27.81121

p <- 6
n * log(mean(model_1$residuals^2)) + 2 * (p + 1)

## [1] -33.21933

n * log(mean(model_1$residuals^2)) + log(n) * (p + 1)

## [1] -27.81121

extractAIC(model_2)[2]

## [1] -31.87863

extractAIC(model_2, k = log(n))[2]

## [1] -28.78827

p <- 3
n * log(mean(model_2$residuals^2)) + 2 * (p + 1)

## [1] -31.87863
```

```
n * log(mean(model_2$residuals^2)) + log(n) * (p + 1)

## [1] -28.78827
```

```
# regresja krokowa
step(model_1)
```

```

## Start: AIC=-33.22

## Employed ~ GNP.deflator + GNP + Unemployed + Armed.Forces + Population +
##      Year

## 

##          Df Sum of Sq    RSS     AIC
## - GNP.deflator 1  0.00292 0.83935 -35.163
## - Population   1  0.00475 0.84117 -35.129
## - GNP          1  0.10631 0.94273 -33.305
## <none>           0.83642 -33.219
## - Year         1  1.49881 2.33524 -18.792
## - Unemployed   1  1.59014 2.42656 -18.178
## - Armed.Forces 1  2.16091 2.99733 -14.798
## 

## Step: AIC=-35.16

## Employed ~ GNP + Unemployed + Armed.Forces + Population + Year

## 

##          Df Sum of Sq    RSS     AIC
## - Population   1  0.01933 0.8587 -36.799
## <none>           0.8393 -35.163
## - GNP          1  0.14637 0.9857 -34.592
## - Year         1  1.52725 2.3666 -20.578
## - Unemployed   1  2.18989 3.0292 -16.628
## - Armed.Forces 1  2.39752 3.2369 -15.568
## 

## Step: AIC=-36.8

## Employed ~ GNP + Unemployed + Armed.Forces + Year

## 

##          Df Sum of Sq    RSS     AIC
## <none>           0.8587 -36.799
## - GNP          1  0.4647 1.3234 -31.879
## - Year         1  1.8980 2.7567 -20.137
## - Armed.Forces 1  2.3806 3.2393 -17.556
## - Unemployed   1  4.0491 4.9077 -10.908

```

```
##  
## Call:  
## lm(formula = Employed ~ GNP + Unemployed + Armed.Forces + Year,  
##      data = longley)  
  
##  
## Coefficients:  
## (Intercept)          GNP    Unemployed  Armed.Forces        Year  
## -3.599e+03 -4.019e-02 -2.088e-02 -1.015e-02 1.887e+00  
  
# step(model_1, direction = "backward")  
step(model_1, k = log(nrow(longley)))
```

```

## Start: AIC=-27.81

## Employed ~ GNP.deflator + GNP + Unemployed + Armed.Forces + Population +
##      Year

## 

##          Df Sum of Sq    RSS     AIC
## - GNP.deflator 1  0.00292 0.83935 -30.528
## - Population   1  0.00475 0.84117 -30.493
## - GNP          1  0.10631 0.94273 -28.669
## <none>           0.83642 -27.811
## - Year         1  1.49881 2.33524 -14.156
## - Unemployed   1  1.59014 2.42656 -13.542
## - Armed.Forces 1  2.16091 2.99733 -10.162
## 

## Step: AIC=-30.53

## Employed ~ GNP + Unemployed + Armed.Forces + Population + Year

## 

##          Df Sum of Sq    RSS     AIC
## - Population   1  0.01933 0.8587 -32.936
## - GNP          1  0.14637 0.9857 -30.729
## <none>           0.8393 -30.528
## - Year         1  1.52725 2.3666 -16.715
## - Unemployed   1  2.18989 3.0292 -12.765
## - Armed.Forces 1  2.39752 3.2369 -11.705
## 

## Step: AIC=-32.94

## Employed ~ GNP + Unemployed + Armed.Forces + Year

## 

##          Df Sum of Sq    RSS     AIC
## <none>           0.8587 -32.936
## - GNP            1  0.4647 1.3234 -28.788
## - Year           1  1.8980 2.7567 -17.046
## - Armed.Forces   1  2.3806 3.2393 -14.466
## - Unemployed    1  4.0491 4.9077 -7.818

```

```
##  
## Call:  
## lm(formula = Employed ~ GNP + Unemployed + Armed.Forces + Year,  
##      data = longley)  
##  
## Coefficients:  
## (Intercept)          GNP    Unemployed  Armed.Forces        Year  
## -3.599e+03 -4.019e-02 -2.088e-02 -1.015e-02 1.887e+00  
  
  
model_0 <- lm(Employed ~ 1, data = longley)  
step(model_0, direction = "forward", scope = formula(model_1))
```

```

## Start: AIC=41.17

## Employed ~ 1

## 

##          Df Sum of Sq      RSS      AIC
## + GNP      1  178.973  6.036 -11.597
## + Year     1  174.552 10.457 -2.806
## + GNP.deflator 1  174.397 10.611 -2.571
## + Population 1  170.643 14.366  2.276
## + Unemployed 1   46.716 138.293 38.509
## + Armed.Forces 1   38.691 146.318 39.411
## <none>           185.009  41.165

## 

## Step: AIC=-11.6

## Employed ~ GNP

## 

##          Df Sum of Sq      RSS      AIC
## + Unemployed 1   2.45708 3.5791 -17.9598
## + Population  1   2.16178 3.8744 -16.6913
## + Year        1   1.12520 4.9109 -12.8980
## <none>           6.0361 -11.5972
## + GNP.deflator 1   0.21194 5.8242 -10.1691
## + Armed.Forces 1   0.07665 5.9595 -9.8017

## 

## Step: AIC=-17.96

## Employed ~ GNP + Unemployed

## 

##          Df Sum of Sq      RSS      AIC
## + Armed.Forces 1   0.82235 2.7567 -20.137
## <none>           3.5791 -17.960
## + Year         1   0.33980 3.2393 -17.556
## + Population   1   0.09682 3.4822 -16.399
## + GNP.deflator 1   0.01884 3.5602 -16.044

## 

## Step: AIC=-20.14

## Employed ~ GNP + Unemployed + Armed.Forces

## 

##          Df Sum of Sq      RSS      AIC
## + Year       1   1.89803 0.85868 -36.799

```

```

## + Population    1  0.39011 2.36660 -20.578
## <none>                      2.75671 -20.137
## + GNP.deflator  1  0.07288 2.68383 -18.566
##
## Step: AIC=-36.8
## Employed ~ GNP + Unemployed + Armed.Forces + Year
##
##          Df Sum of Sq      RSS      AIC
## <none>             0.85868 -36.799
## + Population    1  0.019332 0.83935 -35.163
## + GNP.deflator  1  0.017507 0.84117 -35.129

##
## Call:
## lm(formula = Employed ~ GNP + Unemployed + Armed.Forces + Year,
##     data = longley)
##
## Coefficients:
## (Intercept)           GNP      Unemployed   Armed.Forces        Year
## -3.599e+03 -4.019e-02 -2.088e-02 -1.015e-02 1.887e+00

step(model_0, direction = "forward", scope = formula(model_1), k = log(nrow(longley)))

```

```

## Start: AIC=41.94
## Employed ~ 1
##
##          Df Sum of Sq      RSS      AIC
## + GNP      1  178.973  6.036 -10.052
## + Year     1  174.552 10.457 -1.261
## + GNP.deflator 1  174.397 10.611 -1.025
## + Population 1  170.643 14.366  3.822
## + Unemployed 1   46.716 138.293 40.054
## + Armed.Forces 1   38.691 146.318 40.956
## <none>           185.009 41.938
##
## Step: AIC=-10.05
## Employed ~ GNP
##
##          Df Sum of Sq      RSS      AIC
## + Unemployed 1   2.45708 3.5791 -15.6420
## + Population  1   2.16178 3.8744 -14.3736
## + Year        1   1.12520 4.9109 -10.5802
## <none>           6.0361 -10.0520
## + GNP.deflator 1   0.21194 5.8242 -7.8513
## + Armed.Forces 1   0.07665 5.9595 -7.4839
##
## Step: AIC=-15.64
## Employed ~ GNP + Unemployed
##
##          Df Sum of Sq      RSS      AIC
## + Armed.Forces 1   0.82235 2.7567 -17.046
## <none>           3.5791 -15.642
## + Year         1   0.33980 3.2393 -14.466
## + Population   1   0.09682 3.4822 -13.308
## + GNP.deflator 1   0.01884 3.5602 -12.954
##
## Step: AIC=-17.05
## Employed ~ GNP + Unemployed + Armed.Forces
##
##          Df Sum of Sq      RSS      AIC
## + Year       1   1.89803 0.85868 -32.936

```

```

## <none>                      2.75671 -17.046
## + Population      1   0.39011 2.36660 -16.715
## + GNP.deflator   1   0.07288 2.68383 -14.703
##
## Step: AIC=-32.94
## Employed ~ GNP + Unemployed + Armed.Forces + Year
##
##          Df Sum of Sq    RSS    AIC
## <none>             0.85868 -32.936
## + Population      1  0.019332 0.83935 -30.528
## + GNP.deflator   1  0.017507 0.84117 -30.493

##
## Call:
## lm(formula = Employed ~ GNP + Unemployed + Armed.Forces + Year,
##     data = longley)
##
## Coefficients:
## (Intercept)           GNP      Unemployed  Armed.Forces        Year
## -3.599e+03 -4.019e-02 -2.088e-02 -1.015e-02 1.887e+00

```

10.2 Zadania

Zadanie 1. Zbiór danych w pliku [Automobile.csv](#) zawiera dane charakteryzujące różne typy samochodów.

```

## symboling normalized.losses      make fuel.type aspiration
## 1      3          NA alfa-romero    gas    std
## 2      3          NA alfa-romero    gas    std
## 3      1          NA alfa-romero    gas    std
## 4      2         164 audi        gas    std
## 5      2         164 audi        gas    std
## 6      2          NA audi        gas    std

## num.of.doors body.style drive.wheels engine.location wheel.base length
## 1      two convertible     rwd   front  88.6  168.8
## 2      two convertible     rwd   front  88.6  168.8
## 3      two hatchback       rwd   front  94.5  171.2
## 4      four sedan          fwd   front  99.8  176.6
## 5      four sedan          4wd   front  99.4  176.6
## 6      two sedan           fwd   front  99.8  177.3

## width height curb.weight engine.type num.of.cylinders engine.size
## 1 64.1 48.8    2548 dohc      four    130
## 2 64.1 48.8    2548 dohc      four    130
## 3 65.5 52.4    2823 ohcv      six     152
## 4 66.2 54.3    2337 ohc       four    109
## 5 66.4 54.3    2824 ohc       five    136
## 6 66.3 53.1    2507 ohc       five    136

## fuel.system bore stroke compression.ratio horsepower peak.rpm city.mpg
## 1      mpfi 3.47  2.68          9.0    111  5000    21
## 2      mpfi 3.47  2.68          9.0    111  5000    21
## 3      mpfi 2.68  3.47          9.0    154  5000    19
## 4      mpfi 3.19  3.40         10.0    102  5500    24
## 5      mpfi 3.19  3.40          8.0    115  5500    18
## 6      mpfi 3.19  3.40          8.5    110  5500    19

## highway.mpg price
## 1      27 13495
## 2      27 16500
## 3      26 16500
## 4      30 13950
## 5      22 17450
## 6      25 15250

```

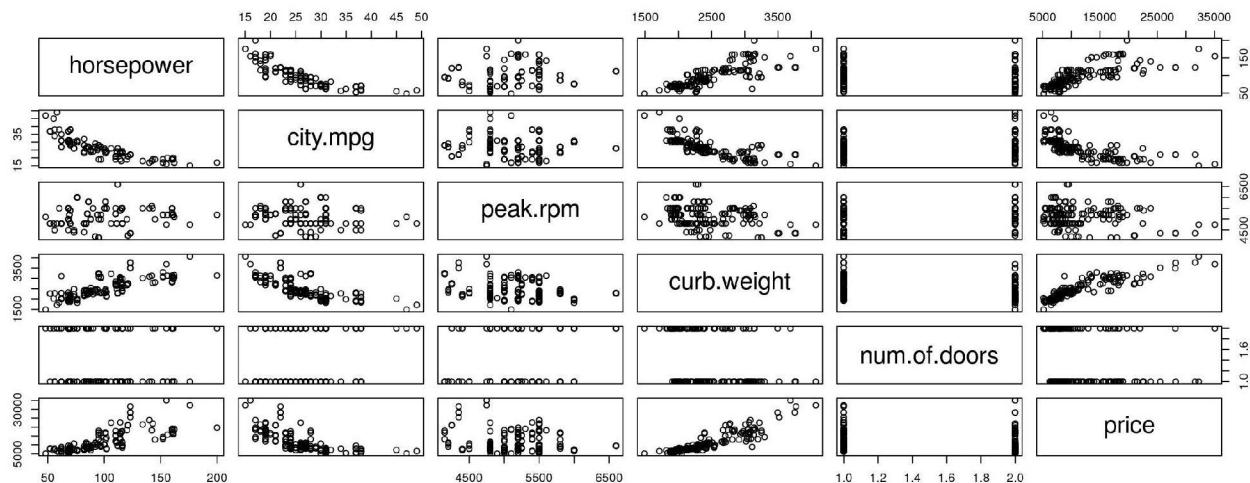
1. W tym zestawie danych występują braki danych. Usuń wszystkie obserwacje, dla których nie mamy pełnych informacji o wszystkich zmiennych zawartych w zbiorze danych, używając funkcji `na.omit()`.

```
## wymiar nowych danych
```

```
## [1] 159 26
```

2. Interesuje nas zbudowanie modelu opisującego cenę samochodów w zależności od pewnych ich cech. Weźmy pod uwagę następujące zmienne: `horsepower`, `city.mpg`, `peak.rpm`, `curb.weight` i `num.of.doors` jako zmienne niezależne.

- Dopasuj model regresji liniowej do tych danych.



```
##  
## Call:  
## lm(formula = price ~ horsepower + city.mpg + peak.rpm + curb.weight +  
##      num.of.doors, data = auto_wna)  
##  
## Coefficients:  
## (Intercept)      horsepower      city.mpg      peak.rpm  
## -2.185e+04      2.743e+01      7.733e+01      4.847e-01  
## curb.weight  num.of.doorstwo  
## 1.053e+01      5.516e+02
```

- Jakie są wartości estymatorów współczynników regresji i przedziały ufności? Które zmienne są stymulantami a które destymulantami?

11 Regresja logistyczna i Poissona

11.1 Przykłady

Regresja logistyczna

Przykład. Rozważmy przykład dotyczący badania szansy ponownego ataku serca w ciągu roku od pierwszego ataku, w zależności od *treatment of anger* oraz *trait anxiety*. Zmienna zależna ma wartość 1, jeśli nastąpił ponowny atak, a 0 w przeciwnym razie.

```
y <- c(1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
x1 <- c(1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0)
x2 <- c(70, 80, 50, 60, 40, 65, 75, 80, 70, 60, 65, 50, 45, 35, 40, 50, 55, 45, 50, 60)
data_set <- data.frame(y, x1, x2)
head(data_set)

##   y x1 x2
## 1 1  1 70
## 2 1  1 80
## 3 1  1 50
## 4 1  0 60
## 5 1  0 40
## 6 1  0 65

# model Logistyczny
model_1 <- glm(y ~ x1 + x2, data = data_set, family = 'binomial')
model_1
```

```
##  
## Call: glm(formula = y ~ x1 + x2, family = "binomial", data = data_set)  
##  
## Coefficients:  
## (Intercept)          x1          x2  
##       -6.363       -1.024        0.119  
##  
## Degrees of Freedom: 19 Total (i.e. Null); 17 Residual  
## Null Deviance:      27.73  
## Residual Deviance: 18.82      AIC: 24.82  
  
  
  
# podsumowanie modelu  
# tj. reszty, estymacja punktowa, testy istotności dla współczynników regresji, AIC  
summary(model_1)
```

```

##  

## Call:  

## glm(formula = y ~ x1 + x2, family = "binomial", data = data_set)  

##  

## Deviance Residuals:  

##      Min       1Q   Median       3Q      Max  

## -1.52106 -0.68746  0.00424  0.70625  1.88960  

##  

## Coefficients:  

##              Estimate Std. Error z value Pr(>|z|)  

## (Intercept) -6.36347   3.21362 -1.980   0.0477 *  

## x1          -1.02411   1.17101 -0.875   0.3818  

## x2           0.11904   0.05497  2.165   0.0304 *  

## ---  

## Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  

##  

## (Dispersion parameter for binomial family taken to be 1)  

##  

## Null deviance: 27.726  on 19  degrees of freedom  

## Residual deviance: 18.820  on 17  degrees of freedom  

## AIC: 24.82  

##  

## Number of Fisher Scoring iterations: 4

```

```

# zredukowany model logistyczny  

model_2 <- glm(y ~ x2, data = data_set, family = 'binomial')  

summary(model_2)

```

```

## 
## Call:
## glm(formula = y ~ x2, family = "binomial", data = data_set)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.62461 -0.83983 -0.01232  0.64540  2.10801
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -7.0925    3.1709  -2.237  0.0253 *
## x2          0.1246    0.0553   2.254  0.0242 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 27.726 on 19 degrees of freedom
## Residual deviance: 19.601 on 18 degrees of freedom
## AIC: 23.601
##
## Number of Fisher Scoring iterations: 4

```

regresja krokowa

AIC(model_1, model_2)

```

##        df      AIC
## model_1 3 24.82037
## model_2 2 23.60052

```

step(model_1)

```

## Start: AIC=24.82
## y ~ x1 + x2
##
##          Df Deviance    AIC
## - x1     1  19.601 23.601
## <none>   18.820 24.820
## - x2     1  25.878 29.878
##
## Step: AIC=23.6
## y ~ x2
##
##          Df Deviance    AIC
## <none>   19.601 23.601
## - x2     1  27.726 29.726

##
## Call: glm(formula = y ~ x2, family = "binomial", data = data_set)
##
## Coefficients:
## (Intercept)      x2
## -7.0925       0.1246
##
## Degrees of Freedom: 19 Total (i.e. Null);  18 Residual
## Null Deviance:    27.73
## Residual Deviance: 19.6  AIC: 23.6

#
# # iloraz szans (ręcznie)
# exp(coef(model_2)[2])

##
##      x2
## 1.132734

```

```

# Wartość ta oznacza, że wraz ze wzrostem wartości zmiennej x2 o jedną jednostkę,
# przewidywane ryzyko ponownego zawału serca wzrasta o 13%.
# do krzywych ROC

library(ROCR)

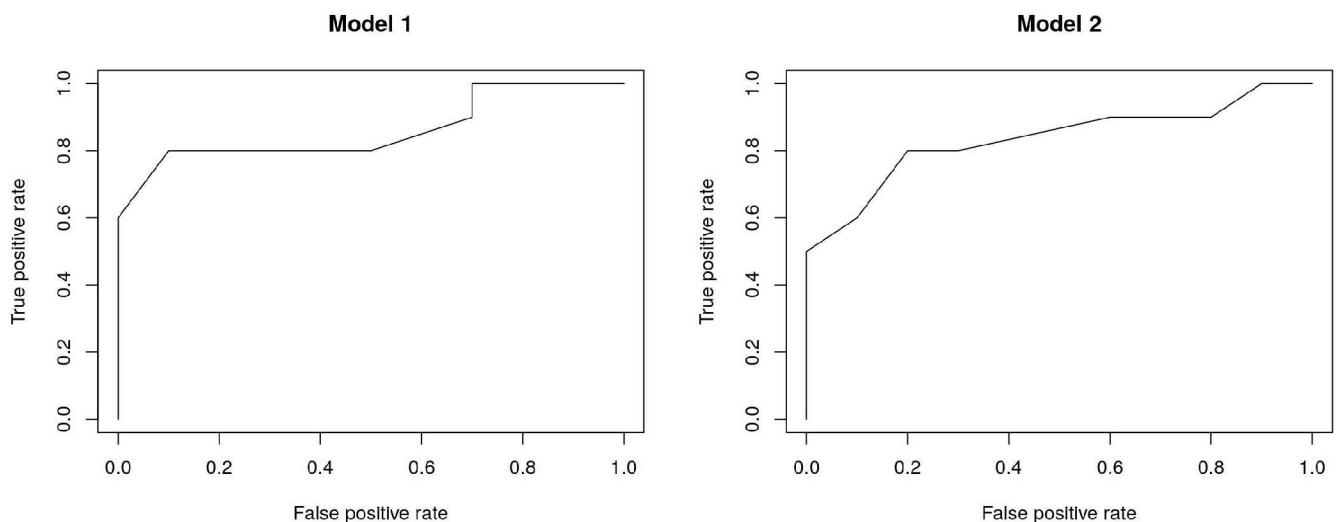
pred_1 <- prediction(model_1$fitted, y)
pred_2 <- prediction(model_2$fitted, y)

# krzywe ROC

par(mfrow = c(1, 2))

plot(performance(pred_1, 'tpr', 'fpr'), main = "Model 1")
plot(performance(pred_2, 'tpr', 'fpr'), main = "Model 2")

```



```

par(mfrow = c(1, 1))

# AUC

performance(pred_1, 'auc')@y.values

## [[1]]
## [1] 0.86

performance(pred_2, 'auc')@y.values

## [[1]]
## [1] 0.835

```

```

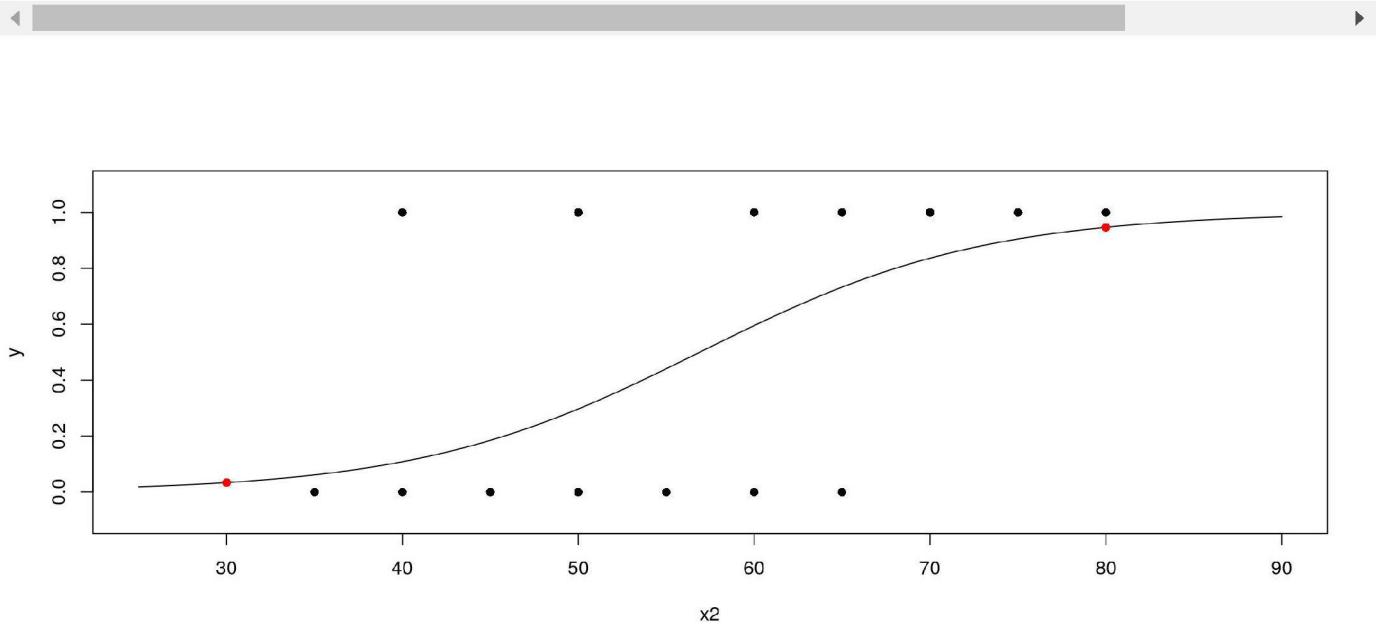
# predykcja
(predict_glm <- predict(model_2,
                         data.frame(x2 = c(30, 80)),
                         type = 'response'))

##          1         2
## 0.03378247 0.94676209

# Uwzględniamy argument type = 'response' w celu uzyskania przewidywanego prawdopodobieństwa
# Domyślne przewidywane są z logarytmowane i lorazy szans (prawdopodobieństwa w skali logitowej)
x_temp <- seq(min(x2) - 10, max(x2) + 10, length.out = 100)
y_temp <- exp(coef(model_2)[1] + coef(model_2)[2] * x_temp) /
  (1 + exp(coef(model_2)[1] + coef(model_2)[2] * x_temp))

plot(x_temp, y_temp, type = "l", xlab = "x2", ylab = "y", ylim = c(-0.1, 1.1))
points(x2, y, pch = 16)
points(c(30, 80), predict_glm, pch = 16, col = "red")

```



Regresja Poissona

Nie zawsze interesuje nas prawdopodobieństwo sukcesu. Dość często jesteśmy zainteresowani liczbą sukcesów (ogólnie liczebnościami). W tej sytuacji najbardziej popularny jest model Poissona, który zakłada, że zmienna zależna ma rozkład Poissona i

$$h(x) = \ln(x), \quad E(Y) = \exp(\mathbf{X}\boldsymbol{\beta}).$$

Przykład. W zbiorze danych `student_award.RData`, zmienna `num_awards` podaje liczbę nagród zdobytych przez uczniów szkoły średniej przez rok, zmienna `math` jest zmienną ciągłą i reprezentuje wyniki uczniów na końcowym egzaminie z matematyki, a zmienna `prog` jest zmienną jakościową z trzema poziomami wskazującymi rodzaj programu, ma który uczniowie byli zapisani ("General" - ogólny, "Academic" - akademicki, "Vocational" - zawodowy). Chcemy opisać związek między liczbą nagród a wynikiem egzaminu z matematyki i programem.

```
load(url("http://ls.home.amu.edu.pl/data_sets/student_award.RData"))
head(student_award)
```

```
##   num_awards math      prog
## 1          0   41 Vocational
## 2          0   41    General
## 3          0   44 Vocational
## 4          0   42 Vocational
## 5          0   40 Vocational
## 6          0   42    General
```

```
model_1 <- glm(num_awards ~ math + prog, data = student_award, family = "poisson")
model_1
```

```
##
## Call:  glm(formula = num_awards ~ math + prog, family = "poisson", data = student_award)
##
## Coefficients:
## (Intercept)           math      progAcademic  progVocational
##           -5.24712       0.07015       1.08386       0.36981
##
## Degrees of Freedom: 199 Total (i.e. Null);  196 Residual
## Null Deviance:      287.7
## Residual Deviance: 189.4      AIC: 373.5
```

```
summary(model_1)
```

```

## 
## Call:
## glm(formula = num_awards ~ math + prog, family = "poisson", data = student_award)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.2043 -0.8436 -0.5106  0.2558  2.6796
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -5.24712   0.65845 -7.969 1.60e-15 ***
## math         0.07015   0.01060  6.619 3.63e-11 ***
## progAcademic 1.08386   0.35825  3.025  0.00248 **
## progVocational 0.36981   0.44107  0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
## Null deviance: 287.67 on 199 degrees of freedom
## Residual deviance: 189.45 on 196 degrees of freedom
## AIC: 373.5
##
## Number of Fisher Scoring iterations: 6

```

Możemy również przetestować ogólny efekt programu, porównując pełny model z modelem bez zmiennej program. Test chi-kwadrat wskazuje, że program, jest statystycznie istotnym predyktorem liczby nagród.

```

model_2 <- update(model_1, . ~ . - prog)
anova(model_1, model_2, test = "Chisq")

```

```
## Analysis of Deviance Table
##
## Model 1: num_awards ~ math + prog
## Model 2: num_awards ~ math
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      196    189.45
## 2      198    204.02 -2   -14.572 0.0006852 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
AIC(model_1, model_2)
```

```
##          df      AIC
## model_1  4 373.5045
## model_2  2 384.0762
```

```
step(model_1)
```

```
## Start:  AIC=373.5
## num_awards ~ math + prog
##
##          Df Deviance     AIC
## <none>     189.45 373.50
## - prog    2   204.02 384.08
## - math    1   234.46 416.51
```

```
##  
## Call: glm(formula = num_awards ~ math + prog, family = "poisson", data = student_awards)  
##  
## Coefficients:  
## (Intercept) math progAcademic progVocational  
## -5.24712 0.07015 1.08386 0.36981  
##  
## Degrees of Freedom: 199 Total (i.e. Null); 196 Residual  
## Null Deviance: 287.7  
## Residual Deviance: 189.4 AIC: 373.5
```



```
(data_new <- data.frame(math = mean(student_award$math),  
                        prog = factor(1:3, levels = 1:3,  
                        labels = levels(student_award$prog))))
```

```
## math prog  
## 1 52.645 General  
## 2 52.645 Academic  
## 3 52.645 Vocational
```

```
(pred <- predict(model_1, data_new, type = "response"))
```

```
## 1 2 3  
## 0.2114109 0.6249446 0.3060086
```

```

student_award$num_award_hat <- predict(model_1, type = "response")
# sortowanie według programu, a następnie według wyniku z matematyki
student_award <- student_award[with(student_award, order(prog, math)), ]
par(mfrow = c(1, 3))

plot(student_award$math[student_award$prog == "General"],
      student_award$num_award_hat[student_award$prog == "General"],
      type = "l", lwd = 2, col = "red",
      xlim = c(min(student_award$math), max(student_award$math)), ylim = c(0, 6),
      xlab = "Math score", ylab = "Award number", main = "General")

points(student_award$math[student_award$prog == "General"],
       student_award$num_awards[student_award$prog == "General"], pch = 16)

points(mean(student_award$math), pred[1], pch = 16, col = "blue", lwd = 4)

plot(student_award$math[student_award$prog == "Academic"],
      student_award$num_award_hat[student_award$prog == "Academic"],
      type = "l", lwd = 2, col = "red",
      xlim = c(min(student_award$math), max(student_award$math)), ylim = c(0, 6),
      xlab = "Math score", ylab = "Award number", main = "Academic")

points(student_award$math[student_award$prog == "Academic"],
       student_award$num_awards[student_award$prog == "Academic"], pch = 16)

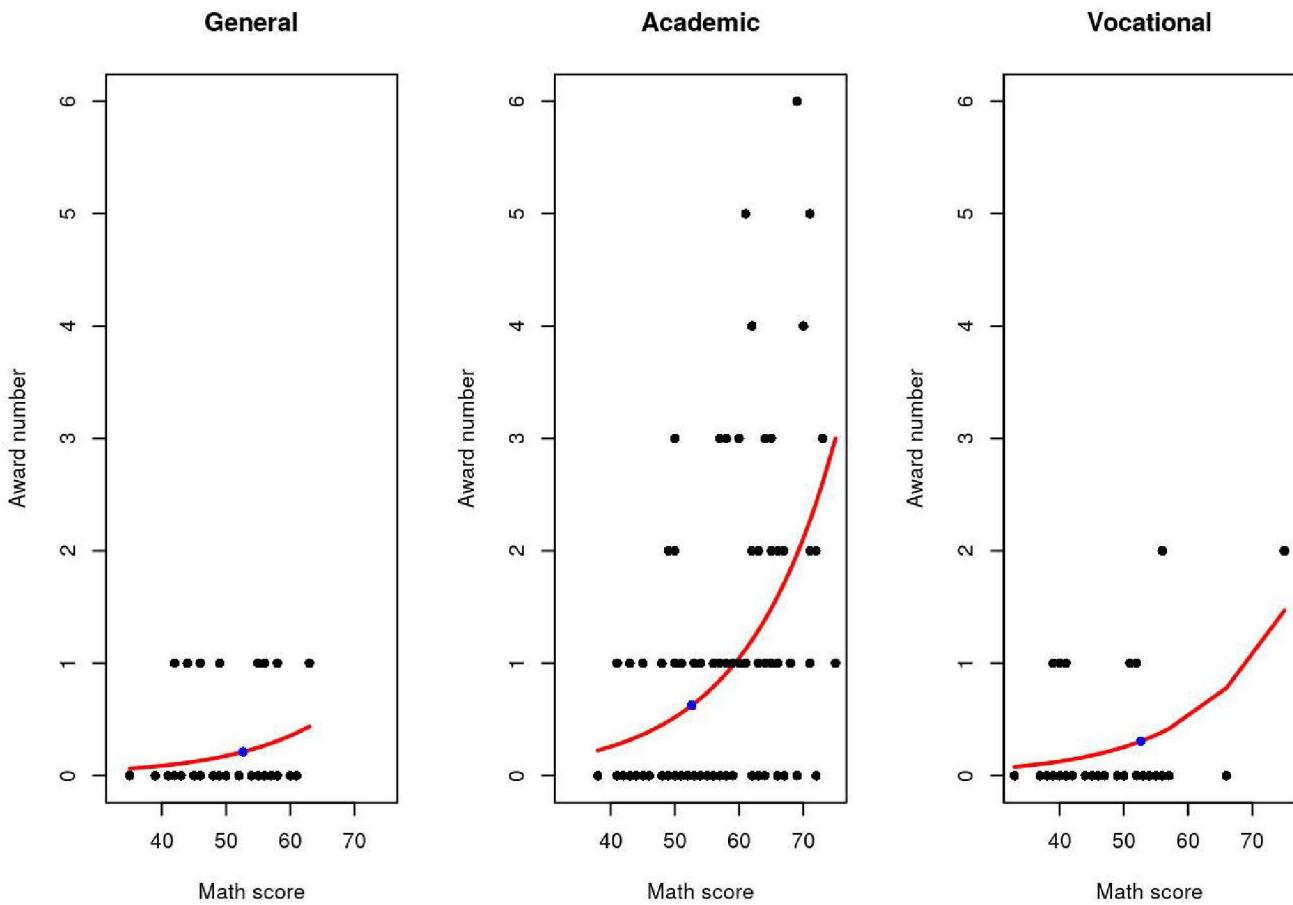
points(mean(student_award$math), pred[2], pch = 16, col = "blue", lwd = 4)

plot(student_award$math[student_award$prog == "Vocational"],
      student_award$num_award_hat[student_award$prog == "Vocational"],
      type = "l", lwd = 2, col = "red",
      xlim = c(min(student_award$math), max(student_award$math)), ylim = c(0, 6),
      xlab = "Math score", ylab = "Award number", main = "Vocational")

points(student_award$math[student_award$prog == "Vocational"],
       student_award$num_awards[student_award$prog == "Vocational"], pch = 16)

points(mean(student_award$math), pred[3], pch = 16, col = "blue", lwd = 4)

```



```
par(mfrow = c(1, 1))
```

11.2 Zadania

Zadanie 1. W jednym badaniu klinicznym oceniono wpływ poziomów enzymu LDH i zmian poziomów bilirubiny na zdrowie pacjentów z przewlekłym zapaleniem wątroby. Uzyskane wyniki są zawarte w pliku `liver_data.RData`. Zmienne to: `bilirubin` - zmiana stężenia bilirubiny we krwi, `ldh` - stężenie enzymu LDH w ciele pacjenta, `condition` - zmiana stanu pacjenta (`Yes` - pogorszenie, `No` - brak pogorszenia).

```
## bilirubin ldh condition
## 1      0.9  75      No
## 2      0.8 150      No
## 3      0.6 250      No
## 4      0.8 375     Yes
## 5      3.2 160     Yes
## 6      1.7 106      No
```

1. Dopasuj model regresji logistycznej do tych danych. Jakie są wartości estymatorów współczynników regresji?

```
##  
## Call: glm(formula = condition ~ bilirubin + ldh, family = "binomial",  
##      data = liver_data)  
##  
## Coefficients:  
## (Intercept)    bilirubin          ldh  
## -8.13113      2.88050       0.02464  
##  
## Degrees of Freedom: 38 Total (i.e. Null); 36 Residual  
## Null Deviance: 54.04  
## Residual Deviance: 33.11      AIC: 39.11
```

2. Które współczynniki są statystycznie istotne w skonstruowanym modelu? Jakie jest dopasowanie modelu?

```

##  

## Call:  

##   glm(formula = condition ~ bilirubin + ldh, family = "binomial",  

##       data = liver_data)  

##  

## Deviance Residuals:  

##      Min        1Q    Median        3Q       Max  

## -2.05593 -0.79191  0.04353  0.57765  2.11829  

##  

## Coefficients:  

##              Estimate Std. Error z value Pr(>|z|)  

## (Intercept) -8.131132  2.639959 -3.080  0.00207 **  

## bilirubin    2.880497  1.105836  2.605  0.00919 **  

## ldh         0.024635  0.008764  2.811  0.00494 **  

## ---  

## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  

##  

## (Dispersion parameter for binomial family taken to be 1)  

##  

## Null deviance: 54.040  on 38  degrees of freedom  

## Residual deviance: 33.114  on 36  degrees of freedom  

## AIC: 39.114  

##  

## Number of Fisher Scoring iterations: 6

```

3. Czy model ten może być zredukowany za pomocą regresji krokowej?

```

## Start:  AIC=39.11  

## condition ~ bilirubin + ldh  

##  

##              Df Deviance     AIC  

## <none>            33.114 39.114  

## - ldh             1   46.989 50.989  

## - bilirubin      1   48.726 52.726

```

```

## 
## Call: glm(formula = condition ~ bilirubin + ldh, family = "binomial",
##           data = liver_data)
## 
## Coefficients:
## (Intercept)    bilirubin          ldh
## -8.13113      2.88050       0.02464
## 
## Degrees of Freedom: 38 Total (i.e. Null); 36 Residual
## Null Deviance:      54.04
## Residual Deviance: 33.11      AIC: 39.11

```

4. Zinterpretuj współczynniki modelu.

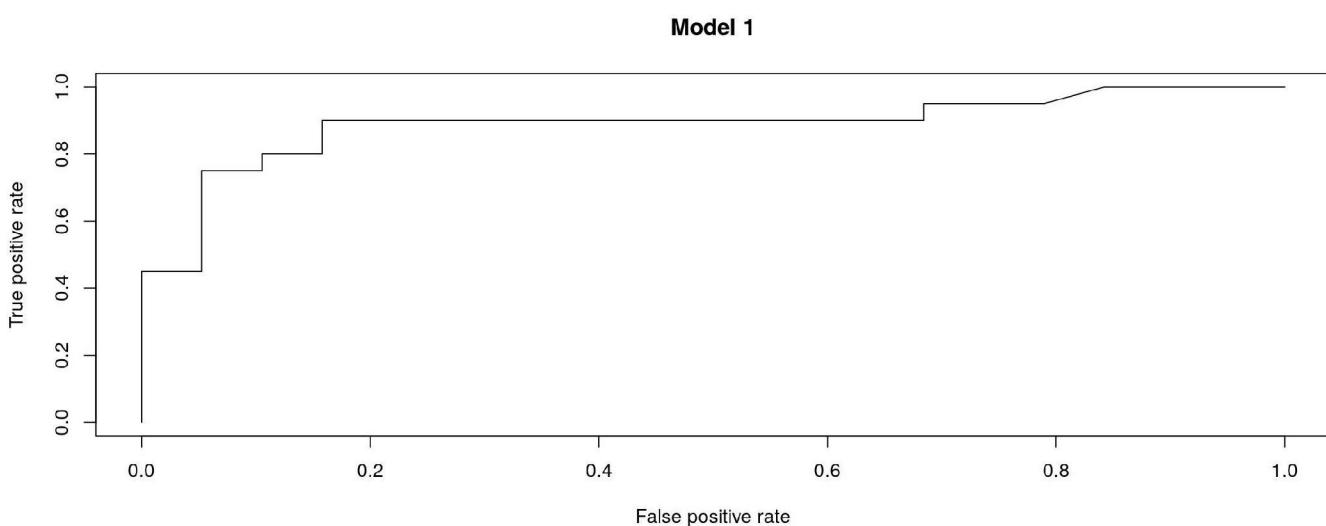
```

## bilirubin
## 17.82313

## ldh
## 1.024941

```

5. Narysuj krzywą ROC i oblicz AUC dla modelu.



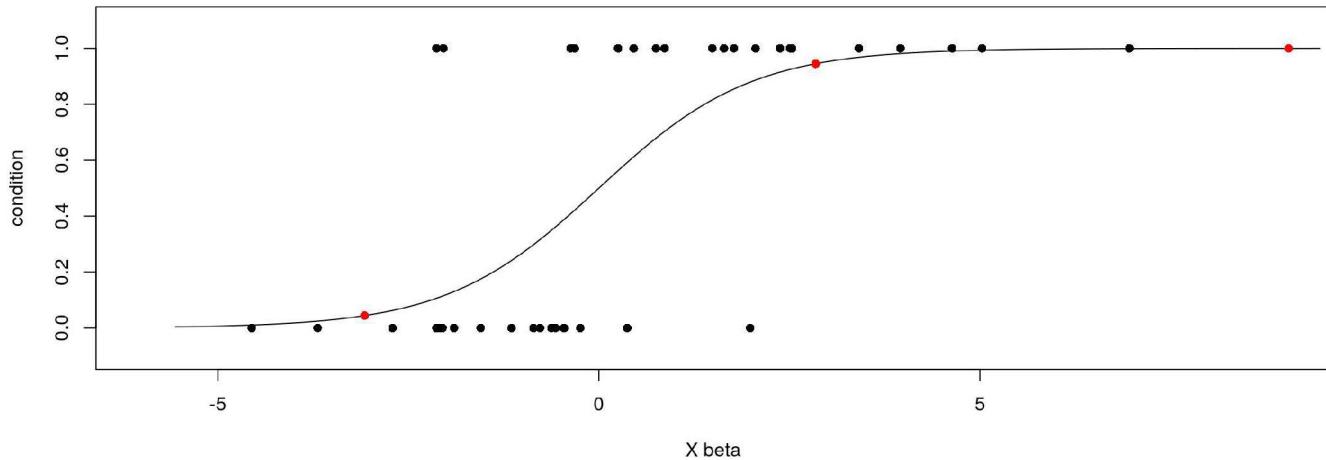
```

## [[1]]
## [1] 0.8881579

```

6. Dokonaj przedykcji zmiennej `condition` dla trzech pacjentów scharakteryzowanych następująco: (`bilirubin`, `ldh`) = (0.9, 100), (2.1, 200), (3.4, 300). Zilustruj wyniki na wykresie.

```
##           1            2            3
## 0.04414365 0.94505776 0.99988299
```



7. Powyższy wykres pokazuje, że istnieją dwie obserwacje odstające dla pacjentów z pogorszeniem i jedna obserwacja odstająca dla pacjentów bez pogorszenia. Zidentyfikuj je i wykonaj powyższą analizę dla danych bez tych trzech wartości odstających. Jak zmieniają się wyniki?

1.

```
##
## Call: glm(formula = condition ~ bilirubin + ldh, family = "binomial",
##           data = liver_data_wo)
##
## Coefficients:
## (Intercept)    bilirubin         ldh
## -72.7256      30.2781       0.1947
##
## Degrees of Freedom: 35 Total (i.e. Null); 33 Residual
## Null Deviance: 49.91
## Residual Deviance: 6.207   AIC: 12.21
```

2.

```

##  

## Call:  

##   glm(formula = condition ~ bilirubin + ldh, family = "binomial",  

##     data = liver_data_wo)  

##  

## Deviance Residuals:  

##       Min        1Q    Median        3Q       Max  

## -0.93161 -0.01879  0.00000  0.00047  1.89807  

##  

## Coefficients:  

##             Estimate Std. Error z value Pr(>|z|)  

## (Intercept) -72.7256   45.3298 -1.604   0.109  

## bilirubin    30.2781   18.9417  1.598   0.110  

## ldh          0.1947    0.1235  1.577   0.115  

##  

## (Dispersion parameter for binomial family taken to be 1)  

##  

## Null deviance: 49.9066 on 35 degrees of freedom  

## Residual deviance: 6.2068 on 33 degrees of freedom  

## AIC: 12.207  

##  

## Number of Fisher Scoring iterations: 10

```

3.

```

## Start: AIC=12.21  

## condition ~ bilirubin + ldh  

##  

##             Df Deviance    AIC  

## <none>           6.207 12.207  

## - ldh            1   38.422 42.422  

## - bilirubin     1   44.216 48.216

```

```

## 
## Call: glm(formula = condition ~ bilirubin + ldh, family = "binomial",
##           data = liver_data_wo)
## 
## Coefficients:
## (Intercept)    bilirubin          ldh
## -72.7256      30.2781       0.1947
## 
## Degrees of Freedom: 35 Total (i.e. Null); 33 Residual
## Null Deviance: 49.91
## Residual Deviance: 6.207      AIC: 12.21

```

4.

```

##    bilirubin
## 1.411294e+13

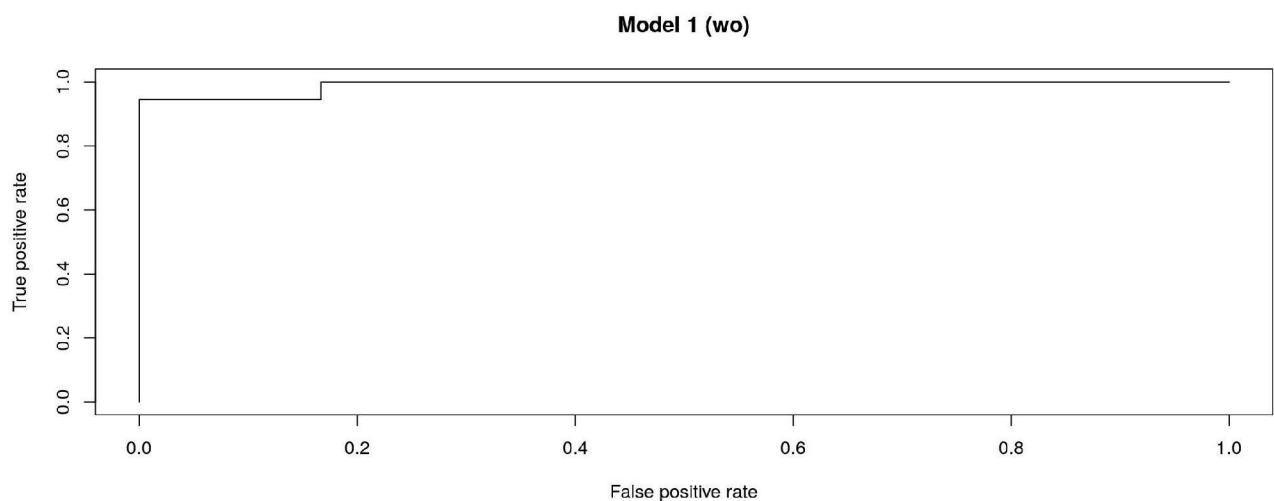
```

```

##    ldh
## 1.214999

```

5.



```

## [[1]]
## [1] 0.9907407

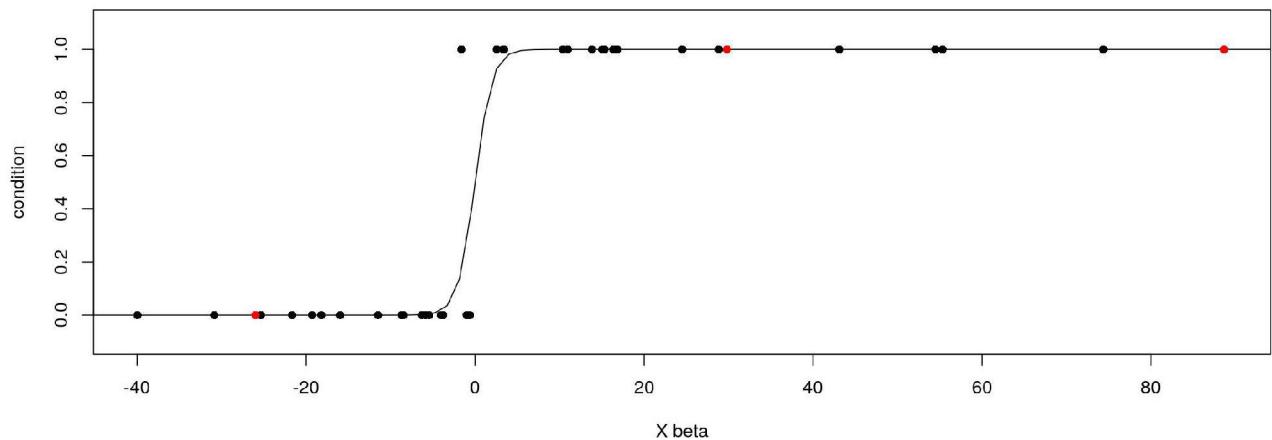
```

6.

```

##          1          2          3
## 5.104082e-12 1.000000e+00 1.000000e+00

```



Zadanie 2. Użyj modelu regresji Poissona do zestawu danych `moths` (wpływ siedliska na liczbę moli) z pakietu `DAAG`. Użyj zlogarytmowanej zmiennej `meters` jako zmiennej objaśniającej, a liczby moli `A` jako zmiennej objaśnianej.

```

##   meters A P   habitat
## 1     25 9 8   NWsoak
## 2     37 3 20  SWsoak
## 3    109 7 9 LowerSide
## 4     10 0 2 LowerSide
## 5    133 9 1 Upperside
## 6     26 3 18 Disturbed

```

1. Dopasuj model regresji Poissona do tych danych. Jakie są wartości estymatorów współczynników regresji?

12 Analiza korelacji

12.1 Przykład

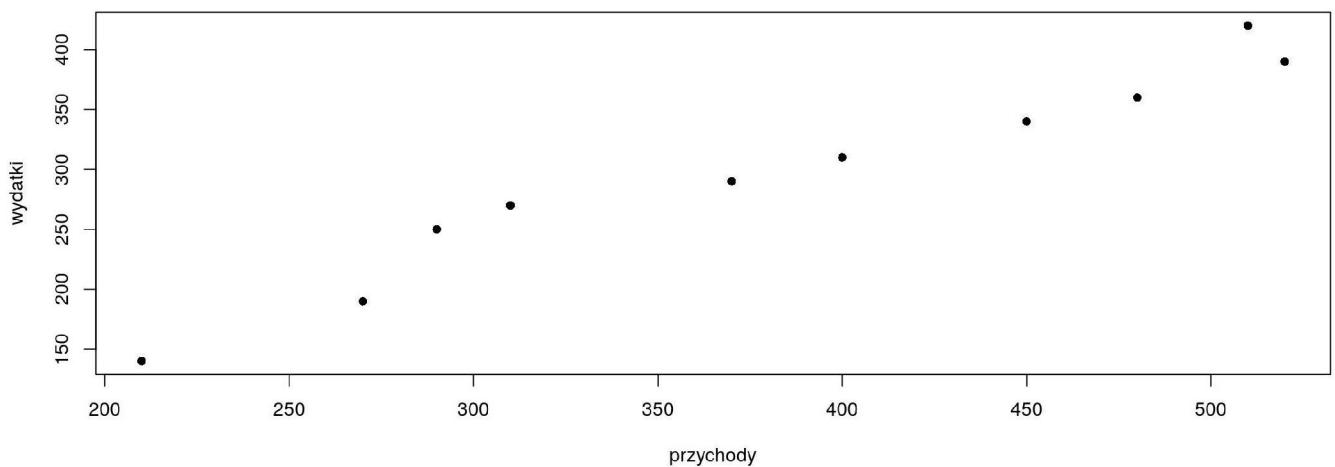
Przykład. Chcemy zbadać, czy istnieje zależność między miesięcznym dochodem rodziny na jedną osobę a miesięczną wartością wydatków na jedną osobę. Dane dotyczące tych dwóch cech dla dziesięciu rodzin podano w poniższej tabeli.

rodzina	1	2	3	4	5	6	7	8	9	10
przychody	210	270	290	310	370	400	450	480	510	520
wydatki	140	190	250	270	290	310	340	360	420	390

```
przychody <- c(210, 270, 290, 310, 370, 400, 450, 480, 510, 520)
wydatki <- c(140, 190, 250, 270, 290, 310, 340, 360, 420, 390)
data_set <- data.frame(przychody = przychody, wydatki = wydatki)
head(data_set)

##  przychody wydatki
## 1      210     140
## 2      270     190
## 3      290     250
## 4      310     270
## 5      370     290
## 6      400     310

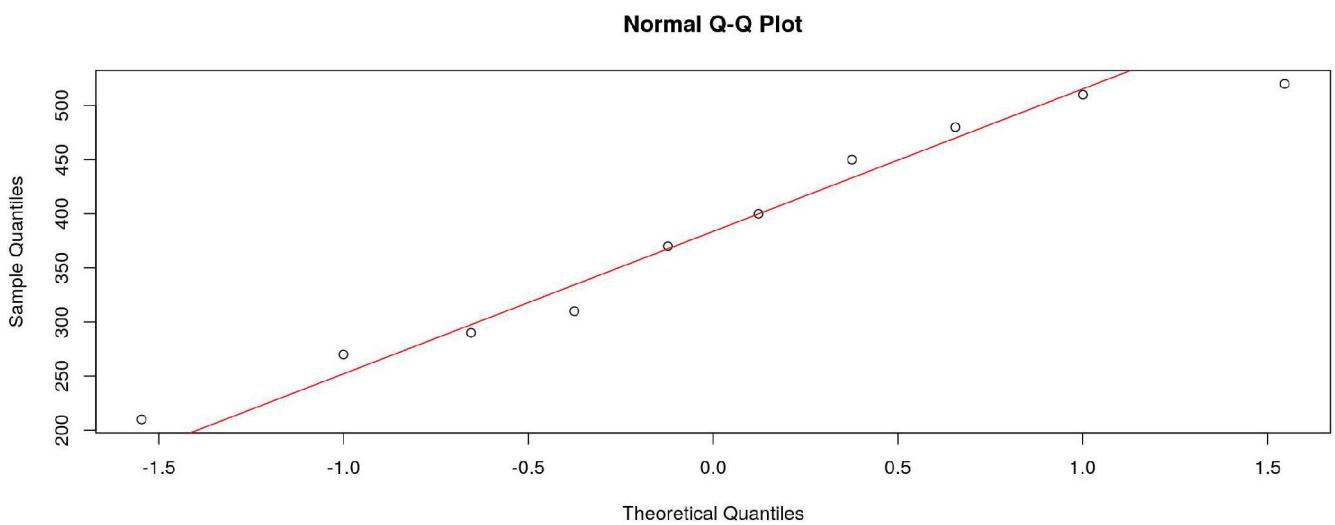
# wykres rozrzutu
plot(data_set$przychody, data_set$wydatki, xlab = "przychody", ylab = "wydatki", pch = :)
```



```
# założenia
shapiro.test(data_set$przychody)

##
## Shapiro-Wilk normality test
##
## data: data_set$przychody
## W = 0.94256, p-value = 0.5819

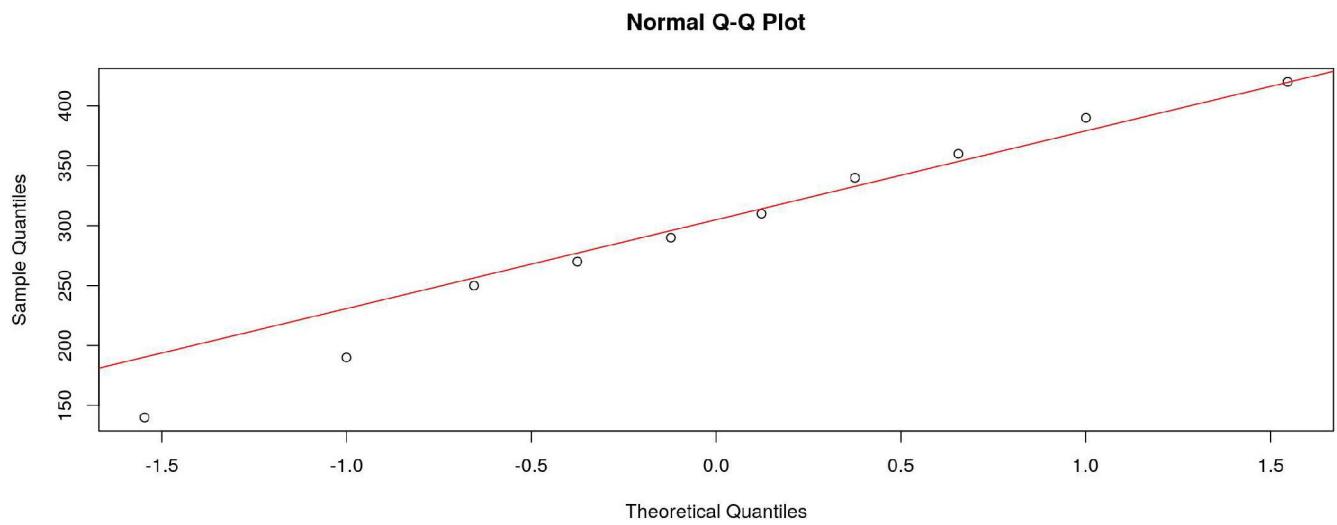
qqnorm(data_set$przychody)
qqline(data_set$przychody, col = "red")
```



```
shapiro.test(data_set$wydatki)

##
##  Shapiro-Wilk normality test
##
## data: data_set$wydatki
## W = 0.97753, p-value = 0.9506
```

```
qqnorm(data_set$wydatki)
qqline(data_set$wydatki, col = "red")
```



```
# testy
cor.test(data_set$przychody, data_set$wydatki, method = "pearson")
```

```
##  
## Pearson's product-moment correlation  
##  
## data: data_set$przychody and data_set$wydatki  
## t = 12.399, df = 8, p-value = 1.67e-06  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.8942997 0.9942521  
## sample estimates:  
## cor  
## 0.9749541
```

```
cor.test(data_set$przychody, data_set$wydatki, method = "kendall")
```

```
##  
## Kendall's rank correlation tau  
##  
## data: data_set$przychody and data_set$wydatki  
## T = 44, p-value = 5.511e-06  
## alternative hypothesis: true tau is not equal to 0  
## sample estimates:  
## tau  
## 0.9555556
```

```
cor.test(data_set$przychody, data_set$wydatki, method = "spearman")
```

```

## Spearman's rank correlation rho
##
## data: data_set$przychody and data_set$wydatki
## S = 2, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##      rho
## 0.9878788

```

12.2 Zadania

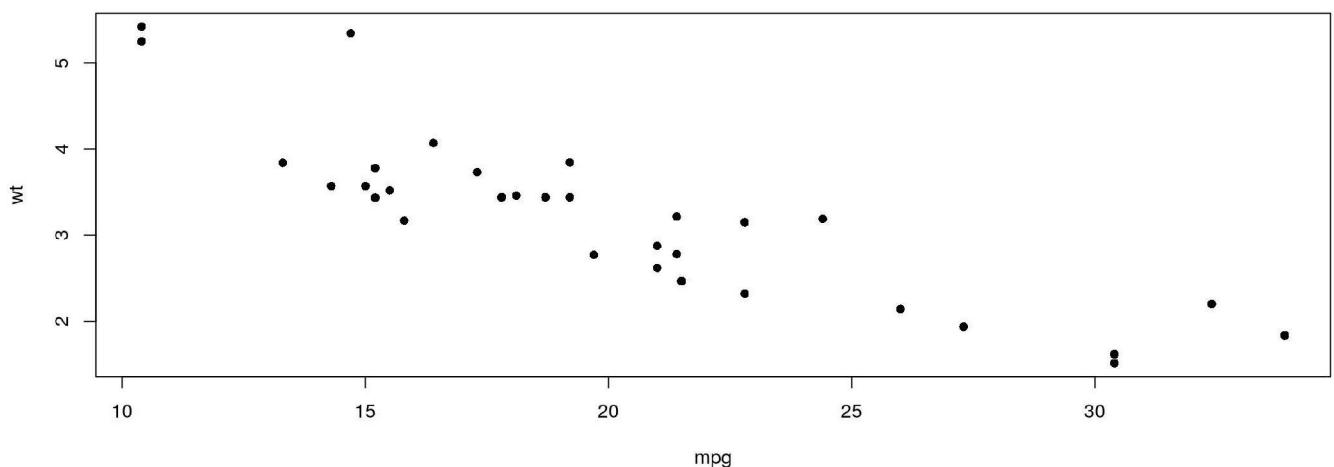
Zadanie 1. Zbiór danych `mtcars` zawiera dane dotyczące pewnych cech samochodów. Interesuje nas zbadanie korelacji między zmiennymi `mpg` i `wt`.

```

##          mpg cyl disp hp drat    wt  qsec vs am gear carb
## Mazda RX4     21.0   6 160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag 21.0   6 160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710    22.8   4 108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive 21.4   6 258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360 175 3.15 3.440 17.02  0  0    3    2
## Valiant       18.1   6 225 105 2.76 3.460 20.22  1  0    3    1

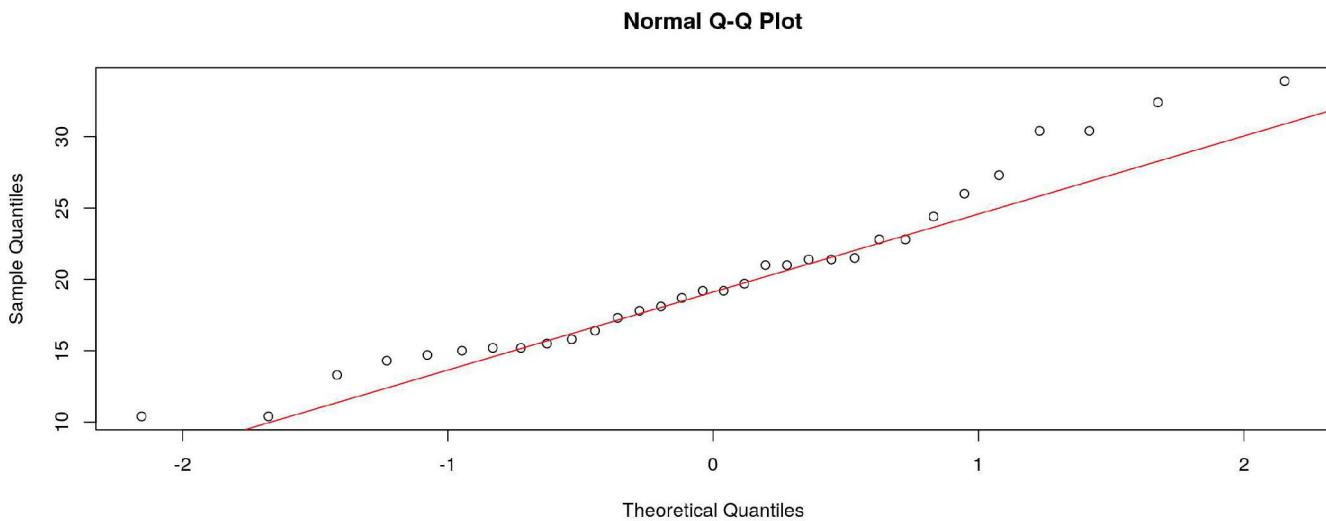
```

1. Wykonaj wykres rozrzutu dla badanych cech.

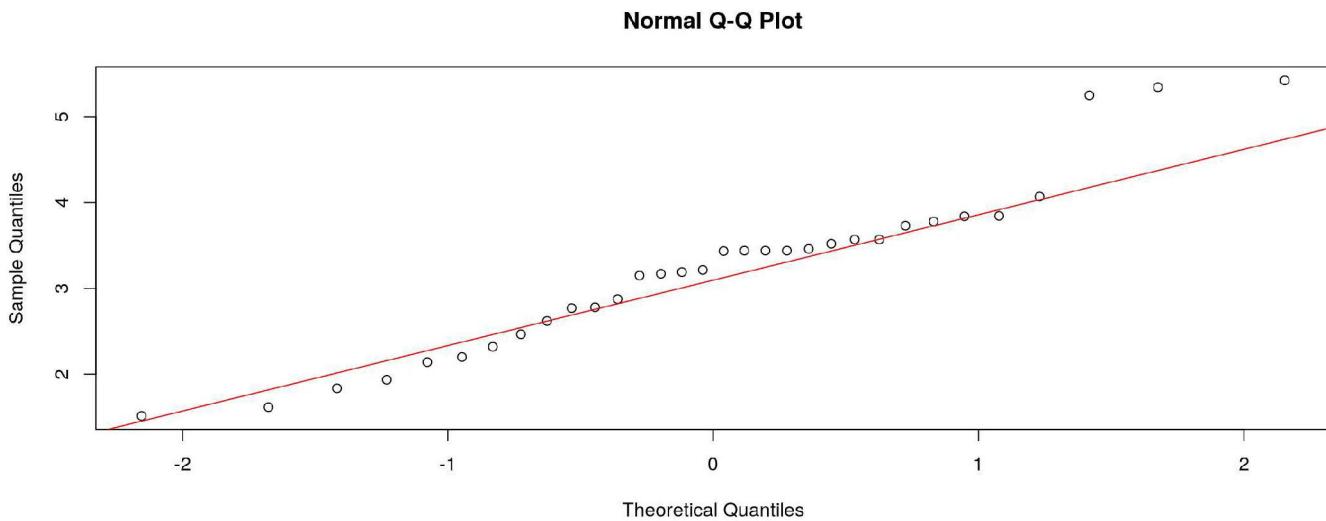


2. Sprawdź założenia testu istotności dla współczynnika korelacji.

```
## [1] 0.1228814
```



```
## [1] 0.09265499
```



3. Wykonaj test istotności dla współczynnika korelacji dla zmiennych `mpg` i `wg`. Oszacuj punktowo i przedziałowo współczynnik korelacji.

```
## [1] 1.293959e-10
```

```
##          cor
## ## -0.8676594
```

```
## [1] -0.9338264 -0.7440872
## attr(,"conf.level")
## [1] 0.95
```

4. Wykonaj polecenie punktu 3 korzystając ze współczynników Kendalla i Spearmana.

```
## [1] 6.70577e-09
```

```
##      tau
## -0.7278321
```

```
## [1] 1.487595e-11
```

```
##      rho
## -0.886422
```

13 Analiza składowych głównych

13.1 Przykład

Przykład. Zbiór danych `USArrests` zawiera informacje dotyczące liczby morderstw, napadów, gwałtów przypadających na 100,000 osób w poszczególnych stanach USA w roku 1973 oraz procent ludności mieszkającej w miastach. Chcielibyśmy się dowiedzieć, czy stany są do siebie w pewien sposób zbliżone oraz spróbować zwizualizować je na płaszczyźnie.

```
head(USArrests)
```

```
##           Murder Assault UrbanPop Rape
## Alabama     13.2     236      58 21.2
## Alaska      10.0     263      48 44.5
## Arizona     8.1      294      80 31.0
## Arkansas    8.8      190      50 19.5
## California   9.0      276      91 40.6
## Colorado     7.9      204      78 38.7
```

```
dim(USArrests)
```

```
## [1] 50  4
```

- przygotowanie danych do analizy składowych głównych

```
# sprawdzamy czy wariancje (,,zmiennosci'') zmiennych sa bardzo zróżnicowane
var(USArrests)
```

```

##           Murder   Assault  UrbanPop      Rape
## Murder    18.970465 291.0624  4.386204 22.99141
## Assault   291.062367 6945.1657 312.275102 519.26906
## UrbanPop   4.386204  312.2751 209.518776 55.76808
## Rape      22.991412  519.2691  55.768082 87.72916

```

```

# tak się, więc centrujemy i skalujemy funkcją scale
USArrests_scale <- scale(USArrests)
var(USArrests_scale)

```

```

##           Murder   Assault  UrbanPop      Rape
## Murder    1.00000000 0.8018733 0.06957262 0.5635788
## Assault   0.80187331 1.0000000 0.25887170 0.6652412
## UrbanPop  0.06957262 0.2588717 1.00000000 0.4113412
## Rape     0.56357883 0.6652412 0.41134124 1.0000000

```

- model analizy składowych głównych w R i procent wyjaśnianej wariancji zmiennych oryginalnych przez poszczególne składowe główne

```

pca <- prcomp(USArrests, scale = TRUE)
# Lub
# pca <- prcomp(USArrests_scale)
pca

## Standard deviations (1, ..., p=4):
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
##
## Rotation (n x k) = (4 x 4):
##           PC1        PC2        PC3        PC4
## Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
## Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
## Rape     -0.5434321 -0.1673186  0.8177779  0.08902432

```

```

# bez skalowania

prcomp(USArrests)

## Standard deviations (1, ..., p=4):
## [1] 83.732400 14.212402  6.489426  2.482790
##
## Rotation (n x k) = (4 x 4):
##          PC1        PC2        PC3        PC4
## Murder   0.04170432 -0.04482166  0.07989066 -0.99492173
## Assault   0.99522128 -0.05876003 -0.06756974  0.03893830
## UrbanPop 0.04633575  0.97685748 -0.20054629 -0.05816914
## Rape     0.07515550  0.20071807  0.97408059  0.07232502

```

```
summary(pca)
```

```

## Importance of components:
##          PC1        PC2        PC3        PC4
## Standard deviation   1.5749  0.9949  0.59713  0.41645
## Proportion of Variance 0.6201  0.2474  0.08914  0.04336
## Cumulative Proportion  0.6201  0.8675  0.95664  1.00000

```

- wyniki (ang. scores) - współrzędne obserwacji w nowym układzie współrzędnych utworzonym przez składowe główne (to one najczęściej podlegają wizualizacji)

```
head(pca$x)
```

```

##          PC1        PC2        PC3        PC4
## Alabama   -0.9756604  1.1220012 -0.43980366  0.154696581
## Alaska    -1.9305379  1.0624269  2.01950027 -0.434175454
## Arizona   -1.7454429 -0.7384595  0.05423025 -0.826264240
## Arkansas   0.1399989  1.1085423  0.11342217 -0.180973554
## California -2.4986128 -1.5274267  0.59254100 -0.338559240
## Colorado   -1.4993407 -0.9776297  1.08400162  0.001450164

```

- ładunki (ang. *loadings*) - współczynniki pokazujące wkład poszczególnych zmiennych bazowych w tworzenie składowych głównych (im wartość bezwzględna z ładunku jest większa, tym zmienna ma większy wkład w budowę składowej głównej)

```
pca$rotation
```

	PC1	PC2	PC3	PC4
## Murder	-0.5358995	0.4181809	-0.3412327	0.64922780
## Assault	-0.5831836	0.1879856	-0.2681484	-0.74340748
## UrbanPop	-0.2781909	-0.8728062	-0.3780158	0.13387773
## Rape	-0.5434321	-0.1673186	0.8177779	0.08902432

- wykres osypiska (piargowy, ang. *scree plot*) - wykres wariancji poszczególnych składowych głównych

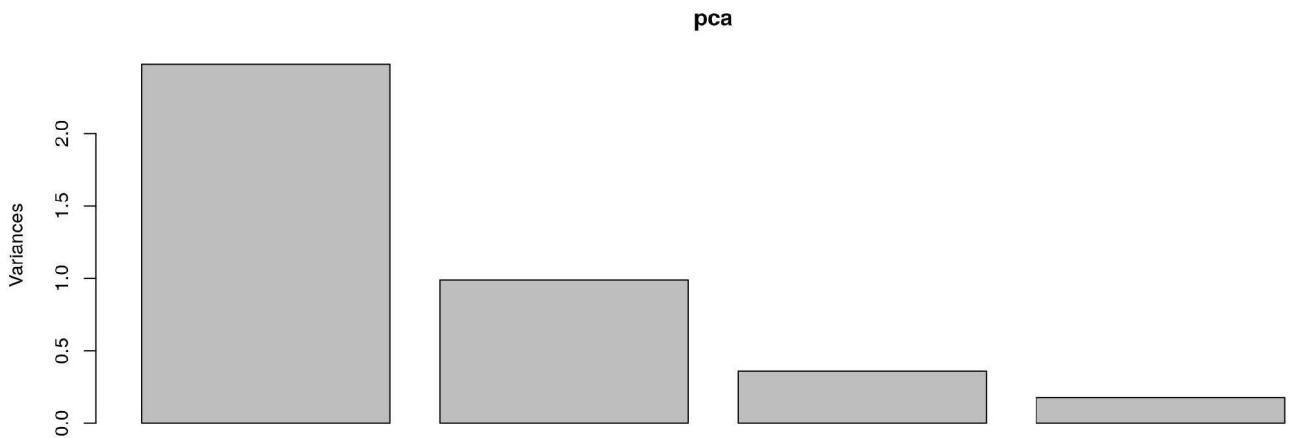
```
pca$sdev^2
```

```
## [1] 2.4802416 0.9897652 0.3565632 0.1734301
```

```
apply(pca$x, 2, var)
```

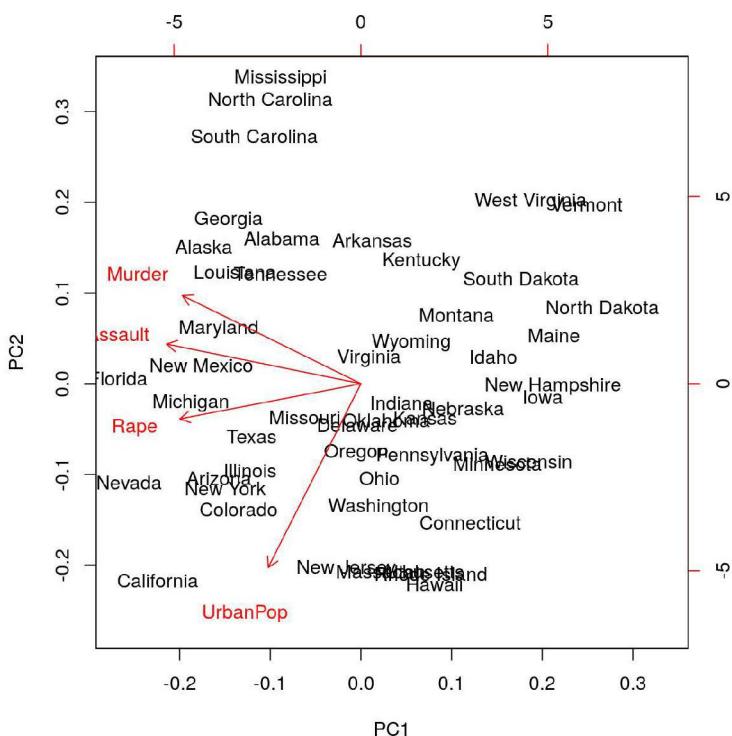
	PC1	PC2	PC3	PC4
##	2.4802416	0.9897652	0.3565632	0.1734301

```
plot(pca)
```



- biplot (ang. *biplot*) - wykres, na którym punkty przedstawiają poszczególne obserwacje w nowym układzie dwóch pierwszych składowych głównych, a strzałki oznaczają zmienne. Kierunek strzałek pokazuje wpływ tych zmiennych odpowiednio na pierwszą i drugą składową główną. Kąt przecięcia strzałek jest proporcjonalny do zależności pomiędzy zmiennymi, a ich długość odzwierciedla odchylenie standardowe.

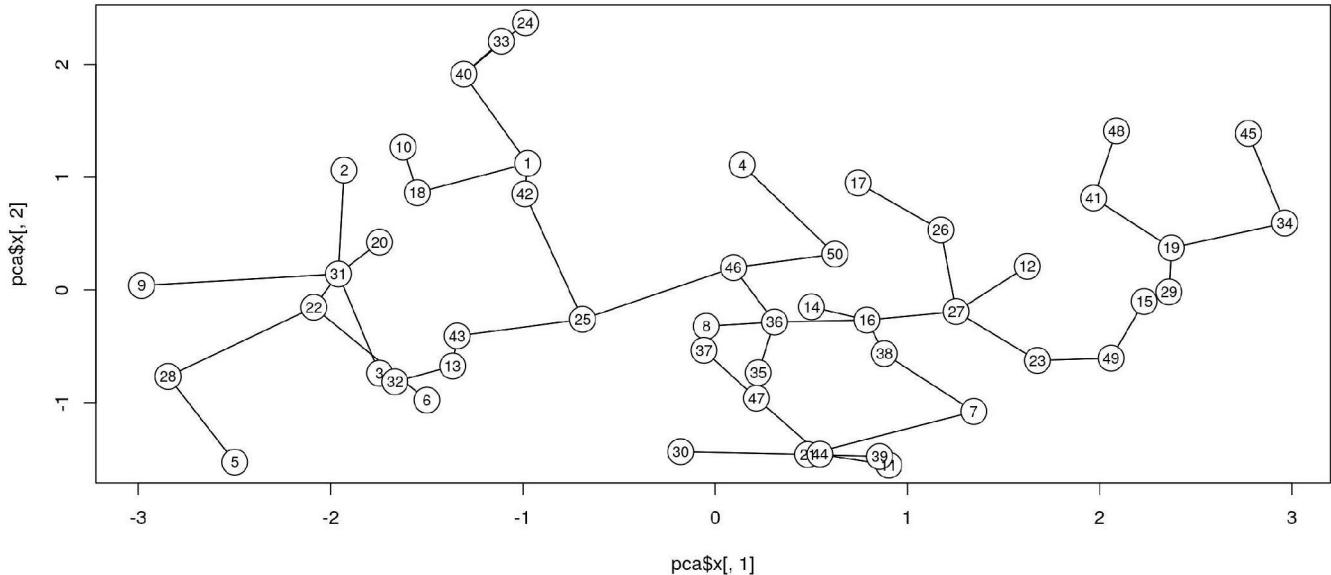
biplot(pca)



- Żeby stwierdzić, czy taki wykres jest adekwatnym odzwierciedleniem położenia oryginalnych punktów, można na niego nanieść minimalne drzewo rozpinające (MST). MST to graf, którego wierzchołkami są obserwacje, dwa punkty połączone są dokładnie

jedną ścieżką, a suma krawędzi jest minimalna. Punkty połączone krawędziami powinny być blisko siebie na wykresie.

```
library(ape)
plot(mst(dist(USArrests_scale)), x1 = pca$x[, 1], x2 = pca$x[, 2])
```



```
# odczytywanie nazw obserwacji
row.names(USArrests_scale[c(24, 33),])
```



```
## [1] "Mississippi"    "North Carolina"
```

13.2 Zadania

Zadanie 1. W powyższym przykładzie do analizy składowych głównych zostały wykorzystane wszystkie zmienne. Jednak jedna z nich jest bardzo słabo skorelowana z pozostałymi. Ustal tę zmienną, a następnie wykonaj poniższe polecenia bez jej uwzględnienia:

1. Dokonaj analizy składowych głównych.

```

## Standard deviations (1, .., p=3):
## [1] 1.5357670 0.6767949 0.4282154
##
## Rotation (n x k) = (3 x 3):
##          PC1      PC2      PC3
## Murder -0.5826006 0.5339532 -0.6127565
## Assault -0.6079818 0.2140236 0.7645600
## Rape    -0.5393836 -0.8179779 -0.1999436

```

2. Jaki procent wariancji tłumaczony jest przez poszczególne składowe?

```

## Importance of components:
##          PC1      PC2      PC3
## Standard deviation     1.5358 0.6768 0.42822
## Proportion of Variance 0.7862 0.1527 0.06112
## Cumulative Proportion   0.7862 0.9389 1.00000

```

3. Wyznacz współrzędne obserwacji w nowym układzie współrzędnych utworzonym przez składowe główne.

```

##          PC1      PC2      PC3
## Alabama -1.1980278 0.8338118 -0.16217848
## Alaska  -2.3087473 -1.5239622 0.03833574
## Arizona -1.5033307 -0.4983038 0.87822311
## Arkansas -0.1759894 0.3247326 0.07111174
## California -2.0452358 -1.2725770 0.38153933
## Colorado -1.2634133 -1.4264063 -0.08369314

##
## ...

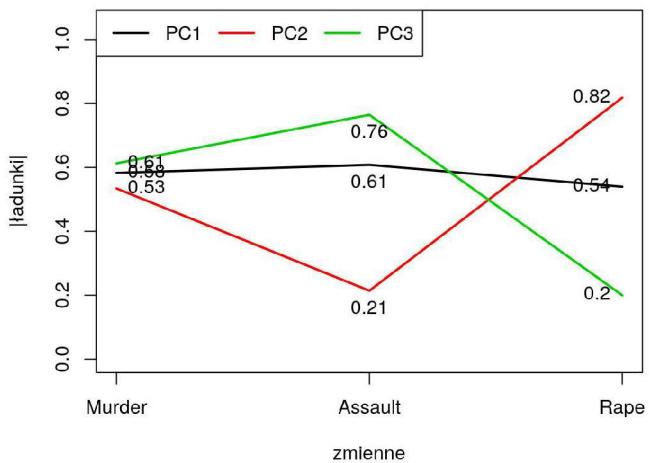
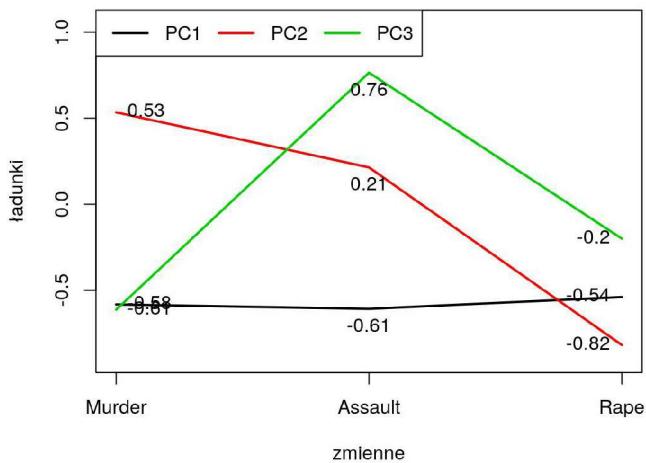
```

4. Dokonaj interpretacji ładunków i zilustruj je na wykresie.

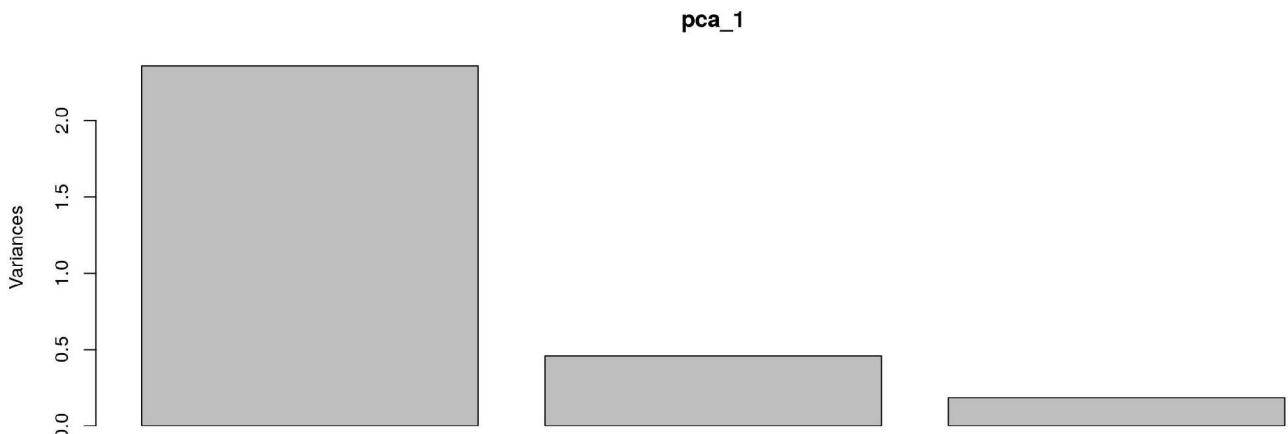
```

##          PC1      PC2      PC3
## Murder -0.5826006 0.5339532 -0.6127565
## Assault -0.6079818 0.2140236 0.7645600
## Rape    -0.5393836 -0.8179779 -0.1999436

```

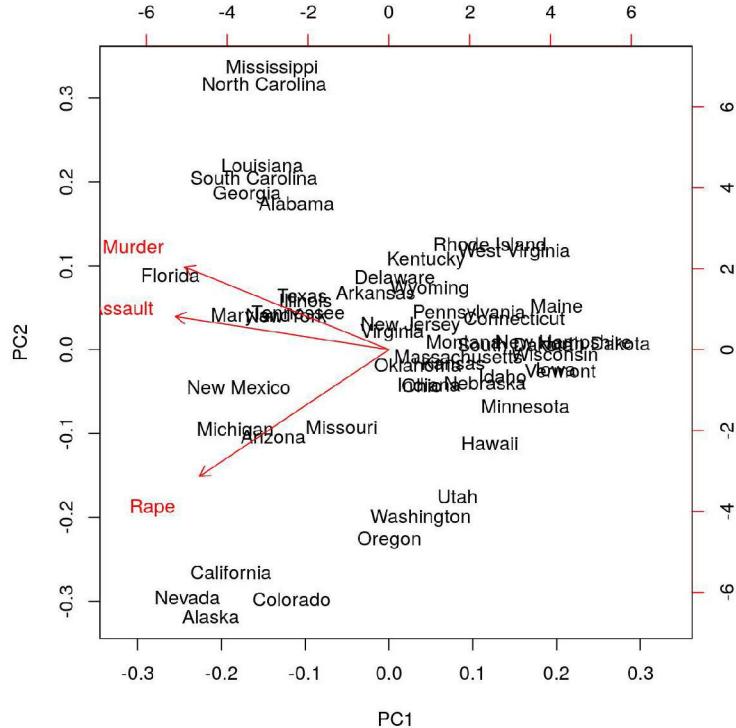


5. Narysuj wykres osypiska i zaproponuj optymalną liczbę składowych głównych w oparciu o trzy kryteria.

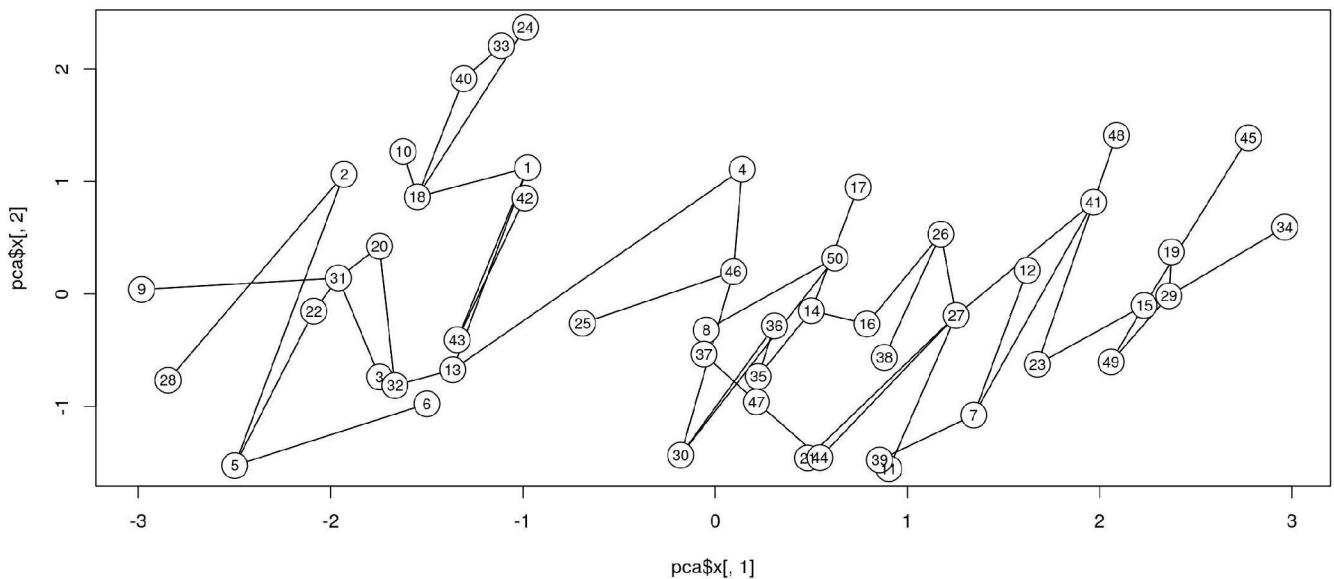


```
## 1 lub 2
```

6. Przedstaw stany w układzie dwóch pierwszych składowych głównych (dokładniej narysuj biplot i dokonaj jego interpretacji).



7. Przedstaw stany za pomocą minimalnego drzewa rozpinającego.



Zadanie 2. Zbiór danych `mtcars` zawiera informacje na temat 32 samochodów z roku 1974.

- Dokonaj analizy składowych głównych biorąc pod uwagę cechy: `mpg`, `disp`, `hp`, `drat`, `wt`, `qsec`.

```

## Standard deviations (1, .., p=6):
## [1] 2.0463129 1.0714999 0.5773705 0.3928874 0.3532648 0.2279872
##
## Rotation (n x k) = (6 x 6):
##          PC1       PC2       PC3       PC4       PC5       PC6
## mpg   -0.4586835  0.05867609 -0.19479235  0.78205878 -0.1111533 -0.35249327
## disp   0.4660354 -0.06065296  0.09688406  0.60001871  0.2946297  0.56825752
## hp     0.4258534  0.36147576  0.14613554  0.12301873 -0.8057408 -0.04771555
## drat  -0.3670963  0.43652537  0.80049152  0.02259258  0.1437714  0.11277675
## wt     0.4386179 -0.29953457  0.41776208  0.10438337  0.2301541 -0.69246040
## qsec  -0.2528320 -0.76284877  0.34059066  0.04268124 -0.4218755  0.24152663

```

2. Jaki procent wariancji tłumaczony jest przez poszczególne składowe?

```
## Importance of components:
```

```

##          PC1       PC2       PC3       PC4       PC5       PC6
## Standard deviation 2.0463 1.0715 0.57737 0.39289 0.3533 0.22799
## Proportion of Variance 0.6979 0.1913 0.05556 0.02573 0.0208 0.00866
## Cumulative Proportion 0.6979 0.8892 0.94481 0.97054 0.9913 1.00000

```

3. Wyznacz współrzędne obserwacji w nowym układzie współrzędnych utworzonym przez składowe główne.

```

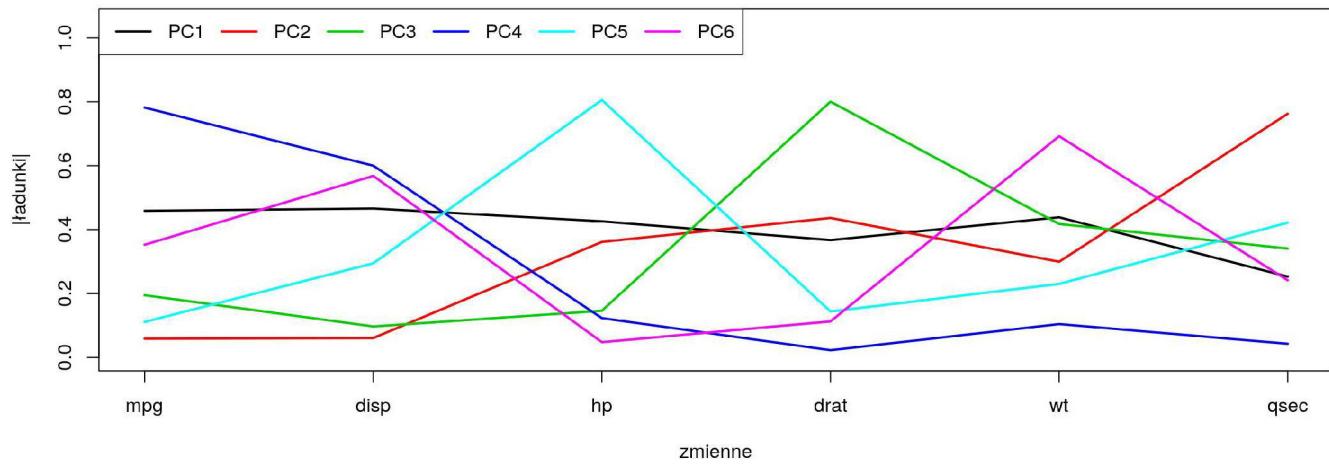
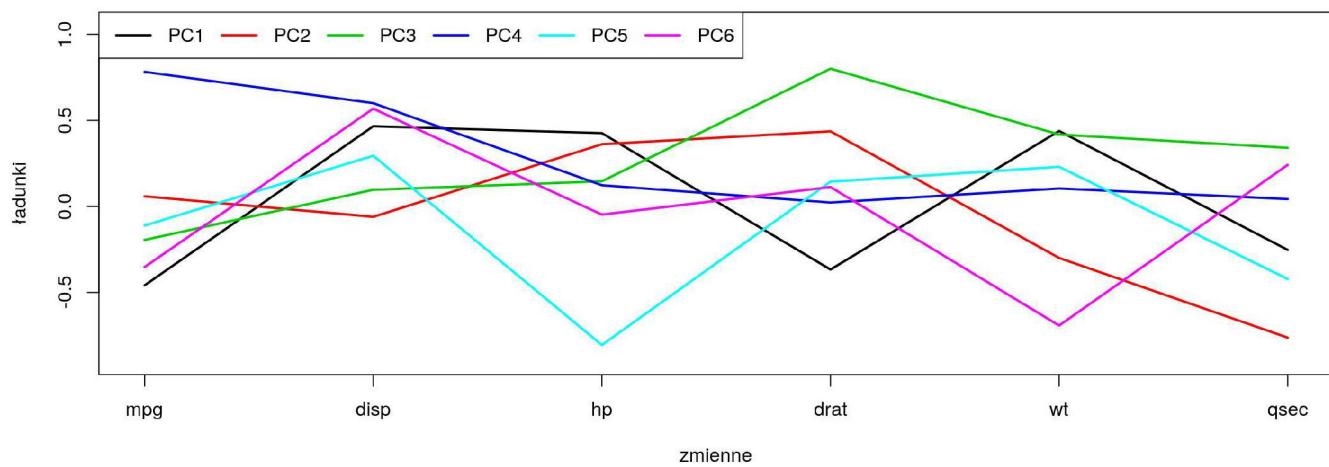
##          PC1       PC2       PC3       PC4
## Mazda RX4      -0.8425806  0.873469391 -0.2282783 -0.3742725
## Mazda RX4 Wag  -0.8075041  0.556341552 -0.0126678 -0.3336931
## Datsun 710     -1.6850448 -0.040006569 -0.1564937 -0.4057157
## Hornet 4 Drive -0.0964443 -1.294377904 -0.5702297  0.2520788
## Hornet Sportabout 1.2915096 -0.006516693 -0.5250741  0.4813192
## Valiant        0.2187309 -2.005957905 -0.7258399 -0.3136170
##          PC5       PC6
## Mazda RX4      0.51522641 -0.05293884
## Mazda RX4 Wag  0.44299870 -0.15771326
## Datsun 710     -0.03340433  0.10756126
## Hornet 4 Drive -0.04326023  0.18173489
## Hornet Sportabout 0.12822104  0.29051949
## Valiant        -0.21465335  0.09145688

```

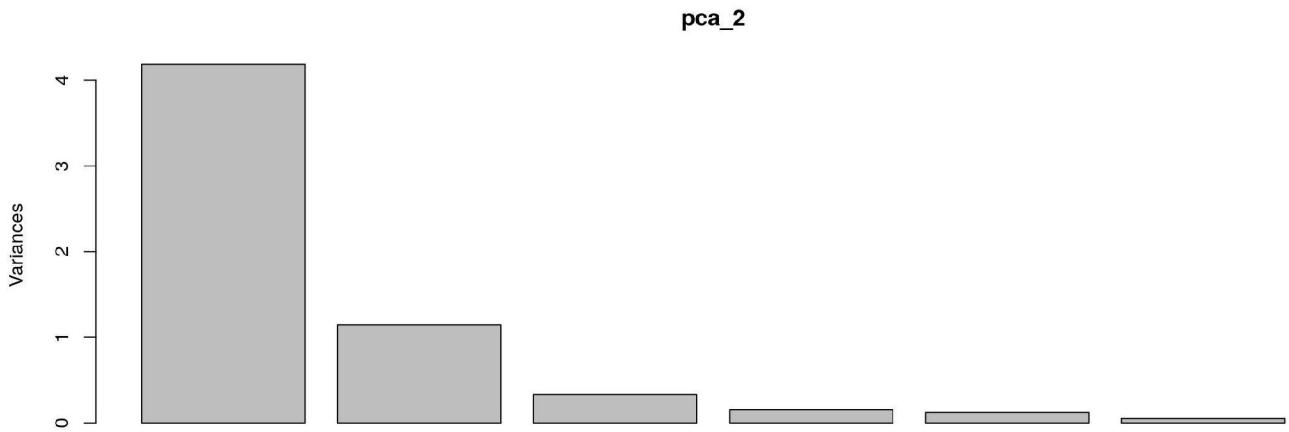
```
## ...
```

4. Dokonaj interpretacji ładunków i zilustruj je na wykresie.

```
##          PC1        PC2        PC3        PC4        PC5        PC6
## mpg -0.4586835  0.05867609 -0.19479235  0.78205878 -0.1111533 -0.35249327
## disp  0.4660354 -0.06065296  0.09688406  0.60001871  0.2946297  0.56825752
## hp    0.4258534  0.36147576  0.14613554  0.12301873 -0.8057408 -0.04771555
## drat -0.3670963  0.43652537  0.80049152  0.02259258  0.1437714  0.11277675
## wt    0.4386179 -0.29953457  0.41776208  0.10438337  0.2301541 -0.69246040
## qsec -0.2528320 -0.76284877  0.34059066  0.04268124 -0.4218755  0.24152663
```

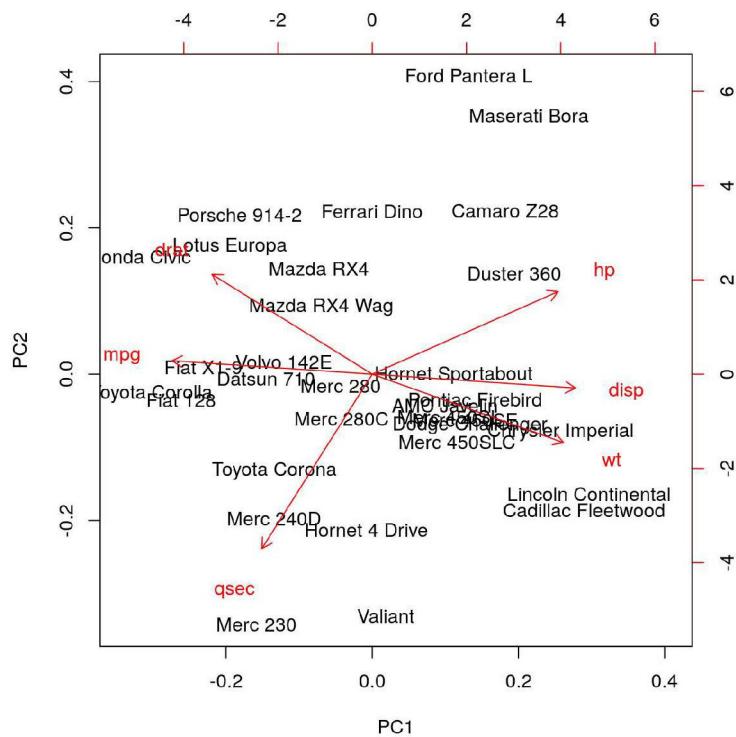


5. Narysuj wykres osypiska i zaproponuj optymalną liczbę składowych głównych w oparciu o trzy kryteria.

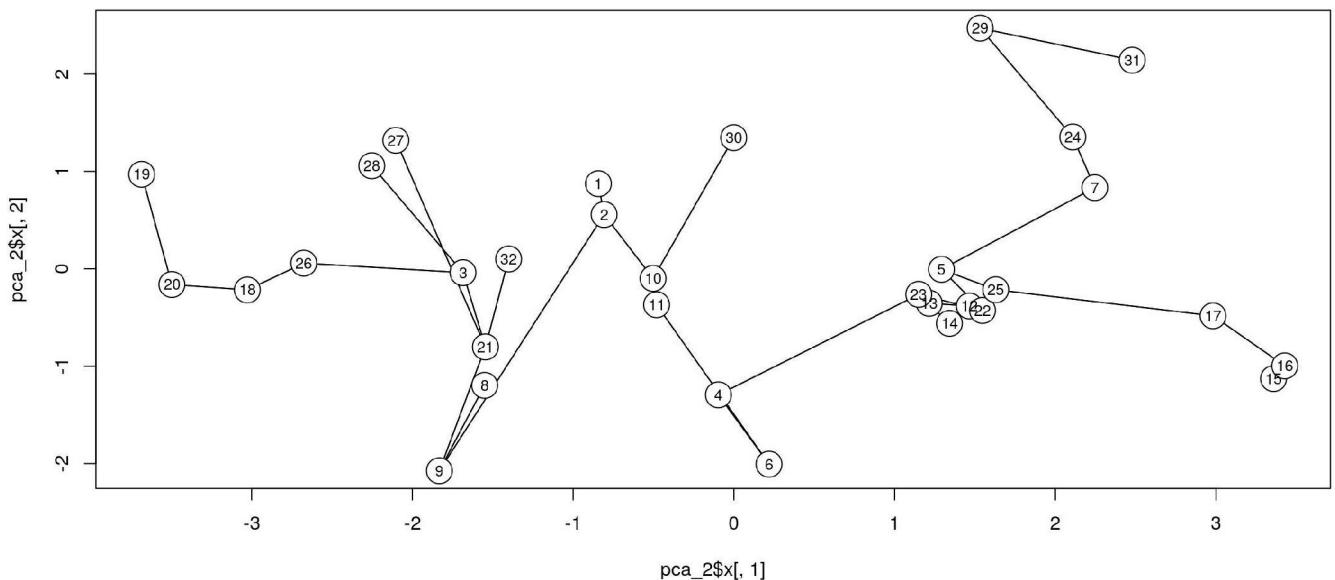


```
## 2 lub 3
```

6. Przedstaw samochody w układzie dwóch pierwszych składowych głównych (dokładniej narysuj biplot i dokonaj jego interpretacji).



7. Przedstaw samochody za pomocą minimalnego drzewa rozpinającego.



8. Jak bardzo będą różniły się wyniki, jeśli nie wykonamy skalowania danych?

Ad. 1

```
## Standard deviations (1, .., p=6):
## [1] 310.0207637 40.8471739 15.7168252 2.1068823 0.3894500 0.2969505
##
## Rotation (n x k) = (6 x 6):
##          PC1        PC2        PC3        PC4        PC5
## mpg   -0.05193468  0.121255352 -0.82446804  0.540735371 -0.064362234
## disp  -0.85253108 -0.522102198 -0.00915689  0.022137483  0.001587345
## hp    -0.51734213  0.841835388  0.15361995 -0.004990023 -0.006795464
## drat -0.01010286  0.021298587 -0.10869056 -0.033506518  0.982931599
## wt    -0.01067910  0.001369032 -0.04162846 -0.192177061  0.129755288
## qsec -0.05132793  0.059700171 -0.53199901 -0.817945952 -0.113215907
##          PC6
## mpg    0.0794678281
## disp  -0.0048593900
## hp    -0.0003699391
## drat -0.1426655136
## wt     0.9717935462
## qsec -0.1700734209
```

Ad. 2.

```

## Importance of components:

##                      PC1        PC2        PC3        PC4        PC5        PC6
## Standard deviation   310.0208  40.84717  15.71683  2.10688  0.3895  0.297
## Proportion of Variance  0.9804  0.01702  0.00252  0.00005  0.0000  0.000
## Cumulative Proportion  0.9804  0.99743  0.99995  1.00000  1.0000  1.000

```

Ad. 3.

```

##                      PC1        PC2        PC3        PC4        PC5
## Mazda RX4          -195.3155  12.68122 -11.170400  0.2509678  0.46472555
## Mazda RX4 Wag      -195.3469  12.71500 -11.478935 -0.2560871  0.43441224
## Datsun 710         -142.3892  25.86447 -15.915699 -1.5412826  0.05036709
## Hornet 4 Drive     -279.0353  -38.27504 -13.918563  0.1123864 -0.47164240
## Hornet Sportabout  -399.3594  -37.28023 -1.370742  2.5199166 -0.20567766
## Valiant            -248.1831  -25.61490 -12.054118 -3.0519863 -0.64870858
##                      PC6
## Mazda RX4           0.04092377
## Mazda RX4 Wag       0.19349001
## Datsun 710          -0.20711953
## Hornet 4 Drive      -0.21512523
## Hornet Sportabout   -0.32914754
## Valiant             -0.16407438

## ...

```

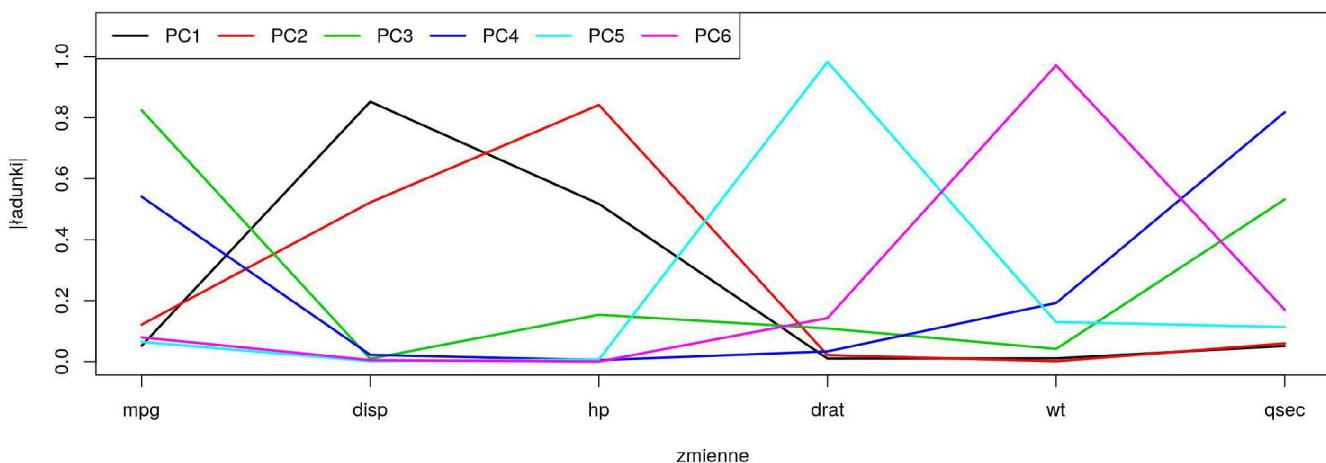
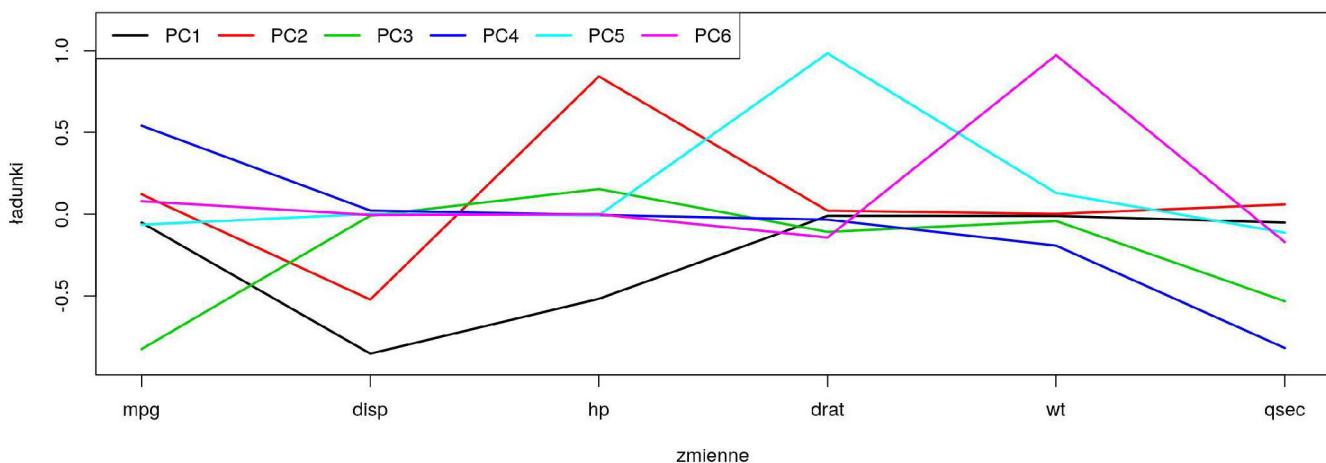
Ad. 4.

```

##          PC1        PC2        PC3        PC4        PC5
## mpg -0.05193468 0.121255352 -0.82446804 0.540735371 -0.064362234
## disp -0.85253108 -0.522102198 -0.00915689 0.022137483 0.001587345
## hp   -0.51734213 0.841835388 0.15361995 -0.004990023 -0.006795464
## drat -0.01010286 0.021298587 -0.10869056 -0.033506518 0.982931599
## wt   -0.01067910 0.001369032 -0.04162846 -0.192177061 0.129755288
## qsec -0.05132793 0.059700171 -0.53199901 -0.817945952 -0.113215907

##          PC6
## mpg  0.0794678281
## disp -0.0048593900
## hp   -0.0003699391
## drat -0.1426655136
## wt   0.9717935462
## qsec -0.1700734209

```

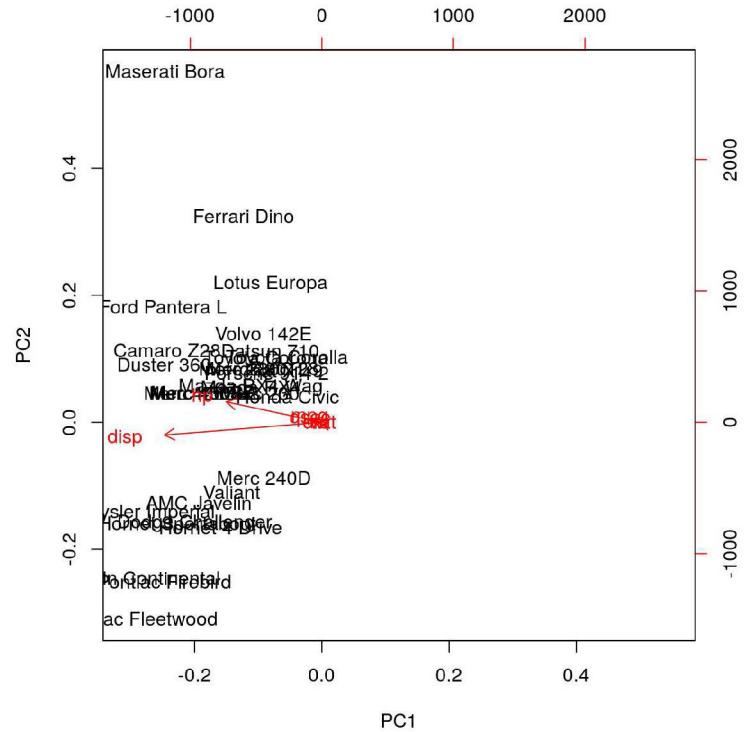


Ad. 5.

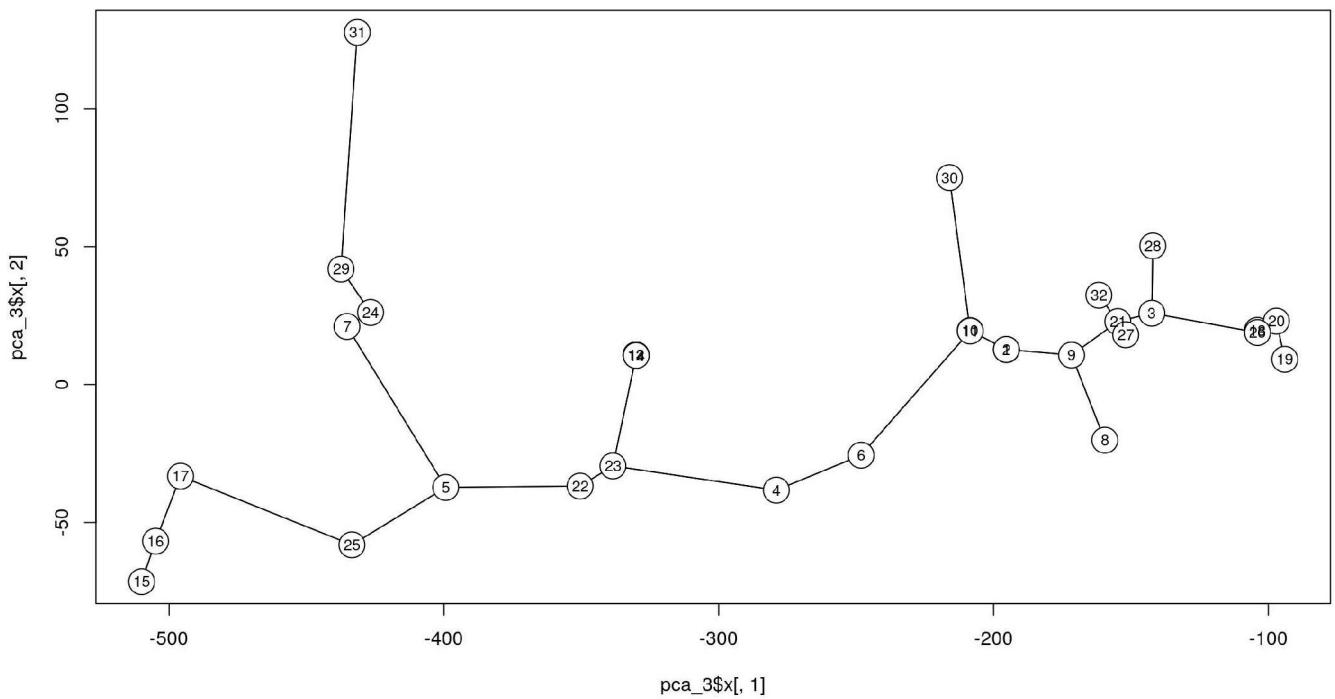


1

Ad. 6.



Ad. 7.



14 Analiza skupień

14.1 Przykład

Przykład. Zbiór danych `USArrests` zawiera informacje dotyczące liczby morderstw, napadów, gwałtów przypadających na 100,000 osób w poszczególnych stanach USA w roku 1973 oraz procent ludności mieszkającej w miastach. Chcielibyśmy się dowiedzieć, czy i które stany są do siebie w pewien sposób zbliżone.

```
head(USArrests)
```

```
##           Murder Assault UrbanPop Rape
## Alabama     13.2     236      58 21.2
## Alaska      10.0     263      48 44.5
## Arizona     8.1      294      80 31.0
## Arkansas    8.8      190      50 19.5
## California   9.0      276      91 40.6
## Colorado     7.9      204      78 38.7
```

```
dim(USArrests)
```

```
## [1] 50  4
```

1. metoda hierarchiczna

```
(skupienia_1 <- hclust(dist(USArrests)))
```

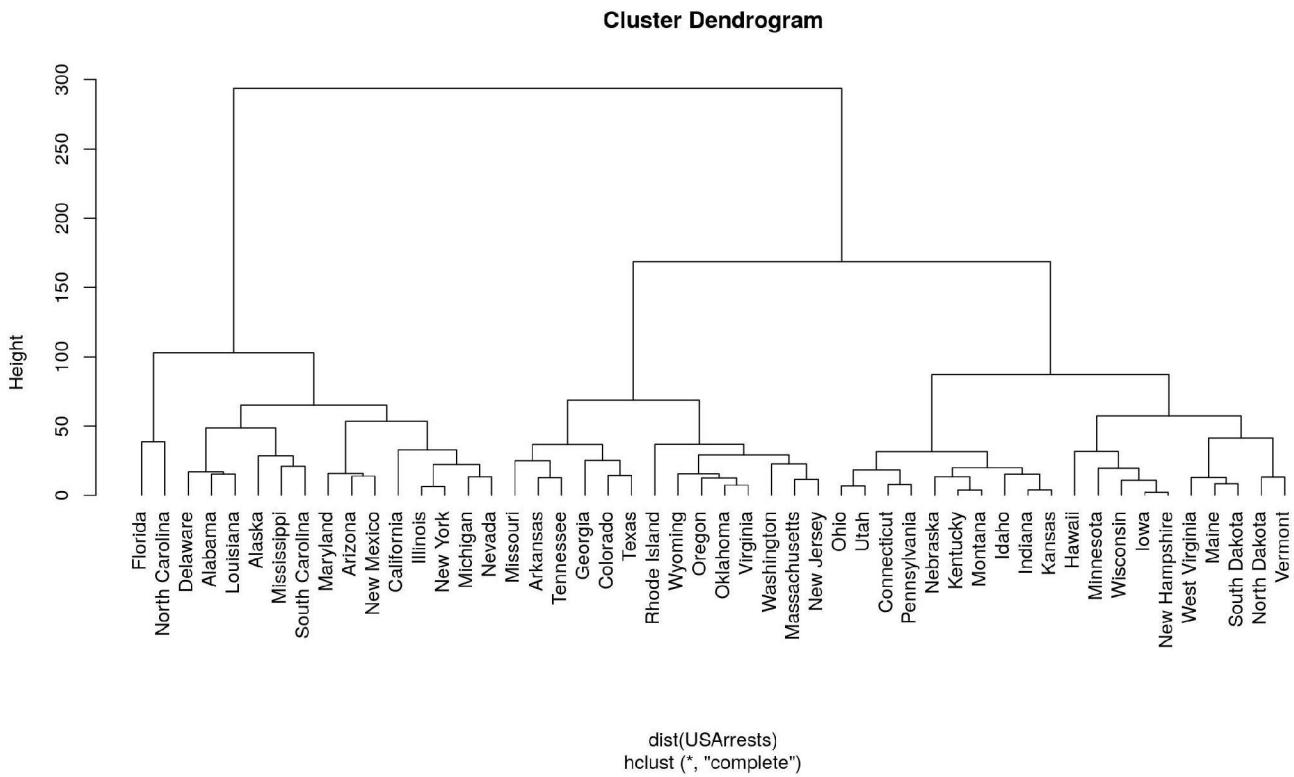
```

## 
## Call:
## hclust(d = dist(USArrests))
##
## Cluster method : complete
## Distance       : euclidean
## Number of objects: 50

```

- dendrogram

```
plot(skupienia_1, hang = -1)
```

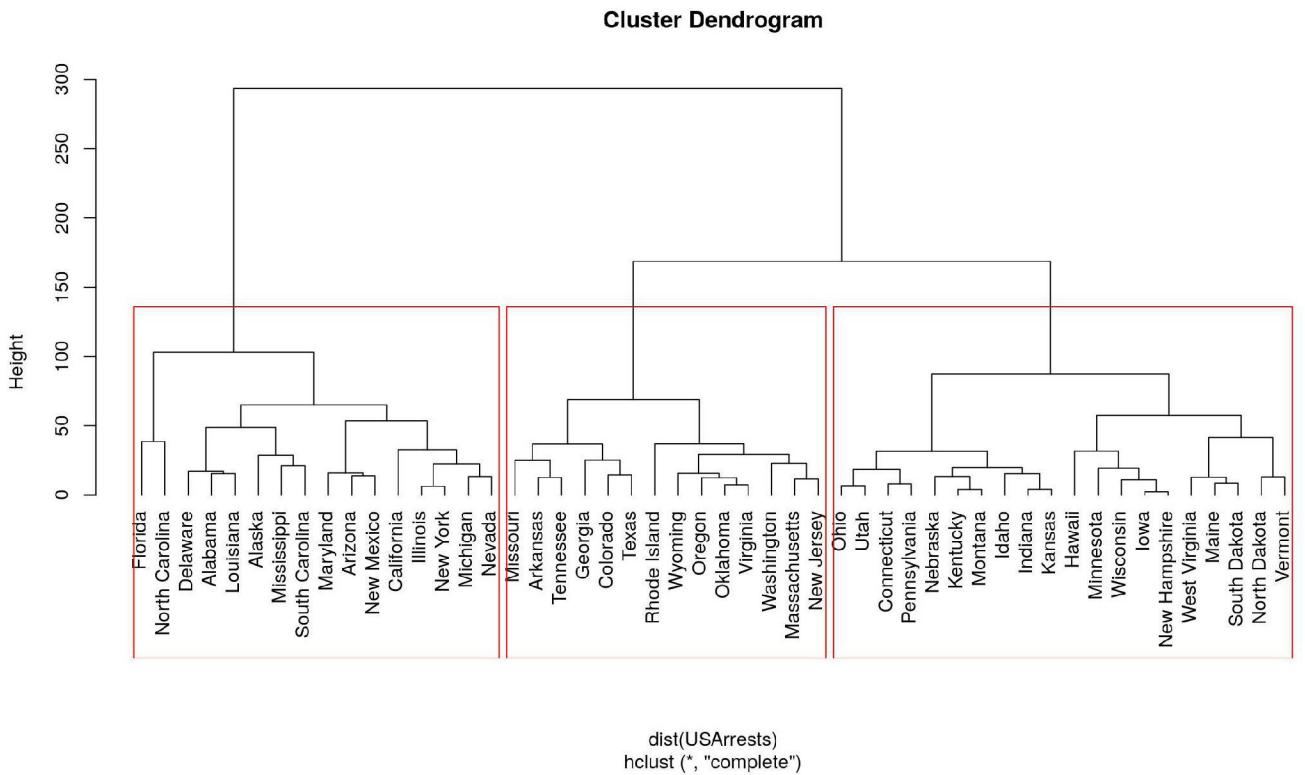


- automatyczny podział na skupienia i nanoszenie ich na dendrogram

```

plot(skupienia_1, hang = -1)
(podzial_1 <- rect.hclust(skupienia_1, k = 3))

```



```

## [[1]]

##      Alabama          Alaska        Arizona       California       Delaware
##      1                 2              3                  5                  8
##      Florida         Illinois    Louisiana     Maryland   Michigan
##      9                 13             18                  20                  22
##      Mississippi    Nevada  New Mexico  New York North Carolina
##      24                 28             31                  32                  33
## South Carolina
##      40
##
## [[2]]

##      Arkansas        Colorado      Georgia Massachusetts      Missouri
##      4                  6             10                  21                  25
##      New Jersey    Oklahoma    Oregon Rhode Island Tennessee
##      30                 36             37                  39                  42
##      Texas          Virginia    Washington      Wyoming
##      43                 46             47                  50
##
## [[3]]

##      Connecticut      Hawaii        Idaho       Indiana       Iowa
##      7                  11             12                  14                  15
##      Kansas          Kentucky      Maine Minnesota Montana
##      16                 17             19                  23                  26
##      Nebraska New Hampshire North Dakota      Ohio Pennsylvania
##      27                 29             34                  35                  38
##      South Dakota      Utah        Vermont West Virginia Wisconsin
##      41                 44             45                  48                  49

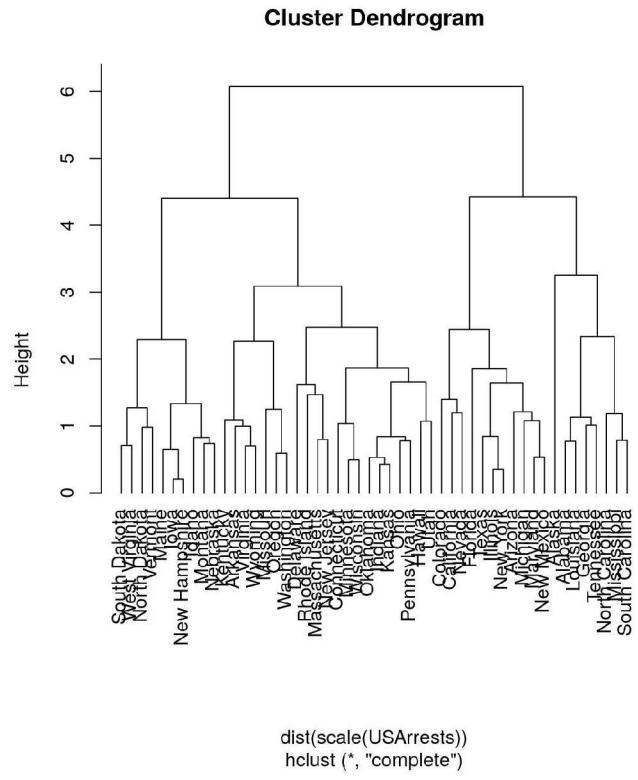
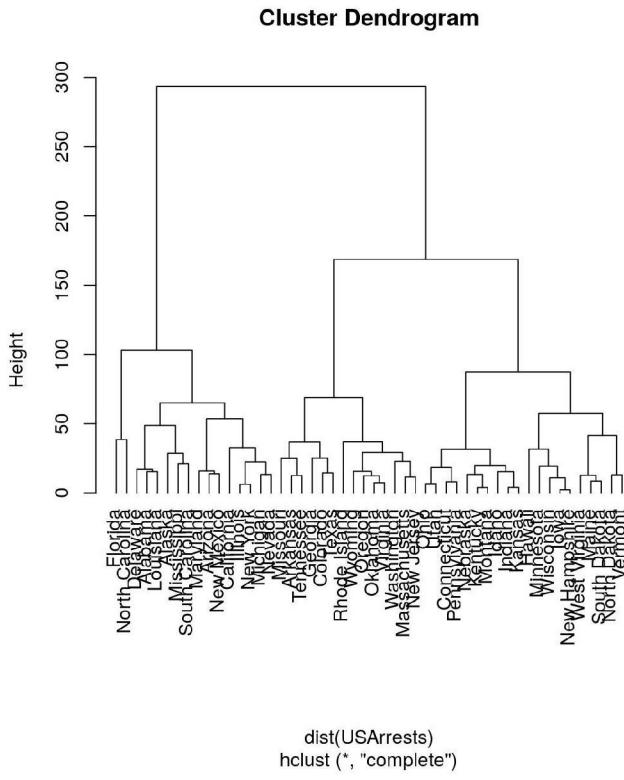
(podzial_2 <- cutree(skupienia_1, k = 3))

```

##	Alabama	Alaska	Arizona	Arkansas	California
##	1	1	1	2	1
##	Colorado	Connecticut	Delaware	Florida	Georgia
##	2	3	1	1	2
##	Hawaii	Idaho	Illinois	Indiana	Iowa
##	3	3	1	3	3
##	Kansas	Kentucky	Louisiana	Maine	Maryland
##	3	3	1	3	1
##	Massachusetts	Michigan	Minnesota	Mississippi	Missouri
##	2	1	3	1	2
##	Montana	Nebraska	Nevada	New Hampshire	New Jersey
##	3	3	1	3	2
##	New Mexico	New York	North Carolina	North Dakota	Ohio
##	1	1	1	3	3
##	Oklahoma	Oregon	Pennsylvania	Rhode Island	South Carolina
##	2	2	3	2	1
##	South Dakota	Tennessee	Texas	Utah	Vermont
##	3	2	2	3	3
##	Virginia	Washington	West Virginia	Wisconsin	Wyoming
##	2	2	3	3	2

- zmiana skali ma wpływ na analizę skupień

```
par(mfrow = c(1, 2))
plot(hclust(dist(USArrests)), hang = -1)
plot(hclust(dist(scale(USArrests))), hang = -1)
```



```
par(mfrow = c(1, 1))
```

- parametry metody hierarchicznej

```
# inna miara niepodobieństwa
(skupienia_2 <- hclust(dist(USArrests, method = 'manhattan')))
```

```
##
## Call:
## hclust(d = dist(USArrests, method = "manhattan"))
##
## Cluster method : complete
## Distance       : manhattan
## Number of objects: 50
```

```
# inna miara niepodobieństwa i inna metoda wiązania skupień
(skupienia_3 <- hclust(dist(USArrests, method = 'manhattan'), 'ward.D'))
```

```

## Call:
## hclust(d = dist(USArrests, method = "manhattan"), method = "ward.D")
##
## Cluster method : ward.D
## Distance       : manhattan
## Number of objects: 50

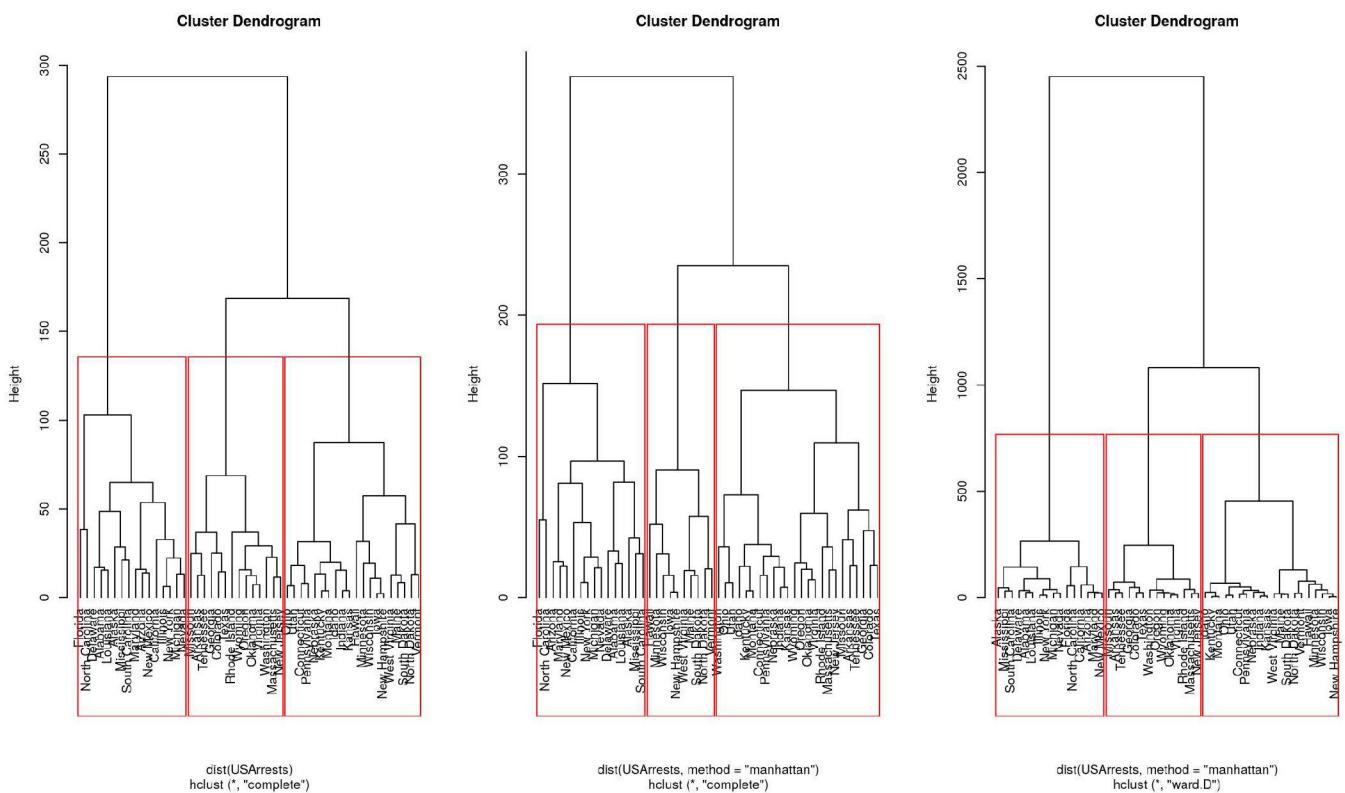
```

porównanie dendrogramów

```

par(mfrow = c(1, 3))
plot(skupienia_1, hang = -1)
rect.hclust(skupienia_1, k = 3)
plot(skupienia_2, hang = -1)
rect.hclust(skupienia_2, k = 3)
plot(skupienia_3, hang = -1)
rect.hclust(skupienia_3, k = 3)

```



```
par(mfrow = c(1, 1))
```

2. metoda K -średnich

```
set.seed(1234)
(skupienia_4 <- kmeans(USAarrests, centers = 3, nstart = 1000))
```

```

## K-means clustering with 3 clusters of sizes 14, 20, 16
##
## Cluster means:
##      Murder Assault UrbanPop      Rape
## 1  8.214286 173.2857 70.64286 22.84286
## 2  4.270000  87.5500 59.75000 14.39000
## 3 11.812500 272.5625 68.31250 28.37500
##
## Clustering vector:
##      Alabama      Alaska      Arizona      Arkansas      California
##            3          3          3          1          3
##      Colorado Connecticut Delaware Florida Georgia
##            1          2          3          3          1
##      Hawaii Idaho Illinois Indiana Iowa
##            2          2          3          2          2
##      Kansas Kentucky Louisiana Maine Maryland
##            2          2          3          2          3
##      Massachusetts Michigan Minnesota Mississippi Missouri
##            1          3          2          3          1
##      Montana Nebraska Nevada New Hampshire New Jersey
##            2          2          3          2          1
##      New Mexico New York North Carolina North Dakota Ohio
##            3          3          3          2          2
##      Oklahoma Oregon Pennsylvania Rhode Island South Carolina
##            1          1          2          1          3
##      South Dakota Tennessee Texas Utah Vermont
##            2          1          1          2          2
##      Virginia Washington West Virginia Wisconsin Wyoming
##            1          1          2          2          1
##
## Within cluster sum of squares by cluster:
## [1] 9136.643 19263.760 19563.863
## (between_SS / total_SS =  86.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"       "totss"        "withinss"

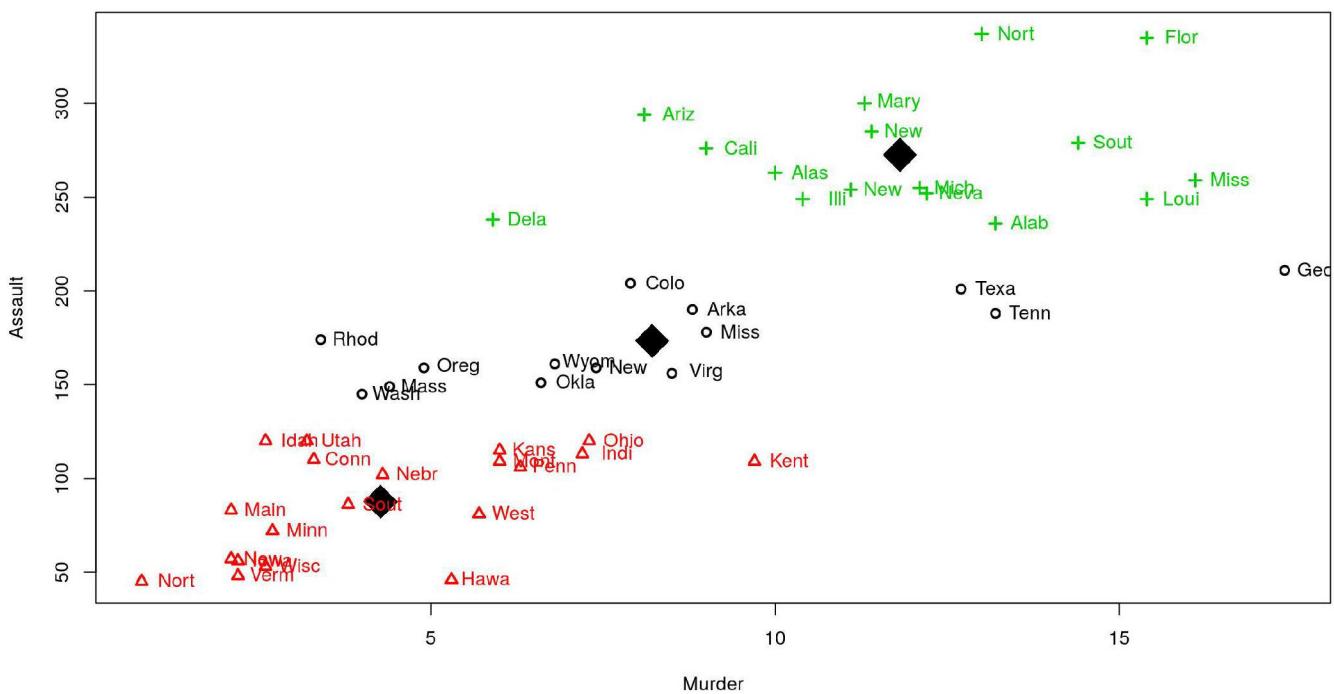
```

```

## [5] "tot.withinss" "betweenss"      "size"           "iter"
## [9] "ifault"

# wykres danych w układzie Murder-Assault z podziałem na
# otrzymane skupienia i centrami skupień
plot(USArrests[, 1:2], pch = skupienia_4$cluster,
     col = skupienia_4$cluster, lwd = 2)
points(skupienia_4$centers, pch = 18, cex = 4)
text(USArrests[, 1:2] + 0.5, substring(row.names(USArrests), 1, 4),
     col = skupienia_4$cluster)

```



- metoda K -średnich z wyborem optymalnej liczby skupień poprzez indeks Calińskiego-Harabasza

```

library(vegan)
set.seed(1234)
(model <- cascadeKM(USArrests, 2, 5))

```

```
## $partition  
##          2 groups 3 groups 4 groups 5 groups  
## Alabama      1        1        2        1  
## Alaska       1        1        2        1  
## Arizona      1        1        2        3  
## Arkansas     1        3        3        4  
## California   1        1        2        1  
## Colorado     1        3        3        4  
## Connecticut  2        2        4        5  
## Delaware    1        1        2        1  
## Florida      1        1        2        3  
## Georgia      1        3        3        4  
## Hawaii       2        2        1        2  
## Idaho        2        2        4        5  
## Illinois     1        1        2        1  
## Indiana      2        2        4        5  
## Iowa         2        2        1        2  
## Kansas       2        2        4        5  
## Kentucky     2        2        4        5  
## Louisiana    1        1        2        1  
## Maine        2        2        1        2  
## Maryland     1        1        2        3  
## Massachusetts 2        3        3        4  
## Michigan     1        1        2        1  
## Minnesota   2        2        1        2  
## Mississippi  1        1        2        1  
## Missouri     2        3        3        4  
## Montana      2        2        4        5  
## Nebraska     2        2        4        5  
## Nevada       1        1        2        1  
## New Hampshire 2        2        1        2  
## New Jersey   2        3        3        4  
## New Mexico   1        1        2        1  
## New York     1        1        2        1  
## North Carolina 1        1        2        3  
## North Dakota 2        2        1        2  
## Ohio         2        2        4        5  
## Oklahoma     2        3        3        4
```

```

## Oregon          2     3     3     4
## Pennsylvania    2     2     4     5
## Rhode Island    2     3     3     4
## South Carolina  1     1     2     1
## South Dakota   2     2     1     2
## Tennessee       1     3     3     4
## Texas           1     3     3     4
## Utah            2     2     4     5
## Vermont         2     2     1     2
## Virginia        2     3     3     4
## Washington      2     3     3     4
## West Virginia   2     2     1     2
## Wisconsin       2     2     1     2
## Wyoming         2     3     3     4

##
## $results

##             2 groups 3 groups 4 groups 5 groups
## SSE      96399.0281 47964.2654 34728.6294 24417.0235
## calinski 129.1675 150.8274 141.7624 152.6864
##
## $criterion
## [1] "calinski"
##
## $size

##             2 groups 3 groups 4 groups 5 groups
## Group 1     21       16       10       12
## Group 2     29       20       16       10
## Group 3     NA       14       14       4
## Group 4     NA       NA       10       14
## Group 5     NA       NA       NA       10

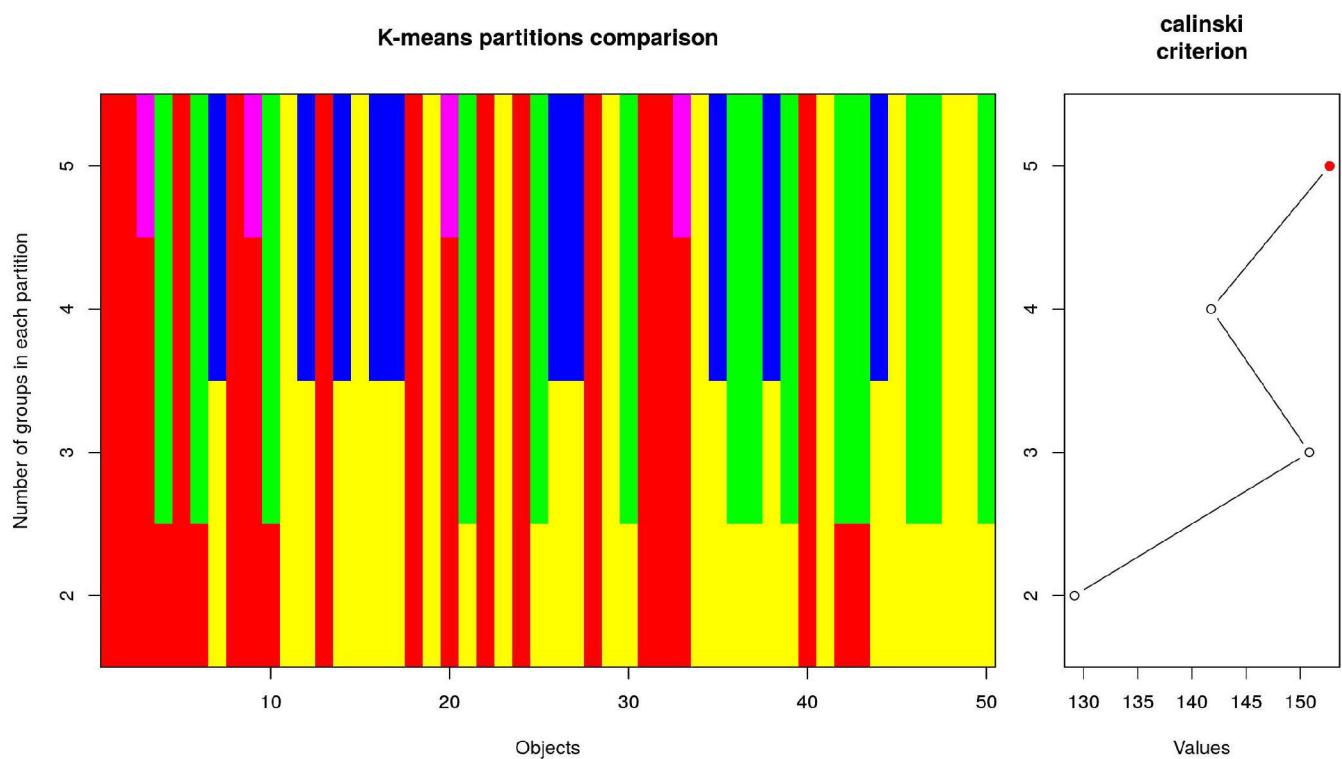
##
## attr(,"class")
## [1] "cascadeKM"

```

```

# wykres podziału na grupy
# (na osi x obserwacje, na osi y liczba skupień, kolory oznaczają skupienia)
# oraz wykres wartości indeksu Calińskiego-Harabasza dla
# poszczególnych Liczb skupień (czerwona kropka oznacza
# optymalną liczbę skupień według tego kryterium)
plot(model)

```



14.2 Zadania

Zadanie 1. Plik `wojewodztwa.txt` zawiera dane dotyczące następujących cech województw w Polsce: współczynnik aktywności zawodowej (w %), wskaźnik zatrudnienia (w %), stopa bezrobocia rejestrowanego (w %), śmiertelność niemowląt (na 1000 urodzeń żywych), oczekiwana dalsza długość życia w momencie narodzin, gęstość zaludnienia (osoby na 1 km kwadratowy), produkt krajowy brutto na mieszkańca. Celem badania jest wyznaczenie podobieństw w województwach Polski.

```

##      wojewodztwo wspaktzaw wskzatr bezrobrej smniemowl lifeexp
## 1    Dolnoslaskie    54.3    41.5    20.6    6.9    74.6
## 2  KujawskoPomorskie    56.2    45.1    22.3    6.6    74.8
## 3    Lubelskie    56.1    48.7    17.0    7.3    74.9
## 4    Lubuskie    53.2    42.2    23.0    6.2    74.6
## 5    Lodzkie    55.9    45.7    17.9    6.1    73.5
## 6   Malopolskie    54.8    46.1    13.8    5.8    76.2
## gestzaludn pkbcap
## 1    144.8  26620
## 2    115.1  22474
## 3    86.8   17591
## 4    72.1   23241
## 5    141.5  23666
## 6    215.0  21989

## ...
## [1] 16  8

```

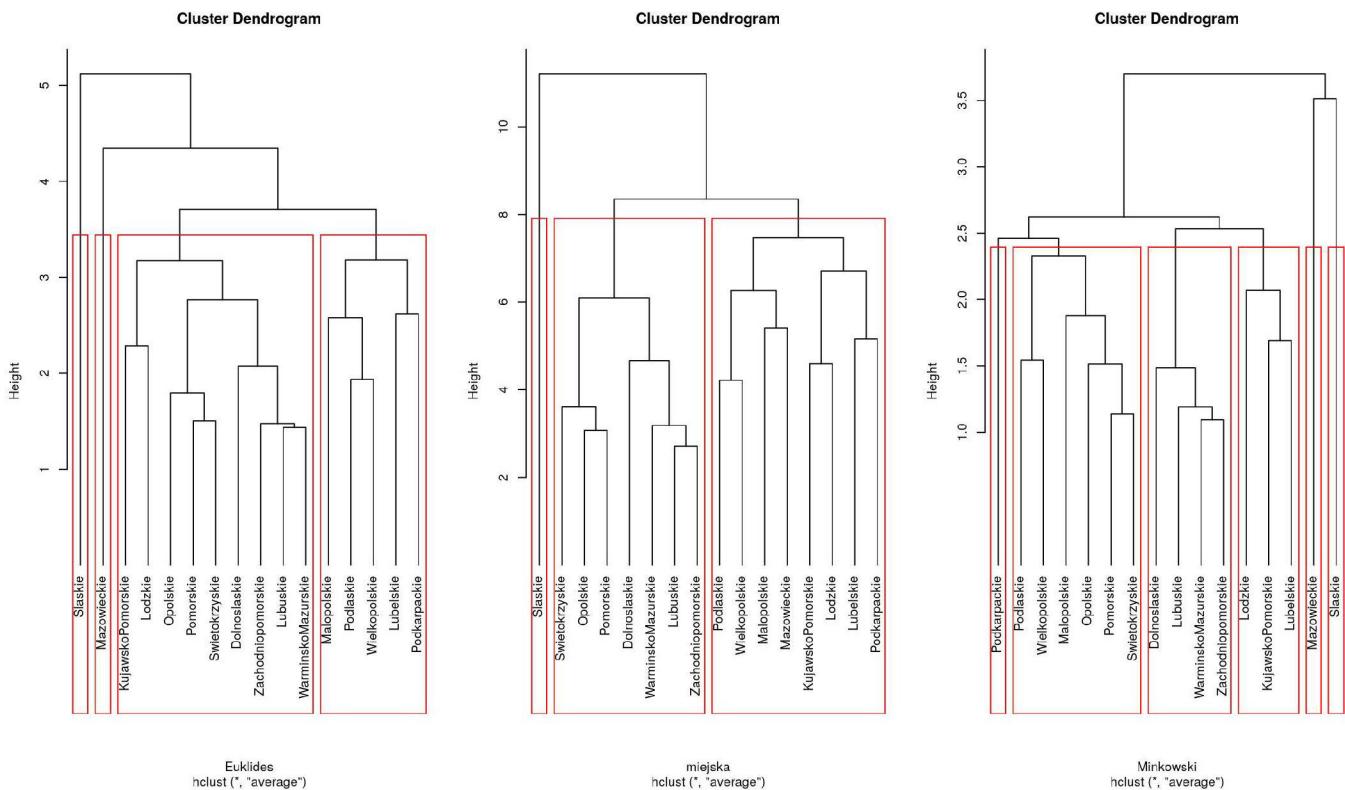
- Zauważmy, że jedna ze zmiennych przyjmuje znacznie większe wartości niż pozostałe zmienne. Czy w takim przypadku powinniśmy dokonać standaryzacji wszystkich zmiennych?

```

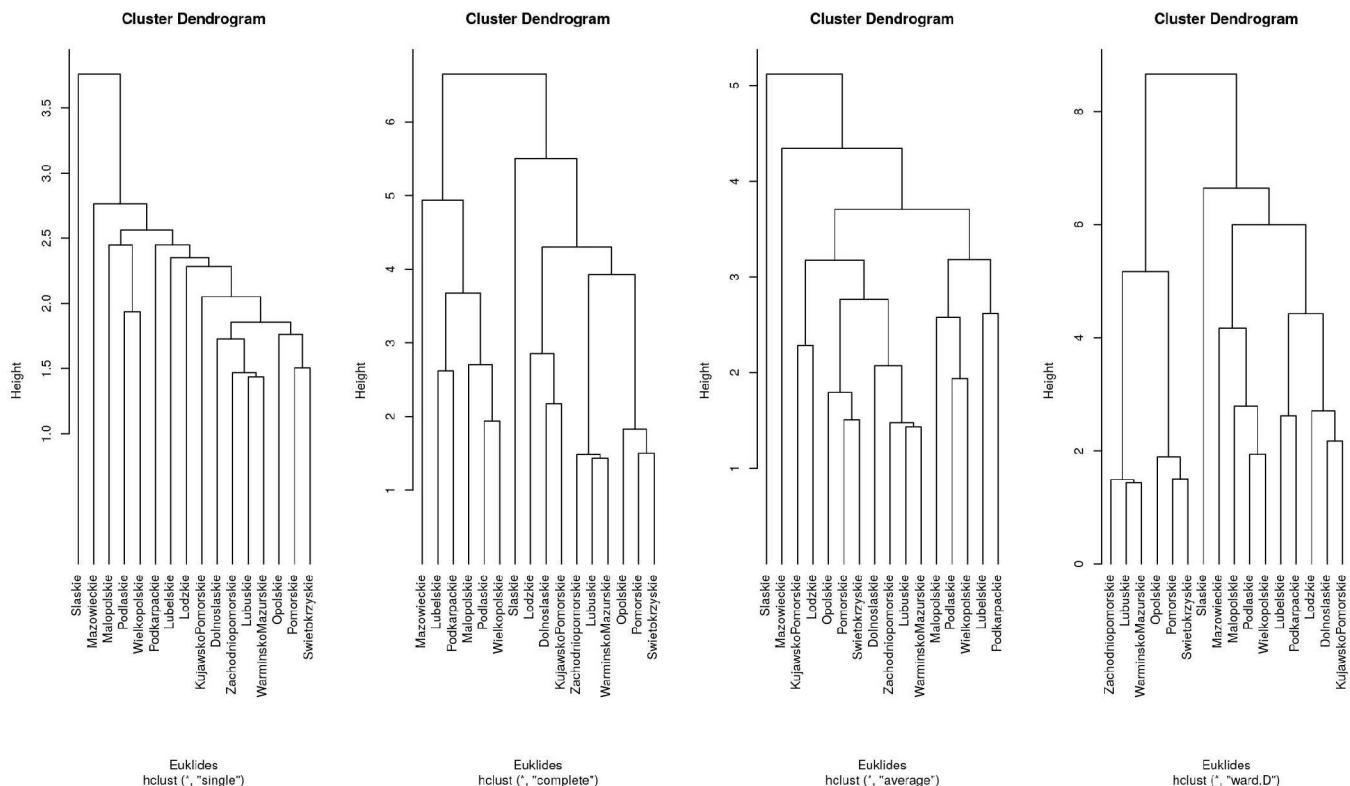
##      wojewodztwo wspaktzaw wskzatr bezrobrej smniemowl
## 1    Dolnoslaskie -0.08925698 -1.1278399 0.3969181 0.7935138
## 2  KujawskoPomorskie 1.20284403  0.2602707 0.8170027 0.3703064
## 3     Lubelskie 1.13483871  1.6483813 -0.4926727 1.3577903
## 4     Lubuskie -0.83731545 -0.8579295 0.9899787 -0.1939700
## 5     Lodzkie 0.99882808  0.4916225 -0.2702750 -0.3350392
## 6   Malopolskie 0.25076960  0.6458570 -1.2834201 -0.7582465
##      lifeexp gestzaludn pkbcap
## 1 -0.7398300  0.2031423 0.530348302
## 2 -0.4613058 -0.1799261 -0.206399582
## 3 -0.3220437 -0.5449373 -1.074113021
## 4 -0.7398300 -0.7345368 -0.070103001
## 5 -2.2717134  0.1605792  0.005419877
## 6  1.4883640  1.1085766 -0.292584513
## ...

```

2. Wykorzystując odległość euklidesową, miejską i Minkowskiego z potęgą cztery jako miary niepodobieństwa oraz metodę średniego wiązania skupień wykonaj hierarchiczną analizę skupień. Narysuj dendrogramy. Przy ich pomocy określ jaka liczba skupień wydaje się najbardziej sensowna. Zaznacz te skupienia na wykresie.

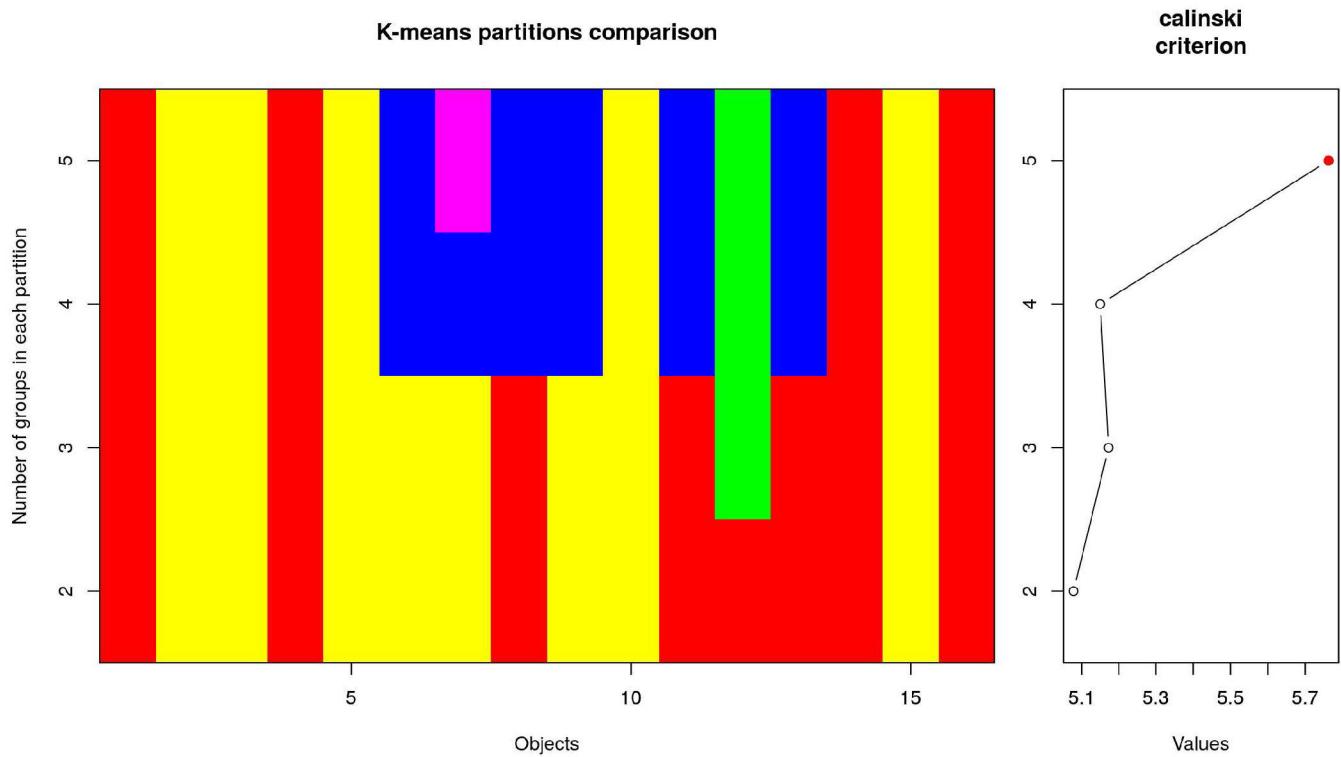


3. Wykorzystując odległość euklidesową jako miarę niepodobieństwa oraz metody pojedynczego, kompletnego, średniego wiązania skupień oraz metodę Warda łączenia skupień wykonaj hierarchiczną analizę skupień. Narysuj dendrogramy.



4. Jaką optymalną liczbę skupień proponuje indeks Calińskiego-Harabasza? Rozważ $K = 2, 3, 4, 5$.

```
##          2 groups   3 groups   4 groups   5 groups
## SSE      77.049773 58.469889 45.900765 33.919492
## calinski 5.078577 5.172675 5.150174 5.762804
```



5. Wykonaj analizę skupień korzystając z metody K -średnich oraz hierarchicznej analizy skupień (odległość Minkowskiego z potągą cztery, metoda średniego wiązania skupień) dla liczby skupień wyznaczonej przez indeks Calińskiego-Harabasza. Przedstaw obserwacje w układzie dwóch pierwszych składowych głównych z podziałem na otrzymane skupienia.

```
## K-średnich
```

```
## [1] KujawskoPomorskie Lubelskie      Lodzkie        Podlaskie
## [5] Wielkopolskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] Mazowieckie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] Malopolskie    Opolskie       Podkarpackie   Pomorskie
## [5] Swietokrzyskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie
```

```

## [1] Slaskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] Dolnoslaskie      Lubuskie      WarminskoMazurskie
## [4] Zachodniopomorskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## metoda hierarchiczna

## [1] Dolnoslaskie      Lubuskie      WarminskoMazurskie
## [4] Zachodniopomorskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] KujawskoPomorskie Lubelskie      Lodzkie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

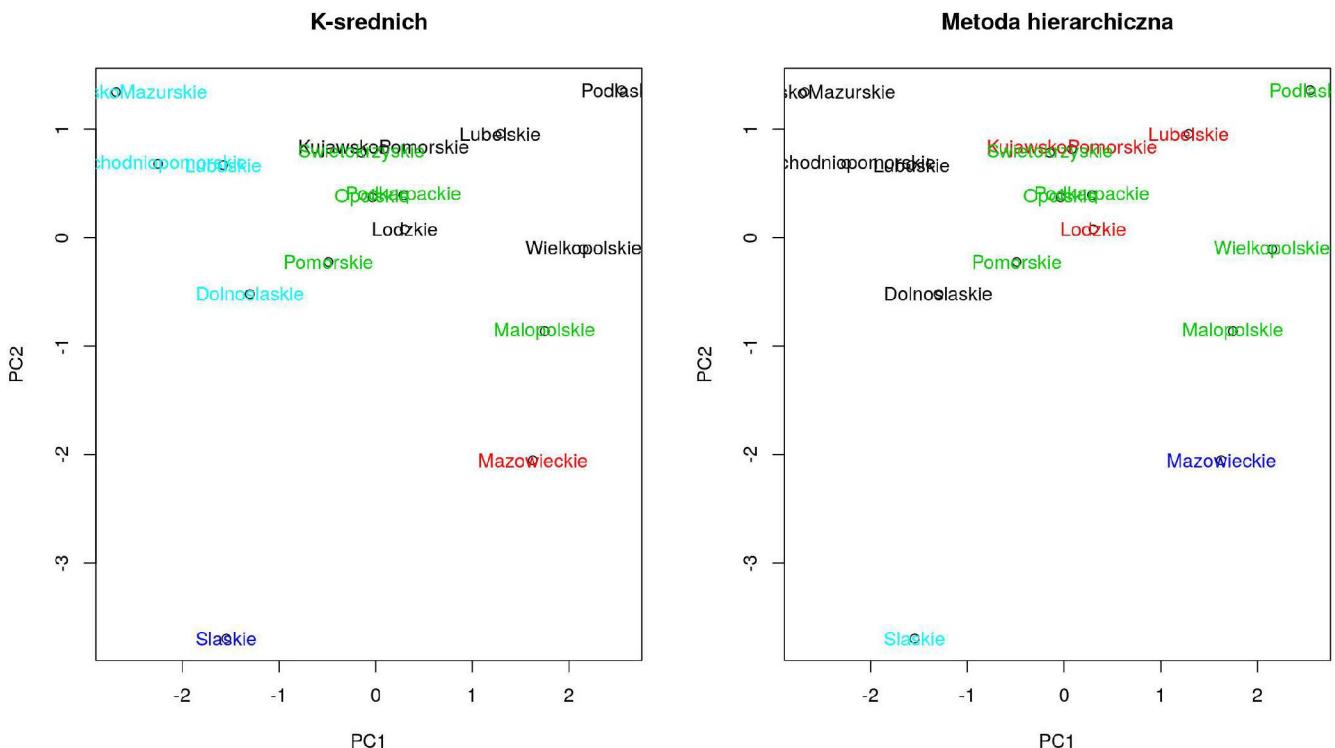
## [1] Malopolskie      Opolskie      Podkarpackie  Podlaskie
## [5] Pomorskie        Swietokrzyskie Wielkopolskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] Mazowieckie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## [1] Slaskie
## 16 Levels: Dolnoslaskie KujawskoPomorskie Lodzkie Lubelskie ... Zachodniopomorskie

## Importance of components:
##                  PC1    PC2    PC3    PC4    PC5    PC6    PC7
## Standard deviation 1.5860 1.3168 1.1021 0.9503 0.70586 0.31884 0.18202
## Proportion of Variance 0.3593 0.2477 0.1735 0.1290 0.07118 0.01452 0.00473
## Cumulative Proportion 0.3593 0.6070 0.7805 0.9096 0.98074 0.99527 1.00000

```



Zadanie 2. W pliku [wina.txt](#) zawarto informację o trzynastu cechach różnych gatunków win. Co więcej obserwacje podzielone są na trzy grupy.

```
##      V1     V2     V3     V4     V5     V6     V7     V8     V9     V10    V11    V12    V13    V14
## 1 14.23 1.71 2.43 15.6 127 2.80 3.06 0.28 2.29 5.64 1.04 3.92 1065    1
## 2 13.20 1.78 2.14 11.2 100 2.65 2.76 0.26 1.28 4.38 1.05 3.40 1050    1
## 3 13.16 2.36 2.67 18.6 101 2.80 3.24 0.30 2.81 5.68 1.03 3.17 1185    1
## 4 14.37 1.95 2.50 16.8 113 3.85 3.49 0.24 2.18 7.80 0.86 3.45 1480    1
## 5 13.24 2.59 2.87 21.0 118 2.80 2.69 0.39 1.82 4.32 1.04 2.93  735    1
## 6 14.20 1.76 2.45 15.2 112 3.27 3.39 0.34 1.97 6.75 1.05 2.85 1450    1
```

```
## ...
```

```
## [1] 178 14
```

```
##  
## 1 2 3  
## 59 71 48
```

1. Czy powinniśmy dokonać standaryzacji zmiennych?

```

##          V1          V2          V3          V4          V5          V6
## 1 1.5143408 -0.56066822  0.2313998 -1.1663032 1.90852151 0.8067217
## 2 0.2455968 -0.49800856 -0.8256672 -2.4838405 0.01809398 0.5670481
## 3 0.1963252  0.02117152  1.1062139 -0.2679823 0.08810981 0.8067217
## 4 1.6867914 -0.34583508  0.4865539 -0.8069748 0.92829983 2.4844372
## 5 0.2948684  0.22705328  1.8352256  0.4506745 1.27837900 0.8067217
## 6 1.4773871 -0.51591132  0.3043010 -1.2860793 0.85828399 1.5576991
##          V7          V8          V9          V10         V11         V12
## 1 1.0319081 -0.6577078  1.2214385  0.2510088 0.3611585 1.8427215
## 2 0.7315653 -0.8184106 -0.5431887 -0.2924962 0.4049085 1.1103172
## 3 1.2121137 -0.4970050  2.1299594  0.2682629 0.3174085 0.7863692
## 4 1.4623994 -0.9791134  1.0292513  1.1827317 -0.4263410 1.1807407
## 5 0.6614853  0.2261576  0.4002753 -0.3183774 0.3611585 0.4483365
## 6 1.3622851 -0.1755994  0.6623487  0.7298108 0.4049085 0.3356589
##          V13         V14
## 1 1.01015939 1
## 2 0.96252635 1
## 3 1.39122370 1
## 4 2.32800680 1
## 5 -0.03776747 1
## 6 2.23274072 1

## ...

```

2. Wykonaj hierarchiczną analizę skupień. Narysuje dendrogram z podziałem na skupienia w liczbie równej liczbie grup wyszczególnionych w danych. Jaki jest błąd otrzymanego podziału?

15 Klasyfikacja

15.1 Przykład

Przykład. Zbiór danych `iris` zawiera informacje na temat czterech cech trzech gatunków irysa.

```
head(iris)
```

```
##   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
```

```
dim(iris)
```

```
## [1] 150    5
```

```
table(iris$Species)
```

```
##
##      setosa versicolor  virginica
##          50          50          50
```

Na przykładzie tego zbioru danych przedstawimy liniową analizę dyskryminacyjną (LDA).

- model liniowej analizy dyskryminacyjnej w R

```

library(MASS)

(model_lda <- lda(Species ~ ., data = iris))

## Call:
## lda(Species ~ ., data = iris)
##
## Prior probabilities of groups:
##   setosa versicolor virginica
## 0.3333333 0.3333333 0.3333333
##
## Group means:
##           Sepal.Length Sepal.Width Petal.Length Petal.Width
## setosa          5.006      3.428      1.462      0.246
## versicolor      5.936      2.770      4.260      1.326
## virginica       6.588      2.974      5.552      2.026
##
## Coefficients of linear discriminants:
##           LD1        LD2
## Sepal.Length 0.8293776 0.02410215
## Sepal.Width  1.5344731 2.16452123
## Petal.Length -2.2012117 -0.93192121
## Petal.Width   -2.8104603  2.83918785
##
## Proportion of trace:
##    LD1     LD2
## 0.9912 0.0088

```

Lub

```

# model_lda <- lda(iris[, 1:4], grouping = iris$Species)

```

- tablica kontyngencji

```
head(predict(model_lda)$posterior)
```

```
##   setosa  versicolor  virginica
## 1      1 3.896358e-22 2.611168e-42
## 2      1 7.217970e-18 5.042143e-37
## 3      1 1.463849e-19 4.675932e-39
## 4      1 1.268536e-16 3.566610e-35
## 5      1 1.637387e-22 1.082605e-42
## 6      1 3.883282e-21 4.566540e-40
```

```
head(predict(model_lda)$class)
```

```
## [1] setosa setosa setosa setosa setosa setosa
## Levels: setosa versicolor virginica
```

```
(conf_matrix <- table(predict(model_lda)$class, iris$Species))
```

```
##
##          setosa versicolor virginica
## setosa     50         0         0
## versicolor    0        48         1
## virginica     0         2        49
```

- błąd klasyfikacji metodą ponownego podstawiania

```
(1 - sum(diag(conf_matrix)) / nrow(iris))
```

```
## [1] 0.02
```

- błąd klasyfikacji metodą sprawdzania krzyżowego z $v = 1$ (1-CV, LOO, ang. *leave one out*)

```

pred_loo <- numeric(nrow(iris))
for (i in 1:nrow(iris)) {
  model_lda_i <- lda(Species ~ ., data = iris[-i, ])
  pred_loo[i] <- predict(model_lda_i, iris[i, ])$class
}
table(iris$Species, pred_loo)

##          pred_loo
##            1 2 3
## setosa    50 0 0
## versicolor 0 48 2
## virginica   0 1 49

(1 - sum(diag(table(iris$Species, pred_loo))) / nrow(iris))

## [1] 0.02

```

- predykcja

```

new_data <- data.frame(Sepal.Length = 5.1,
                       Sepal.Width = 3.5,
                       Petal.Length = 1.3,
                       Petal.Width = 0.3)
predict(model_lda, new_data)

```

```

## $class
## [1] setosa
## Levels: setosa versicolor virginica
##
## $posterior
##   setosa  versicolor  virginica
## 1 1 4.850575e-22 6.605032e-42
##
## $x
##      LD1      LD2
## 1 8.000875 0.6775315

```

15.2 Zadania

Zadanie 1. Kontynuujemy przykład dotyczący zbioru danych `iris`.

1. Wyznacz błąd klasyfikacji liniowej analizy dyskryminacyjnej metodą sprawdzania krzyżowego z $v = 10$ (10-CV).

```
## [1] 0.02
```

2. Błąd klasyfikacji można oszacować również następującą metodą bootstrapową.

- Przyjmijmy, że zbiór danych ma n obserwacji.
- Krok 1. Losujemy ze zwracaniem n obserwacji ze zbioru danych tworzących próbę bootstrapową.
- Krok 2. Konstruujemy klasyfikator na bazie próby bootstrapowej.
- Krok 3. Liczymy błąd klasyfikatora wyznaczonego w kroku 2 dla obserwacji, które nie znalazły się w próbie bootstrapowej.
- Krok 4. Powtarzamy kroki 1-3 n_{boot} razy, otrzymując błędy $b_1, \dots, b_{n_{boot}}$.
- Krok 5. Obliczamy błąd klasyfikacji metodą bootstrapową według wzoru

$$\frac{1}{n_{boot}} \sum_{i=1}^{n_{boot}} b_i.$$

Wyznacz błąd klasyfikacji liniowej analizy dyskryminacyjnej metodą bootstrapową. Przyjmij $n_{boot} = 100$.

```
## [1] 0.0259815
```

Zadanie 2. W pliku [wina.txt](#) zawarto informację o trzynastu cechach różnych gatunków win. Co więcej obserwacje podzielone są na trzy grupy.

```
##      V1      V2      V3      V4      V5      V6      V7      V8      V9      V10     V11     V12     V13     V14
## 1 14.23 1.71 2.43 15.6 127 2.80 3.06 0.28 2.29 5.64 1.04 3.92 1065 1
## 2 13.20 1.78 2.14 11.2 100 2.65 2.76 0.26 1.28 4.38 1.05 3.40 1050 1
## 3 13.16 2.36 2.67 18.6 101 2.80 3.24 0.30 2.81 5.68 1.03 3.17 1185 1
## 4 14.37 1.95 2.50 16.8 113 3.85 3.49 0.24 2.18 7.80 0.86 3.45 1480 1
## 5 13.24 2.59 2.87 21.0 118 2.80 2.69 0.39 1.82 4.32 1.04 2.93 735 1
## 6 14.20 1.76 2.45 15.2 112 3.27 3.39 0.34 1.97 6.75 1.05 2.85 1450 1
```

```
## ...
```

1. Jaki jest wymiar tych danych? Jakie są etykiety klas i ich liczebności?

```
## [1] 178 14
```

```
##
## 1 2 3
## 59 71 48
```

2. Wykonaj liniową analizę dyskryminacyjną bazując na trzech pierwszych zmiennych w tym zbiorze danych.

```
##      1          2          3
## 0.3314607 0.3988764 0.2696629
```

```
##      V1          V2          V3
## 1 13.74475 2.010678 2.455593
## 2 12.27873 1.932676 2.244789
## 3 13.15375 3.333750 2.437083
```

```
##          LD1        LD2
## V1 -1.8725417 -0.2943580
## V2 -0.0862327  1.0473192
## V3 -1.4493443  0.1419408
```

3. Wyznacz oceny prawdopodobieństw a posteriori i przewidywaną przynależność do klas obserwacji oraz tablicę kontyngencji otrzymanego klasyfikatora.

```
##          1        2        3
## 1 0.9705550 0.0006735689 0.02877140
## 2 0.3933512 0.3924750849 0.21417373
## 3 0.5316537 0.0682685490 0.40007778
## 4 0.9723331 0.0002235964 0.02744332
## 5 0.5798070 0.0197639349 0.40042907
## 6 0.9668517 0.0007345077 0.03241381
```

```
## [1] 1 1 1 1 1 1
## Levels: 1 2 3
```

```
##
##          1 2 3
## 1 51 5 7
## 2 4 62 8
## 3 4 4 33
```

4. Wyznacz błąd klasyfikacji metodą ponownego podstawiania.

```
## [1] 0.1797753
```

5. Wyznacz błąd klasyfikacji metodą sprawdzania krzyżowego z $v = 1$.

```
## pred_loo
##   1  2  3
## 1 49  5  5
## 2  5 61  5
## 3 10  8 30
```

```
## [1] 0.2134831
```

6. Wyznacz błąd klasyfikacji metodą sprawdzania krzyżowego z $v = 10$.

```
## [1] 0.2078652
```

7. Wyznacz błąd klasyfikacji metodą bootstrapową. Przyjmij $n_{boot} = 100$.

```
## [1] 0.2111531
```

8. Do których klas i z jakimi prawdopodobieństwami a posteriori należy zaklasyfikować poniższe nowe obserwacje?

V1	V2	V3
13.64	3.10	2.56
13.94	1.73	2.27
13.08	3.90	2.36
12.29	3.17	2.21

```
## $class
## [1] 1 1 3 2
## Levels: 1 2 3
##
## $posterior
##          1          2          3
## 1 0.531302523 0.007133455 0.46156402
## 2 0.924346812 0.007006399 0.06864679
## 3 0.061216479 0.054434582 0.88434894
## 4 0.005015639 0.810915785 0.18406858
##
## $x
##          LD1          LD2
## 1 -1.5435449  0.6390430
## 2 -1.5668588 -0.9252545
## 3 -0.2740389  1.6133507
## 4  1.4856206  1.0600594
```