

# Programowanie w języku R. Funkcje w R

Aleksander Denisiuk  
Uniwersytet Warmińsko-Mazurski  
Olsztyn, ul. Słoneczna 54  
[denisjuk@matman.uwm.edu.pl](mailto:denisjuk@matman.uwm.edu.pl)

18 maja 2025

# ***Funkcje w R***

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

Najnowsza wersja tego dokumentu dostępna jest pod adresem

<http://wmii.uwm.edu.pl/~denisjuk/uwm>

## Składowe funkcji

❖ Trzy składowe

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

# Składowe funkcji

# Trzy składowe

Składowe funkcji

❖ Trzy składowe

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- `body()` — kod funkcji
- `formals()` — lista argumentów
- `environment()` — „kontekst” funkcji

♦ jeżeli nie podany, to globalny

```
f <- function(x) x^2
```

```
f
```

```
formals(f)
```

```
body(f)
```

```
environment(f)
```

- Można użyć `body()`, `formals()`, `environment()` do modyfikacji funkcji

```
body(f) <- expression({x^3})
```

# Funkcje prymitywne

Składowe funkcji

❖ Trzy składowe

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Zaimplementowane w C, nie mają kodu w R
  - ✦ wykorzystują `.Primitive()`

```
sum
formals (sum)
body (sum)
environment (sum)
```

Składowe funkcji

**Zasięg**

❖ Lokalizacja  
zmiennych

❖ Domknięcia

❖ Dynamiczna  
lokalizacja



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

# Zasięg

# Lokalizacja zmiennych

Składowe funkcji

Zasięg

❖ Lokalizacja zmiennych

❖ Domknięcia

❖ Dynamiczna lokalizacja



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Jednostką syntaktyczną jest funkcja

```
f <- function() {  
  x <- 1  
  y <- 2  
  c(x, y)  
}  
f()  
rm(f)
```

# Decyduje włożoność

Składowe funkcji

Zasięg

❖ Lokalizacja  
zmiennych

❖ Domknięcia

❖ Dynamiczna  
lokalizacja



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
● x <- 2
  g <- function() {
    y <- 1
    c(x, y)
  }
  g()

● h <- function() {
  y <- 2
  i <- function() {
    z <- 3
    c(x, y, z)
  }
  i()
}
h()
rm(x, h, g)
```



# Inicjalizacja

Składowe funkcji

Zasięg

❖ Lokalizacja  
zmiennych

❖ Domknięcia

❖ Dynamiczna  
lokalizacja



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
j <- function() {  
  if (!exists("a")) {  
    a <- 1  
  } else {  
    a <- a + 1  
  }  
  a  
}  
j()  
j()  
rm(j)
```

# Domknięcia

Składowe funkcji

Zasięg

❖ Lokalizacja  
zmiennych

❖ Domknięcia

❖ Dynamiczna  
lokalizacja



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
j <- function(x) {  
  y <- 2  
  function() {  
    c(x, y)  
  }  
}  
k <- j(1)  
k()  
rm(j, k)
```

# ***Dynamiczna lokalizacja***

Składowe funkcji

Zasięg

❖ Lokalizacja  
zmiennych

❖ Domknięcia

❖ **Dynamiczna  
lokalizacja**



Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
f <- function() x
```

```
x <- 15
```

```
f()
```

```
x <- 20
```

```
f()
```



## Składowe funkcji

### Zasięg

- ❖ Lokalizacja zmiennych
- ❖ Domknięcia
- ❖ Dynamiczna lokalizacja



### Operatory

### Argumenty funkcji

### Fukcje specjalne

### Zwracane wartości

### Superpozycja

### Pakiety

### Case study

- R szuka funkcji spośród funkcji, nie zmiennych

```
n <- function(x) x / 2
o <- function() {
  n <- 10
  n(n)
}
o()
```

- Można zmienić definicje standardowych funkcji
  - ◆ nigdy tak nie robić

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

❖ Operatory  
a funkcje

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

# Operatory

# Operator a funkcje

[Składowe funkcji](#)

[Zasięg](#)

[Operator](#)

❖ Operator  
a funkcje

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

To understand computations in R, two slogans are helpful:

- Everything that exists is an object.
- Everything that happens is a function call.

— *John Chambers*

# Przykłady

Składowe funkcji

Zasięg

Operatory

❖ Operatory  
a funkcje

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
x <- 10; y <- 5
```

```
x + y
```

```
`+`(x, y)
```

```
for (i in 1:2) print(i)
```

```
`for`(i, 1:2, print(i))
```

```
if (i == 1) print("yes!") else print("no.")
```

```
`if`(i == 1, print("yes!"), print("no."))
```

```
x[3]
```

```
`[`(x, 3)
```

```
{ print(1); print(2); print(3) }
```

```
`{`(print(1), print(2), print(3))
```

# Operatory zamiast funkcji

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

❖ Operatory  
a funkcje

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

```
add <- function(x, y) x + y
sapply(1:10, add, 3)
sapply(1:5, `+`, 3)
sapply(1:5, "+", 3)
```

- `sapply()` może mieć jako argument funkcję albo jej nazwę
  - ◆ zobaczyć w kodzie

```
x <- list(1:3, 4:9, 10:12)
sapply(x, function(x) x[2])
sapply(x, "[", 2)
```





# ***Redefinicja operatorów***

Składowe funkcji

Zasięg

Operatory

❖ Operatory  
a funkcje

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Można zmienić definicję operatorów
  - ◆ nigdy tego nie robić

Składowe funkcji

Zasięg

Operatory

**Argumenty funkcji**

- ❖ Przekazywanie argumentów
- ❖ Lista argumentów
- ❖ Wartości domyślne
- ❖ Wartości pominięte
- ❖ Wartościowanie
- ❖ Argument . . .
- ❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

# Argumenty funkcji

# Przekazywanie argumentów

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument ...

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Według pełnej nazwy
- Według skrótu nazwy
- Według pozycji

```
f <- function(abcdef, bcde1, bcde2) {  
  list(a = abcdef, b1 = bcde1, b2 = bcde2)  
}
```

```
str(f(1, 2, 3))
```

```
str(f(2, 3, abcdef = 1))
```

```
str(f(2, 3, a = 1))
```

```
str(f(1, 3, b = 1))
```

# Reguły dobrego stylu

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument . . .

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Według pozycji tylko dla pierwszego (pierwszych dwóch) argumentów
  - ◆ oczywistych, powszechnie znanych
- Jeżeli skróty, to czytelne
- Jeżeli to moduł dla CRAN, to tylko pełna nazwa
- Argumenty po . . . tylko pełna nazwa

# Lista argumentów

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument . . .

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
args <- list(1:10, na.rm = TRUE)
do.call(mean, args)
```

```
mean(1:10, na.rm = TRUE)
```

# Wartości domyślne

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument ...

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

## ● Wartości domyślne

```
f <- function(a = 1, b = 2) {  
  c(a, b)  
}  
f()
```

## ● Zależne od poprzednich

```
g <- function(a = 1, b = a * 2) {  
  c(a, b)  
}  
g()  
g(10)
```

# Wartości pominięte

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ **Wartości pominięte**

❖ Wartościowanie

❖ Argument ...

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

## ● Funkcja `missing()`

```
i <- function(a, b) {  
  c(missing(a), missing(b))  
}  
i()  
i(b=2)  
i(2, 4)
```

## ● Inne podejście

```
g <- function(a = NULL, b = NULL) {  
  c(is.null(a), is.null(b))  
}  
g()  
g(b=2)  
g(1, 2)
```

# Wartościowanie leniwe

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

**❖ Wartościowanie**

❖ Argument ...

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Argumenty są obliczane w momencie, gdy są potrzebne

```
f <- function(x) {  
  10  
}  
f(stop("This is an error!"))
```

- Funkcja `force()`

```
f <- function(x) {  
  force(x)  
  10  
}  
f(stop("This is an error!"))
```



# Argumenty domyśle

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

**❖ Wartościowanie**

❖ Argument . . .

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- są obliczane wewnątrz funkcji
  - ✦ wynik może być inny, niż w globalnym kontekście

```
f <- function(x = ls()) {  
  a <- 1  
  x  
}
```

```
f()
```

```
f(ls())
```

# Wartościowanie leniwe w wyrażeniach

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

**❖ Wartościowanie**

❖ Argument . . .

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

```
if (!is.null(x) && x > 0) {
```

```
!is.null(a) || stop("a is null")
```

# Argument . . .

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument . . .

❖ Pytania

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- . . . pasuje do argumentów niedopasowanych w inny sposób
  - ◆ może zostać przekazany do innych funkcji
  - ◆ na przykład, `plot(x, y, . . .)`
  - ◆ literówki nie powodują błędu:  
`sum(1, 2, NA, na.mr = TRUE)`
- Odwołać się do . . . w programie można za pomocą listy:

```
f <- function(. . .) {  
  names(list(. . .))  
}  
f(a = 1, b = 2)
```

# Pytania

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

❖ Przekazywanie argumentów

❖ Lista argumentów

❖ Wartości domyślne

❖ Wartości pominięte

❖ Wartościowanie

❖ Argument ...

❖ **Pytania**

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

- Jaki będzie wynik obliczenia:

```
f1 <- function(x = {y <- 1; 2}, y = 0) {  
  x + y  
}  
f1()
```

- Jaki będzie wynik obliczenia:

```
f2 <- function(x = z) {  
  z <- 100  
  x  
}  
f2()
```

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

**Fukcje specjalne**

❖ Notacja infiksowa

❖ Funcje  
zastępujące

Zwracane wartości

Superpozycja

Pakiety

Case study

# Fukcje specjalne

# Notacja infiksowa

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

❖ Notacja infiksowa

❖ Funcje zastępujące

Zwracane wartości

Superpozycja

Pakiety

Case study

- Operatory standardowe: `% %`, `% * %`, `% / %`, `% in %`, `% o %`, `% x %`,  
`:`, `::`, `:::`, `$`, `@`, `^`, `*`, `/`, `+`, `-`, `>`, `>=`, `<`,  
`<=`, `==`, `!=`, `!`, `&`, `&&`, `|`, `||`, `~`, `<-`, `<<-`
- Operatory użytkownika powinny się rozpoczynać od `%`  
``%+%` <- function(a, b) paste0(a, b)`  
`"new" %+% " string"`
- W nazwach operatorów można używać znaków specjalnych  
``% %` <- function(a, b) paste(a, b)`  
``%'%` <- function(a, b) paste(a, b)`  
``%/\\%` <- function(a, b) paste(a, b)`  
`"a" % % "b"`  
`"a" %' % "b"`  
`"a" %/\\ % "b"`
- Operatory obliczane są od lewej strony

# Funcje zastępujące

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

❖ Notacja infiksowa

❖ Funcje  
zastępujące

Zwracane wartości

Superpozycja

Pakiety

Case study

- Modyfikują argumenty

- ❖ mają postać `xxx<-`

```
`second<-` <- function(x, value) {  
  x[2] <- value  
  x  
}
```

```
x <- 1:10
```

```
second(x) <- 5L
```

```
`second<-` (x, 5L)
```

- Jest tworzona kopia danych

- ❖ funkcje prymitywne nie kopiują danych

- `second(x) <- 5L` **vs** `x[2] <- 5L`

- ◆ `tracemem(x)`

# Dodatkowe argumenty

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

❖ [Notacja infiksowa](#)

❖ [Funcje zastępujące](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

- Między `x` `value`

```
`modify<-` <- function(x, position, value) {  
  x[position] <- value  
  x  
}
```

```
modify(x, 1) <- 10
```

```
`modify<-`(x, 1, 10)
```



Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

**Zwracane wartości**

- ❖ Wynik funkcji
- ❖ `return`
- ❖ Efekty uboczne
- ❖ Funkcje niewidoczne
- ❖ `on.exit()`

Superpozycja

Pakiety

Case study

# Zwracane wartości

# Wynik funkcji

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ `return`

❖ Efekty uboczne

❖ Funkcje niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

- Ostatnie obliczane wyrażenie

```
f <- function(x) {  
  if (x < 10) {  
    0  
  } else {  
    10  
  }  
}  
f(5)  
f(15)
```

- Tylko jeden obiekt

- ◆ może być lista

# *return*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ **return**

❖ Efekty uboczne

❖ Funkcje niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

- Funkcja `return()` zazwyczaj w przypadku błędów, bądź wcześniejszego końca obliczeń

```
f <- function(x, y) {  
  if (!x) return(y)  
  .....  
}
```

# ***Efekty uboczne***

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ `return`

❖ **Efekty uboczne**

❖ Funkcje niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

- Większość funkcji nie ma efektów ubocznych
  - ✦ zawsze taki sam wynik dla tych samych danych
- Modyfikacja argumentów funkcji nie jest możliwa

```
f <- function(x) {  
  x$a <- 2  
  x  
}  
  
x <- list(a = 1)  
f(x)  
  
x$a
```

- ✦ wyjątek: kontekst funkcji oraz klasy referencyjne

# *Funkcje mające efekty uboczne*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ `return`

❖ **Efekty uboczne**

❖ Funkcje niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

- `library()`
- `setwd()`, `Sys.setenv()`, `Sys.setlocale()`
- `plot()`
- `write()`, `write.csv()`, `saveRDS()`
- `options()` oraz `par()`
- Niektóre funkcje klas S4
- Generator liczb pseudolosowych

# Funkcje niewidoczne

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ `return`

❖ Efekty uboczne

❖ Funkcje  
niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

- Funkcja, która zwraca `invisible()` nie wyświetla wyniku

```
f1 <- function() 1
f2 <- function() invisible(1)

f1()
f2()

f1() == 1
f2() == 1

(f2())
a <- 2
(a <- 2)

a <- b <- c <- d <- 2
(a <- (b <- (c <- (d <- 2))))
```

# *on.exit()*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

❖ Wynik funkcji

❖ `return`

❖ Efekty uboczne

❖ Funkcje niewidoczne

❖ `on.exit()`

Superpozycja

Pakiety

Case study

## ● Działania po zakończeniu funkcji

```
in_dir <- function(dir, code) {  
  old <- setwd(dir)  
  on.exit(setwd(old))  
  
  force(code)  
}  
getwd()
```

- ◆ w razie wielu `on.exit()` dodać parameter `add = TRUE`

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

**Superpozycja**

❖ Superpozycja  
funkcji

❖ Superpozycja

❖ Kolejne wywołania

❖ Pipe

Pakiety

Case study

# Superpozycja



# Superpozycja funkcji

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

❖ Superpozycja funkcji

❖ Superpozycja

❖ Kolejne wywołania

❖ Pipe

[Pakiety](#)

[Case study](#)

- Odchylenie standardowe

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^m (x_i - \bar{x})^2}$$

- Implementacja z wykorzystaniem `mean()`, `sqrt()` oraz dodatkowych dwóch funkcji:

```
square <- function(x) x^2  
deviation <- function(x) x - mean(x)
```

# Superpozycja

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

❖ Superpozycja  
funkcji

❖ Superpozycja

❖ Kolejne wywołania

❖ Pipe

Pakiety

Case study

```
x <- runif(100)
sqrt(mean(square(deviation(x))))
```

- Krótkie wyrażenia, czyta się z prawej, kanapka Dagwooda



# Kolejne wywołania

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

❖ Superpozycja  
funkcji

❖ Superpozycja

❖ Kolejne wywołania

❖ Pipe

Pakiety

Case study

```
out <- deviation(x)
out <- square(out)
out <- mean(out)
out <- sqrt(out)
out
```

- Trzeba mianować pośrednie wyniki

# Potok (pipe)

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

❖ Superpozycja  
funkcji

❖ Superpozycja

❖ Kolejne wywołania

❖ Pipe

[Pakiety](#)

[Case study](#)

```
library(magrittr)
```

```
x %>%
```

```
  deviation() %>%
```

```
  square() %>%
```

```
  mean() %>%
```

```
  sqrt()
```

- $x \%>\% f(y) \iff f(x, y)$
- Zewnętrzny moduł, tylko liniowe potoki

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

**Pakiety**

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ `devtools`

❖ **Prawdziwy pakiet**

❖ `DESCRIPTION`

❖ `NAMESPACE`

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

# Pakiety

# Własne pakiety

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ **Własne pakiety**

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

- Ponowne wykorzystanie swoich funkcji
- Rozpowszechnienie danych oraz oprogramowania
  - ❖ CRAN
  - ❖ kod używanych w eksperymentach dotyczących artykułu (pracy magisterskiej)
- Pakiet nie musi być duży
- Nie ma obowiązku rozpowszechniania
- R`ozygen2` znacznie ułatwia napisanie dokumentacji

# Minimalny pakiet

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

## ● Struktura katalogów:

```
adscore
├── R
│   ├── confusionmatrix.r
│   └── f1score.r
└── DESCRIPTION
```

## ● Plik DESCRIPTION:

Package: `adscore`

Version: `0.1`

# Kompilacja

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ **Kompilacja**

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

- W katalogu nadrzędnym do `adscore`:  
`R CMD build adscore`
- Tworzony jest domyślny plik `NAMESPACE`
- Pakiet zapisywany jest do pliku `adscore_0.1.tar.gz`



# Wykorzystanie devtools

- [Składowe funkcji](#)
- [Zasięg](#)
- [Operatory](#)
- [Argumenty funkcji](#)
- [Funkcje specjalne](#)
- [Zwracane wartości](#)
- [Superpozycja](#)
- [Pakiety](#)
  - ❖ Własne pakiety
  - ❖ Minimalny pakiet
  - ❖ Kompilacja
  - ❖ **devtools**
  - ❖ Prawdziwy pakiet
  - ❖ DESCRIPTION
  - ❖ NAMESPACE
  - ❖ Dokumentacja
  - ❖ Licencja
  - ❖ weryfikacja
- [Case study](#)

- Można skorzystać z pakietu `devtools`
  - ✦ Debian Linux i pochodne:

```
$ sudo apt install r-cran-devtools
```
- Ustawić bieżący katalog na `adscore`

```
install.packages('devtools')  
library(devtools)  
build()  
install()
```
- Trzeba jednak mieć plik `NAMESPACE`

# Prawdziwy pakiet

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

## Dodatkowe czynności

- Uzupełnić plik DESCRIPTION
- Utworzyć plik NAMESPACE
- Dodać dokumentację
- Dodać licencję

# Uzupełnienie *DESCRIPTION*

<a href="#">Składowe funkcji</a>
<a href="#">Zasięg</a>
<a href="#">Operatory</a>
<a href="#">Argumenty funkcji</a>
<a href="#">Funkcje specjalne</a>
<a href="#">Zwracane wartości</a>
<a href="#">Superpozycja</a>
<a href="#">Pakiety</a>
❖ Własne pakiety
❖ <b>Minimalny pakiet</b>
❖ Kompilacja
❖ devtools
❖ <b>Prawdziwy pakiet</b>
❖ <b>DESCRIPTION</b>
❖ NAMESPACE
❖ Dokumentacja
❖ Licencja
❖ weryfikacja
<a href="#">Case study</a>

- Dodać co najmniej taką informację

Package: `adscore`

Version: `0.1`

Date: `2024-03-11`

Title: `A. D. Score`

Description: `Functions that Aleksander Denisiuk  
uses for computing F1score.`

Author: `Aleksander Denisiuk <denisiuk@matman.uwm.edu.pl>`

Maintainer: `Aleksander Denisiuk <denisiuk@matman.uwm.edu.pl>`

Encoding: `UTF-8`

- Tytuł ma być krótki
- Opisanie może być na kilka linii
- Dla autora email jest opcjonalny
- Kodowanie — dla Roxygen2

# Utworzenie *NAMESPACE*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ **NAMESPACE**

❖ Dokumentacja

❖ Licencja

❖ weryfikacja

Case study

- Minimalny plik

```
# Export all names  
exportPattern(".")
```

- Pierwszy wiersz to komentarz
- Już można instalować z devtools

# *Dokumentacja*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

- ❖ Własne pakiety
- ❖ **Minimalny pakiet**
- ❖ Kompilacja
- ❖ `devtools`
- ❖ **Prawdziwy pakiet**
- ❖ DESCRIPTION
- ❖ NAMESPACE
- ❖ **Dokumentacja**
- ❖ Licencja
- ❖ weryfikacja

Case study

- W katalogu `man`
- Format `Rd`
- Roxygen2

# Format Rd

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ **Dokumentacja**

❖ Licencja

❖ weryfikacja

[Case study](#)

```
\name{f1core}
\alias{f1core}
\title{Calculation of the F1 Score}
\usage{
  f1score( precision, recall )
}
\arguments{
\item{precision}{the value of precision}
\item{recall}{the value of recall}
\value{the correspondent f1score value}
\description{
  Calculates the value of F1 Score
}
\examples{
  f1score( precision, recall )
}
```

# Format Roxygen2

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ **Dokumentacja**

❖ Licencja

❖ weryfikacja

Case study

```
# ' Calculation of the F1 Score
# '
# ' Calculates the value of F1 Score
# '
# ' @param precision the value of precision
# ' @param recall the value of recall
# '
# ' @return the correspondent f1score value
# '
# ' @examples
# ' f1score( precision, recall )
# '
# ' @export

f1score <- function( precision, recall){
  2* (precision*recall) / (precision+recall)
}
```

# *kompilacja dokumentacji*

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ **Dokumentacja**

❖ Licencja

❖ weryfikacja

Case study

document ()

## ● Utworzenie pakietu:

```
setwd('adscore')  
library(devtools)  
document()  
build()  
install()
```



# Dodatkowe znaczniki Roxygen2

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ **Dokumentacja**

❖ Licencja

❖ weryfikacja

Case study

```
#' @author Aleksander Denisiuk,  
#'\email{denisiuk@matman.uwm.edu.pl}  
#'\@references  
#'\url{https://en.wikipedia.org/wiki/F-score}  
#'\@seealso \code{\link[MLmetrics]{F1_Score}}  
#'\@keywords flscore  
#'\dots
```

● etc

# Licencja

- Składowe funkcji
- Zasięg
- Operatory
- Argumenty funkcji
- Fukcje specjalne
- Zwracane wartości
- Superpozycja
- Pakiety
  - ❖ Własne pakiety
  - ❖ **Minimalny pakiet**
  - ❖ Kompilacja
  - ❖ devtools
  - ❖ **Prawdziwy pakiet**
  - ❖ DESCRIPTION
  - ❖ NAMESPACE
  - ❖ Dokumentacja
  - ❖ **Licencja**
  - ❖ weryfikacja
- Case study

- W pliku DESCRIPTION

- ◆ GPL-3

- License: GPL-3

- ◆ MIT

- License: MIT + file LICENSE

- W pliku LICENSE

- YEAR: 2014

- COPYRIGHT HOLDER: Aleksander Denisiuk

- ◆ Może być LICENSE

- ◆ Albo nawet inna nazwa

# Weryfikacja pakietu

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

❖ Własne pakiety

❖ **Minimalny pakiet**

❖ Kompilacja

❖ devtools

❖ **Prawdziwy pakiet**

❖ DESCRIPTION

❖ NAMESPACE

❖ Dokumentacja

❖ Licencja

❖ **weryfikacja**

Case study

- W wierszu poleceń:

```
R CMD check adscore_0.1.tar.gz
```

- ❖ Dodatkowa weryfikacja dla pakietów, które są ładowane na CRAN

```
R CMD check --as-cran adscore_0.1.tar.gz
```

- Z devtools:

```
check()
```

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

**Case study**

- ❖ Dane ciągłe: PCA
- ❖ Dane nominalne
- ❖ Implementacja
- ❖ Wyniki testów
- ❖ Bibliografia

# Redukcja wymiarowości dla danych nominalnych

# Dane ciągłe: PCA

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

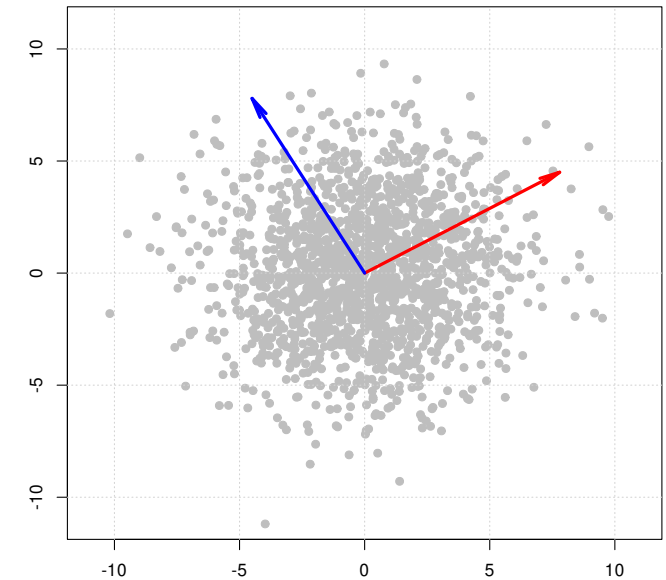
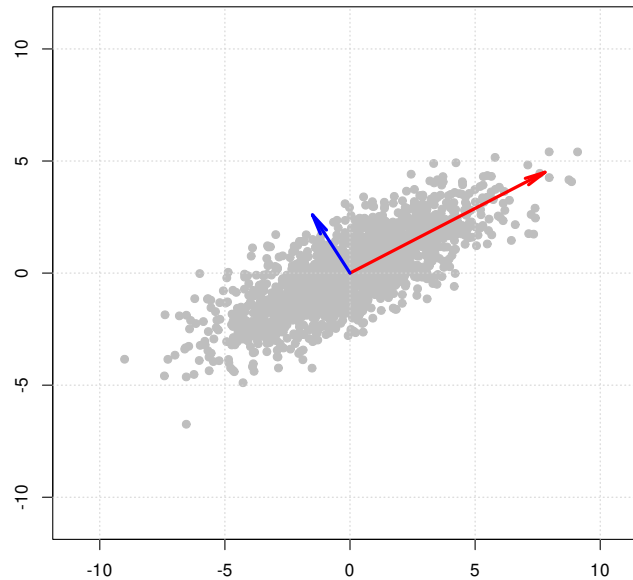
❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia



- Najmniej istotna składowa ma największy czynnik przy minimalizacji sumy wzajemnych odległości

# Dane nominalne

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

❖ Dane ciągłe: PCA

❖ **Dane nominalne**

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

- Dany jest zbiór  $\mathbb{X}$  z  $M$  jednostek, gdzie każda ma  $n$  cech nominalnych:  $\mathbb{X} \ni x = (x_1, \dots, x_n)$ .
- Metryka Hamminga: dla  $x, y \in \mathbb{X}$ :

$$\text{dist}_h(x, y) = \sum_{i=1}^n \text{diff}(x_i, y_i),$$

◆ gdzie

$$\text{diff}(\alpha, \beta) = \begin{cases} 1, & \text{if } \alpha \neq \beta, \\ 0, & \text{if } \alpha = \beta. \end{cases}$$

- Sumaryczna wzajemna odległość:

$$G = \frac{1}{M^2} \sum_{x, y \in \mathbb{X}} \text{dist}_h(x, y).$$

# Skalowanie

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

❖ Dane ciągłe: PCA

❖ **Dane nominalne**

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

- Wektor wag  $u = (u_1, \dots, u_n) \in \mathbb{R}^n$ , gdzie  $u_i \geq 0$ .
- Ważona metryka Hamminga:

$$\text{dist}_{h,u}(x, y) = \sum_{i=1}^n u_i \text{diff}(x_i, y_i).$$

# Sumaryczna wewnątrzklasowa wzajemna odległość:

- Dane podzielone są na klasy:  $\mathbb{X} = C_1 \cup \dots \cup C_c$ .
- Sumaryczna wewnątrzklasowa wzajemna odległość:

$$\begin{aligned} G(u) &= \frac{1}{M^2} \sum_{k=1}^c \sum_{x,y \in C_k} \text{dist}_{h,u}(x, y) \\ &= \frac{1}{M^2} \sum_{k=1}^c \sum_{x,y \in C_k} \sum_{i=1}^n u_i \text{diff}(x_i, y_i) \\ &= \sum_{i=1}^n u_i \left( \frac{1}{M^2} \sum_{k=1}^c \sum_{x,y \in C_k} \text{diff}(x_i, y_i) \right) = \sum_{i=1}^n s_i u_i, \end{aligned}$$

◆ gdzie

$$s_i = \frac{1}{M^2} \sum_{k=1}^c \sum_{x,y \in C_k} \text{diff}(x_i, y_i).$$

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

❖ Dane ciągłe: PCA

❖ **Dane nominalne**

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia



# Zagadnienie minimalizacji

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

❖ [Dane ciągłe: PCA](#)

❖ [Dane nominalne](#)

❖ [Implementacja](#)

❖ [Wyniki testów](#)

❖ [Bibliografia](#)



$$\begin{cases} \sum_{i=1}^n u_i s_i \rightarrow \min, \\ \sum_{i=1}^n u_i = 1, \quad u_i \geq 0 \text{ dla } i = 1, \dots, n \end{cases}$$



Programowanie liniowe:

- ❖ rozwiązanie  $u_{\text{opt}} = (0, \dots, 0, 1, 0, \dots, 0)$ ,
- ❖ cecha  $k$  (odpowiadająca jedynce) jest najmniej istotną.

# Algorytm

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

**Require:** wymiar zbioru danych  $\mathbb{X}$  jest  $n$ ,  $n > m$

**Ensure:** wymiar zbioru danych  $\mathbb{X}$  jest  $n - m$

$s \leftarrow 0$

**while**  $s < m$  **do**

rozwiązać zagadnienie optymalizacji i wyeliminować  
najmniej istotną cechę

$s \leftarrow s + 1$

**end while**

# Obliczenie $s_i$

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ **Implementacja**

❖ Wyniki testów

❖ Bibliografia

$$s_i = \frac{1}{M^2} \sum_{k=1}^c \sum_{x,y \in C_k} \text{diff}(x_i, y_i).$$

```
sum.diff <- function(x, v) {  
  sum(v != x)  
}
```

```
sum.sum.diff <- function(v) {  
  apply(v, function(x) {sum.diff(x, v) })  
}
```

```
calc.s.i<- function(data, clusters) {  
  x<-lapply(split(data, clusters),  
    function(x) { apply(x, sum.sum.diff) })  
  M <- nrow(clusters)  
  rowSums(apply(x, colSums)) / (M*M)  
}
```

# Zagadnienie optymalizacji

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

$$\begin{cases} \sum_{i=1}^n u_i s_i \rightarrow \min, \\ \sum_{i=1}^n u_i = 1, \quad u_i \geq 0 \text{ dla } i = 1, \dots, n \end{cases}$$

```
require(lpSolve)
```

```
calc.opt <- function(data, clusters){  
  s.i <- calc.s.i(data, clusters)  
  n <- ncol(data)  
  constr1 <- matrix(1, nrow = 1, ncol=n,  
    byrow = TRUE)  
  constr <- rbind(constr1, diag(n))  
  opt <- lp("min", s.i, constr,  
    const.dir=c("=", rep(">=", n)),  
    const.rhs=c(1, rep(0, n)) )  
  which.max(opt$solution)  
}
```

# Ustawiane cech w odpowiedniej kolejności

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

```
m <- ncol(train.nom.data)
```

```
opt.first <- calc.opt(train.nom.data,  
                      train.class.data)
```

```
indices <- 1:m
```

```
opt <- opt.first
```

```
indices[opt.first] = 0;
```

```
for(i in 2:(m-1)) {  
  opt.n <- calc.opt(train.nom.data[-opt],  
                   train.class.data)  
  opt.next <- indices[indices>0][opt.n]  
  indices[opt.next] = 0;  
  opt <- c(opt, opt.next)  
}
```

```
opt.last <- which.max(indices)
```

```
opt <- c(opt, opt.last)
```

# Car Evaluation

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

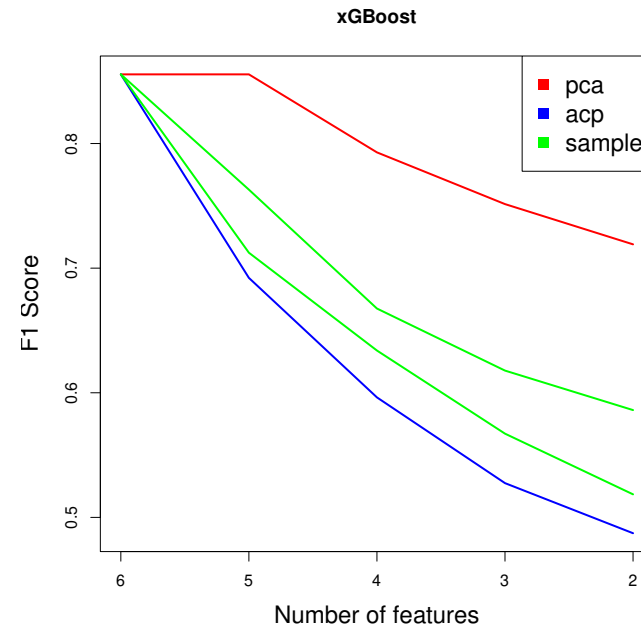
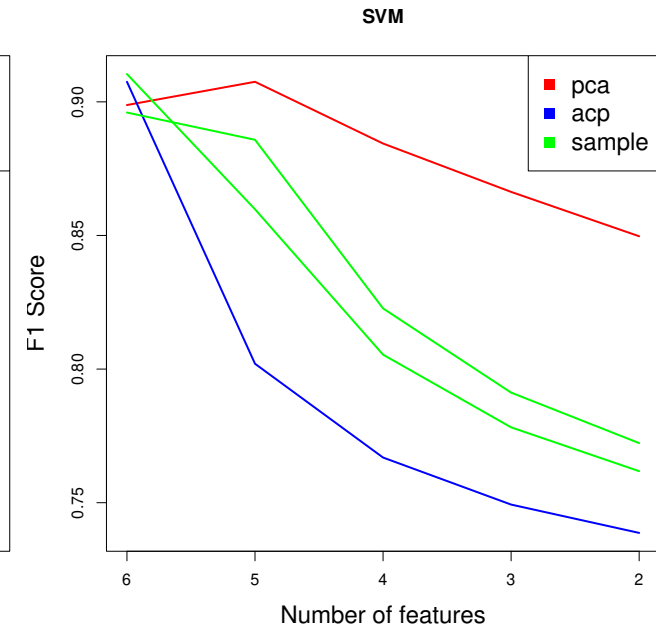
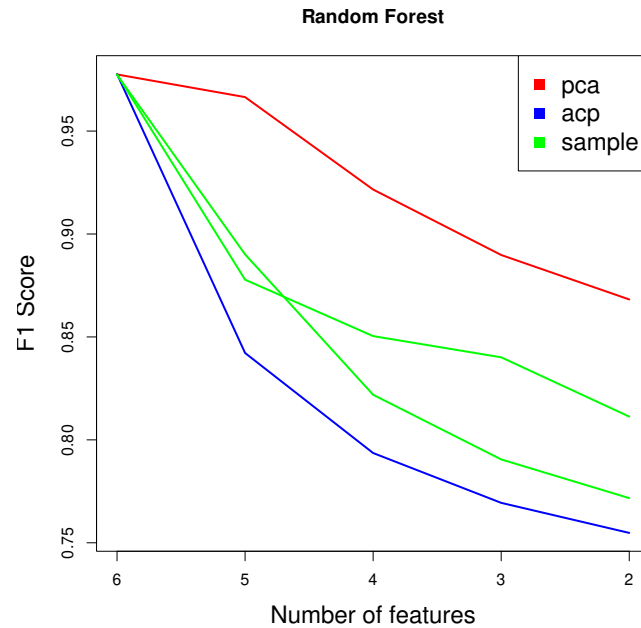
❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia



# Congressional Voting Records

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

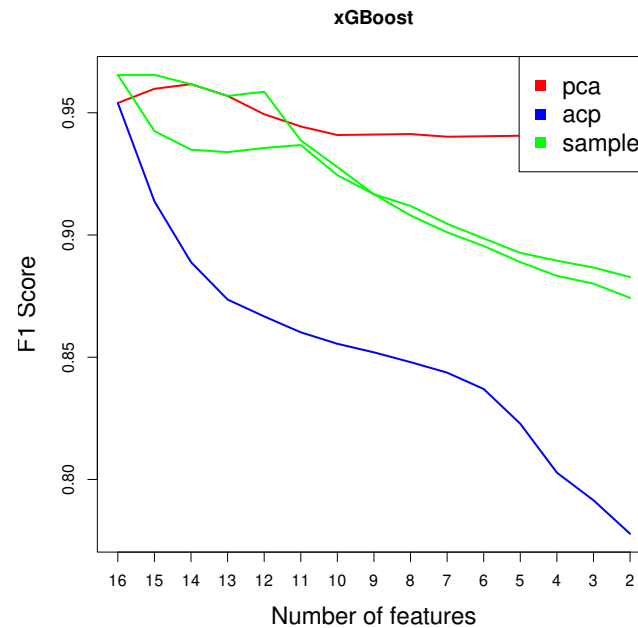
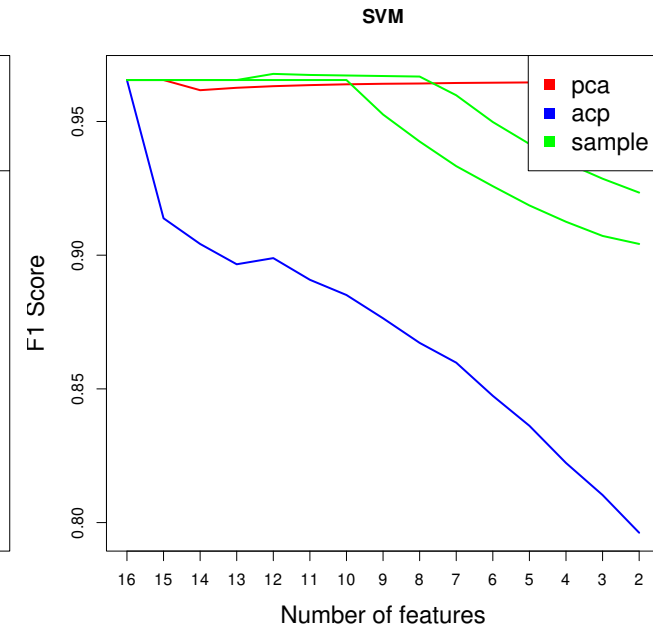
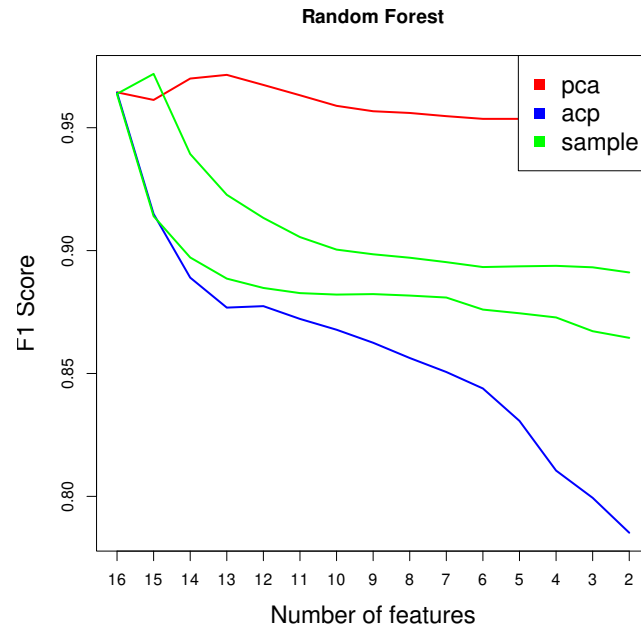
❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia



# Tic-Tac-Toe Endgame

[Składowe funkcji](#)

[Zasięg](#)

[Operatory](#)

[Argumenty funkcji](#)

[Funkcje specjalne](#)

[Zwracane wartości](#)

[Superpozycja](#)

[Pakiety](#)

[Case study](#)

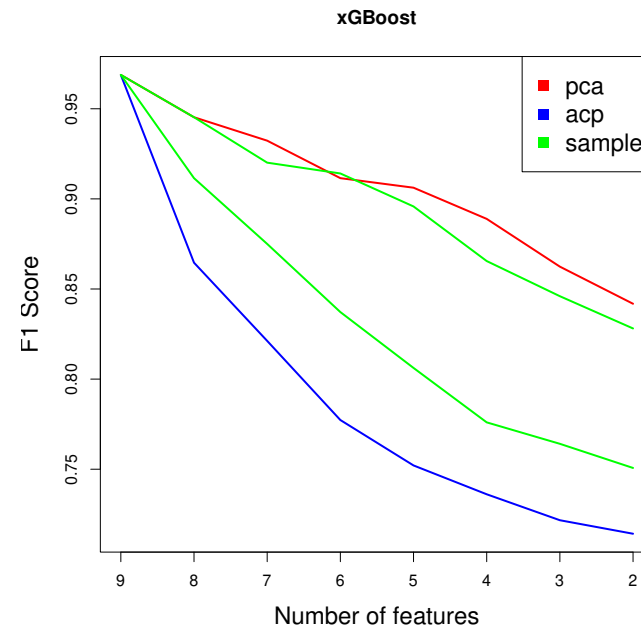
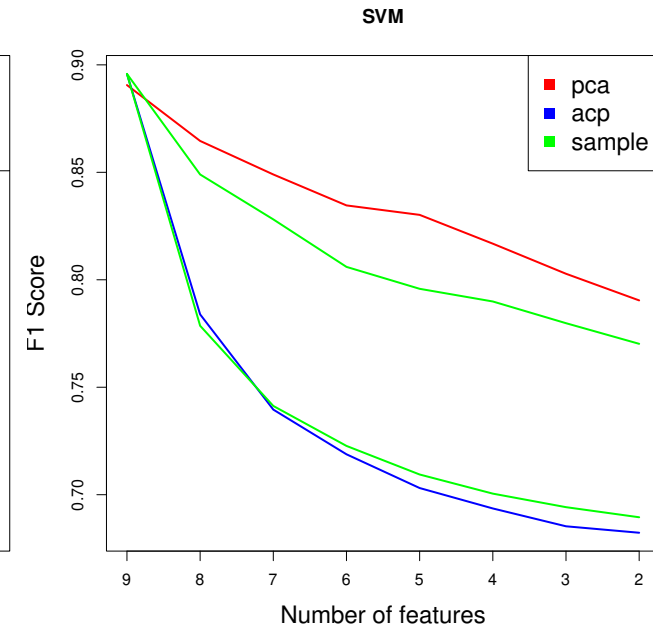
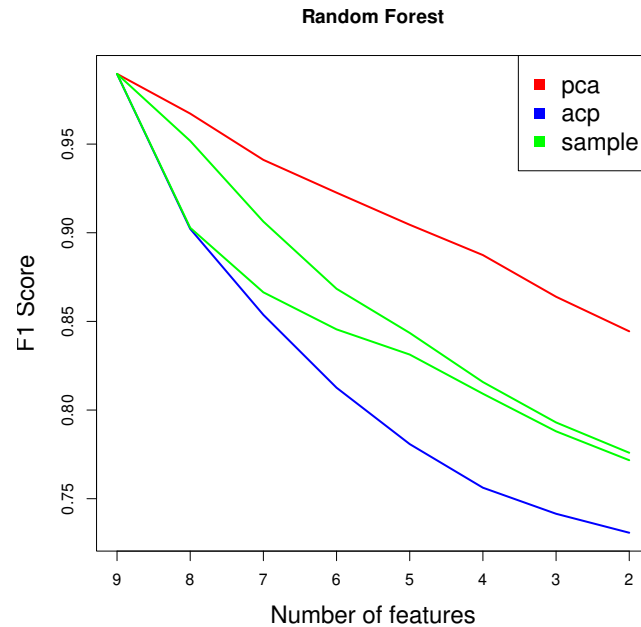
❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia





# Bibliografia

Składowe funkcji

Zasięg

Operatory

Argumenty funkcji

Fukcje specjalne

Zwracane wartości

Superpozycja

Pakiety

Case study

❖ Dane ciągłe: PCA

❖ Dane nominalne

❖ Implementacja

❖ Wyniki testów

❖ Bibliografia

1. ALEKSANDER DENSIUK: *PCA Dimensionality Reduction for Categorical Data*. In: Franco, L., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds) *Computational Science – ICCS 2024. ICCS 2024. Lecture Notes in Computer Science, vol 14834*. Springer, Cham.  
[https://doi.org/10.1007/978-3-031-63759-9\\_22](https://doi.org/10.1007/978-3-031-63759-9_22)
2. Projekt: <https://gitlab.com/adenisiuk/pca>